Adaptive Metric Nearest Neighbor Classification

Carlotta Domeniconi
Computer Science Department
University of California
Riverside, CA 92521
carlotta@cs.ucr.edu

Jing Peng
Computer Science Department
Oklahoma State University
Stillwater, OK 74078
jpeng@cs.okstate.edu

Dimitrios Gunopulos [*]
Computer Science Department
University of California
Riverside, CA 92521
dg@cs.ucr.edu

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 tend to be 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 a variety of simulated and real world data.*

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, $L_j, j = 1, \ldots, J$. The goal is to predict the class label of a given query $\mathbf{x}_0$.

The K nearest neighbor classification method [3, 8, 9, 10] 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 so that all points in the neighborhood are close to the query. In addition, it has been shown [4, 5] that the one nearest neighbor rule has asymptotic error rate that is at most twice the Bayes error rate, independent of the distance metric used.

[*]This research was supported by NSF IIS-9907477 and the US Dept. of Defense.

The nearest neighbor rule becomes less appealing with finite training samples, however. This is due to the curse-of-dimensionality [2]. Severe bias can be introduced in the nearest neighbor 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 nearest neighbor classification [6, 7, 9]. The commonly used Euclidean distance measure, while simple computationally, implies that the input space is isotropic. However, the assumption for isotropy is often invalid and generally undesirable in many practical applications. This implies that distance computation does not vary with equal strength 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 high dimensional settings.

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

2 Local Feature Relevance Measure

Kernel methods are based on the assumption of 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 $\mathbf{x}_0$ approaches class boundaries. In the following, we describe a nearest neighbor classification 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.

Our technique is motivated as follows. Let $\mathbf{x}_0$ be the test point whose class membership we are predicting. In the one nearest neighbor classification rule, a single nearest neighbor $\mathbf{x}$ is found according to a distance metric $D(\mathbf{x}, \mathbf{x}_0)$. Let $p(j|\mathbf{x})$ be the class conditional probability at point $\mathbf{x}$. Consider the weighted *Chi-squared* distance [7, 11]

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

which measures the distance between the test point $\mathbf{x}_0$ and the point $\mathbf{x}$, in terms of the difference between the class posterior probabilities at the two points. Small $D(\mathbf{x}, \mathbf{x}_0)$ indicates that the classification error rate will be close to the asymptotic error rate for one nearest neighbor. In general, this can be achieved when $\Pr(j|\mathbf{x}) = \Pr(j|\mathbf{x}_0)$, which states that if $\Pr(j|\mathbf{x})$ can be sufficiently well approximated at $\mathbf{x}_0$, the asymptotic 1-NN error rate might result in finite sample settings.

Equation (1) computes the distance between the true and estimated posteriors. Now, imagine we replace $\Pr(j|\mathbf{x}_0)$ with a quantity that attempts to predict $\Pr(j|\mathbf{x})$ under the constraint that the quantity is conditioned at a location along a particular feature dimension. Then, the *Chi-squared* distance (1) tells us the extent to which that dimension can be relied on to predict $\Pr(j|\mathbf{x})$. Thus, Equation (1) provides us with a foundation upon which to develop a theory of feature relevance in the context of pattern classification.

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 $p(j|\mathbf{x})$, 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,

$$\begin{aligned} \overline{\Pr}(j|x_i = z) &= E[\Pr(j|\mathbf{x})|x_i = z] \\ &= \int \Pr(j|\mathbf{x}) p(\mathbf{x}|x_i = z) d\mathbf{x} \end{aligned} \tag{2}$$

Here $p(\mathbf{x}|x_i = z)$ is the conditional density of the other input variables. 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 captured 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.

The relative relevance, as a weighting scheme, can then be given by $w_i(\mathbf{x}_0) = (1/\bar{r}_i(\mathbf{x}_0))^t / \sum_{l=1}^{q} (1/\bar{r}_l(\mathbf{x}_0))^t$, where $t = 1, 2$, giving rise to linear and quadratic weightings, respectively. In this paper we propose the following exponential weighting scheme

$$w_i(\mathbf{x}_0) = \exp(c \frac{1}{\bar{r}_i(\mathbf{x}_0)}) / \sum_{l=1}^{q} \exp(c \frac{1}{\bar{r}_l(\mathbf{x}_0)}) \tag{5}$$

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 $w_i = 1/q$, thereby ignoring any difference between 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 (4) and gives rise to better performance improvement. Thus, (5) can be used as weights associated with features for weighted distance computation

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

These weights enable the neighborhood to elongate less important feature dimensions, and to constrict the most influential ones. Note that the technique is *query-based* because weightings depend on the query [1].

3 Estimation

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.

```
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) through Equations (7) and (8);
  4. Update $\mathbf{w}$ according to (5);
  5. Iterate steps 2, 3, and 4;
  6. At completion, use $\mathbf{w}$, hence (6), for $K$-nearest neighbor classification at the test point $\mathbf{x}_0$.
```

Figure 1: The ADAMENN algorithm

To compute $\overline{\Pr}(j|x_i = z) = E[\Pr(j|\mathbf{x})|x_i = z]$, we introduce a dummy variable g_j such that if $y = j$, then $g_j|\mathbf{x} = 1$, otherwise $g_j|\mathbf{x} = 0$, where $j = 1, \cdots, J$. 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 $E[g_j|x_i = z]$. In details, 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.

4 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 $N_1(\mathbf{z})$ for estimation (7); K_2: the size of the neighborhood $N_2(\mathbf{z})$ for each of the K_0 neighbors for estimation (8); L: the number of points within the Δ intervals; K: the number of neighbors in the final nearest neighbor rule; and c: the positive factor for the exponential weighting scheme (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 both a linear and an exponential weighting scheme. 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, the resulting $\mathbf{w}$ is plugged in (6) to compute nearest neighbors at the test point $\mathbf{x}_0$.

In all our experiments we obtained optimal performance for small values (one or three) of parameters K_1 and K. Optimal values for parameters K_0 and K_2 are in a range close to respectively 10% and 15% of the number of training points. The value for L is usually set to be roughly half the value of K_2. Different values of the c factor turned out to be optimal for different problems (5, 11, and 16). An outline of the ADAMENN algorithm is shown in Figure 1.

5 Empirical Results

We compare the following classification methods, using both simulated and real data: (1) ADAMENN-adaptive metric nearest neighbor described in Figure 1 (one iteration), coupled with the exponential weighting scheme (5); (2) i-ADAMENN-adaptive metric nearest neighbor with five iterations; (3) Simple K-NN method using the Euclidean distance measure; (4) C4.5 decision tree method [12]; (5) Machete [6]: it 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 equation (9); and (6) Scythe [6]: it is a generalization of the machete algorithm, in which the input variables influence each split in proportion to their estimated local relevance, rather than the winner-take-all strategy of the machete.

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.

5.1 Experiments on Simulated Data

For all simulated data, 20 independent training samples (of size N) 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 10000 such classifications are reported in Table 1.

5.1.1 The Problems

1. This problem is taken from [6], and designed to be favorable to the adaptive methods (ADAMENN, scythe, machete, C4.5), and unfavorable to the regular K-NN procedure. There are $q = 10$ input features, $N = 200$ training data, and $J = 2$ classes. Data for the first class are generated from a standard normal distribution $\mathbf{x}_n \sim N(\mathbf{0}, \mathbf{1})$. Data for the second class are also generated from a normal distribution $\mathbf{x}_n \sim N(\mathbf{m}, \mathbf{C})$, with the coordinate mean values and covariance matrix given by $\{m_i = \sqrt{i}/2\}_{i=1}^{q}$, $\mathbf{C} = diag\{1/\sqrt{i}\}_{i=1}^{q}$. Although all input variables are relevant, the ones with higher coordinate number i are more so. The first column of Table 1 shows the results for the six methods under comparison.
2. This problem is taken from [6], and it is designed to be more favorable to the K-NN procedure. There are $q = 10$ input features, $N = 200$ training data, and $J = 2$ classes. Data for the first class are generated from a standard normal distribution. Data for the second class are also normal with the same covariance matrix as in the previous problem, and mean values given by $\{m_i = \sqrt{q - i + 1}/2\}_{i=1}^{q}$. Although the variances in the first problem are most different between the two classes on the high coordinates, their means now are most separated in the lower coordinates, so that all variables contain substantial discriminating information. The second column of Table 1 shows the results for this problem.
3. This problem is adapted from [7], and consists of four dimensional spheres with 6 noise features. There are $q = 10$ input features, $N = 200$ training data, and $J = 2$ classes. The last 6 features are noise variables, with standard Gaussian distributions, independent of each other and the class membership. The data for both classes are generated from a standard normal distribution. The data for class one have the property that the radius, computed from the first four features, is greater than 1.85 while the data for class two do not have such restriction. Class one basically surrounds class two in the subspace spanned by the first four features. Results are shown in the second column of Table 1.
4. This example is also taken from [6]. It is designed to be more favorable to the K-NN procedure, since all the input variables have the same global relevance. As before there are $q = 10$ input features and $J = 2$ classes, but $N = 500$ training data. The data for both classes are generated from a standard normal distribution $\mathbf{x}_n \sim N(\mathbf{0}, \mathbf{1})$, and the classes are defined by $\sum_{i=1}^{10} x_i^2 \leq 9.8 \Rightarrow class1, \quad otherwise \Rightarrow class2$. The fourth column of Table 1 shows the results for this example.
5. This example is again taken from [6]. It is constructed so that all input variables have equal local relevance everywhere in the input space. However, there is a single direction in the space that contains all the discriminant information. There are $q = 10$ input features, $N = 200$ training data, and $J = 2$ classes defined by $\sum_{i=1}^{10} x_i \leq 0 \Rightarrow class1, \quad otherwise \Rightarrow class2$. Results are shown in the third column of Table 1.

5.1.2 Results

Table 1 shows that, for each method, there is at least one example for which it has the best performance, or close to the best. Therefore, it seems natural to ask the question of robustness. That is, how well a particular method m performs on average in situations that are most favorable to other procedures. Following Friedman [6], we capture robustness by computing the ratio b_m of its error rate e_m and the smallest error rate over all methods being compared in a particular example: $b_m = \frac{e_m}{min_{1 \leq k \leq 6} e_k}$. Thus, the best method m^* for that example has $b_{m^*} = 1$, and all other methods have larger values $b_m \geq 1$, for $m \neq m^*$. The distribution of the b_m values for each method m over all the examples, therefore, seems to be a good indicator concerning its robustness.

Table 1: Average classification error rates for simulated data.

	Ex1	Ex2	Ex3.	Ex4	Ex5
ADAMENN	9.9	7.3	23.9	33.7	20.8
i-ADAMENN	8.3	6.3	23.1	33.7	20.3
K-NN	14.7	10.4	33.9	36.1	18.1
C4.5	10.3	11.5	14.6	30.6	30.1
Machete	7.1	6.9	21.7	33.0	25.7
Scythe	7.9	6.6	25.6	32.7	22.2

Figure 2 plots the distribution of b_m for each method over the five simulated 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. It is clear that the most robust method over the simulated data is i-ADAMENN. In 4/5 of the data its error rate was no worse than 17% higher than the best error rate. In the worst case it was 58%. In contrast, K-NN has the worst distribution, where the corresponding numbers are 107% and 132%.

5.2 Experiments on Real Data

In this section we examine the performance of the competing classification methods using real world data.

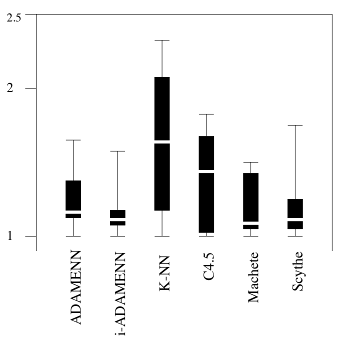

Figure 2: Performance distributions for simulated data.

In our experiments we used five different real data sets. The Iris, Sonar, Vowel, and Glass data are taken from UCI Machine Learning Repository at http://www.cs.uci.edu/ mlearn/MLRepository.html. The Image data are obtained from MIT Media Lab at ftp://whitechapel.media.mit.edu/pub/VisTex. For the Iris, Sonar, and Glass data we perform leave-one-out cross-validation to measure performance. On 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. Table 2 shows the cross-validated error rates for the six methods under consideration on the five real data.

5.2.1 The Problems

1. Iris data. This data set consists of $q = 4$ measurements made on each of $N = 100$ iris plants of $J = 2$ species. The problem is to classify each test point to its correct species based on the four measurements. The results on this data set are shown in the first column of Table 2.
2. Sonar data. This data set consists of $q = 60$ frequency measurements made on each of $N = 208$ data of $J = 2$ classes ("mines" and "rocks"). The problem is to classify each test point in the 60-dimensional feature space to its correct class. The results on this data set are shown in the second column of Table 2.
3. Vowel data. This example has $q = 10$ measurements and $J = 11$ classes. There are $N = 528$ samples in this example. Results are shown in the third column of Table 2.
4. Glass data. This data set consists of $q = 9$ chemical attributes measured for each of $N = 214$ data of $J = 6$ classes. The problem is to classify each test point in the 9-dimensional space to its correct class. Results are shown in the fourth column of Table 2.
5. Image data. This data set consists of 640 images of 15 classes. The number of images in each class varies from 16 to 80. The images in this database are represented by $q =$ 16 dimensional feature vectors (8 Gabor filters: 2 scales and 4 orientations). Results are shown in the fifth column of Table 2.

Table 2: Average classification error rates for real data.

	Iris	Sonar	Vowel	Glass	Image
ADAMENN	4.0	9.1	10.7	24.8	5.2
i-ADAMENN	5.0	9.6	10.9	24.8	5.2
K-NN	6.0	12.5	11.8	28.0	6.1
C4.5	8.0	23.1	36.7	31.8	21.6
Machete	5.0	21.2	20.2	28.0	12.3
Scythe	4.0	16.3	15.5	27.1	6.1

5.2.2 Results

Table 2 shows that ADAMENN achieved the best performance over the real data sets, followed by i-ADAMENN. As shown in Figure 3, the spread of the error distribution for ADAMENN is zero, and the spread for i-ADAMENN is narrow and close to 1. The results clearly demonstrate that they obtained the most robust performance over these data sets. Similar characteristics were also observed for the two methods over the simulated data sets. 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 [6].

6 Related Work

Friedman [6] describes an approach 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 from a reduction in prediction error given the query's value along that dimension. This method performs well on a number of classification tasks. In our notations, the reduction in prediction error can be described by

$$I_i^2(\mathbf{z}) = \sum_{j=1}^{J}(\overline{\Pr}(j) - \overline{\Pr}(j|x_i = z_i)])^2, \tag{9}$$

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 the particular point $x_i = z$. In this case, the most informative input variable is the one that gives the largest deviation from $\overline{\Pr}(j)$.

The main difference, however, between our relevance measure (4) and Friedman's (9) 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. As a result, a feature dimension is more relevant than others when it minimizes (3) in case of our relevance measure, whereas when it maximizes (9) in case of Friedman's. Furthermore, we take into account not only the test point $\mathbf{x}_0$ itself, but also its K nearest neighbors, resulting in a relevance measure (4) that is in general more robust.

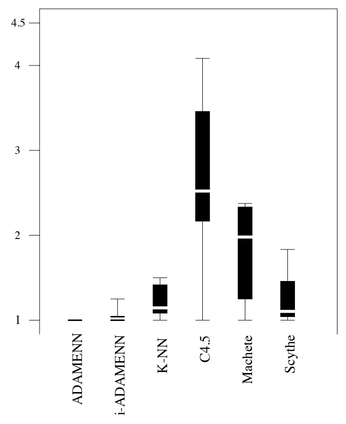

Figure 3: Performance distributions for real data.

In [7], 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 *Chi-squared* distance (1) by a Taylor series expansion. While sound in theory, the method is quite limited in practice. The main concern 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 serve as a potential bridge between Friedman's and that of Hastie and Tibshirani.

7 Summary and Conclusions

This paper presents an adaptive nearest neighbor 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 in 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. The results are also in favor of ADAMENN over other adaptive methods such as machete.

References

[1] C. Atkeson, A.W. Moore, and S. Schaal, "Locally Weighted Learning," *AI Review*. 11:11-73. 1997.

[2] R.E. Bellman, *Adaptive Control Processes*. Princeton Univ. Press, 1961.

[3] W.S. Cleveland and S.J. Devlin, "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *J. Amer. Statist. Assoc.* **83**, 596-610, 1988

[4] T.M. Cover and P.E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. on Information Theory*, pp. 21-27, 1967.

[5] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., 1973.

[6] J.H. Friedman "Flexible Metric Nearest Neighbor Classification," Tech. Report, Dept. of Statistics, Stanford University, 1994.

[7] T. Hastie and R. Tibshirani, "Discriminant Adaptive Nearest Neighbor Classification", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 6, pp. 607-615, 1996.

[8] T.K. Ho, "Nearest Neighbors in Random Subspaces," *Lecture Notes in Computer Science: Advances in Pattern Recognition*, pp. 640-648, 1998.

[9] D.G. Lowe, "Similarity Metric Learning for a Variable-Kernel Classifier," *Neural Computation* **7**(1):72-85, 1995.

[10] G.J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*. New York: Wiley, 1992.

[11] 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.

[12] J.R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan-Kaufmann Publishers, Inc., 1993.