# LDA/SVM Driven Nearest Neighbor Classification

Jing Peng
Electrical Engineering & Computer Science Dept.
Tulane University
New Orleans, LA 70118
jp@eecs.tulane.edu

Douglas R. Heisterkamp & H.K. Dai
Computer Science Dept.
Oklahoma State University
Stillwater, OK 74078
{doug,dai}@cs.okstate.edu

## Abstract

*Nearest neighbor classification relies on the assumption that class conditional probabilities are locally constant. This assumption becomes false in high dimensions with finite samples due to the curse of dimensionality. The nearest neighbor rule introduces severe bias under these conditions. We propose a locally adaptive neighborhood morphing classification method to try to minimize bias. We use local support vector machine learning to estimate an effective metric for producing neighborhoods that are elongated along less discriminant feature dimensions and constricted along most discriminant ones. As a result, the class conditional probabilities can be expected to be approximately constant in the modified neighborhoods, whereby better classification performance can be achieved. The efficacy of our method is validated and compared against other competing techniques using a number of data sets.*

## 1. Introduction

In classification, a feature vector $\mathbf{x} = (x_1, \cdots, x_n)^t \in \Re^n$, representing an object, is assumed to be in one of $J$ classes $\{i\}_{i=1}^J$, and the objective is to build classifier machines that assign $\mathbf{x}$ to the correct class from a given set of $l$ training samples.

$K$ nearest neighbor (NN) classification methods [3, 4, 5, 7, 8, 10, 11, 13, 16] are a simple and attractive approach to this problem. 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. Furthermore, empirical evaluation to date shows that the KNN rule is a rather robust method. In addition, it has been shown [6] that the one NN rule has asymptotic error rate that is at most twice the Bayes error rate, independent of the distance metric used.

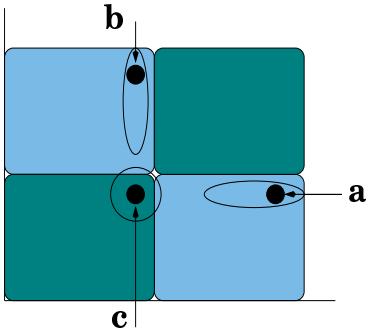


**Figure 1. Feature relevance varies with query locations.**

The nearest neighbor rule becomes less appealing in finite training samples, however. This is due to the curse-of-dimensionality [2]. Severe bias can be introduced in the 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 nearest neighbor classification. The commonly used Euclidean distance measure implies that the input space is isotropic or homogeneous. However, the assumption for isotropy is often invalid in many practical applications. Figure 1 illustrates a case in point. For query $a$, the horizontal coordinate is more relevant, because a slight move along that axis may change the class label, while for query $b$, the vertical coordinate is more relevant. For query $c$, however, both coordinates are equally relevant. This implies that 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 high dimensional settings.

In this paper we propose an adaptive neighborhood morphing classification method to try to minimize bias in high dimensions. We estimate an effective local metric for com-

puting neighborhoods based on local Support Vector Machines (SVMs), which have been successfully used as a classification tool in a number of areas, ranging from object recognition to classification of cancer morphologies. The resulting neighborhoods are highly adaptive to query locations. Moreover, the neighborhoods are elongated along less relevant (discriminant) 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. Related Work

Friedman [7] describes an approach to learning local feature relevance that 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, local relevance can be described by

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

where $\overline{\Pr}(j)$ represents the expected value of $\Pr(j|\mathbf{x})$, and $\overline{\Pr}(j|x_i = z_i)$ the conditional expectation of $\Pr(j|\mathbf{x})$, given that $x_i$ assumes value $z_i$. This measure reflects the influence of the $i$th input variable on the variation of $\Pr(j|\mathbf{x})$ at the particular point $x = z$. In this case, the most informative dimension is the one that deviates the most from $\overline{\Pr}(j)$.

The main difference, however, between our relevance measure to be described in section 3.2 and Friedman's (1) is that a feature dimension is more relevant if it is most discriminanting in case of our relevance measure, whereas it varies most in case of Friedman's. As a result, our measure is more informative than Friedman's.

Hastie and Tibshirani [8] 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 by a Taylor series expansion, given that class densities are Gaussian and have the same covariance matrix. While sound in theory, the method has limitations. The main concern is that in high dimensions we may never have sufficient data to locally fill in $n \times n$ within and between sum-of-squares matrices.

Amari and Wu [1] describe a method for improving SVM performance by increasing spatial resolution around the decision boundary surface based on the Riemannian geometry. The method first trains a SVM with an initial kernel that is then modified from the resulting set of support vectors and a quasiconformal mapping. A new SVM is built using the new kernel. Viewed under the same light, our technique can be regarded as a way to increase spatial resolution around the separating hyperplane in a local fashion. However, our technique varies spatial resolution judiciously in that it increases spatial resolution along discriminanting directions, while decreasing spatial resolution along less discriminant ones.

Weston et al. [15] propose a technique for feature selection for SVMs to improve generalization performance. In their technique, a feature is either completely relevant or completely irrelevant. Clearly, feature importance as such is non-local, and therefore, insensitive to query locations. In addition, these global relevance techniques usually do not work well on tasks that exhibit local feature differential relevance, as evidenced by the example shown in Figure 1.

## 3. Feature Relevance

Our technique is motivated as follows. In linear discriminant analysis (for $J = 2$), data are projected onto a single dimension where class label assignment is made for a given input query. This dimension is computed according to $\mathbf{w} = \mathbf{W}^{-1}(\mu_1 - \mu_2)$ where $\mathbf{W}$ denotes the within sum-of-squares matrix, and $\mu_i$ the class means. The vector $\mathbf{w}$ represents the same direction as the discriminant in the Bayes classifier along which the data have the maximum separation. Furthermore, any direction, $\mathbf{\Theta}$, whose dot product with $\mathbf{w}$ is large, also carries discriminant information. The larger $|\mathbf{w} \cdot \mathbf{\Theta}|$ is, the more discriminant information that $\mathbf{\Theta}$ captures. State it differently, if we transform $\mathbf{\Theta}$ via $\mathbf{\Theta}^{new} = \mathbf{W}^{-1/2}\mathbf{\Theta}^{old}$, then in the transformed space, any direction $\mathbf{\Theta}^{new}$ close to $\mathbf{W}^{-1}(\mu_1 - \mu_2)$ provides discriminant information.

In particular, when $\mathbf{\Theta}$ is restricted to the feature axes, i.e.,

$$\mathbf{\Theta} \in \{\mathbf{e}_1, \cdots, \mathbf{e}_n\}, \qquad (2)$$

where $\mathbf{e}_i$ is a unit vector along the $i$th feature, the value of $|\mathbf{w} \cdot \mathbf{\Theta}|$ measures the degree of relevance of feature dimension $\mathbf{\Theta}$ in providing class discriminant information. It thus seems natural to associate, when $\Theta = \mathbf{e}_i$,

$$\mathbf{w}_i = \mathbf{w} \cdot \mathbf{\Theta}, \qquad (3)$$

as a weight, with each dimension $\mathbf{\Theta}$ in a weighted nearest neighbor rule. Now imagine for each input query we compute $\mathbf{w}$ locally, from which to induce a new neighborhood for the final classification of the query. In this case, large $|\mathbf{w} \cdot \mathbf{\Theta}|$ enables the shape of neighborhood to constrict along $\mathbf{\Theta}$, while small $|\mathbf{w} \cdot \mathbf{\Theta}|$ enlongates the neighborhood along the $\mathbf{\Theta}$ direction. Figure 1 illustrates a case in point, where for query $a$ the discriminant direction is parallel to

the vertical axis, and as such, the shape of the neighborhood is squashed along that direction and enlongated along the horizontal axis.

While $\mathbf{w}$ points to a direction along which projected data can be well separated, the corresponding hyperplane may be far from optimal with respect to margin maximization. In general, such a hyperplane does not yield the maximum margin of separation between the data, which has direct bearing on its generalization performance. In terms of weighted nearest neighbor computation discussed above, this implies that the class (conditional) probability tends to vary in the neighborhood induced by $\mathbf{w}$. Furthermore, the assumption on equal covariance structures for all classes is often invalid in practice. Computationally, if the dimension of the feature space is large, there will be insufficient data to locally estimate the $O(n^2)$ elements of the covariance matrix, thereby making them highly biased. This motivates us to consider the SVM approach to feature relevance estimation.

### 3.1. Support Vector Machines

Let $R(\mathbf{w}) = \int \frac{1}{2}|y - f(\mathbf{x}, \mathbf{w})|dP(\mathbf{x}, y)$ be the expected generalization error (risk) for a learning machine $f(\mathbf{x}, \mathbf{w})$, where $\mathbf{w}$ is an adjustable parameter that determines $f$, and $P(\mathbf{x}, y)$ the (unknown) probability distribution. The empirical risk is defined as

$$R_{emp}(\mathbf{w}) = \frac{1}{2l}\sum_{i=1}^{l}|y_i - f(\mathbf{x}_i, \mathbf{w})| \tag{4}$$

for a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$.

In the SVM framework, unlike typical classification methods that simply minimize $R_{emp}(\mathbf{w})$, SVMs minimize the following upper bound of the expected generalization error $R(\mathbf{w}) \leq R_{emp}(\mathbf{w}) + C(h)$, where $C$ represents the "VC confidence," and $h$ the VC dimension. This can be accomplished by maximizing the margin between the separating plane and the data, which can be viewed as realizing the Structure Risk Minimization principle [14].

Now we examine general SVMs having basis functions $\Phi : \Re^n \rightarrow \Re^N$, where $n \leq N$. SVMs search for a linear function: $f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b$ in space $\Re^N$. An input query $\mathbf{x}$ is classified according to the algebraic sign of $f(\mathbf{x})$. The SVM solution produces a hyerplane having the maximum margin, where the margin is defined as $2/\|\mathbf{w}\|$. It is shown [14] that this hyperplane is optimum with respect to generalization error. The hyperplane, determined by its normal vector $\mathbf{w}$, can be explicitly written as $\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \Phi(\mathbf{x}_i)$, where $SV$ is the set of support vectors determined by the SVM.

In a simple case in which $\Phi$ is the identity function on $\Re^n$: $\Phi(\mathbf{x}) = \mathbf{x}$, we have

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \tag{5}$$

and

$$\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i. \tag{6}$$

Here the normal $\mathbf{w}$ is perpendicular to the separating hyperplane. Similar to linear discriminant analysis, the normal $\mathbf{w}$ points to the direction that is most discriminant and yields the maximum margin of separation between the data.

We note that real data are often highly non-linear. In such situations, linear machines can not be expected to work well. As such, $\mathbf{w}$ is unlikely to provide any useful discriminant information. On the other hand, piecewise local hyperplanes can approximate any decision boundaries, thereby enabling $\mathbf{w}$ to capture local discriminant information.

### 3.2. Discriminant Feature Relevance

Based on the above discussion, we now propose a measure of feature relevance for an input query $\mathbf{x}_0$ as

$$R_i(\mathbf{x}_0) = |\mathbf{w}_i| \tag{7}$$

where $\mathbf{w}_i$ denotes the $i$th component of $\mathbf{w}$ in (6) computed locally at $\mathbf{x}_0$. One attractive property of (7) is that $\mathbf{w}$ enables $R_i$'s to capture relevance information that may not otherwise be attainable should relevance estimates been conducted along each individual dimension one at a time, as in [7].

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

$$\mathbf{r}_i(\mathbf{x}_0) = \exp(CR_i(\mathbf{x}_0))/\sum_{l=1}^{n}\exp(CR_l(\mathbf{x}_0)) \tag{8}$$

where $C$ is a parameter that can be chosen to maximize (minimize) the influence of $R_i$ on $\mathbf{r}_i$. When $C = 0$ we have $\mathbf{r}_i = 1/n$, thereby ignoring any difference between the $R_i$'s. On the other hand, when $C$ is large a change in $R_i$ will be exponentially reflected in $\mathbf{r}_i$. The exponential weighting is more stable because it prevents neighborhoods from extending infinitely in any direction, i.e., zero weight. Thus, (8) can be used as weights associated with features for weighted distance computation

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n}\mathbf{r}_i(x_i - y_i)^2}. \tag{9}$$

These weights enable the neighborhood to elongate along feature dimensions that run more or less parallel to the separating hyperplane, and, at the same time, to constrict along feature coordinates that have small angles with $\mathbf{w}$. This can be considered highly desirable in nearest neighbor search. It might be argued that a softmax for relative weightings tend to favor just one local dimension over the others to influence the distance calculation. In two-class classification

problems, however, this one dimension is sufficient to provide discriminant information needed.

We desire that the parameter $C$ in (8) increases with decreasing perpendicular distance between the input query and the decision boundary in an adaptive fashion. The advantage of doing so is that any difference among $\mathbf{w}_i$'s will be magnified exponentially in $\mathbf{r}$, thereby making the neighborhood highly elliptical as the input query approaches the decision boundary. Figure 2 illustrates this situation.

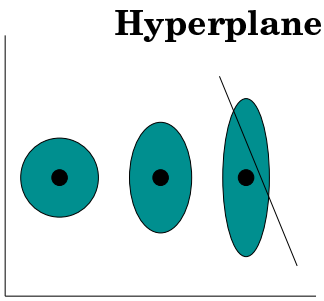


**Figure 2. Neighborhood changes with decreasing distance between the query and the decision boundary.**

In general, however, the boundary is unknown. By using the knowledge that Equation (5) computes an approximate locally linear boundary, we can potentially solve the problem by computing the following: $|\mathbf{w} \cdot \mathbf{x} + b|$. After normalizing $\mathbf{w}$ to unit length, the above equation returns the perpendicular distance between $\mathbf{x}$ and the local separating hyperplance. We can set $C$ to be inversely proportional to $\frac{|\mathbf{w} \cdot \mathbf{x} + b|}{||\mathbf{w}||} = |\mathbf{w} \cdot \mathbf{x} + b|$. In practice, we find it more effective to set $C$ to a fixed constant. In the experiments reported here, $C$ is determined through cross-validation.

Instead of axis-parallel elongation and constriction, one might attempt to use general Mahalanobis distance and have an ellipsoid whose main axes are parallel to the separating hyperplane, and whose width in other dimensions is determined by the distance of $\mathbf{x}$ from the hyperplane. The main concern with such an approach is that in high dimensions there may be insufficient data to locally fill in $n \times n$ within sum-of-squares matrices. Moreover, very often features may be locally independent. Therefore, to effectively compute general Mahalanobis distance some sort of local clustering has to be done. In such situations, without local clustering, general Mahalanobis distance reduces to weighted Euclidean distance.

Let us examine the relevance measure (7) in the context of the Riemannian geometry proposed by Amari and Wu [1]. A large component of $\mathbf{w}$ along a direction $\mathbf{\Theta}$, i.e., a large value of $\mathbf{\Theta} \cdot \mathbf{w}$, implies that data points along that direction become far apart in terms of Equation. (9). Likewise, data points are moving closer to each other along directions that have a small dot product with $\mathbf{w}$. That is, (7) and (9) can be viewed as approximating a local qausiconformal transformation around the separating boundary surface. This transformation is more judicious than that proposed by Amari and Wu [1], because this local mapping increases spatial resolution along discriminant directions around the separating boundary. In contrast, the quasiconformal mapping introduced by Amari and Wu [1] does not attend to directions.

## 4. Neighborhood Morphing Nearest Neighbor Algorithm

The neighborhood *mor*phing nearest neighbor algorithm (MORF) has three adjustable procedural parameters: $K$: the number of nearest neighbors in the final nearest neighbor rule; $K_L$: the number of nearest neighbors in the neighborhood $N_{K_l}$ for local SVM computation; and $C$: the positive factor for the exponential weighting scheme (8).

We note that the parameter $K$ is common to all nearest neighbor rules. Our algorithm however has added two new parameters. Arguably, there is no strong theoretic foundation upon which to determine their selection. The value of $K_L$ should be a reasonable number to support local SVM computation. To be consistent, $K_L$ has to be a diminishing fraction of $l$, the number of training points. The value of $C$ should increase as the input query moves close to the decision boundary, so that highly stretched neighborhoods will result. We have empirically tested different ranges of values, and cross validation is used to choose best values for these parameters, which is what we do in the examples in the next section.

At the beginning, a nearest neighborhood of $K_L$ points around the query $\mathbf{x}_0$ is computed using the simple Euclidean distance. From these $K_L$ points a local linear SVM is built, whose $\mathbf{w}$ (normal to the separating hyperplance) is employed in (7) and (8) to obtain an exponential feature weighting scheme $\mathbf{r}$. Finally, the resulting $\mathbf{r}$ is used in (9) to compute $K$ nearest neighbors at the query point $\mathbf{x}_0$ to classify $\mathbf{x}_0$.

## 5. Empirical Evaluation

In the following we compare several competing classification methods using a number of data sets: 1. MORF - boundary adjusted local metric method described above, coupled with the exponential weighting scheme (8); SVMlight [9] was used to build local SVMs. 2. SVM-L - local linear SVM classifier. For each input query, a linear SVM from $K_L$ nearest points to the query is built to classify the input. We used SVMlight [9]. 3. SVM-R - SVM classifier

**Table 1. Average classification error rates.**

| | Iris | Sonar | Liver | Pima | Vote | OQ | Cancer | Ionosphere | Unstructured |
|---|---|---|---|---|---|---|---|---|---|
| SVM-L | 5.0 | 9.6 | 24.6 | 21.4 | 3.4 | **3.0** | **2.4** | 5.7 | 15.2 |
| SVM-R | 4.0 | 12.5 | 16.1 | 21.3 | 3.0 | **3.0** | **2.4** | **3.0** | 13.1 |
| MORF | **3.0** | 10.6 | **15.1** | **20.3** | **2.6** | 3.7 | 2.6 | 3.4 | 6.6 |
| KNN | 6.0 | 12.5 | 16.6 | 21.7 | 7.8 | 6.0 | 2.7 | 5.8 | 19.0 |
| DANN | 6.0 | **7.7** | 15.9 | 22.2 | 3.2 | 4.2 | 2.5 | 4.8 | 13.6 |
| Scythe | 4.0 | 16.3 | 20.8 | 21.1 | 14.7 | 5.4 | 2.5 | 7.2 | 5.8 |
| Machete | 5.0 | 21.2 | 20.1 | 21.5 | 6.5 | 6.7 | 2.9 | 6.0 | 2.9 |
| C4.5 | 8.0 | 23.1 | 19.5 | 22.4 | 3.5 | 9.4 | 4.7 | 5.4 | **2.2** |

using using radial basis kernels. Again we used SVMlight [9]. 4. KNN - simple K nearest neighbor method using the Euclidean distance. 5. C4.5 - decision tree method [12]. 6. Machete - an adaptive NN procedure [7], in which the input variable used for splitting at each step is the one that maximizes the estimated local relevance (1). 7. Scythe - a generalization of the Machete algorithm [7], in which the input variables influence each split in proportion to their estimated local relevance, rather than the winner-take-all strategy of Machete. 8. DANN - discriminant adaptive nearest neighbor classification [8].

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. Also, in all the experiments where SVMlight was involved, we did not attempt to estimate optimal values for $eps$. We instead used its default value (0.001). The values of $\gamma$ in the radial basis kernel $\exp(-\gamma\|\mathbf{x}-\mathbf{x}'\|)$ and $c$ (the soft-margin case) that affect the performance of SVM related algorithms (SVM-L, SVM-R, MORF) were chosen through cross-validation for each problem. Similarly, optimal procedural parameters for each method are selected through experiments for each problem.

### 5.1. The Problems

The first 8 data sets are taken from UCI Repository of Machine Learning Database. The last one is a simulated (Unstructured) data set that is taken from [8]. The data set has two classes ($J = 2$) and is measured by $n = 10$ features. Each class contains six spherical bivariate normal subclasses (in first two dimensions), having standard deviation 0.25. The rest of eight feature dimensions follow independent standard Gaussian distributions. They serve as noise. The means of the 12 subclasses are chosen at random without replacement from the integers $[1, 2, \ldots, 5] \times [1, 2, \ldots, 5]$. For each class, data are evenly drawn from each of the six normal subclasses.

For the Iris, Sonar, and Vote 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 six cases. For the Liver, Pima, OQ, Cancer, Ionosphere and Unstructured data we randomly select 200 points as training data and 200 as testing data (145 for the Liver data and 151 for the Ionosphere data). We repeat this process 10 times independently, and report the average error rates for these data sets in Table 1. Standard deviations by the corresponding methods are: Liver: 3.3, 4.3, 3.3, 3.9, 2.7, 3.1, 3.6 and 3.2; Pima: 3.7, 1.9, 2.8, 1.9, 2.3, 2.5, 2.4 and 2.6; OQ: 1.3, 1.4, 1.4, 1.9, 1.2, 1.2, 1.7 and 2.5; Cancer: 0.9, 1.2, 1.0, 1.2, 1.1, 1.1, 1.0 and 1.7; Ionosphere: 1.6, 1.5, 1.4, 1.5, 1.8, 1.2, 1.6 and 1.8; and Unstructured: 11.4, 10.2, 8.7, 8.7, 8.6, 4.0, 2.1 and 1.9, respectively.

### 5.2. Results

Table 1 shows clearly that MORF registered competitive performance over the example problems. While the results improvement may not be significant, they nonetherless are in favor of the MORF algorithm. 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 [7], 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: $b_m = \frac{e_m}{\min_{1 \le k \le 8} e_k}$. Thus, the best method $m^*$ for that example has $b_{m^*} = 1$, and all other methods have larger values $b_m \ge 1$, for $m \ne m^*$. The larger the value of $b_m$, the worse the performance of the $mth$ method is in relation to the best one for that example, among the methods being compared. The distribution of the $b_m$ values for each method $m$ over all the problems, therefore, seems to be a good indicator of robustness.

Fig. 3 plots the distribution of $b_m$ for each method over the nine data sets. The dark area represents the lower and upper quartiles of the distribution that are separated by the

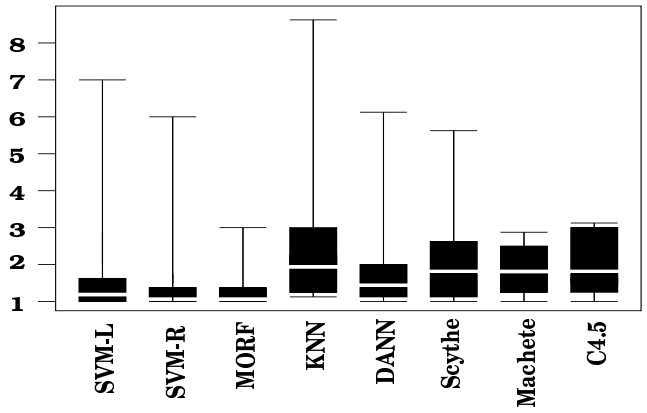


**Figure 3. Performance distributions.**

median. The outer vertical lines show the entire range of values for the distribution. It is clear that the most robust method over the data sets is MORF. In 4/9 of the problems its error rate was the best (median = 1.08). In 8/9 of them it was no worse than $40\%$ higher than the best error rate. In the worst case it was $300\%$. In contrast, KNN has the worst distribution, where the corresponding numbers are 1.93, $300\%$ and $864\%$.

## 6. Summary and Conclusions

This paper presents an adaptive metric 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 show clearly that the MORF 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 space. The results are also in favor of MORF over similar competing methods such as Machete and DANN.

## References

[1] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

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

[3] L. Bottou and V. Vapnik. Local learning algorithms. *Neural Computation*, 4(6):888–900, 1992.

[4] 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.

[5] C. Domeniconi, J. Peng, and D. Gunopulos. An adaptive metric machine for pattern classification. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 458–464. The MIT Press, 2001.

[6] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., New York, second edition, 2001.

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

[8] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(6):607–615, 1996.

[9] T. Joachims. Making large-scale svm learning practical. In B. Schvlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, volume 13. The MIT Press, 1999.

[10] D. G. Lowe. Similarity metric learning for a variable-kernel classifier. *Neural Computation*, 7(1):72–85, 1995.

[11] J. P. Myles and D. J. Hand. The multi-class metric problem in nearestneighbor discrimination rules. *Pattern Recognition*, 723:1291–1297, 1990.

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

[13] R. Short and K. Fukunaga. ‘optimal distance measure for nearest neighbor classification. *IEEE Transactions on Information Theory*, 27:622–627, 1981.

[14] V. N. Vapnik. *Statistical learning theory*. Adaptive and learning systems for signal processing, communications, and control. Wiley, New York, 1998.

[15] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. The MIT Press, 2001.

[16] Y. Wu, K. Ianakiev, and V. Govindaraju. Improvements in k-nearest neighbor classifications. In W. Kropatsch and N. Murshed, editors, *International Conference on Advances in pattern recognition*, volume 2, pages 222–229. Springer-Verlag, 2001.