AN ADAPTIVE METRIC MACHINE

FOR PATTERN CLASSIFICATION

Carlotta Domeniconi

Computer Science Department

University of California, Riverside

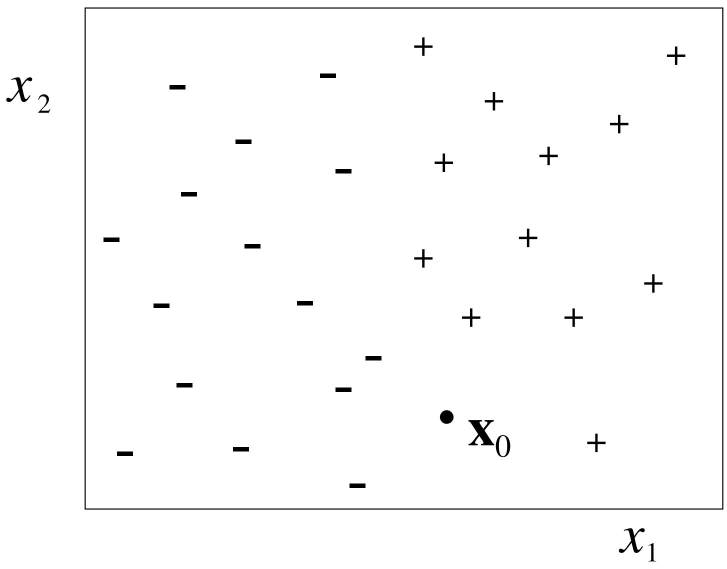
www.cs.ucr.edu/∼carlotta

OUTLINE

- **Classification Problem**
- **Background**
  - Problems facing classification
- **Adaptive Metric Nearest Neighbor Classification**
  - Our Approach
  - The Algorithm
  - Experimental results
- **Final Remarks**

CLASSIFICATION PROBLEM

- **Given** $\{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$
  - $\mathbf{x}_n = (x_1, \cdots, x_q)^t \in \Re^q$
  - $y_n \in \{1, \cdots, J\}$



Predict class label of a given query $\mathbf{x}_0$**.**

THE BAYESIAN CLASSIFIER

- **Loss function** $\lambda(j|k)$**.**

- **Expected loss (***conditional risk***) associated with class** j**:**

$$R(j|\mathbf{x}) = \sum_{k=1}^{J} \lambda(j|k) P(k|\mathbf{x}).$$

- **Bayes rule:**

$$j^* = \arg \min_{1 \le j \le J} R(j|\mathbf{x}).$$

- *Zero-one* **loss function:** $\lambda(j|k) = \begin{cases} 0 & \textbf{if } j = k \\ 1 & \textbf{if } j \neq k \end{cases}$

$$j^* = \arg \max_{1 \le j \le J} P(j|\mathbf{x}).$$

- **Bayes rule achieves** *minimum error rate*

How to estimate the posterior probabilities $\{P(j|\mathbf{x})\}_{j=1}^{J}$?

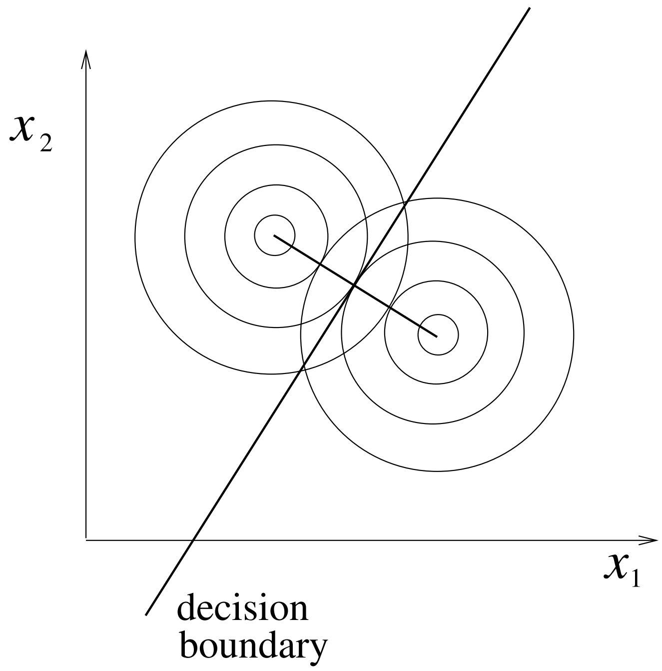
DENSITY ESTIMATION

- **Use Bayes theorem to estimate the posterior probability values:**

$$P(j|\mathbf{x}) = \frac{p(\mathbf{x}|j)P(j)}{\Sigma_{k=1}^{J} p(\mathbf{x}|k)P(k)}$$

  - $p(\mathbf{x}|j) =$probability density function of $\mathbf{x}$ given class j
  - $P(j) =$ prior probability of class j

- **Example:** Discriminant Analysis.



REGRESSION

- Direct estimation of probabilities $\{P(j|\mathbf{x})\}_{j=1}^{J}$
- $y(\mathbf{x})$: random variable $\sim$ multinomial distribution with probabilities $\{P(j|\mathbf{x})\}_{j=1}^{J}$
- Additional output variable g_j : $g_j|\mathbf{x} = \begin{cases} 1 & \textbf{if } y = j \\ 0 & \textbf{otherwise} \end{cases}$
- J separate training sets, one for each class j :

$$\{\mathbf{x}_n, g_j\}_{n=1}^{N}, \;\; j = 1, \cdots, J.$$

$$f_j(\mathbf{x}) \stackrel{\text{def}}{=} P(j|\mathbf{x}) = P(g_j = 1|\mathbf{x}) = E[g_j|\mathbf{x}].$$

- Least-squares problems:

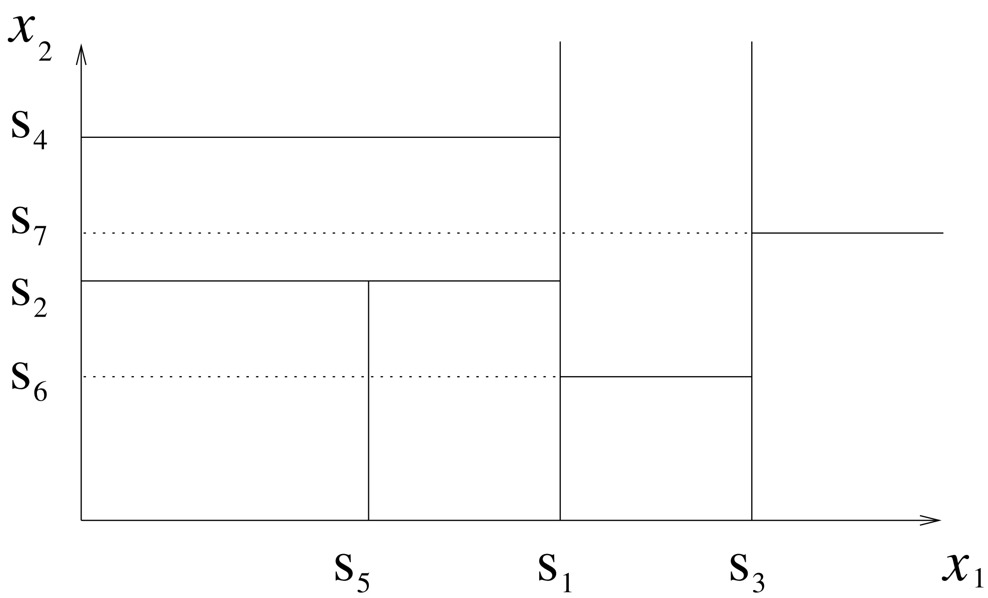
$$f_j(\mathbf{x}) = \arg\min_f E[(g_j - f)^2|\mathbf{x}].$$

- **Examples:** Neural Networks, Decision Trees, K-nearest-neighbors.

DECISION TREES

- Top-down recursive splitting strategy.
- Highly adaptive partitioning of the input space.

for (all regions R)
    $while$ (R does not meet a terminal criterion)
        $Split(R) \rightarrow (R_l, R_r)$;



- Constant class probability estimates within each region.

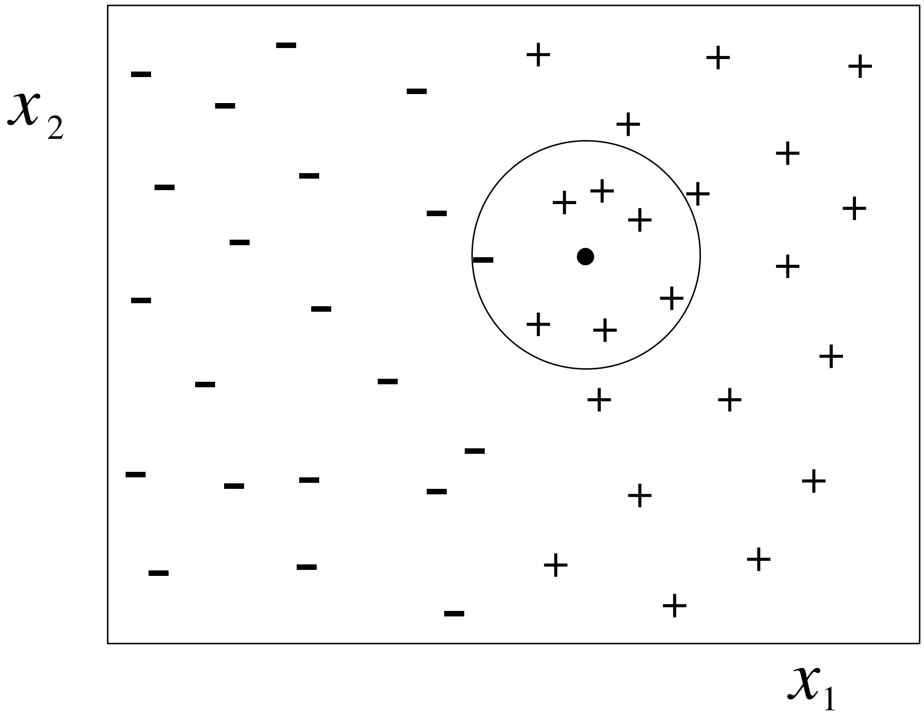
DECISION TREES

- **Advantages:**
  - Variable subset selection;
  - Exploit "local relevance".

- **Disadvantages:**
  - Limited number of meaningful splits;
  - Piecewise-constant estimations $\rightarrow$ high bias;
  - Top-down (greedy) algorithm$\rightarrow$ high variance.

NEAREST-NEIGHBOR METHODS

- Predict the class label of $\mathbf{x}_0$ as the most frequent one occuring in the K neighbors.



- **Basic Assumption:** smoothness of target functions: $f_j(\mathbf{x} + \delta\mathbf{x}) \simeq f_j(\mathbf{x})$ **for** $||\delta\mathbf{x}||$ **small enough**

$$\{f_j(\mathbf{x})\}_{j=1}^{J} = \{P(j|\mathbf{x})\}_{j=1}^{J}.$$

NEAREST-NEIGHBOR METHODS

- **Then**

$$f_j(\mathbf{x}_0) \simeq ave[f_j(\mathbf{x}) | \mathbf{x} \in N(\mathbf{x}_0)]$$

-where $N(\mathbf{x}_0)$ = neighborhood of $\mathbf{x}_0$ that contains points $\mathbf{x} \in \Re^q$ "close" to $\mathbf{x}_0$.

- **Given** $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$:

$$\hat{f}(j|\mathbf{x}_0) = \frac{\Sigma_{n=1}^N 1(\mathbf{x}_n \in N(\mathbf{x}_0)) 1(y_n = j)}{\Sigma_{n=1}^N 1(\mathbf{x}_n \in N(\mathbf{x}_0))}$$

DISTANCE METRIC

ASYMPTOTIC PROPERTIES OF K-NN METHODS

$$\boxed{\lim_{N\to\infty} \hat{f}_j(\mathbf{x}) = f_j(\mathbf{x})}$$

if $\quad \lim_{N\to\infty} K = \infty, \quad \lim_{N\to\infty} K/N = 0.$

- **Also:**
  - $E_1 \overset{def}{=}$ classification error rate of the 1-NN rule
  - $E_\infty \overset{def}{=}$ classification error rate of the Bayes rule

$$\boxed{\lim_{N\to\infty} E_1 = e}$$

where $\quad E_\infty \leq e \leq E_\infty(2-e) \leq 2E_\infty$

- **In the asymptotic limit no decision rule is more than twice as accurate as the 1-NN rule.**

CURSE-OF-DIMENSIONALITY

$$\text{bias } \hat{f}_j(\mathbf{x}) = f_j(\mathbf{x}) - E[\hat{f}_j(\mathbf{x})]$$

- Random sample of size $N \sim$ uniform distribution in the $q-$dimensional unit hypercube.
- Diameter of a $k = 1$ neighborhood using Euclidean distance:

$$\text{diameter} \sim N^{-1/q}$$

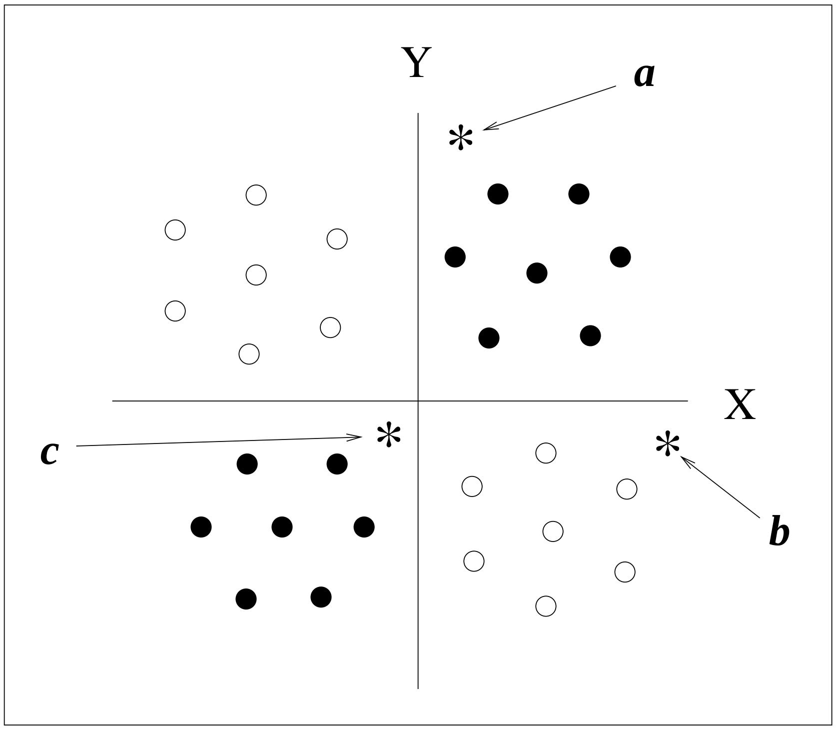
q	4	4	6	6	10	10	20	20	20
N	100	1000	100	1000	1000	10000	10000	10^6	10^{10}
$d(q, N)$	0.42	0.23	0.71	0.48	0.91	0.72	1.51	1.20	0.76

Highly biased estimations

- Why simple nearest neighbor methods still perform well?
  - intrinsic dimensionality
  - $\max_{1 \leq j \leq J} \hat{f}_j(\mathbf{x}) = \max_{1 \leq j \leq J} f_j(\mathbf{x})$

ADAPTIVE METRIC NEAREST NEIGHBOR CLASSIFICATION

- How to reduce the "high bias effect" due to the curse-of-dimensionality?

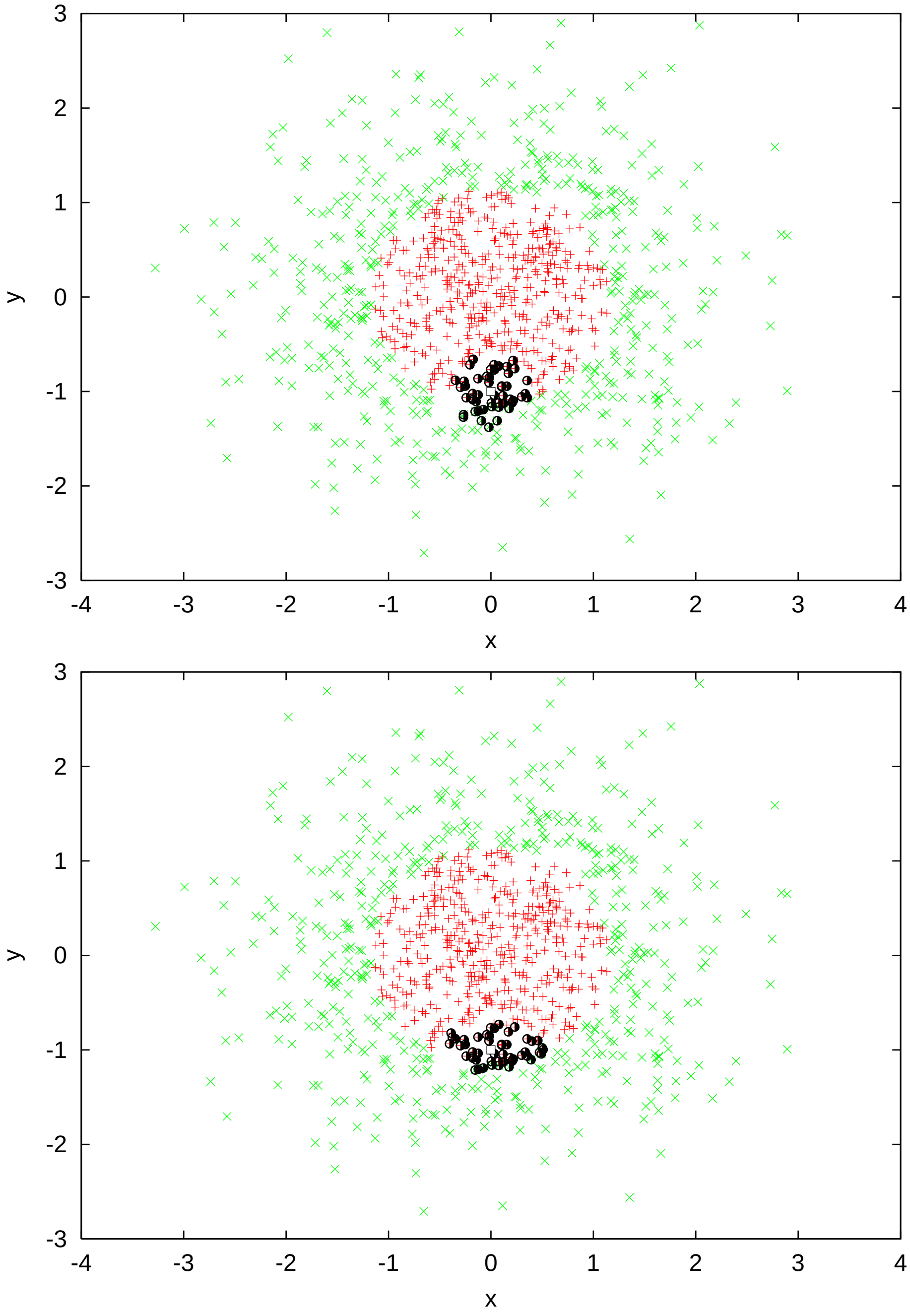


Problems:

- Different features have different degrees of relevance: see a and b
- Feature relevance changes from location to location: compare a, b and c

MODIFIED "WEIGHTED" NEIGHBORHOODS

$$D_p(\mathbf{x}_0, \mathbf{x}) = \{\Sigma_{i=1}^{q} |[W(\mathbf{x}_0)(\mathbf{x}_0 - \mathbf{x})]_i|^p\}^{1/p}$$



ADAMENN

- Adaptive Metric Nearest Neighbor Technique.

- Direct estimate of *weighted Chi-squared distance*.

- Features are weighted accordingly to their local relevance.

- Modified neighborhoods:

  - highly adaptive to query locations

  - with approximately constant posterior probabilities.

CHI-SQUARED DISTANCE

- $\mathbf{x}_0$: query point ;
- $\mathbf{x}$: nearest neighbor of $\mathbf{x}_0$ according to $D(\mathbf{x}, \mathbf{x_0})$;
- Goal: find a metric $D(\mathbf{x}, \mathbf{x_0})$ that minimize $E[r(\mathbf{x_0}, \mathbf{x})]$, where

$$r(\mathbf{x}_0, \mathbf{x}) = \Sigma_{j=1}^{J} \Pr(j|\mathbf{x}_0)(1 - \Pr(j|\mathbf{x}))$$

- Or, we can minimize $E(r^*(\mathbf{x}_0) - r(\mathbf{x}_0, \mathbf{x}))^2$, where

$$r^*(\mathbf{x}_0) = \Sigma_{j=1}^{J} \Pr(j|\mathbf{x}_0)(1 - \Pr(j|\mathbf{x}_0))$$

- As a result

$$D(\mathbf{x}_0, \mathbf{x}) = (\Sigma_{j=1}^{J} \Pr(j|\mathbf{x}_0)(\Pr(j|\mathbf{x}) - \Pr(j|\mathbf{x}_0)))^2$$

- To avoid cancellation effect, we consider the (weighted) *Chi-squared* distance:

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

CHI-SQUARED DISTANCE

- **Weighted Chi-squared distance:**

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

- We want to estimate the relevance of each dimension i by computing its ability to predict $Pr(j|\mathbf{x})$ in the vicinity of $\mathbf{x}_0$.

- We do so by considering a quantity that is conditioned at location x_i along dimension i.

- Then: The weighted Chi-squared distance can measure the deviation of such quantity from $Pr(j|\mathbf{x})$, telling us its capability of predicting it.

FEATURE RELEVANCE MEASURE

- $Pr(j|\mathbf{x})$: function of $\mathbf{x}$
- $\overline{\mathrm{Pr}}(j|x_i = z) = E[\mathrm{Pr}(j|\mathbf{x})|x_i = z]$
- Ability of feature i to predict $Pr(j|\mathbf{z})$ at $x_i = z_i$

$$\boxed{r_i(\mathbf{z}) = \Sigma_{j=1}^{J} \frac{[\mathrm{Pr}(j|\mathbf{z}) - \overline{\mathrm{Pr}}(j|x_i=z_i)]^2}{\overline{\mathrm{Pr}}(j|x_i=z_i)}}$$

- Measure of relevance of feature i at $\mathbf{x}_0$

$$\boxed{\bar{r}_i(\mathbf{x}_0) = \frac{1}{K} \Sigma_{\mathbf{z} \in N(\mathbf{x}_0)} r_i(\mathbf{z})}$$

-where $N(\mathbf{x}_0) = K$-nearest neighbors of $\mathbf{x}_0$

- $\bar{r}_i(\mathbf{x}_0)$ is a function of:
  - test point $\mathbf{x}_0$
  - dimension i

WEIGHTING

- Weight for feature i

$$w_i(\mathbf{x}_0) = (R_i(\mathbf{x}_0))^t / \Sigma_{l=1}^{q} (R_l(\mathbf{x}_0))^t$$

  - $R_i(\mathbf{x}_0) = \max_j\{\bar{r}_j(\mathbf{x}_0)\} - \bar{r}_i(\mathbf{x}_0)$
  - $t = 1$: linear weighting
  - $t = 2$: quadratic weighting

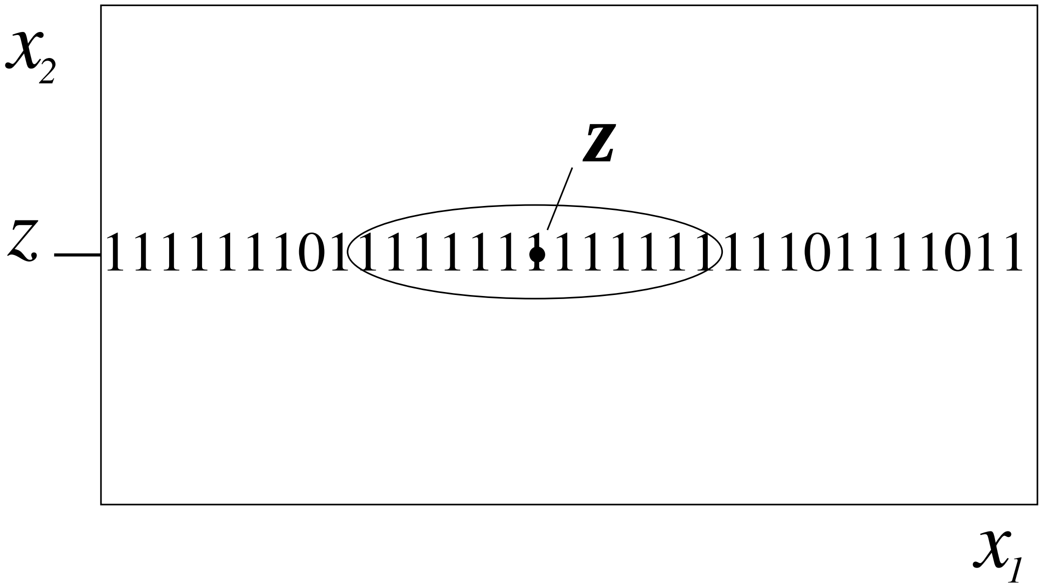
- Exponential weighting scheme

$$\boxed{w_i(\mathbf{x}_0) = \exp(cR_i(\mathbf{x}_0)) / \Sigma_{l=1}^{q} \exp(cR_l(\mathbf{x}_0))}$$

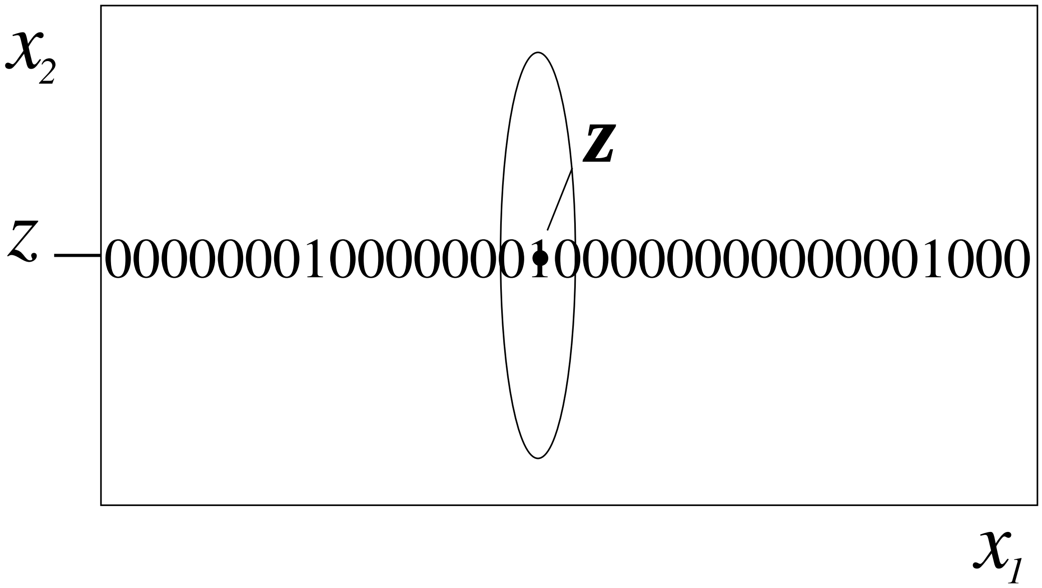
- Weighted distance $D(\mathbf{x}, \mathbf{y}) = \sqrt{\Sigma_{i=1}^{q} w_i (x_i - y_i)^2}$
  - small $\bar{r}_i \rightarrow$ large $w_i \rightarrow$ constriction
  - large $\bar{r}_i \rightarrow$ small $w_i \rightarrow$ elongation

JUSTIFICATION

$r_2(\mathbf{z})$ small $\rightarrow$ Large weight for feature x_2 .



$r_2(\mathbf{z})$ large $\rightarrow$ Small weight for feature x_2 .



ESTIMATION

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

- Training data: $\{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$, $y_n \in \{1, \cdots, J\}$.

- $\hat{\Pr}(j|\mathbf{z}) = \frac{\Sigma_{n=1}^{N} 1(\mathbf{x}_n \in N_1(\mathbf{z})) 1(y_n = j)}{\Sigma_{n=1}^{N} 1(\mathbf{x}_n \in N_1(\mathbf{z}))}$
  - $N_1(\mathbf{z})$ = neighborhood centered at $\mathbf{z}$

- $\hat{\overline{\Pr}}(j|x_i = z_i) = \frac{\Sigma_{\mathbf{x}_n \in N_2(\mathbf{z})} 1(|x_{ni} - z_i| \le \Delta_i) 1(y_n = j)}{\Sigma_{\mathbf{x}_n \in N_2(\mathbf{z})} 1(|x_{ni} - z_i| \le \Delta_i)}$
  - $N_2(\mathbf{z})$ = neighborhood centered at $\mathbf{z}$
  - $\Sigma_{n=1}^{N} 1(|x_{ni} - z_i| \le \Delta_i) 1(\mathbf{x}_n \in N_2(\mathbf{z})) = L$

THE ALGORITHM

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

1. Initialize $\mathbf{w}$ to 1;

2. Compute the K_0 nearest neighbors of $\mathbf{x}_0$ using the weighted distance metric $D(\mathbf{x}, \mathbf{y}) = \sqrt{\Sigma_{i=1}^{q} w_i (x_i - y_i)^2}$;

3. For each dimension i, $i = 1, \ldots, q$, compute relevance estimate $\bar{r}_i(\mathbf{x}_0)$;

4. Update $\mathbf{w}$ according to the weighting scheme in use;

5. Iterate steps 2, 3, and 4;

6. At completion, use $\mathbf{w}$ for K-nearest neighbor classification at the test point $\mathbf{x}_0$.

WHY AVERAGING

- For each given dimension i, our goal is to estimate t_i:

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

- Let $r_i(\mathbf{x}_0, \mathbf{z})$ be the estimator, 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
  $\Sigma_{i=1}^{q} E_{\mathbf{z}}[(t_i(\mathbf{x}_0) - r_i(\mathbf{x}_0, \mathbf{z}))^2] =$
  $\Sigma_{i=1}^{q}(t_i^2(\mathbf{x}_0) - 2t_i(\mathbf{x}_0)E_{\mathbf{z}}[r_i(\mathbf{x}_0, \mathbf{z})] + E_{\mathbf{z}}[r_i^2(\mathbf{x}_0, \mathbf{z})])$
- Applying $E[X^2] \geq E^2[X]$ gives
  $\Sigma_{i=1}^{q} E_{\mathbf{z}}[(t_i(\mathbf{x}_0) - r_i(\mathbf{x}_0, \mathbf{z}))^2] \geq \Sigma_{i=1}^{q}(t_i^2(\mathbf{x}_0) - 2t_i(\mathbf{x}_0)E_{\mathbf{z}}[r_i(\mathbf{x}_0, \mathbf{z})] + E_{\mathbf{z}}^2[r_i(\mathbf{x}_0, \mathbf{z})])$
  $= \Sigma_{i=1}^{q}(t_i(\mathbf{x}_0) - E_{\mathbf{z}}[r_i(\mathbf{x}_0, \mathbf{z})])^2$
  $= \Sigma_{i=1}^{q}(t_i(\mathbf{x}_0) - \bar{r}_i(\mathbf{x}_0))^2$
- Integrating both sides: 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 $\mathbf{z}$.

RELATED WORK

- **The "machete" algorithm - by J. Friedman, 94:**
  - Reduction in prediction error:
    $$I_i^2(\mathbf{z}) = \Sigma_{j=1}^{J}(\overline{\Pr}(j) - \overline{\Pr}(j|x_i = z_i)])^2$$
  - The most informative input variable is the one that gives the largest deviation from the average value of $Pr(j|\mathbf{x})$.
- **The DANN algorithm - by T. Hastie & R. Tibshirani, 96:**
  - Metric $\Sigma = W^{-1}BW^{-1}$: behaves locally like the LDA metric.
  - Approximation of the weighted *Chi-squared distance*
    $$D(\mathbf{x}, \mathbf{x}_0) = \Sigma_{j=1}^{J}\frac{[P(j|\mathbf{x}) - P(j|\mathbf{x}_0)]^2}{P(j|\mathbf{x}_0)}$$
    * Gaussian and equal covariance matrix assumptions.

EMPIRICAL RESULTS

- Methods compared:
  - **ADAMENN**-adaptive metric nearest neighbor coupled with the exponential weighting scheme;
  - **i-ADAMENN**-adaptive metric nearest neighbor with five iterations;
  - Simple **K-NN** method using the Euclidean distance measure;
  - **C4.5** decision tree method;
  - **Machete**;
  - **Scythe**;
  - **DANN**-discriminant adaptive nearest neighbor classification;
  - **i-DANN**-discriminant adaptive nearest neighbor classification with five iterations.

EXPERIMENTS ON SIMULATED DATA

The Problems:

1. $N = 200$, $q = 10$, $J = 2$. Class 1: $\mathbf{x}_n \sim N(\mathbf{0}, \mathbf{1})$. Class 2: $\mathbf{x}_n \sim N(\mathbf{m}, \mathbf{C})$, $\{m_i = \sqrt{i}/2\}_{i=1}^{q}$, $\mathbf{C} = diag\{1/\sqrt{i}\}_{i=1}^{q}$.
2. $N = 200$, $q = 10$, $J = 2$. Two four dimensional standard Gaussians with 6 noise features. Class 1: $radius > 1.85$.
3. $N = 500$, $q = 10$, $J = 2$. Both classes: $\mathbf{x}_n \sim N(\mathbf{0}, \mathbf{1})$: $\Sigma_{i=1}^{10} x_i^2 \leq 9.8 \Rightarrow$ class 1, otherwise $\Rightarrow$ class 2.
4. $N = 200$, $q = 10$, $J = 2$. Both classes: $\mathbf{x}_n \sim N(\mathbf{0}, \mathbf{1})$: $\Sigma_{i=1}^{10} x_i \leq 0 \Rightarrow$ class 1, otherwise $\Rightarrow$ class 2.
5. $N = 200$, $q = 2$, $J = 2$. Each class: six spherical bivariate normal subclasses, $std = 0.25$, $means \in [1, 2, \ldots, 5] \times [1, 2, \ldots, 5]$.
6. $N = 200$, $q = 2$, $J = 4$. Each class: three spherical bivariate normal subclasses, $std = 0.25$, $means \in [1, 2, \ldots, 5] \times [1, 2, \ldots, 5]$.
7. $N = 200$, $q = 10$, $J = 4$. As in 6 with 8 noise features.

RESULTS FOR SIMULATED DATA

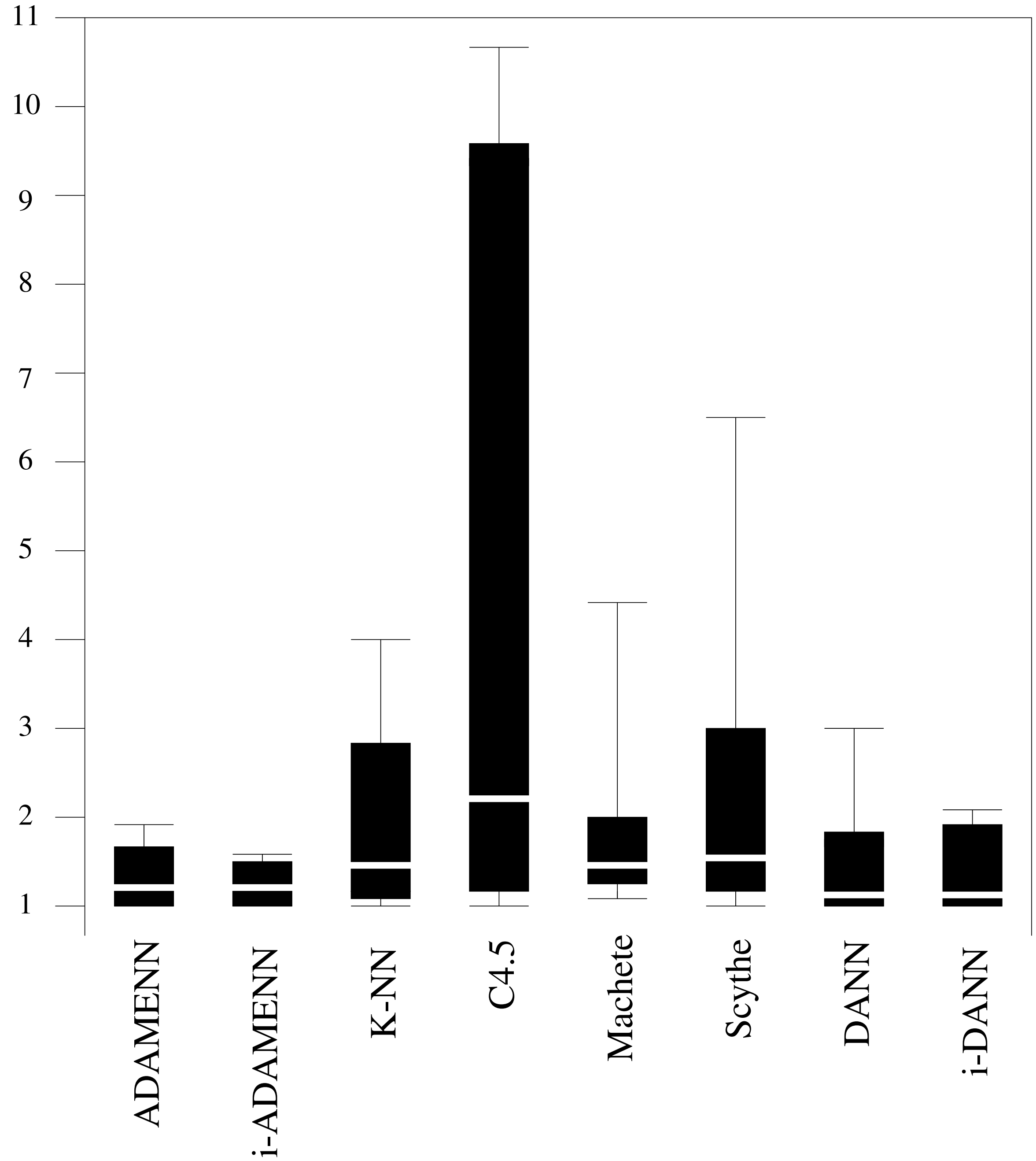
Average classification error rates for simulated data

	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7
ADAMENN	9.9	23.9	33.7	20.8	**2.4**	**3.3**	**12.8**
i-ADAMENN	8.3	23.1	33.7	20.3	**2.4**	**3.3**	14.2
K-NN	14.7	33.9	36.1	18.1	2.5	**3.3**	50.7
C4.5	10.3	**14.6**	30.6	30.1	25.5	31.7	38.2
Machete	7.1	21.7	33.0	25.7	2.6	14.5	20.1
Scythe	7.9	25.6	32.7	22.2	**2.4**	21.3	38.1
DANN	6.2	25.3	26.7	13.3	2.8	4.2	37.6
i-DANN	**5.3**	22.8	**25.4**	**13.2**	3.1	6.1	26.7

- **ADAMENN:**
  - **Best performance in 3/7 data sets.**

RESULTS FOR SIMULATED DATA

Performance distributions of $b_m = e_m / \min_{1 \le k \le 8} e_k$



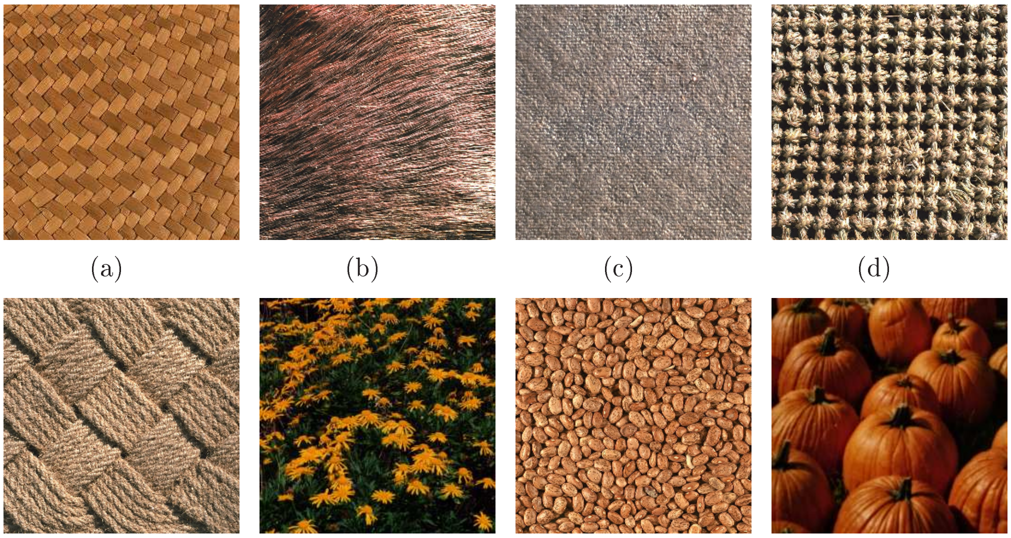
EXPERIMENTS ON REAL DATA

The Problems:

1. *Iris*: $N = 100$, $q = 4$, $J = 2$.
2. *Sonar*: $N = 208$, $q = 60$, $J = 2$.
3. *Vowel*: $N = 528$, $q = 10$, $J = 11$.
4. *Glass*: $N = 214$, $q = 9$, $J = 6$.
5. *Image*: $N = 640$ (images of 128×128), $q = 16$, $J = 15$.
6. *Segmentation*: $N = 2,310$, $q = 19$, $J = 7$.
7. *Letter*: $N = 20,000$, $q = 16$, $J = 26$.
8. *Liver*: $N = 345$, $q = 6$, $J = 2$.
9. *Lung*: $N = 32$, $q = 56$, $J = 3$.

- *Iris*, *Sonar* , *Glass* , *Liver* , and *Lung* data: leave-one-out cross-validation.
- *Vowel*, and *Image* data: ten two-fold cross-validation.
- *Segmentation*, and *Letter* data: two ten-fold cross-validation.

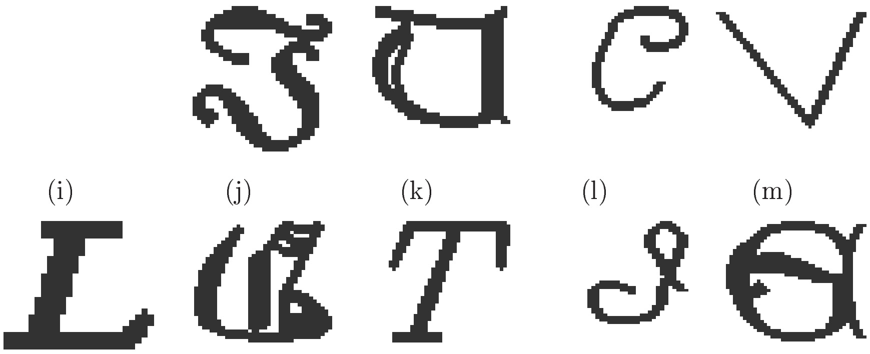
Sample images from the Image database:



(a) (b) (c) (d)

(e) (f) (g) (h)

Sample Letter images:



(i) (j) (k) (l) (m)

(n) (o) (p) (q) (r)

RESULTS FOR REAL DATA

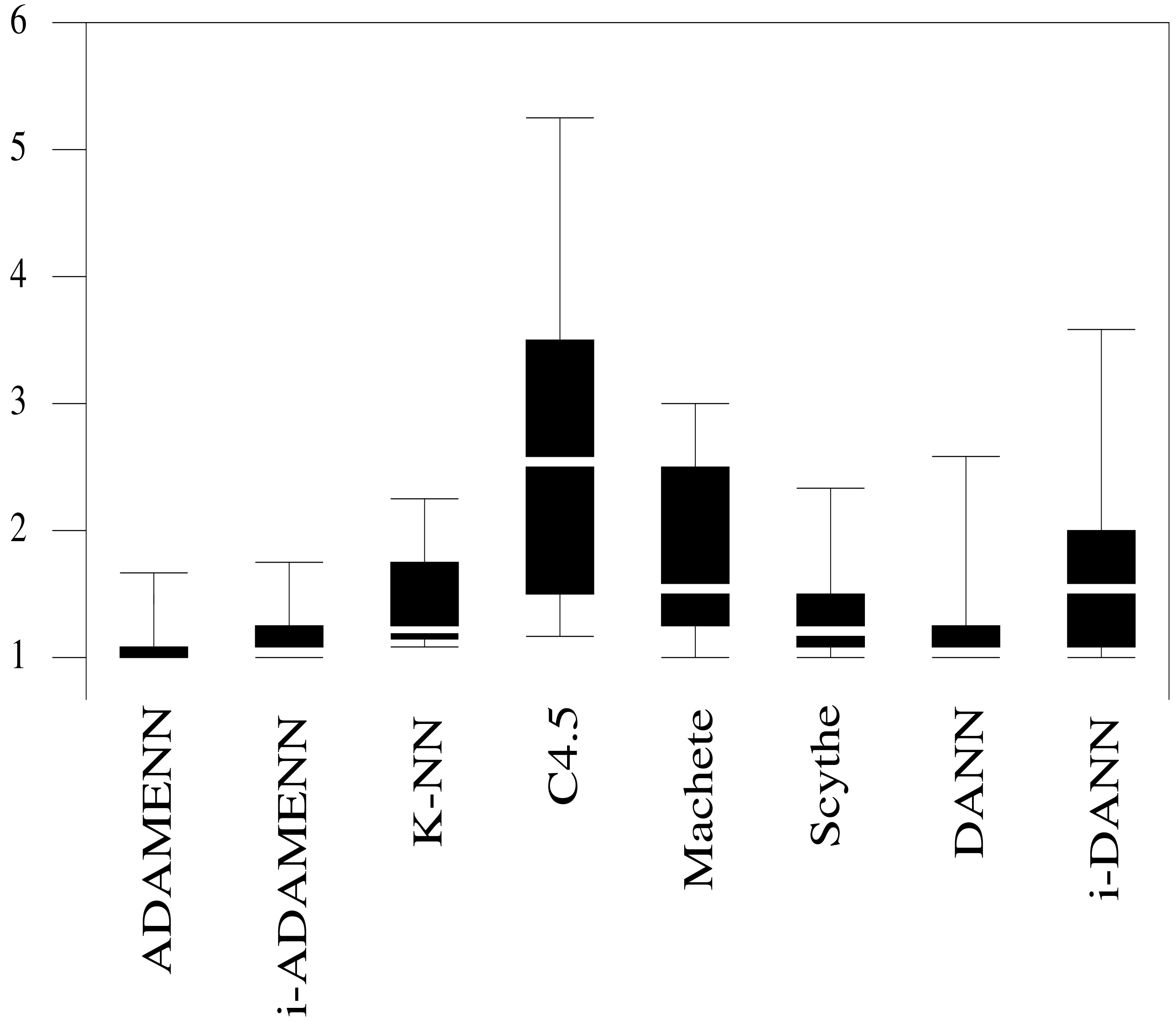
Average classification error rates for real data

	Iris	Sonar	Vowel	Glass	Image	Seg	Letter	Liver	Lung
ADAMENN	**3.0**	9.1	**10.7**	**24.8**	5.2	**2.4**	5.1	30.7	**40.6**
i-ADAMENN	5.0	9.6	10.9	24.8	5.2	2.5	5.3	30.4	**40.6**
K-NN	6.0	12.5	11.8	28.0	6.1	3.6	6.9	32.5	50.0
C4.5	8.0	23.1	36.7	31.8	21.6	3.7	16.4	38.3	59.4
Machete	5.0	21.2	20.2	28.0	12.3	3.2	9.1	**27.5**	50.0
Scythe	4.0	16.3	15.5	27.1	**5.0**	3.3	7.2	**27.5**	50.0
DANN	6.0	**7.7**	12.5	27.1	12.9	2.5	**3.1**	30.1	46.9
i-DANN	6.0	9.1	21.8	26.6	18.1	3.7	6.1	27.8	**40.6**

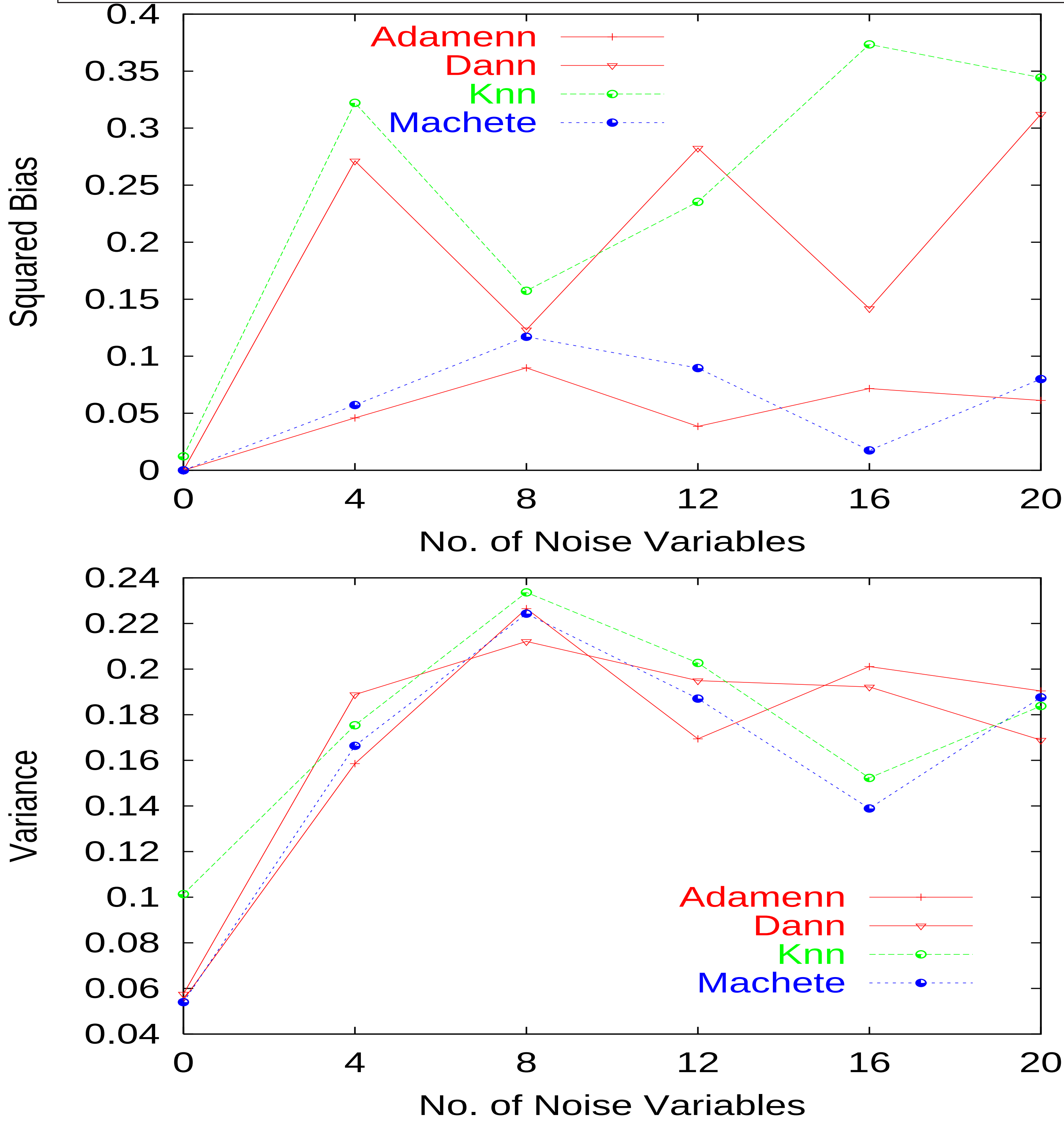
- **ADAMENN:**
  - Best performance in 5/9 data sets
  - Second best performance in 3/9 data sets

RESULTS FOR REAL DATA

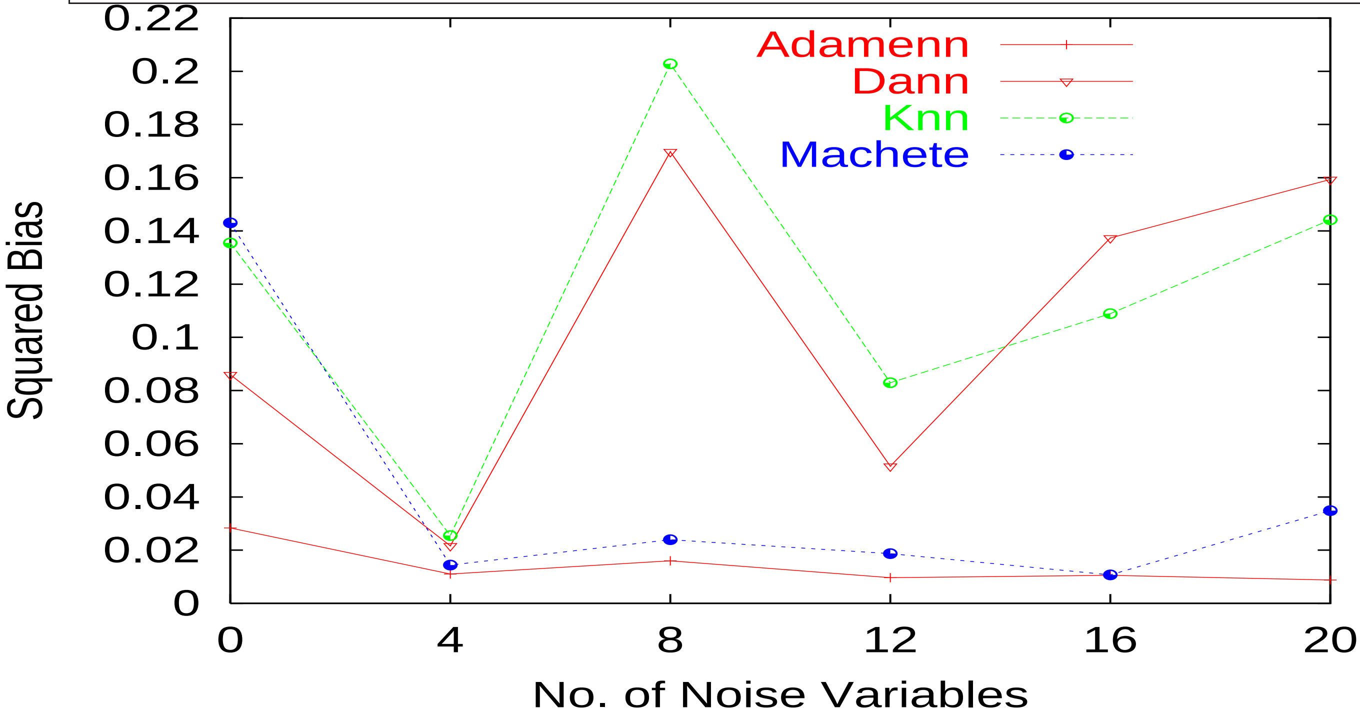
Performance distributions of $b_m = e_m / \min_{1 \le k \le 8} e_k$

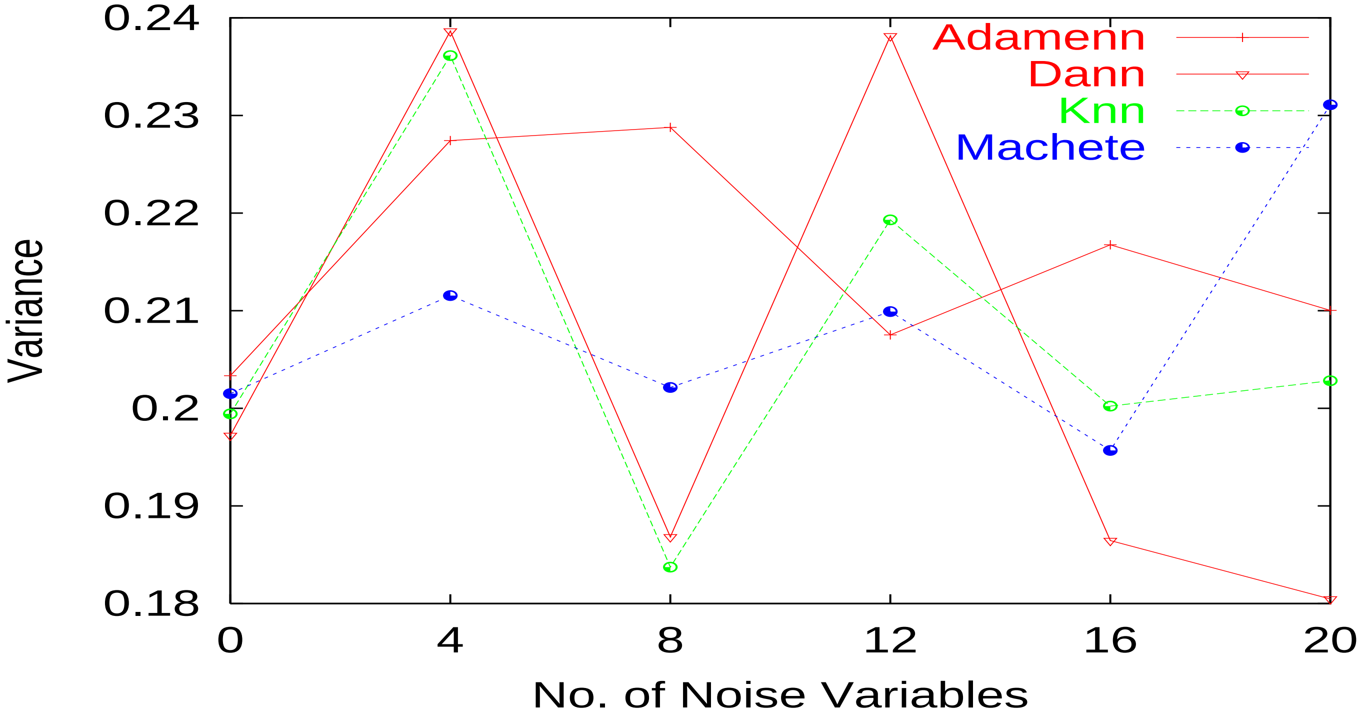


BIAS AND VARIANCE CALCULATIONS

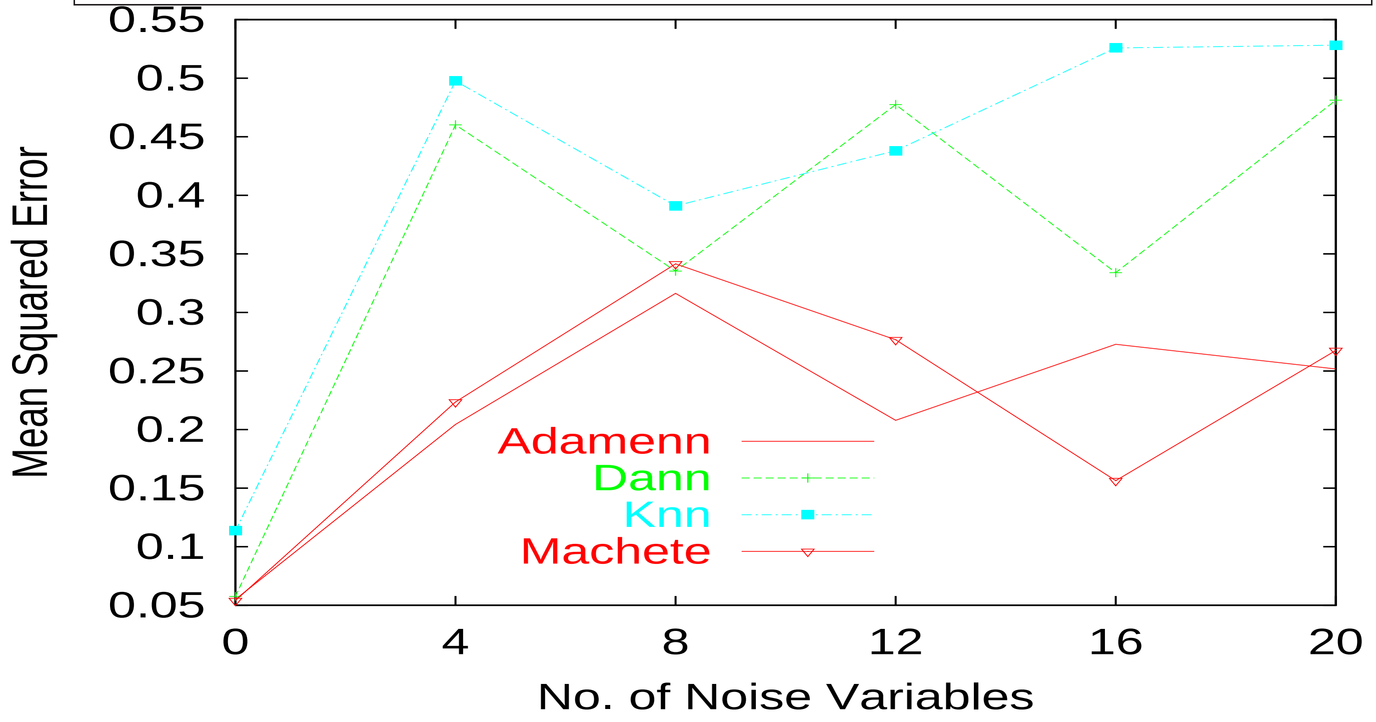


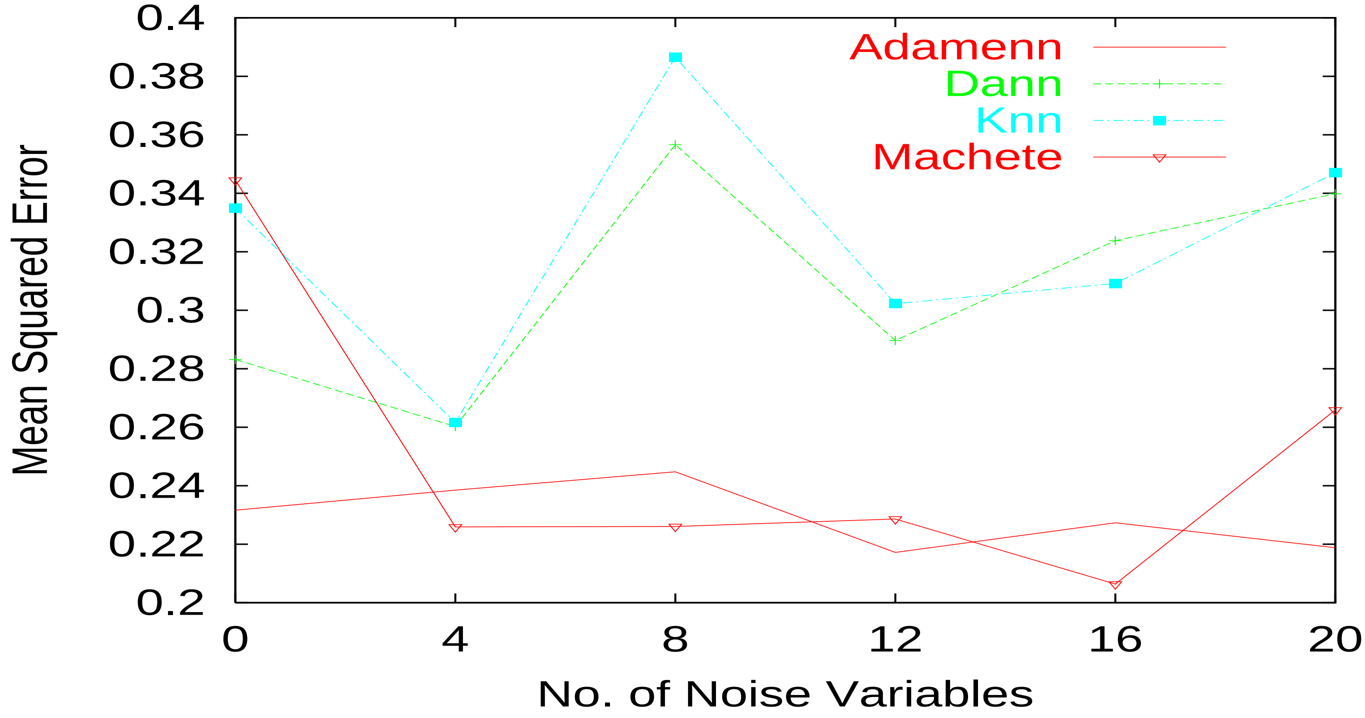
BIAS AND VARIANCE CALCULATIONS





BIAS AND VARIANCE CALCULATIONS





REMARKS

- Adaptive nearest neighbor method for effective pattern classification.

- Estimation of a flexible local metric.

- Smoother class conditional probabilities in the modified neighborhoods.

- Performance improvement validated with experiments using both simulated and real data sets.