CpE 646 Pattern Recognition and Classification

Prof. Hong Man

Department of Electrical and Computer Engineering
Stevens Institute of Technology

STEVENS Institute of Technology

Visual Information Environment Laboratory

Non-Parametric Classification

Chapter 4 (Section 4.4 – 4.6):

- k_n-Nearest-Neighbor Estimation
- The Nearest-Neighbor Rule

k_n-Nearest Neighbor Estimation

- A solution for the problem of the unknown "best" window function. To estimate $p(\mathbf{x})$ from n training samples or prototypes:
  - Let the cell volume be a function of the number of the training data
  - Center a cell about $\mathbf{x}$ and let it grows until it captures k_n samples ($k_n = f(n)$)
  - The included samples are called the k_n nearest-neighbors of $\mathbf{x}$
  - The density is given as $$p_n(x) = \frac{k_n / n}{V_n} \qquad (30)$$

k_n-Nearest Neighbor Estimation

- Two possibilities can occur:
  - If the density is high near ***x***, the cell will be small which provides a good resolution
  - If the density is low around ***x***, the cell will grow large and stop until higher density regions are reached

k_n-Nearest Neighbor Estimation

- It can be proven that $\lim_{n\to\infty} k_n = \infty$ and $\lim_{n\to\infty} k_n / n = 0$ are necessary and sufficient for $p_n(\mathbf{x})$ converge to $p(\mathbf{x})$ at all points where $p(\mathbf{x})$ is continuous
- If $k_n = \sqrt{n}$ and $p_n(\mathbf{x})$ is a good estimate of $p(\mathbf{x})$, i.e. $p_1(\mathbf{x}) = p_n(\mathbf{x}) = p(\mathbf{x})$, from (30) we have

$$V_n = 1/\left(\sqrt{n}\, p(x)\right) \text{ and then } V_n = V_1/\sqrt{n}$$

this becomes similar to the Parzen-window approach except that V_1 is not determined arbitrarily.

k_n-Nearest Neighbor Estimation

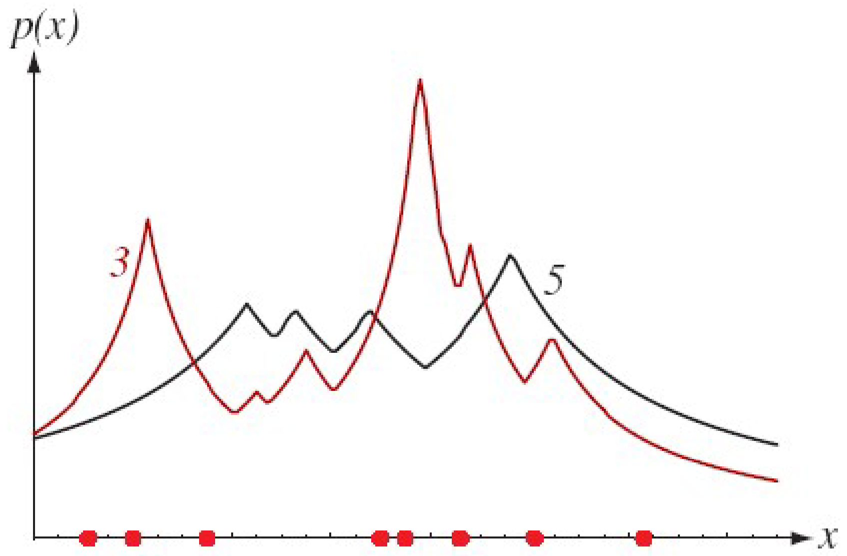


FIGURE 4.10. Eight points in one dimension and the k-nearest-neighbor density estimates, for $k = 3$ and 5. Note especially that the discontinuities in the slopes in the estimates generally lie *away* from the positions of the prototype points. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Peaks are at the middle of regions with *k* prototypes

k_n-Nearest Neighbor Estimation

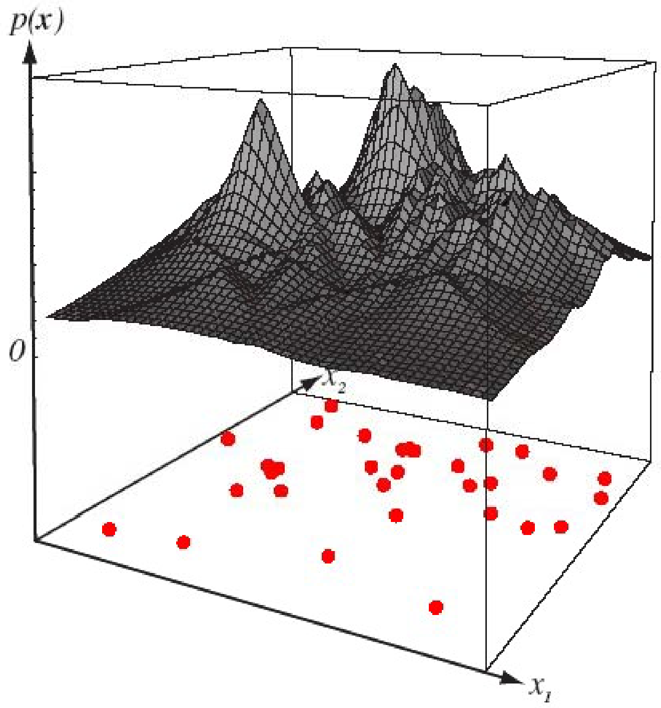


FIGURE 4.11. The k-nearest-neighbor estimate of a two-dimensional density for $k = 5$. Notice how such a finite n estimate can be quite "jagged," and notice that discontinuities in the slopes generally occur along lines away from the positions of the points themselves. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

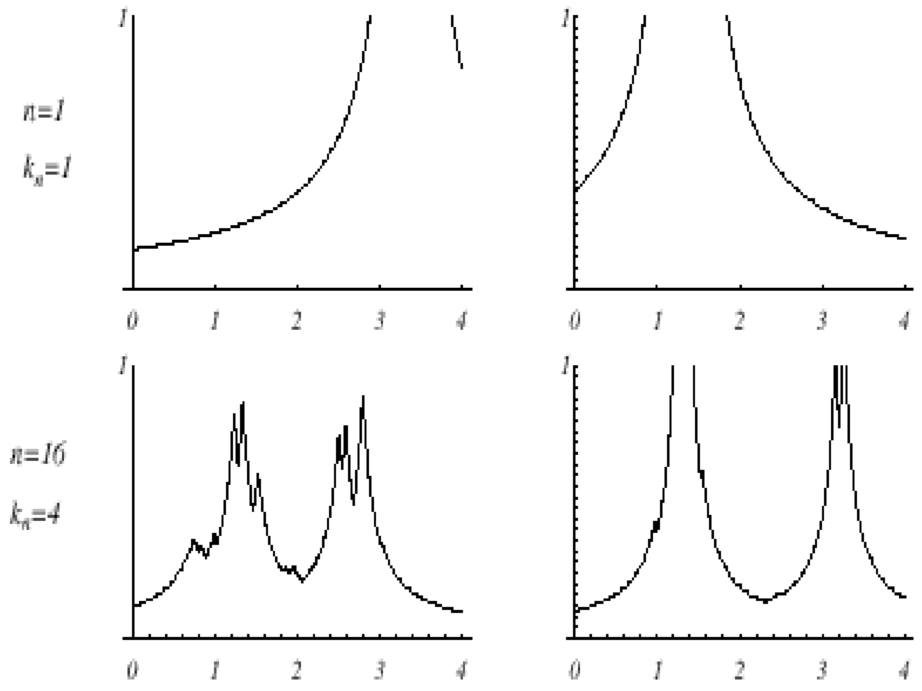
k_n-Nearest Neighbor Estimation

- k_n nearest-neighbor illustration
  - For $k_n = \sqrt{n}$ when n=1, k_1=1, and the estimate is

$$p_n(x) = \frac{1}{2\,|\,x - x_1\,|}$$

    where x_1 is the first training sample, and |·| is a distance measure. 2|x - x_1| is the volume. This is not a good estimate (Figure 4.12)
  - As n increases, the estimate gets better.
  - This method will not generate zero $p(\mathbf{x})$ for any $\mathbf{x}$. (If a fixed window, e.g. Parzen window, is used and no sample falls inside this window, the density estimate for this window will be zero. This will not happen here.)

k_n-Nearest Neighbor Estimation



k_n-Nearest Neighbor Estimation

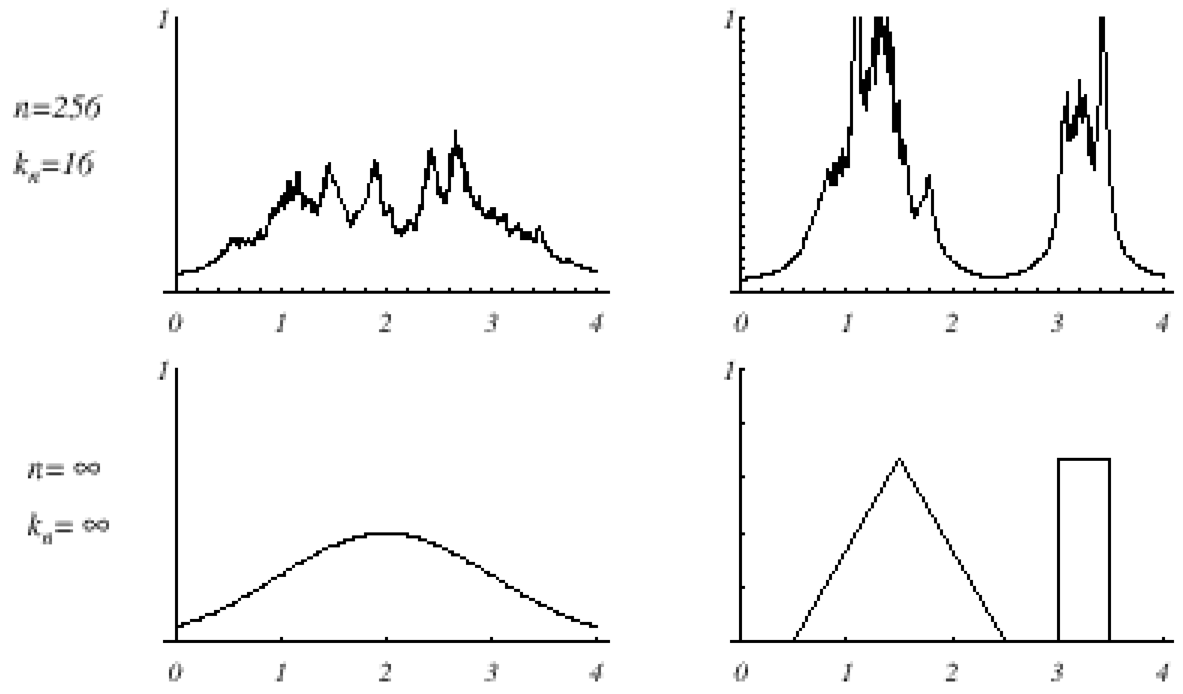


FIGURE 4.12. Several k-nearest-neighbor estimates of two unidimensional densities: a Gaussian and a bimodal distribution. Notice how the finite n estimates can be quite "spiky." From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

k_n-Nearest Neighbor Estimation

- We can obtain a family of estimates by making $k_n = k_1\sqrt{n}$ and adjusting the value k_1.
- Similar to Parzen window method, the choice of k_1 is case dependent.
- Usually k_1 is selected in the way that, when the estimated density is applied to classify new test samples from the same density, it yields the lowest error rate.

k_n-Nearest Neighbor Estimation

- Estimation of *a posteriori* probabilities. We can estimate $P(\omega_i|x)$ from a set of n labeled samples using the window methods
  - We place a cell of volume V around $\mathbf{x}$ and capture k samples
  - If k_i samples among these k samples turn out to be labeled ω_i then the joint probability $p(\mathbf{x},\omega_i)$ can be

$$p_n(x,\omega_i) = \frac{k_i / n}{V}$$

k_n-Nearest Neighbor Estimation

– Then a reasonable estimate of *a posteriori* probability is

$$P_n(\omega_i \mid x) = \frac{p_n(x, \omega_i)}{\sum_{j=1}^{c} p_n(x, \omega_j)} = \frac{k_i}{k}$$

– k_i/k is the fraction of the samples within the cell that are labeled ω_j, i.e. k samples in the cell and k_i out of k are labeled ω_j

– For minimum error rate, the most frequently represented category within the cell is selected for this cell, and any test sample lies in this cell is labeled as this category.

The Nearest Neighbor Rule

- Let $D^n = \{x_1, x_2, \ldots, x_n\}$ be a set of n labeled prototypes
- Let $x' \in D^n$ be the closest prototype to a test point x then the nearest-neighbor rule for classifying x is to assign it the label associated with x'
- The nearest-neighbor rule leads to an error rate greater than the minimum possible -- the Bayes rate
- If the number of prototype is large (unlimited), the error rate of the nearest-neighbor classifier is never worse than twice the Bayes rate

The Nearest Neighbor Rule

- The label θ' associated with the nearest neighbor $\mathbf{x}'$ is a random variable, and the probability that $\theta' = \omega_i$ is the *a posteriori* probability $P(\omega_i|x')$.
- If $n \to \infty$, it is always possible to find $\mathbf{x}'$ sufficiently close to $\mathbf{x}$ so that $P(\omega_i \mid x') \cong P(\omega_i \mid x)$
- We define $\omega_m(x)$ that $P(\omega_m| x) = \max_i P(\omega_i| x)$. Then the Bayes rule always select ω_m for $\mathbf{x}$.

The Nearest Neighbor Rule

- This rule essentially partitions the feature space into cells and each cell containing a prototype $\boldsymbol{x}'$ and all points closer to it than to any other prototypes. All points in a cell are labeled by the category of this $\boldsymbol{x}'$, which is called Voronoi tesselation of the space.
- In each cell,
  - If $P(\omega_m \mid x) \cong P(\omega_i \mid x') \cong 1$, then the nearest neighbor selection is almost always the same as the Bayes selection
  - If $P(\omega_m \mid x) \cong P(\omega_i \mid x') \cong 1/c$, then the nearest neighbor selection is rarely the same as the Bayes selection, but their error rates are similar (i.e. both are random guess)

The Nearest Neighbor Rule

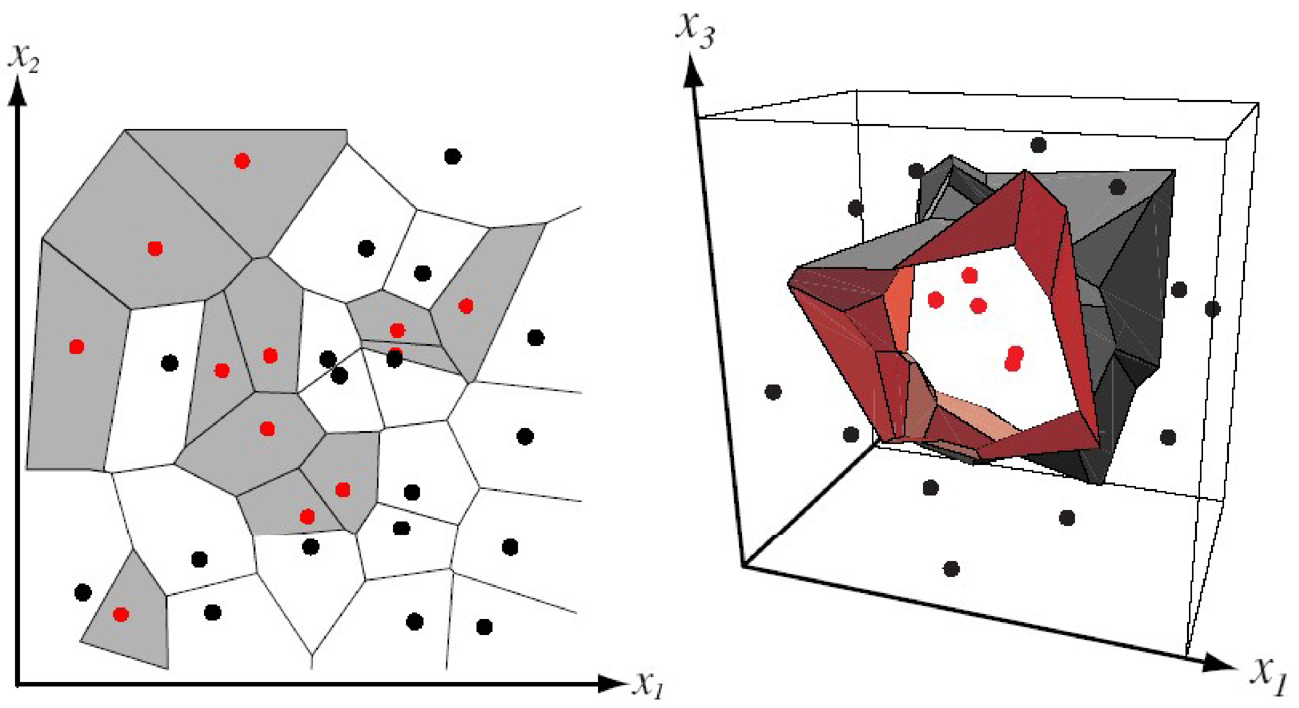


FIGURE 4.13. In two dimensions, the nearest-neighbor algorithm leads to a partitioning of the input space into Voronoi cells, each labeled by the category of the training point it contains. In three dimensions, the cells are three-dimensional, and the decision boundary resembles the surface of a crystal. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

The Nearest Neighbor Rule

- The average probability of error of the nearest-neighbor rule for infinite sample is

$$P(e) = \int P(e \mid x)\, p(x)\, dx$$

- The Bayes decision rule minimizes $P(e)$ by minimizing $P(e|\mathbf{x})$ for every $\mathbf{x}$, then

$$P^*(e \mid x) \triangleq \min\left(P(e \mid x)\right) = 1 - P(\omega_m \mid x)$$

$$P^* \triangleq \min\left(P(e)\right) = \int P^*(e \mid x)\, p(x)\, dx$$

The Nearest Neighbor Rule

- When only n samples are used in nearest neighbor rule, the conditional probability of error becomes

$$P(e|x) = \int P(e|x, x')p(x'|x)dx'$$

where $\mathbf{x}'$ is the nearest neighbor prototype of $\mathbf{x}$.

  – Each time when we take n samples, the nearest neighbor $\mathbf{x}'$ may be different, i.e. $\mathbf{x}'$ is a random variable.

  – As $n \rightarrow \infty$, $p(\mathbf{x}'|\mathbf{x})$ approaches a delta function centered at $\mathbf{x}$, $p(\mathbf{x}'|\mathbf{x}) \rightarrow \delta(\mathbf{x}'-\mathbf{x})$

    (i.e. a nearest neighbor $\mathbf{x}'$ can be always found very close to $\mathbf{x}$).

The Nearest Neighbor Rule

- To solve $P_n(e|\mathbf{x}, \mathbf{x}')$
  - We have n pairs of random variables $\{(\mathbf{x}_1,\theta_1), (\mathbf{x}_2,\theta_2), \ldots, (\mathbf{x}_n,\theta_n)\}$, where θ_j is class label for $\mathbf{x}_j$ and $\theta_j \in \{\omega_1, \omega_2, \ldots, \omega_c\}$
  - Because the state of nature when $\mathbf{x}_n'$ (the nearest neighbor of $\mathbf{x}$ when total sample is n) is drawn is independent of the state of nature when $\mathbf{x}$ is drawn, we have

$$P(\theta, \theta_n' \mid x, x_n') = P(\theta \mid x) P(\theta_n' \mid x_n')$$

The Nearest Neighbor Rule

- If we use the nearest-neighbor decision rule, the error occurs when $\theta \neq \theta'_n$, therefore

$$P_n(e \mid x, x'_n) = 1 - \sum_{i=1}^{c} P(\theta = \omega_i, \theta'_n = \omega_i \mid x, x'_n)$$

$$= 1 - \sum_{i=1}^{c} P(\omega_i \mid x) P(\omega_i \mid x'_n)$$

$$\lim_{n \to \infty} P_n(e \mid x) = \int \left[1 - \sum_{i=1}^{c} P(\omega_i \mid x) P(\omega_i \mid x'_n) \right] \delta(x'_n - x)\, dx'_n$$

$$= 1 - \sum_{i=1}^{c} P^2(\omega_i \mid x)$$

The Nearest Neighbor Rule

- The overall asymptotic nearest-neighbor error rate is

$$\begin{aligned} P &= \lim_{n\to\infty} P_n(e) \\ &= \lim_{n\to\infty} \int P_n(e \mid x)\, p(x)\, dx \\ &= \int \left[1 - \sum_{i=1}^{c} P^2(\omega_i \mid x) \right] p(x)\, dx \end{aligned}$$

- The error rate is bounded (proof in Sec 4.5.3)

$$P^* \le P \le P^* \left(2 - \frac{c}{c-1} P^* \right)$$

The Nearest Neighbor Rule

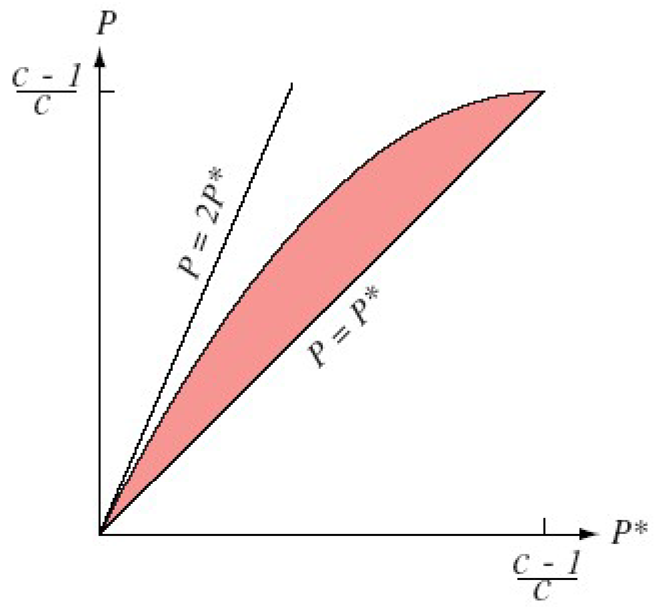


FIGURE 4.14. Bounds on the nearest-neighbor error rate P in a c-category problem given infinite training data, where P^* is the Bayes error (Eq. 52). At low error rates, the nearest-neighbor error rate is bounded above by twice the Bayes rate. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

The *k*-Nearest Neighbor Rule

- The *k*-nearest neighbor rule is an extension of the nearest neighbor rule.
- Classify ***x*** by assigning it the label most frequently represented among the *k* nearest samples and use a voting scheme
- When the total number of prototypes approaches infinity, these *k* neighbors will all converge to ***x***.
- In a two-class case, the *k*-nearest neighbor rule selects ω_m if a majority of the *k* neighbors are labeled ω_m, this event has the probability

$$\sum_{i=(k+1)/2}^{k} \binom{k}{i} P(\omega_m \mid x)^i [1 - P(\omega_m \mid x)]^{k-i}$$

The *k*-Nearest Neighbor Rule

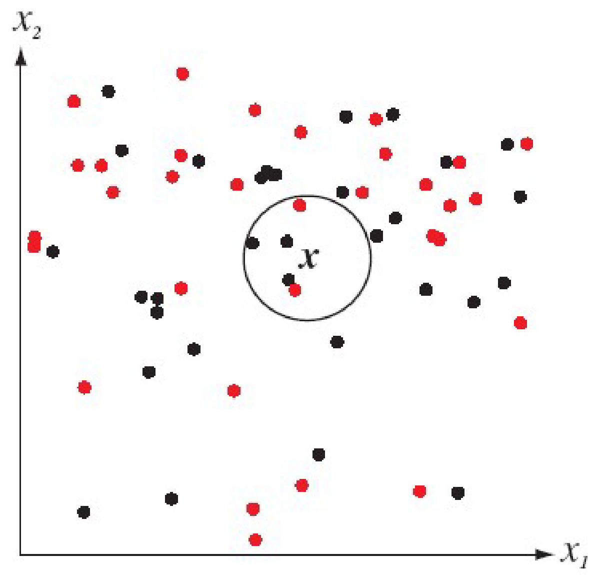


FIGURE 4.15. The k-nearest-neighbor query starts at the test point **x** and grows a spherical region until it encloses k training samples, and it labels the test point by a majority vote of these samples. In this $k = 5$ case, the test point **x** would be labeled the category of the black points. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

The *k*-Nearest Neighbor Rule

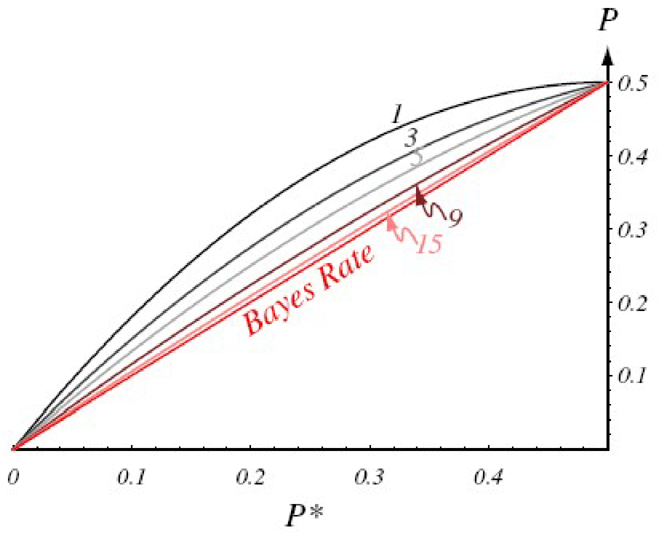


FIGURE 4.16. The error rate for the k-nearest-neighbor rule for a two-category problem is bounded by $C_k(P^*)$ in Eq. 54. Each curve is labeled by k; when $k = \infty$, the estimated probabilities match the true probabilities and thus the error rate is equal to the Bayes rate, that is, $P = P^*$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

The *k*-Nearest Neighbor Rule

- Example

  $k = 3$ (odd value) and

  $\mathbf{x} = (0.10, 0.25)^t$

Prototypes	Labels
(0.15, 0.35)	ω_1
(0.10, 0.28)	ω_2
(0.09, 0.30)	ω_1
(0.12, 0.20)	ω_2

3 closest vectors to $\mathbf{x}$ with their labels are:

$\{(0.10, 0.28; \omega_2); (0.12, 0.20; \omega_2); (0.15, 0.35; \omega_1)\}$

The majority voting scheme will assigns the label ω_2 to $\mathbf{x}$.

Metrics and Nearest Neighbor Classification

- The nearest neighbor classifier relies on certain distance function – metric
- Frequently we assume the metric is Euclidean distance in d dimensions, but it can be a generalized scalar distance between two argument patterns $D(\cdot\ ,\ \cdot)$
- A metric must have four properties: for any given vectors $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$
  - Non-negativity: $D(\boldsymbol{a},\boldsymbol{b})\geq 0$
  - Reflexivity: $D(\boldsymbol{a},\boldsymbol{b})=0$ iff $\boldsymbol{a}=\boldsymbol{b}$
  - Symmetry: $D(\boldsymbol{a},\boldsymbol{b})=D(\boldsymbol{b},\boldsymbol{a})$
  - Triangle inequality: $D(\boldsymbol{a},\boldsymbol{b})+D(\boldsymbol{b},\boldsymbol{c})\geq D(\boldsymbol{a},\boldsymbol{c})$

Metrics and Nearest Neighbor Classification

- The Euclidean distance in *d* dimensions satisfies these properties

$$D(a,b) = \left(\sum_{k=1}^{d} (a_k - b_k)^2 \right)^{1/2}$$

- Euclidean distance is very sensitive to the scales (units) of the coordinates, which has negative impact to the performance of nearest-neighbor classifiers

Metrics and Nearest Neighbor Classification

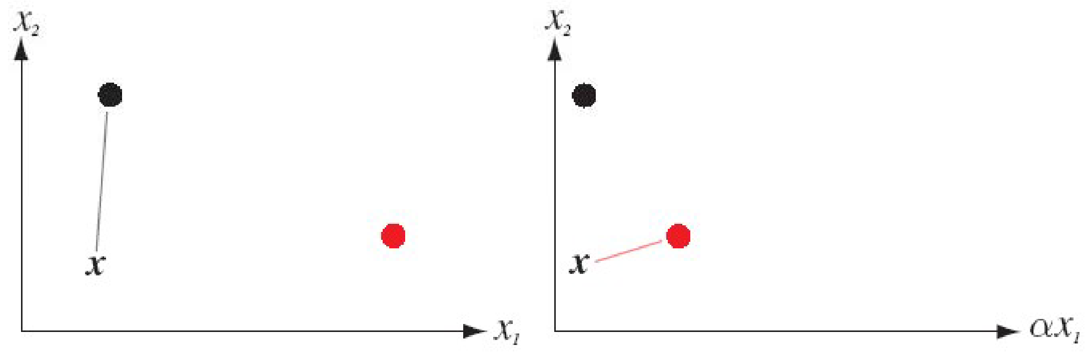


FIGURE 4.18. Scaling the coordinates of a feature space can change the distance relationships computed by the Euclidean metric. Here we see how such scaling can change the behavior of a nearest-neighbor classifer. Consider the test point **x** and its nearest neighbor. In the original space (left), the black prototype is closest. In the figure at the right, the x_1 axis has been rescaled by a factor 1/3; now the nearest prototype is the red one. If there is a large disparity in the ranges of the full data in each dimension, a common procedure is to rescale all the data to equalize such ranges, and this is equivalent to changing the metric in the original space. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Metrics and Nearest Neighbor Classification

- Minkowski metric, also referred to as the L_k norm

$$L_k(a,b) = \left(\sum_{k=1}^{d} | a_k - b_k |^k \right)^{1/k}$$

  - Euclidean distance is the L_2 norm
  - L_1 norm is referred to as the Manhattan distance
  - L_∞ distance between ***a*** and ***b*** is the maximum of the projections of |***a***-***b***| on the d coordinate axes.

Metrics and Nearest Neighbor Classification

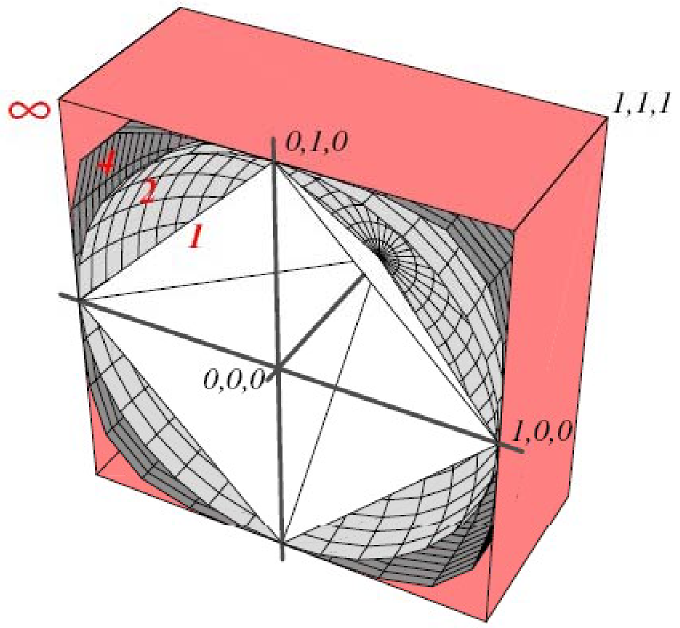


FIGURE 4.19. Each colored surface consists of points a distance 1.0 from the origin, measured using different values for *k* in the Minkowski metric (*k* is printed in red). Thus the white surfaces correspond to the L_1 norm (Manhattan distance), the light gray sphere corresponds to the L_2 norm (Euclidean distance), the dark gray ones correspond to the L_4 norm, and the pink box corresponds to the L_∞ norm. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Metrics and Nearest Neighbor Classification

- Tanimoto metric, for two sets S_1 and S_2

$$D_{Tanimoto}(S_1, S_2) = \frac{n_1 + n_2 - 2n_{12}}{n_1 + n_2 - n_{12}}$$

where n_1 and n_2 are the number of elements in set S_1 and S_2 and n_{12} is the number in both sets.

  – Tanimoto metric is frequently used in taxonomy

Metrics and Nearest Neighbor Classification

- Tanimoto metric examples:
  - Consider four words as sets of unordered letters:

pattern, pat, stop, pots

$$D(pattern, pat) = \frac{7+3-2\times 3}{7+3-3} = \frac{4}{7}, \quad D(pattern, stop) = \frac{7+4-2\times 2}{7+4-2} = \frac{7}{9}$$

$$D(pattern, pots) = \frac{7+4-2\times 2}{7+4-2} = \frac{7}{9}, \quad D(pat, stop) = \frac{3+4-2\times 2}{3+4-2} = \frac{3}{5}$$

$$D(pat, pots) = \frac{3+4-2\times 2}{3+4-2} = \frac{3}{5}, \quad D(stop, pots) = \frac{4+4-2\times 4}{4+4-4} = 0$$

Metrics and Nearest Neighbor Classification

- Uncritical use of a particular metric in nearest-neighbor classifier can cause low performance
  - The metric needs to be invariant to common transforms such as translation, rotation, scaling etc.
  - It is very difficult to make a metric invariant to multiple transforms
  - Typical solutions may include pre-processing two patterns to coalign, shifting the centers and placing in same bounding box etc. Automatic pre-processing can also be difficult and unreliable.

Metrics and Nearest Neighbor Classification

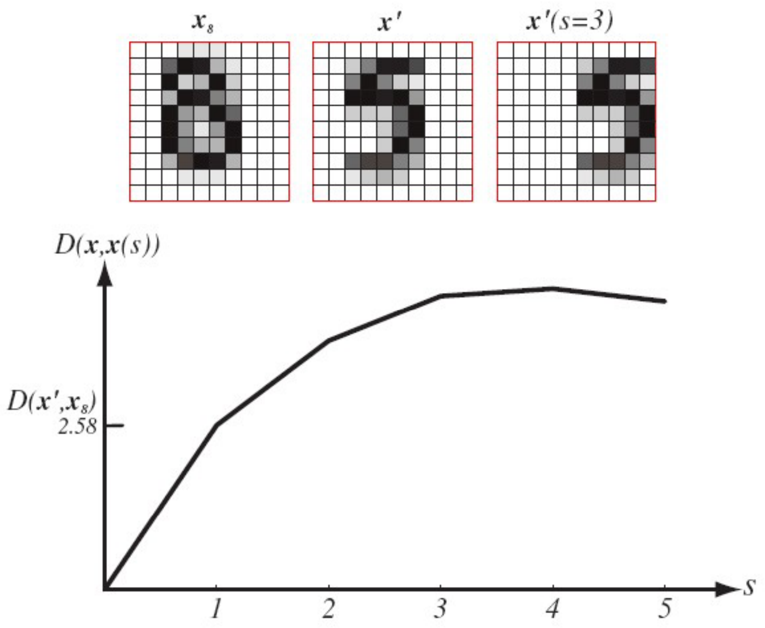


FIGURE 4.20. The uncritical use of Euclidean metric cannot address the problem of translation invariance. Pattern $\mathbf{x}'$ represents a handwritten 5, and $\mathbf{x}'(s = 3)$ represents the same shape but shifted three pixels to the right. The Euclidean distance $D(\mathbf{x}', \mathbf{x}'(s = 3))$ is much larger than $D(\mathbf{x}', \mathbf{x}_8)$, where $\mathbf{x}_8$ represents the handwritten 8. Nearest-neighbor classification based on the Euclidean distance in this way leads to very large errors. Instead, we seek a distance measure that would be insensitive to such translations, or indeed other known invariances, such as scale or rotation. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Metrics and Nearest Neighbor Classification

- Tangent distance classifier is to use a novel distance measure and a linear approximation to the arbitrary transforms.
  - Assume a classifier needs to handle r transforms, such as horizontal translation, vertical translation, shear, rotation, scale and line thinning
  - We take each prototype x' and perform each of the transforms $F_i(x'; \alpha_i)$ where α_i is the parameter associated with this transform, such as the angle in rotation.

Metrics and Nearest Neighbor Classification

– A tangent vector TV_i is constructed for each transform

$$TV_i = F_i(\mathbf{x}' ; \alpha_i) - \mathbf{x}'$$

– For each d-dimensional prototype $\mathbf{x}'$, an $r \times d$ matrix T is generated, consisting of the tangent vectors at $\mathbf{x}'$. These vectors are linearly independent.

– The prototype plus a linear combination of all tangent vectors forms an approximation of an arbitrary transform.

Metrics and Nearest Neighbor Classification

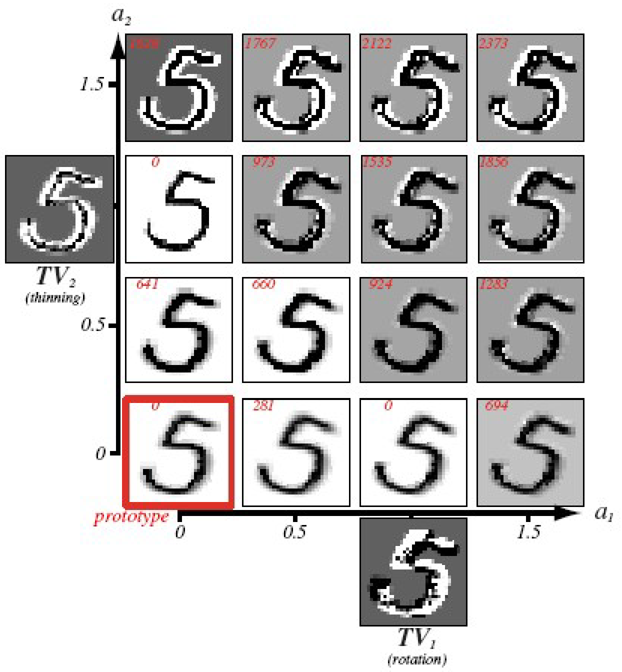


FIGURE 4.21. The pixel image of the handwritten 5 prototype at the lower left was subjected to two transformations, rotation, and line thinning, to obtain the tangent vectors $\mathbf{TV}_1$ and $\mathbf{TV}_2$; images corresponding to these tangent vectors are shown outside the axes. Each of the 16 images within the axes represents the prototype plus linear combination of the two tangent vectors with coefficients a_1 and a_2. The small red number in each image is the Euclidean distance between the tangent approximation and the image generated by the unapproximated transformations. Of course, this Euclidean distance is 0 for the prototype and for the cases $a_1 = 1, a_2 = 0$ and $a_1 = 0, a_2 = 1$. (The patterns generated with $a_1 + a_2 > 1$ have a gray background because of automatic grayscale conversion of images with negative pixel values.) From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Metrics and Nearest Neighbor Classification

- The tangent distance from a test point x to a particular stored prototype x' is defined as

$$D_{tan}(\mathbf{x}', \mathbf{x}) = \min_{\boldsymbol{a}} [||(\mathbf{x}' + \boldsymbol{T}\boldsymbol{a}) - \mathbf{x}||]$$

  where $\boldsymbol{T}$ is a matrix consisting of the r tangent vectors at $\mathbf{x}'$, $\boldsymbol{a}$ is a vector of parameters for linear combination, and $||\cdot||$ can be Euclidean distance.
- In classification of $\mathbf{x}$, we will first find its tangent distance to $\mathbf{x}'$ by finding the optimizing value of $\boldsymbol{a}$. This minimization is quadratic, and can be done using iterative gradient descent.

Metrics and Nearest Neighbor Classification

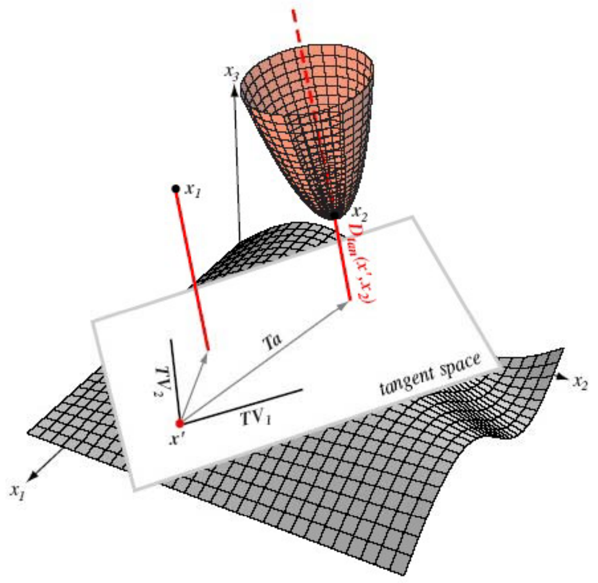


FIGURE 4.22. A stored prototype $\mathbf{x}'$, if transformed by combinations of two basic transformations, would fall somewhere on a complicated curved surface in the full d-dimensional space (gray). The tangent space at $\mathbf{x}'$ is an r-dimensional Euclidean space, spanned by the tangent vectors (here $\mathbf{TV}_1$ and $\mathbf{TV}_2$). The tangent distance $D_{tan}(\mathbf{x}', \mathbf{x})$ is the smallest Euclidean distance from $\mathbf{x}$ to the tangent space of $\mathbf{x}'$, shown in the solid red lines for two points, $\mathbf{x}_1$ and $\mathbf{x}_2$. Thus although the Euclidean distance from $\mathbf{x}'$ to $\mathbf{x}_1$ is less than that to $\mathbf{x}_2$, for the tangent distance the situation is reversed. The Euclidean distance from $\mathbf{x}_2$ to the tangent space of $\mathbf{x}'$ is a quadratic function of the parameter vector $\mathbf{a}$, as shown by the pink paraboloid. Thus simple gradient descent methods can find the optimal vector $\mathbf{a}$ and hence the tangent distance $D_{tan}(\mathbf{x}', \mathbf{x}_2)$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.