Point Estimation and MLE

♢ *parameter space* - $\Omega = \{all\ \theta\}$

♢ *estimator* - a random variable (or vector)

♢ *estimate* - a value (vector) derived from a realization

♢ *(log)-likelihood function* - $\prod_{i=1}^{n} f(x_i; \theta)$.

♢ *maximum likelihood estimator (estimate)*[mle]

Definition: If $E[u(X_1, X_2, \ldots, X_n)] = \theta$, the statistic, $u(X_1, X_2, \ldots, X_n)$ is called an *unbiased estimator* of θ. Otherwise, it is said to be *biased*.

♢ parameter estimation by *method of moments*.

Example 1: mle for the mean and variance of a random sample of size n from $N(\theta_1, \theta_2)$.

Example 2: mle for the mean of a random sample of size n from an exponential distribution with a mean $\alpha > 0$.

□ *Maximum Likelihood Estimation (MLE)*

Let $X_1, X_2, \ldots, X_n$ be a random sample with parameter(s) $\theta \in \Omega$. Observing n independent results $x_1, x_2, \ldots, x_n$, one wants to find a *statistic* $u(X_1, X_2, \ldots, X_n)$ (called an **estimator**) to estimate θ such that $u(x_1, x_2, \ldots, x_n)$ is close to θ.

Example 1: $X_1, X_2, \ldots, X_n \sim b(1, p) \quad \Rightarrow \quad \widehat{p} = \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$

Example 2: $X_1, X_2, \ldots, X_n \sim Exp(\theta) \quad \Rightarrow \quad \widehat{p} = \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$

Example 3: $X_1, X_2, \ldots, X_n \sim Geometric(p) \quad \Rightarrow \quad \widehat{p} = \frac{1}{\overline{X}} = \frac{n}{\sum_{i=1}^{n} X_i}$

Maximum Likelihood Estimator

Let $X_1, X_2, \ldots, X_n$ be a random sample from $N(u, \sigma^2)$. Observing n independent results $x_1, x_2, \ldots, x_n$, the maximum likelihood estimators for mean and variance are given below.

$$\widehat{u} = \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

$$\widehat{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^2$$

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \widehat{s^2} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

Note that $\frac{n}{n-1}\widehat{\sigma^2}$ is an *unbiased* estimator.

Let $X_1, X_2, \ldots, X_n$ be a random sample from $N(\mathbf{u}, C)$. Observing n independent results $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, the maximum likelihood estimators for mean vector and covariance matrix are given below.

$$\widehat{\mathbf{u}} = \bar{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{X}_i$$

$$\widehat{C} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{\mathbf{X}})(X_i - \overline{\mathbf{X}})^2$$

• **Confidence Intervals for Means**

♣ Given a random sample $X_i \sim N(\mu, \sigma^2)$ of size n, we want to know how *close* of the *unbiased estimator*, $\overline{X}$, to the unknown mean μ.

$$P\left[\overline{X} - z_{\alpha/2} * (\frac{\sigma}{\sqrt{n}}) \leq \mu \leq \overline{X} + z_{\alpha/2} * (\frac{\sigma}{\sqrt{n}})\right] = 1 - \alpha \tag{1}$$

$$\left[\overline{X} - z_{\alpha/2} * (\frac{\sigma}{\sqrt{n}}), \quad \overline{X} + z_{\alpha/2} * (\frac{\sigma}{\sqrt{n}})\right] \tag{2}$$

♢ *confidence intervals*

♢ *confidence coefficient*

♢ *The effect of sample size*

♡ See Problem 13 on P.144 of Duda, Hart, and Stork's Book

Bayesian Estimation

Let D denote the set of training samples, our goal is to compute the a posteriori probabilities $P(\omega_i|\mathbf{x}, D)$.

$$P(\omega_i|\mathbf{x}, D) = [P(\mathbf{x}|\omega_i, D)P(\omega_i|D)]/[\sum_{j=1}^{K} P(\mathbf{x}|\omega_j, D)P(\omega_j|D)]$$

$$P(\omega_i|\mathbf{x}) = [P(\mathbf{x}|\omega_i)P(\omega_i)]/[\sum_{j=1}^{K} P(\mathbf{x}|\omega_j)P(\omega_j)]$$

The above discussion is based on the assumption that parameters for the class-conditional distributions are fixed and the samples for each training category are independent of the other training categories. If the parameters are themselves random variables from some distributions, the problem becomes more difficult which is briefly discussed in the *Duda, Hart, and Stork's* book (P.92∼107).

Quadratic Classifiers

Consider n_i training patterns (vectors) $\{\mathbf{v}_1^{(i)}, \mathbf{v}_2^{(i)}, \ \ldots, \ \mathbf{v}_{n_i}^{(i)}\}$, from class ω_i for $i = 1, 2, \ldots, K$. The maximum likelihood estimators of mean vector and covariance matrix for each pattern class can be written as

$$\mathbf{u}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \mathbf{v}_j^{(i)}, \quad C_i = \frac{1}{n_i} \sum_{j=1}^{n_i} [\mathbf{v}_j^{(i)} - \mathbf{u}_i][\mathbf{v}_j^{(i)} - \mathbf{u}_i]^t,$$

for $i = 1, 2, \ \ldots, \ K$, respectively.

A quadratic classifier based on the multivariate normal model can be defined as Assign $\mathbf{x}$ to class ω_i if

$$(\mathbf{x} - \mathbf{u}_j)^t C_j^{-1} (\mathbf{x} - \mathbf{u}_j) \ - \ (\mathbf{x} - \mathbf{u}_i)^t C_i^{-1} (\mathbf{x} - \mathbf{u}_i) \ \geq \ 2ln \left[\frac{P(\omega_j)\sqrt{|C_i|}}{P(\omega_i)\sqrt{|C_j|}} \right],$$

for all $j \neq i$.

where $|C_i| = det(C_i)$ and $P(\omega_i) = n_i/(n_1 + n_2 + \ldots + n_K)$ is the estimated prior probability for class ω_i.

If the underlying class-conditional probability densities are multivariate normal with known parameters, then the form of the above decision rule is *Bayes optimal*.

Fisher's Linear Discriminant (Classifier, FLD)

Consider n_i training patterns (vectors) $\{\mathbf{v}_1^{(i)}, \mathbf{v}_2^{(i)}, \ \ldots, \ \mathbf{v}_{n_i}^{(i)}\}$, from class ω_i for $i = 1, 2, \ldots, K$. The maximum likelihood estimators of mean vector and covariance matrix for each pattern class can be written as

$$\mathbf{u}_i = \frac{1}{n_i}\sum_{j=1}^{n_i}\mathbf{v}_j^{(i)}, \quad C_i = \frac{1}{n_i}\sum_{j=1}^{n_i}[\mathbf{v}_j^{(i)} - \mathbf{u}_i][\mathbf{v}_j^{(i)} - \mathbf{u}_i]^t,$$

for $i = 1, 2, \ldots, K$, respectively.

The pooled mean vector and the within-class scatter matrix are estimated by

$$\mathbf{u} = \frac{1}{n}\sum_{i=1}^{K} n_i\mathbf{u}_i, \quad S = \frac{1}{n}\sum_{i=1}^{K} n_i C_i, \quad where \ \ n = n_1 + n_2 + \ldots + n_K.$$

The squared Mahalanobis distance between a (test) pattern $\mathbf{x}$ and the estimated mean vector of class ω_j is denoted by $g_j(\mathbf{x})$, where

$$g_j(\mathbf{x}) = (\mathbf{x} - \mathbf{u}_j)^t S^{-1}(\mathbf{x} - \mathbf{u}_j), \quad for \ j = 1, 2, \ \ldots, \ K.$$

A Fisher's linear discriminant (classifier, FLD) can be defined as
Assign $\mathbf{x}$ to class ω_i if

$$g_i(\mathbf{x}) = min_{1 \le j \le K}\{g_j(\mathbf{x})\}$$

Note that an alternate name, linear discriminant analysis (LDA), is more frequently used than FLD in the research areas of Face Recognition.

Nearest Neighbor (NN) Decision Rule

The nearest neighbor rule is defined and the relation between the error probability of this non-parametric rule and the error probability of the Bayes rule is derived. *This result is unique because very few theoretical results exist which relate the error probabilities of heuristic rules to the Bayes error.*

The nearest neighbor decision rule, $\delta_{nn}(\bullet)$ assigns the category k to point (pattern) $\mathbf{x}$ in the pattern space if the training pattern closest to $\mathbf{x}$ is from the pattern class ω_k. That is

$$\delta_{nn}(\mathbf{x}) = \omega_k \;\; if \;\; \|\mathbf{x} - \mathbf{x}_j\| < \|\mathbf{x} - \mathbf{x}_i\|, \;\; for\ i = 1, 2, \ldots, n, \; where\ i \neq j \; \mathbf{x}_j \in \omega_k$$

Here, $\|\mathbf{x} - \mathbf{y}\|$ is the distance between $\mathbf{x}$ and $\mathbf{y}$ in the d-dimensional pattern space under some suitable metric, such as Euclidean distance. The nearest neighbor classifications are illustrated below.

The nearest neighbor rule can be justified from intuition, since it matches the training data. It is certainly true that, assuming no training patterns lie on the top of one another, it correctly classifies all the training patterns. However, at first glance, the rule may seem difficult to implement. *Does one really need to classify* $\mathbf{x}$ *by finding the distance to all of the training patterns*?

Asymptotic Error Probability for NN Rule

Let the ramdom vector $(\mathbf{X}, \theta)$ represent $\mathbf{x}$, the pattern to be classified with the true pattern class θ. Denote the nearest neighbor to $\mathbf{X}$ from among the n training patterns of the random vector $(\mathbf{X}', \theta')$. The training patterns and the pattern to be classified are assumed to be statistically independent. The error probability of $\delta_{nn}(\mathbf{x})$ when $\mathbf{x}$ is classified is given below.

$$e[\mathbf{x}, \mathbf{x}'(n)] = 1 - \sum_{j=1}^{K} \{P(\theta = \omega_j | \mathbf{X} = \mathbf{x}) \times P(\theta' = \omega_j | \mathbf{X}'(n) = \mathbf{x}'(n))\}$$

The posterior probability of class ω_j given $\mathbf{x}$ is written as

$$\eta_j(\mathbf{x}) = P(\theta = \omega_j | \mathbf{X} = \mathbf{x})$$

Then

$$e[\mathbf{x}, \mathbf{x}'(n)] = 1 - \sum_{j=1}^{K} \eta_j(\mathbf{x}) \eta_j(\mathbf{x}'(n))$$

If the nearest neighbor random vector $\mathbf{X}'(n)$ converges *in probability* to $\mathbf{X}$ as $n \rightarrow \infty$, we have

$$e[\mathbf{x}, \mathbf{x}'(n)] \rightarrow 1 - \sum_{j=1}^{K} \eta_j^2(\mathbf{x}) \equiv e_1(\mathbf{x}) \quad as \ n \rightarrow \infty$$

Thus, $e_1(\mathbf{x})$ is the asymptotic error probability when $\mathbf{x}$ is classified. Averaging with respect to the (mixture) distribution of $\mathbf{X}$, denoted as $\rho(\cdot)$, produces the following expression for the asymptotic error probability of the nearest neighbor decision rule.

$$E_1 = \int e_1(\mathbf{x}) \rho(\mathbf{x}) d\mathbf{x} = \int \left(1 - \sum_{j=1}^{K} \eta_j^2(\mathbf{x})\right) \rho(\mathbf{x}) d\mathbf{x}$$

$$E_1 = \int e_1(\mathbf{x})\rho(\mathbf{x})d\mathbf{x} = \int \left(1 - \sum_{j=1}^{K} \eta_j^2(\mathbf{x})\right) \rho(\mathbf{x})d\mathbf{x}$$

However, the error probability of the Bayes rule, E^*, is

$$E^* = \int e^*(\mathbf{x})\rho(\mathbf{x})d\mathbf{x}$$

Thus,

$$E_1 \;\leq\; 2E^* - \frac{K}{K-1} \times \int [e^*(\mathbf{x})]^2 \rho(\mathbf{x})d\mathbf{x}$$

Since

$$0 \leq Var[e^*(\mathbf{x})] = \int [e^*(\mathbf{x})]^2 \rho(\mathbf{x})d\mathbf{x} - \left(\int e^*(\mathbf{x})\rho(\mathbf{x})d\mathbf{x}\right)^2$$

Then the inequality becomes

$$E_1 \;\leq\; 2E^* - \frac{K}{K-1} \times (E^*)^2 \;\leq\; 2E^*$$

which shows how the asymptotic error probability of the nearest neighbor rule is bounded above by the Bayes error. The Bayes rule has minimum probability of error. Therefore, we can relate the two error probabilities as

$$E^* \;\leq\; E_1 \;\leq\; 2E^*$$

This result is often abused by the statement: *The nearest neighbor error probability is bounded by twice the Bayes error probability.* It must be mentioned that it is the *asymptotic error probability* that is so bounded, but not the finite-sample error probability.

Relation Between Asymptotic Error of NN Rule and Bayes Error

Suppose the pattern class ω_m has the largest posterior probability when $\mathbf{x}$ is observed, so that

$$\eta_m(\mathbf{x}) \geq \eta_j(\mathbf{x}) \quad for\ j = 1,\ 2,\ \ldots,\ K$$

The Bayes error when $\mathbf{x}$ is observed in the decision problem quoted above, called the conditional Bayes error, is

$$e^*(\mathbf{x}) = 1 - \eta_m(\mathbf{x})$$

Since $\eta_m(\mathbf{x})$ is defined as the largest posterior probability. A Cauchy-Schwarz inequality: $[\sum_{i=1}^K a_i^2][\sum_{i=1}^K b_i^2] \geq [\sum_{i=1}^K (a_i b_i)]^2$ implies that the sum of the squared posterior probabilities, excluding $\eta_m(\mathbf{x})$, can be bounded below as follows.

$$\sum_{j \neq m} \eta_j^2(\mathbf{x}) \geq [e^*(\mathbf{x})]^2/(K-1)$$

Adding $\eta_m^2(\mathbf{x})$ to the left side and its equivalent, $1 - e^*(\mathbf{x})$, to the right side and doing a bit of simplification shows that

$$1 - \sum_{j=1}^K \eta_j^2(\mathbf{x}) \leq 2e^*(\mathbf{x}) - \frac{K}{K-1} \times [e^*(\mathbf{x})]^2$$

This wonderful bound can be used in the equation for E_1, the asymptotic error of the nearest neighbor rule.

$$E_1 = \int \left\{ 2e^*(\mathbf{x}) - \frac{K}{K-1} \times [e^*(\mathbf{x})]^2 \right\} \rho(\mathbf{x}) d\mathbf{x}$$

The k-NN Decision Rule

A natural extension of the nearest neighbor rule is to examine the k nearest neighbors to $\mathbf{x}$ and classify $\mathbf{x}$ according to the pattern class most heavily represented among the k neighbors. In mathematical terms, let $\phi_i(k,n)$ be the number of patterns from class ω_i among the k nearest neighbors of $\mathbf{x}$. The nearest neighbors are computed from the n training patterns. The k-NN rule is defined below and demonstrated in the figure.

$$\delta_{k-NN}(\mathbf{x}) = \omega_j \quad if \quad \phi_j(k,n) > \phi_i(k,n) \quad \forall\, i \neq j$$

Ties must be handled by randomization, so we try to avoid them by choosing k properly. For example, if $K = 2$, we use $k-$NN rule with an odd number of k. A reject option can also be added so no decision is made unless the most frequently represented pattern class receives at least a specified number of votes.

References

[**1**] P.A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Chapter 3, 1982.

[**2**] P.A. Devijver, *A Multiclass k-NN Approach to Bayes Risk Estimation*, Pattern Recognition Letters, vol.3, 1∼6, 1985.

Perceptron Training Algorithm

Let $\mathbf{x}_1,\ \mathbf{x}_2,\ \ldots,\ \mathbf{x}_n\}$ be training patterns from classes ω_1 and ω_2, respectively. The problem of perceptron training is equivalent to the problem of solving a set of n inequalities

$$g(\mathbf{y}_j) = \mathbf{w}^t\mathbf{y}_j > 0 \quad \forall\ j,\ 1 \le j \le n$$

where $\{\mathbf{y}_j\}$ are augmented training patterns defined by

$$\mathbf{y}_i^t = [\ \mathbf{x}_i^t,\ \ 1] \qquad if\ \mathbf{x}_i \in \omega_1$$

$$\mathbf{y}_j^t = [-\mathbf{x}_j^t,\ \ -1] \quad if\ \mathbf{x}_j \in \omega_2$$

♣ *Example*

$$x_i:\ 2.0,\ 3.5,\ 4.0 \in \omega_1,\quad \mathbf{y}_i^t:\ [2.0, 1],\ [3.5, 1],\ [4.0, 1]$$

$$x_j:\ -1.5, 0.5, 1.0 \in \omega_2,\quad \mathbf{y}_j^t:\ [1.5, -1],\ [-0.5, -1], [-1, -1]$$

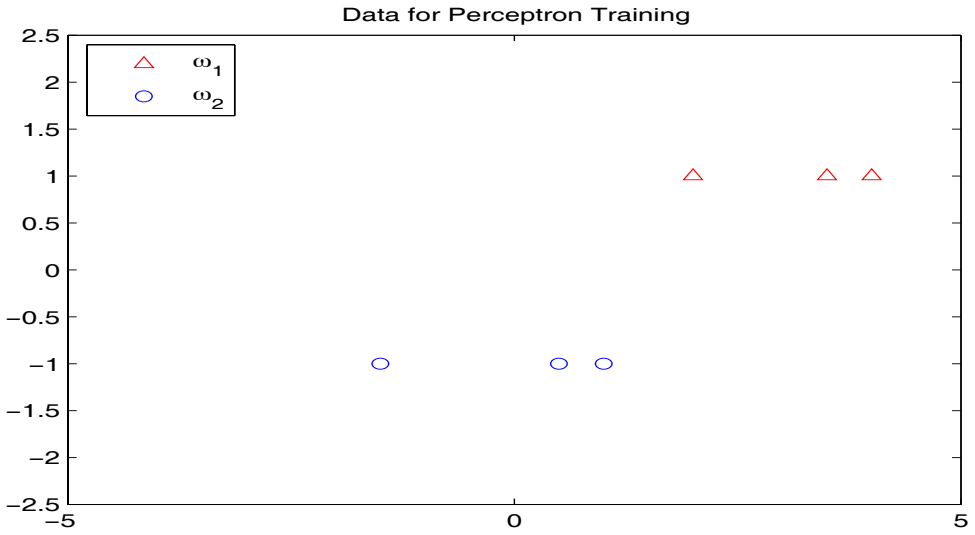

Figure 1: Five points for Perceptron Training

♢ Perceptron Training Algorithm

arbitrarily choose an initial $\mathbf{w}$

repeat

change ← false

for $i = 1$ to n

if $\mathbf{w}^t\mathbf{y}_j < 0$ then

$\mathbf{w} \leftarrow \mathbf{w} + \mathbf{y}_j$

change ← true

endif

endfor

until (not change)

Proof: By contradiction, assume that the the above algorithm may not converge.

Let $\mathbf{y}_1^*,\ \mathbf{y}_2^*,\ \ldots$ be the sequence of training patterns which update the vectors $\mathbf{w}$, and let $\mathbf{w}_0$ be a vector which separates two sets of training patterns. Denote $\beta = max_{1\le j\le n}\{\|\mathbf{y}_j\|^2\}$ and $\gamma = min_{1\le j\le n}\{\mathbf{w}_0^t\mathbf{y}_j\}$. Note that $\gamma > 0$. From $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{y}_k^*$, we have

$$\begin{aligned}\mathbf{w}_0^t\mathbf{w}_{k+1} &= \mathbf{w}_0^t(\mathbf{w}_k + \mathbf{y}_k^*) \\ &= \mathbf{w}_0^t\mathbf{w}_1 + \textstyle\sum_{j=1}^{k}\mathbf{w}_0^t\mathbf{y}_j^* \ge \mathbf{w}_0^t\mathbf{w}_1 + k\gamma,\end{aligned}$$

then

$$\|\mathbf{w}_{k+1}\|^2 \ge [\mathbf{w}_0^t\mathbf{w}_1 + k\gamma]^2/\|\mathbf{w}_0\|^2$$

On the other hand, $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{y}_k^*$ implies that

$$\|\mathbf{w}_{k+1}\|^2 - \|\mathbf{w}_k\|^2 = \|\mathbf{y}_k^*\|^2 + 2\mathbf{w}_k^t\mathbf{y}_k^* < \|\mathbf{y}_k^*\|^2, \quad then$$

$$\|\mathbf{w}_{k+1}\|^2 \le \|\mathbf{w}_1\|^2 + k\beta, \quad thus$$

$$[\mathbf{w}_0^t\mathbf{w}_1 + k\gamma]^2/\|\mathbf{w}_0\|^2 \le \|\mathbf{w}_{k+1}\|^2 \le \|\mathbf{w}_1\|^2 + k\beta.$$

The above inequality holds for k only up to an upper bound, in other words, it does not hold for $k \rightarrow \infty$. This completes the proof.

Nonparametric Density Estimation (1/3)

The main problem in statistical pattern recognition is to determine the class-conditional densities, $p(\mathbf{x}|\omega_i),\ i = 1,\ 2,\ \ldots,\ K$. These densities are seldom completely known in practice. We shall assume that n_i training patterns are available from class ω_i to estimate the densities. Nonparametric density estimates are data-driven and can easily be adapted to different types of data. Unlike the parametric methods, there are no *a priori* assumptions about the densities that suppress the details provided by the training data. A popular approach to nonparametric density estimation is the Parzen window estimator described below.

Let $\{\mathbf{y}_1,\ \mathbf{y}_2,\ \ldots,\ \mathbf{y}_n\}$ be a set of n d-dimensional independent and identically distributed patterns from a density function $p(\mathbf{y})$. The Parzen or kernel estimate of the density at $\mathbf{x}$ is given by [Parzen1962]

$$\hat{p}(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{V_h}\phi\left(\frac{\mathbf{x}-\mathbf{y}_j}{h}\right)$$

where ϕ is a window or kernel function with $\int_{-\infty}^{\infty}\phi(t)dt = 1$, and $h > 0$ is the window width or smoothing parameter.

The shape of the kernel is not as important as its width in density estimation, as long as it is symmetric, continuous, and unimodal. The choice of the bandwidth, h, is very critical to Parzen density estimation. Too small an h would give a *spiky* or noisy estimate of $p(\mathbf{y})$ with each spike corresponding to the kernel itself at the training patterns. When h is very large, each training pattern provides the same contribution towards density estimation at every point $\mathbf{x}$ and the result is an oversmoothed estimate of $p(\mathbf{x})$.

Nonparametric Density Estimation (2/3)

In practice, $h \propto n^{-\alpha/d}$, where $\alpha \in [-0,\ 0.5]$. The routine "parzen" uses the $d-$*dimensional* Gaussian kernel which leads to the following density estimate.

$$\hat{p}(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n}\left[\frac{1}{(h_0 n^{-\alpha/d})^d} \times \frac{1}{(2\pi)^{d/2}|C|^{1/2}} \times Exp\left(-\frac{1}{2(h_0 n^{-\alpha/d})^2}(\mathbf{x}-\mathbf{y}_j)^t C^{-1}(\mathbf{x}-\mathbf{y}_j)\right)\right]$$

where C is the covariance matrix which can be estimated from the training patterns. The subroutine *parzen* is provided for estimating density by the kernel approach.

Another nonparametric density estimate can be achieved based on the K-nn distance. Let r be the $k_n - th$ nearest distance from $\mathbf{x}$ to the training patterns in $\mathbf{Y} = \{\mathbf{y}_1,\ \mathbf{y}_2,\ \ldots,\ \mathbf{y}_n\}$. Denote the volume of a hypersphere with radius r as

$$V_r = \frac{\pi^{d/2} r^d}{\Gamma(\frac{d}{2}+1)}$$

Then the density can be estimated [Loft1965] as

$$\hat{f}(\mathbf{x}) = \frac{k_n - 1}{n} \times \frac{1}{V_r}$$

Note that $limit_{n\to\infty}\frac{k_n}{n} = 0$ is required. Usually, $k_n = \sqrt{n}$ is taken. The subroutine *denknn* is provided for estimating density by K-nn distance approach.

Parzen and Knn Nonparametric Density Estimation (3/3)

```
%  4-14-2008  Script file: denknn.m   <<Plot of density estimation>>
n=500;  Kn=33;          % Kn=ceil(sqrt(n));
h=0.25; a=0.3;  Vh=h;  % Vn=h^d where d=1 here
Y0=random('Normal',0,1,n,1);
T=-3:0.1:3;  K=61;
X0=exp(-T.^2/2)/sqrt(2*pi);
for i=1:K
  s=0; t=T(i);
  for j=1:n   s=s+exp(-(t-Y0(j,1))^2/(2*h*h));  end
  for j=1:n   dist(j)=(t-Y0(j,1))^2;  end
  Xh(i)=s/sqrt(2*pi)/Vh/n;
  H=sort(dist,'ascend');
  r=H(Kn);
  Yh(i)=(Kn-1)/n/(2*sqrt(r));
end
plot(T,X0,'g-',T,Xh,'b-',T,Yh,'r-'); axis([-3 3, 0,0.6]); grid
legend('N(0,1)','Parzen','Knn-Estimate',2)
```

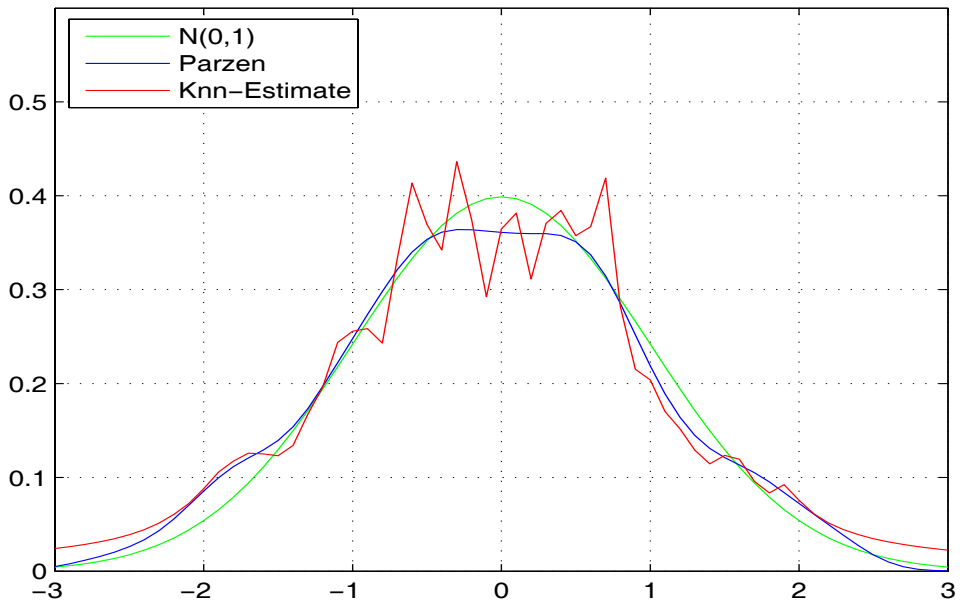

Figure 2: Parzen Window and Knn Density Estimation

.

Tree Classifiers

ω_1 : (2,3), (2,2), (3,2), (3,1).

ω_2 : (3,3), (4,3), (5,3), (4,1), (4,0).

```
V=[1,5, 0 4];
plot([2 2 3 3],[3 2 2 1],'rx',[3 4 5 4 4],[3 3 3 1 0],'b^'); axis(V); grid
```

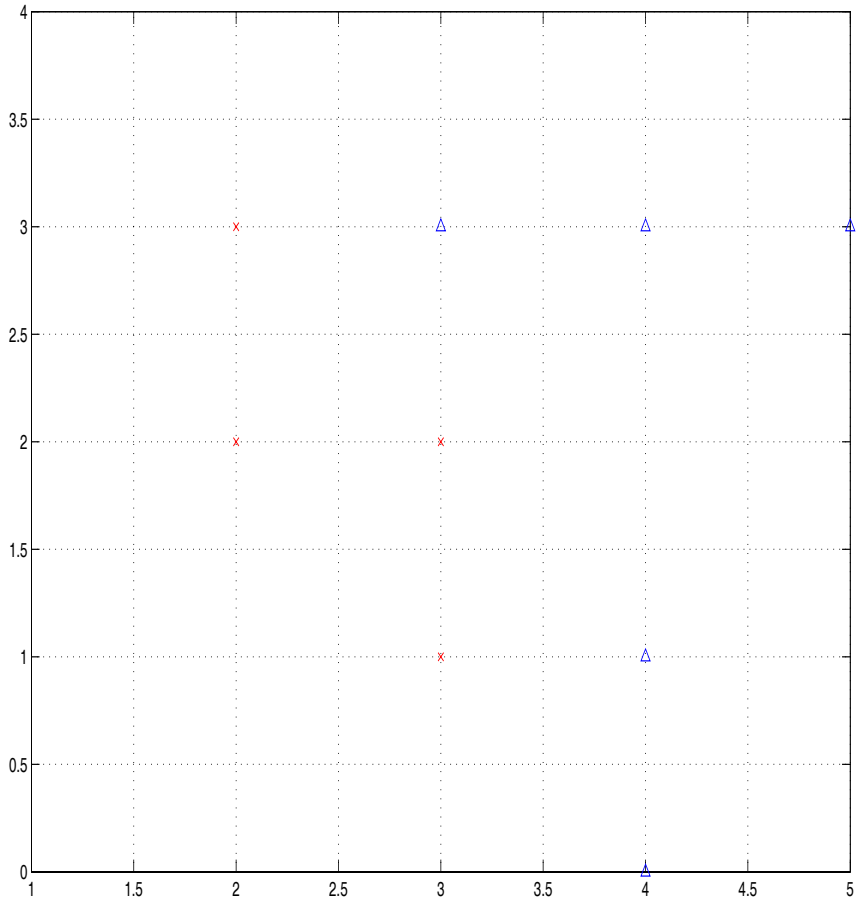

Figure 3: Training patterns for tree classifiers

.