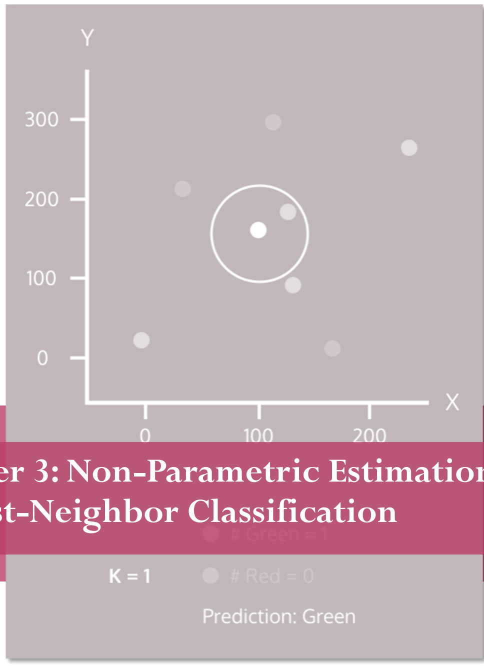
PATTERN RECOGNITION

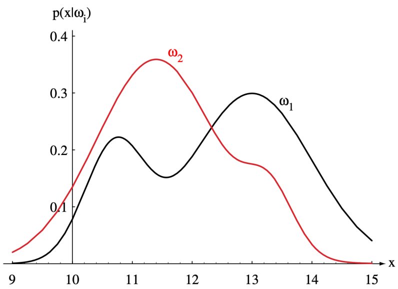


Chapter 3: Non-Parametric Estimation & Nearest-Neighbor Classification

Ch.3 Content

1. Introduction

- **Problems**
  - In most pattern recognition applications it is suspect that that the forms of the density function were known.
  - All Parametric densities are unimodal (have a single local maximum), whereas many practical problems involve multi-modal densities.



- **Nonparametric procedures**
  - Estimating the density functions $p(x)$.
  - It can be used with arbitrary distributions and without the assumption that the forms of the underlying densities are known.

- Nonparametric estimation methods:
  - **Histogram Density Estimation**
  - **Parzen Windows**
  - **kn–Nearest-Neighbor Estimation**

Ch.3 Content

1. Introduction
2. **Histogram Density Estimation**
3. Parzen Windows
4. kn–Nearest-Neighbor Estimation
5. The Nearest-Neighbor Rule

2. Density Estimation

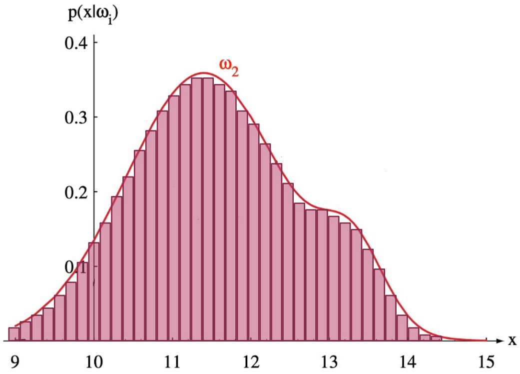
- **Basic idea**

A vector x: $x = (x_1, \dots, x_d)^T$

① The region falled x in is partitioned into a sequence of small regions R_i.

② Count the number k_i of samples falling in each small region R_i.

③ Compute the probality in region R_i as the density estimate at x.



- Probability that a vector x will fall in region R is:

$$P_R = \int_R p(x')dx' \qquad (1)$$

P_R —— *a smoothed (or averaged) version of the density function* $p(x)$;
R —— *the falling region*;
x —— *the vector*;

2. Density Estimation

- **Basic idea**
  - Probability that k of these n fall in R:

$$P_k = \binom{n}{k} {P_R}^k (1 - P_R)^{n-k} \qquad (2)$$

n —— *sample of size*;
k —— k *points fall in* R;

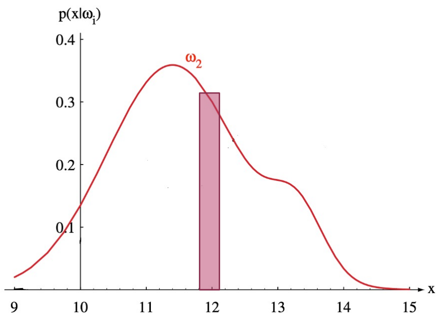
- Expected value for k:

$$E(k) = nP_R = k \qquad (3)$$

k/n —— *a good estimate for the probability* P_R *and for the density function* p.

$$P_R = \int_R p(x')dx' \approx p(x)V \qquad (4)$$

V —— *the volume enclosed by region* R;



—$\mu(\mathfrak{R})$: *a hypervolume in the Euclidean space* $\mathfrak{R}^n$

When that the region $\mathfrak{R}$ is so small, p is constant within it.

$$\int_{\mathfrak{R}} p(x')dx' \approx p(x) \int_{\mathfrak{R}} dx'$$

$$= p(x) \int_{\mathfrak{R}} 1_R(x)dx'$$

$$= p(x)V(\mathfrak{R})$$

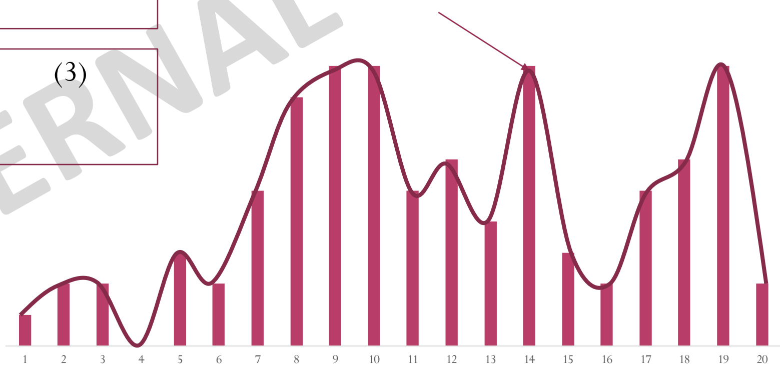
2. Density Estimation

- **Basic idea:**

Combining equation (1) , (3) and (4) yields:

$$\left\{ \begin{array}{ll} \boxed{\begin{array}{ll} P_R = \int_R p(x')dx' \cong p(x)V & (1,4) \\ \boldsymbol{\hat{p}(x) = \frac{P_R}{V}} & \end{array}} \\ \boxed{\begin{array}{ll} E(k) = nP_R = k & (3) \\ \boldsymbol{\hat{p}_R = \frac{k}{n}} & \end{array}} \end{array} \right.$$

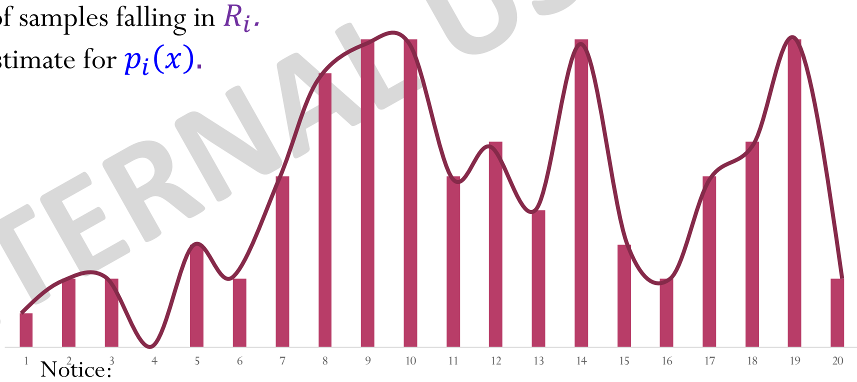
$$\boxed{\hat{p}(x) \cong \frac{k/n}{V}}$$



2. Density Estimation

- To estimate the density at x, we form a sequence of regions
  - A sequence of regions $R_1, R_2, \dots, R_N$
  - Let V_i be the volume of R_i,
    — k_i be the number of samples falling in R_i.
    — $\hat{p}_i(x)$ be the i^{th} estimate for $p_i(x)$.

$$\boxed{\hat{p}_i(x) \cong \frac{k_i/n}{V_i}}$$



Notice:

— *The volume $v \longrightarrow 0$ anyway if we want to use this estimation.*

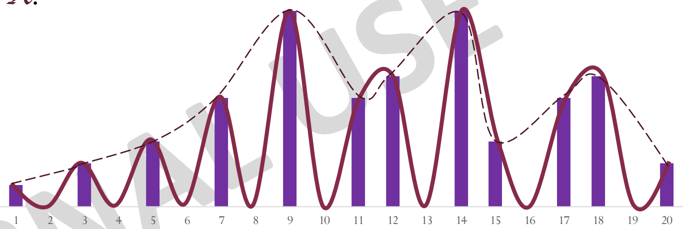
— *V cannot be allowed to become small since the number of samples is always limited.*

2. Density Estimation

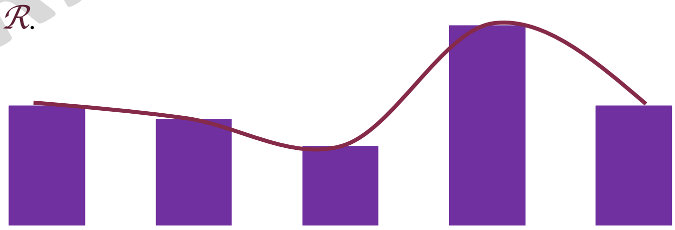
- **Uninteresting case**:

① There is no sample included in $\mathcal{R}$.

$$\lim_{\substack{v \longrightarrow 0 \\ k=0}} \hat{p}(x) = 0 \quad \text{if} \quad n = \text{fixed}$$



② There are unlimited samples in $\mathcal{R}$.



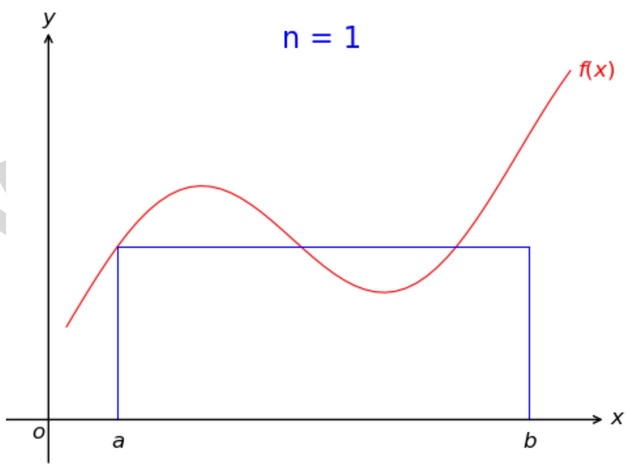
2. Density Estimation

- Three necessary conditions should apply if we want $\hat{p}_i(x)$ to converge to $p(x)$:

① $\lim_{n \longrightarrow \infty} V_i = 0$

② $\lim_{n \longrightarrow \infty} k_i = \infty$

③ $\lim_{n \longrightarrow \infty} k_i/n = 0$



2. Density Estimation

- There are two different ways of obtaining sequences of regions that satisfy these conditions:

$$\boxed{\hat{p}_i(x) \xrightarrow[n \to \infty]{} p(\mathbf{x})}$$

- **Parzen-window estimation method**
- **The k_n-nearest neighbor estimation method**

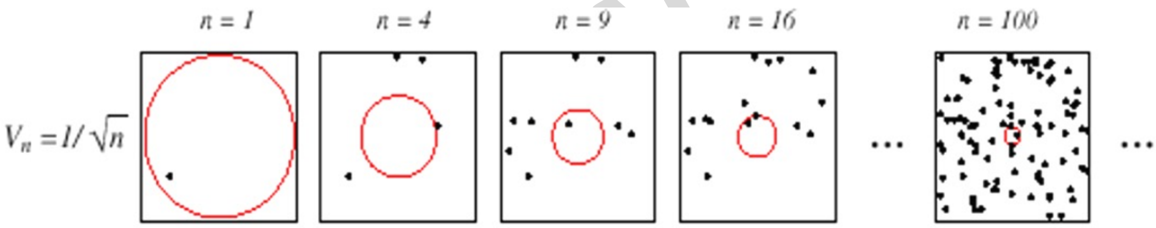
Ch.3 Content

3. Parzen Windows

Parzen-window estimation method

$\hat{p}_n(x) \xrightarrow[n\to\infty]{} \mathrm{p(x)}$

- **GOAL**: a solution for the problem of the unknown "best" window function.
- **METHOD**: shrink an initial region by specifying the volume V_n as some function of n, such as $V_n = 1/\sqrt{n}$.



3. Parzen Windows

① Parzen-window approach to estimate densities by temporarily assuming that the region $\Re^d$ is in a d-dimensional **hypercube** V_n. ($V = f(n)$, such as $h = 1/\sqrt{n}$)

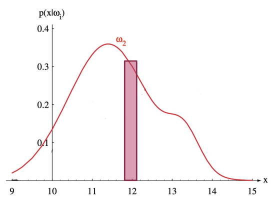
$$V = h^d \qquad (h\text{: the length of an edge of } \Re^d)$$

② Set the window function $\varphi(\mu)$: $\varphi(\mu) = \begin{cases} 1 & |\mu_j| \le \frac{1}{2}, j = 1, \dots, d \\ 0 & otherwise \end{cases}$, $\quad u = (\mu_1, \mu_2, \dots, \mu_d)^T$

The number k_n of samples in this hypercube V_n is:

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{x - x_i}{h}\right)$$

— Dataset: $x_1, \dots x_i \dots, x_n$.
— $\varphi((x - x_i)/h)$ *is equal to unity if* x_i *falls within the hypercube of volumn* V *centered at* x *and equal to zero otherwise.*



Kernel function:

$$K(x, x_i) = \frac{1}{V}\varphi\left(\frac{x - x_i}{h}\right)$$

3. Parzen Windows

③ We obtain the estimate:

$\hat{p}(x) \approx (k_n/n)/V$

$k_n = \sum_{i=1}^{n} \varphi\left(\frac{x - x_i}{h}\right)$ → $$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V}\varphi\left(\frac{x - x_i}{h}\right)$$

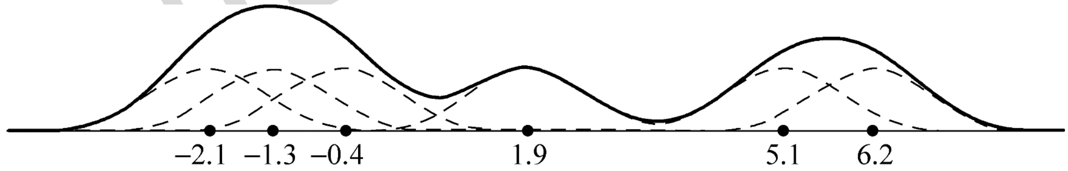
— $\hat{p}(x)$ *estimates* $p(x)$ *as an average of functions of* x *and the samples* $(x_i, i = 1, \dots, n)$.
— *These functions* φ *can be general.*

$K(x, x_i) = \frac{1}{V}\varphi\left(\frac{x - x_i}{h}\right)$ → $$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} K(x, x_i)$$

$K(x, x_i) \geq 0, \quad \int K(x, x_i)\, dx = 1$

Example:

Gaussian kernel function



3. Parzen Windows

- **Kernel function**
  - Hypercube kernel function:

$$k(x, x_i) = \begin{cases} \frac{1}{h^d} & \left|x^j - x_i^j\right| \leq \frac{h}{2}, \; j = 1, \dots, d \\ 0 & otherwise \end{cases}$$

  - Gaussian kernel function:

$$k(x, x_i) = \frac{1}{\sqrt{(2\pi)^d \rho^{2d} |Q|}} \exp\left[-\frac{1}{2}\frac{(x - x_i)^T Q^{-1}(x - x_i)}{\rho^2}\right]$$

  - Hypersphere kernel function:

$$k(x, x_i) = \begin{cases} V^{-1} & \|x - x_i\| \leq \rho \\ 0 & otherwise \end{cases}$$

3. Parzen Windows

- **Case:** $p(x) \sim N(0,1)$

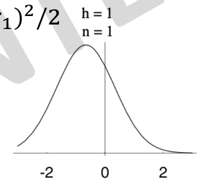
Let $\begin{cases} \varphi(\mu) = \frac{1}{\sqrt{2\pi}} e^{-\mu^2/2} \\ h_n = h_1/\sqrt{n} \end{cases}$, thus: $\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h_n} \varphi\left(\frac{x - x_i}{h_n}\right)$

——*an average of normal densities centered at the samples* x_i.

—— *the contributions of the individual samples are clearly observable.*

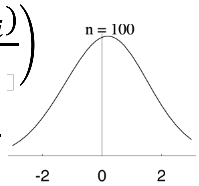
- For $n = 1$ and $h_1 = 1$

$$\hat{p}_1(x) = \varphi(x - x_1)$$
$$= \frac{1}{\sqrt{2\pi}} e^{-(x-x_1)^2/2}$$
$$\hat{p}_1(x) \sim N(x_1, 1)$$



- For $n = 100$ and $h_1 = 1$

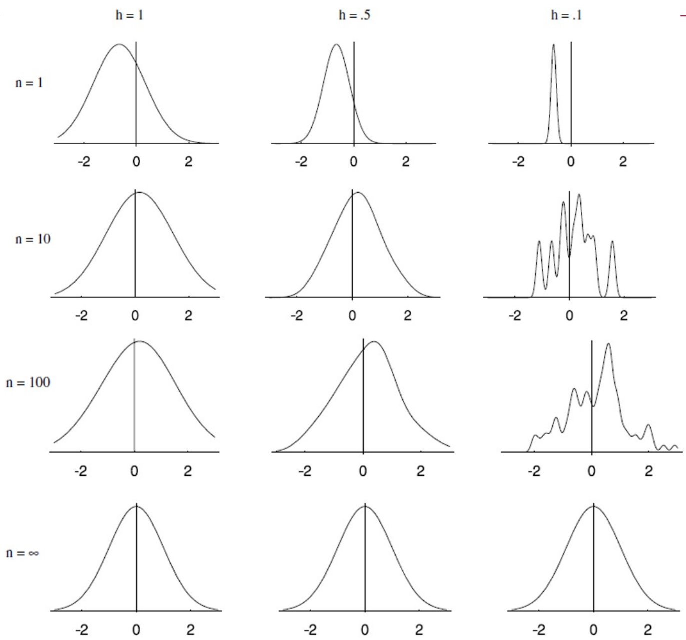
$$\hat{p}_{100}(x) = \frac{1}{100}\sum_{i=1}^{100} \frac{1}{0.1} \cdot \varphi\left(\frac{(x - x_i)}{0.1}\right)$$
$$= \frac{1}{100}\sum_{i=1}^{100} \frac{1}{0.1\sqrt{2\pi}} e^{-\frac{(x-x_i)^2}{2\times 0.1^2}}$$



3. Parzen Windows

Example:

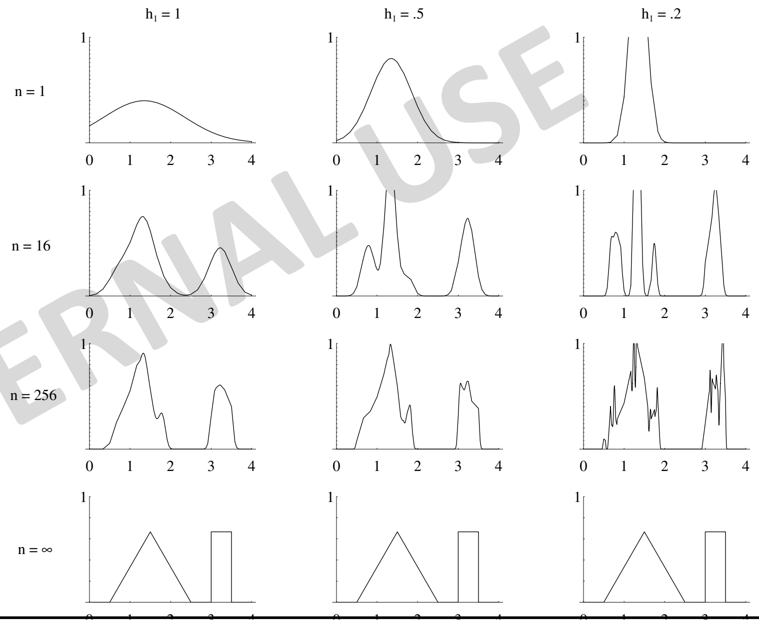
Case $p(x) \sim N(0,1)$



3. Parzen Windows

Example:

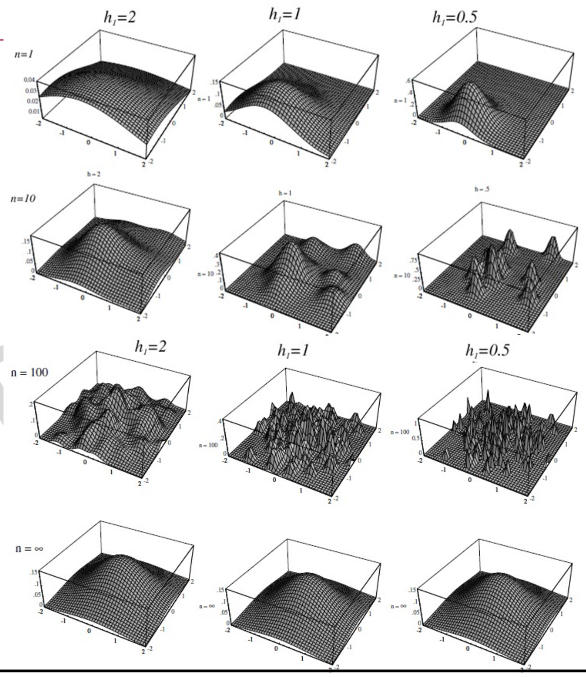
$p(x) = \lambda_1 \cdot U(a,b) + \lambda_2 \cdot T(c,d)$



3. Parzen Windows

Example:

Normal distribution in 2D

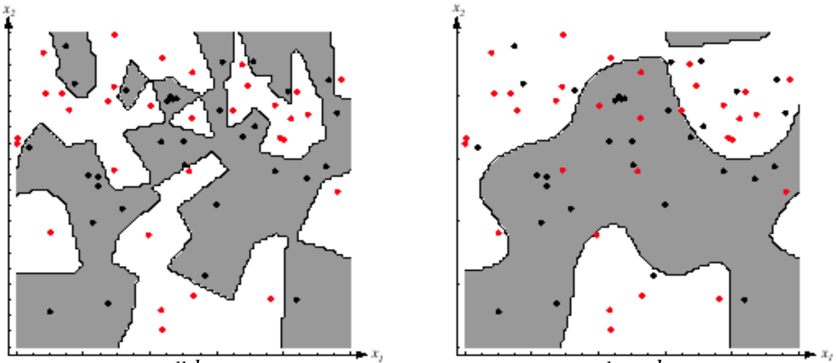


3. Parzen Windows

- **Classifiers based on Parzen-window estimation**
  - Estimate the densities for each category and classify a test point by the label corresponding to the maximum posterior.
  - The decision region for a Parzen-window classifier depends upon the choice of window function.

Example:

Classification example in 2D Parzen-window



small h large h

——*A small h leads to boundaries that are complicated than for large h on same data set.*

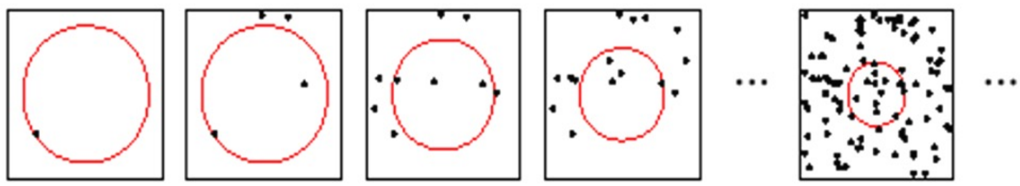
Ch.3 Content

4. k_n - Nearest neighbor estimation

$\hat{p}_n(x) \xrightarrow[n \to \infty]{} p(x)$

- k_n - **Nearest neighbor estimation**
  - **GOAL**: a solution for the problem of the unknown "best" window function.
  - **METHOD**: let the cell volume be a function of the training data number.
    - Specify k_n as some function of n, such as $k_n = \sqrt{n}$.
    - Grow the volume V_n (a cell V_n about x) until it encloses k_n neighbor-samples of x ($k_n = f(n)$, such as $k_n = k_1\sqrt{n}$).
    - k_n are called the k_n nearest-neighbors of x .

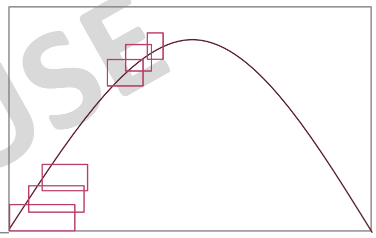
$k_n = \sqrt{n}$



4. k_n - Nearest neighbor estimation

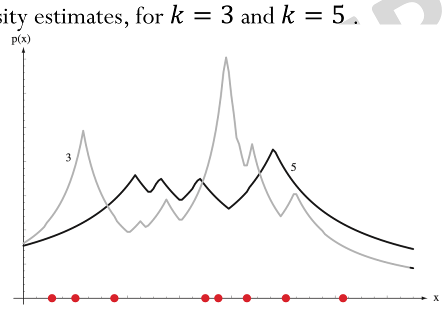
$\hat{p}(x) \cong (k_n/n)/V_n$

- 2 possibilities can occur:
  - Density is **high** near x; therefore the cell will be **small** which provides a good resolution.
  - Density is **low**; therefore the cell will grow **large** and stop until higher density regions are reached.

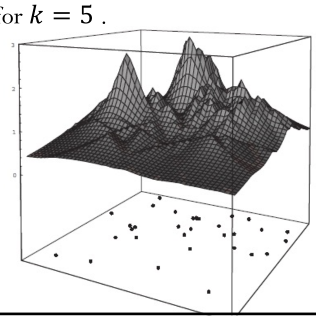


Example:

Eight points in one-dimension and the k -nearest-neighbor density estimates, for $k = 3$ and $k = 5$.

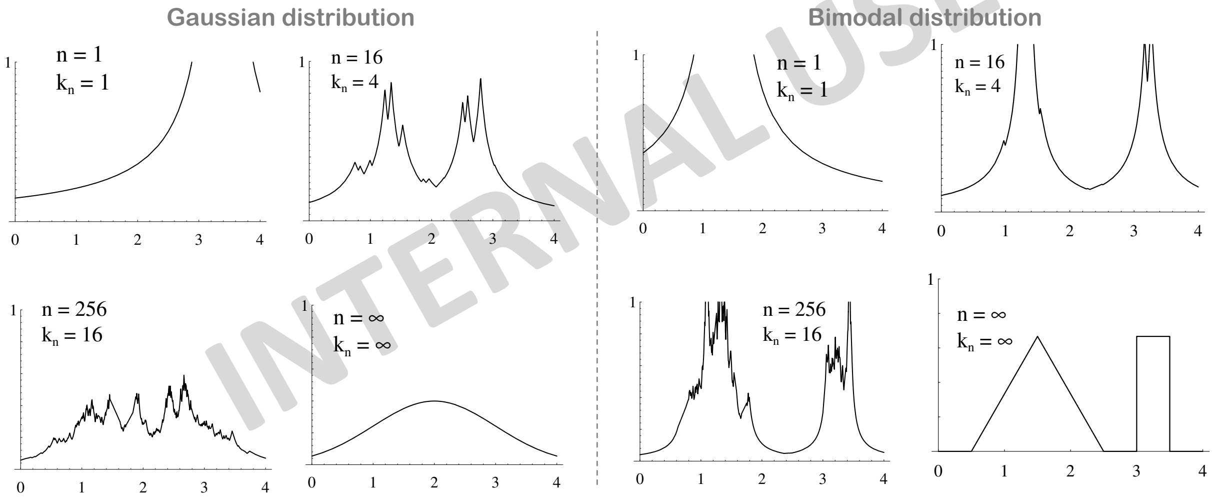


The k -nearest-neighbor estimate of a two-dimensional density for $k = 5$.



4. k_n - Nearest neighbor estimation

Example: $\begin{cases} k_1 = 1 \\ k_n = k_1\sqrt{n} = \sqrt{n} \end{cases}$



4. k_n - Nearest neighbor estimation

- **Estimation of a posterior probability $P(\omega_i|x)$**
  - **GOAL:** estimate $P(\omega_i|x)$ from a set of n labeled samples, $\omega_i \in \{\omega_1, \omega_2, \ldots, \omega_c\}$.
  - **METHOD:**
    - Let's place a cell of volume V around x and capture k samples.
    - k_i samples amongst k turned out to be labeled ω_i then:

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

An estimate for $P_n(\omega_i|x)$ is:

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

— k_i/k is the fraction of the samples within the cell that are labeled ω_i.

— For minimum error rate, the most frequently represented category within the cell is selected.

— If k is large and the cell sufficiently small, the performance will approach the best possible.

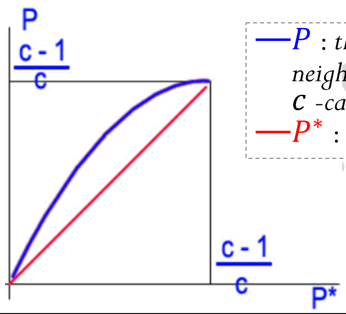
Ch.3 Content

5. The nearest –neighbor rule

- **The nearest –neighbor rule**
  - **GIVEN**: $\mathcal{D}_c = \{x_1, x_2, \dots x_c\}$, c labeled prototypes, $x_1 \in \omega_1, \dots, x_c \in \omega_c$; a test point x;
  - **METHOD**: Let $x_i \in \mathcal{D}_c$ 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_i.

$$g_i(x) = d_i(x, x_i), i = 1, \dots, c \longrightarrow j = \underset{i}{argmin}\, g_i(x) \longrightarrow x \in \omega_j$$

- **Error Bounds**: $P^* \le P \le P^*(2 - \frac{c}{c-1} P^*)$



—P : the nearest-neighbor error rate in a c -category problem
—P^* : the Bayes error.

—The nearest-neighbor rule leads to an error rate greater than the minimum possible (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 (it can be demonstrated!)
—If $n \to \infty$, it is always possible to find x_i sufficiently close so that: $P(\omega|x_i) \cong P(\omega_i|x)$

5. The nearest –neighbor rule

- **The nearest –neighbor rule**

Exercise： Try to use the nearest neighbor rule to label $x=(0.68,0.60)$.

Prototypes	Labels
(0.50, 0.30)	ω3
(0.70, 0.65)	ω5

$x=(0.68,0.60)$

$x_1=(0.50,0.30),\ x_1\in\omega_3$

$x_2=(0.70,0.65),\ x_2\in\omega_5$

$d_1=|x-x_1|=\sqrt{(0.68-0.5)^2+(0.6-0.3)^2}\approx 0.35$

$d_2=|x-x_2|=\sqrt{(0.68-0.7)^2+(0.6-0.65)^2}\approx 0.05$

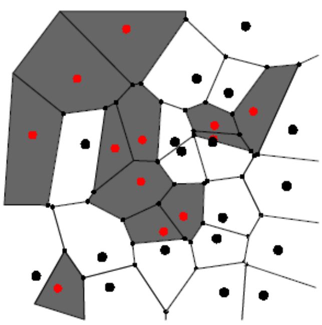
x_2 is the nearest neighbor of x, $x\in\omega_5$

Decision: ω_5 **is the label assigned to** x

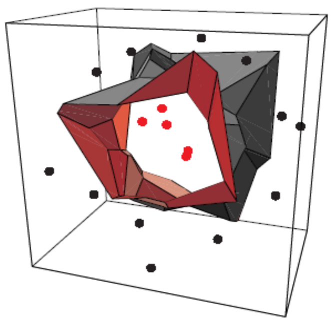
5. The nearest –neighbor rule

- **The nearest –neighbor rule**——Voronoi tesselation
  - Partition the feature space into cells consisting of all points closer to a given training point x than to any other training points.
  - All points in such a cell are thus labelled by the category— a so-called *Voronoi* of the space.

Example:



Voronoi cells in two dimensions



Voronoi cells in three dimensions

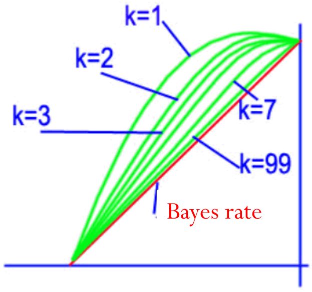
5. The nearest –neighbor rule

- **The k – nearest-neighbor rule**
  - **GOAL**: Classify x by assigning it the label most frequently represented among the k-nearest samples and use a voting scheme.
  - **METHOD**:

$$g_i(x) = k_i \longrightarrow j = \underset{i}{argmax}\, g_i(x), i = 1, \dots, c$$

—k_i: *the number of neighbor samples with* i *label*

- **Error Bounds**:



—P^*: *Bayes rate*

—$C_k(P^*)$: *the error-rate for the* k *-nearest-neighbor rule for a two-category problem*.

—$k = 1 \Rightarrow$ *the nearest neighbor rule*

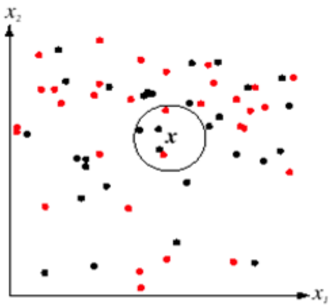
—$k = \infty \Rightarrow$ *Bayes decision*

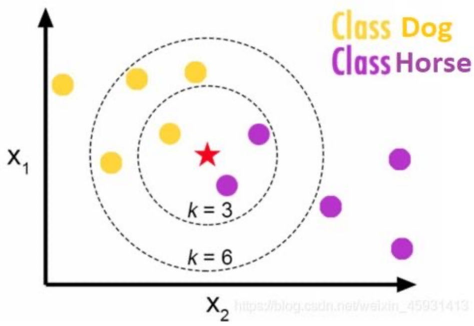
5. The nearest –neighbor rule

- **The k – nearest-neighbor rule**

Example:

In this $k = 5$ case, the test point x ?





x would be labelled the category of the black points.

The test sample would be labelled Class Horse.

5. The nearest –neighbor rule

- The k – nearest-neighbor rule

Exercise: $k = 3$ (odd value) and $x = (0.10, 0.25)$ are given. Try to use the k-nearest neighbor rule to label x.

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

$x_1 = (0.15, 0.35), x_1 \in \omega_1$; $x_2 = (0.10, 0.28), x_2 \in \omega_2$

$x_3 = (0.15, 0.35), x_3 \in \omega_1$; $x_4 = (0.15, 0.35), x_4 \in \omega_2$

$$d_1 = \sqrt{(0.1-0.15)^2 + (0.25-0.35)^2} \approx 0.11$$

$$d_2 = \sqrt{(0.1-0.1)^2 + (0.25-0.28)^2} \approx 0.03$$

$$d_3 = \sqrt{(0.1-0.09)^2 + (0.25-0.3)^2} \approx 0.051$$

$$d_4 = \sqrt{(0.1-0.12)^2 + (0.25-0.2)^2} \approx 0.054$$

Closest vectors to x with their labels are:

$x_2 \in \omega_2; x_3 \in \omega_1; x_4 \in \omega_2$

One voting scheme assigns the label ω_2 to x.

5. The nearest –neighbor rule

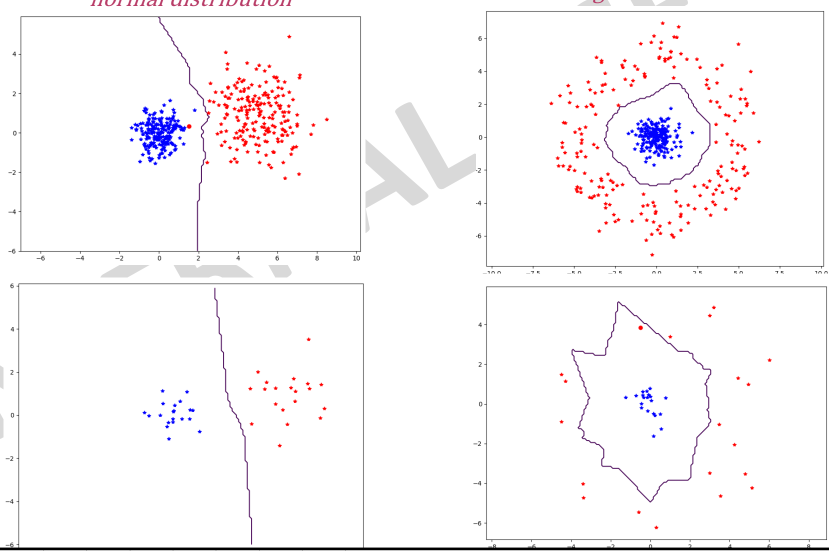
- **The k – nearest-neighbor rule**

Example:

normal distribution *ring distribution*

$n = 200, k = 3$

$n = 20, k = 3$



5. The nearest –neighbor rule

- **The k – nearest-neighbor rule**

Example

Iris Plants Database

- Dataset:
  - Training set: 0.8(120)
  - Test set: 0.2(30)
- Feature Extraction :
  - sepal length, sepal width, petal length, petal width
- Classifier: knn

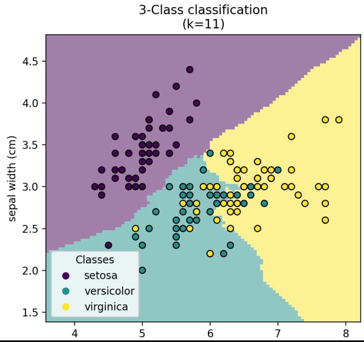
Accuracy	kn=1	kn=3	kn=5	kn=7
Training	0.967	0.958	0.95	0.942
Test	0.967	0.9	0.9	0.833

- **Results:**
  - Precision on training dataset: 0.967
  - Precision on test dataset: 0.967

	precision	recall	f1-score	support
Setosa	1	1	1	11
Versicolour	1	0.91	0.95	11
Virginica	0.89	1	0.94	8

Confusion-Matrix

	Setosa	Versicolour	Virginica
Setosa	11	0	0
Versicolour	0	10	1
Virginica	0	0	8

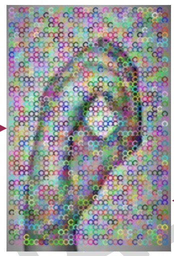


5. The nearest –neighbor rule

- **The k – nearest-neighbor rule**

Example:

Hand



Data: 3773
— Training set: 2053
— Test set: 1720
Feature Extraction: SIFT
Classifier: knn, $k = 1$

- *Accuracy*: 99.65%
- *Confusion Matrix*:

Confusion Matrix:

'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'
167	0	0	0	0	0	0	0	0	0
0	171	0	0	0	0	0	0	0	0
0	1	171	0	0	0	0	0	0	0
0	0	0	172	0	0	0	0	0	0
0	0	1	0	177	0	0	1	0	0
0	0	0	0	0	169	0	0	0	0
0	0	1	0	0	0	173	0	0	0
0	0	0	0	0	1	0	171	1	0
0	0	0	0	0	0	0	0	174	0
0	0	0	0	0	0	0	0	0	169

Prerequisite Knowledge

- **Prerequisite Knowledge in Ch.4**
  - Calculate the extremum by Lagrange function.
  - Inverse Matrix

Chapter 3

END