8 Understand Pattern Recognition & Decision Theory

Outline

- Understand what the Pattern recognition is
- Understand the process of pattern recognition
- Understand how the decision theory is developed from an intuitive process that everyone can do to a mathematical abstract theory
- The optimal decision rule: MAP and Bayesian decision rules
- Evaluation a pattern recognition system: Recognition accuracy or error rate

8 Understand Pattern Recognition & Decision Theory

- What is on earth the Pattern Recognition?
- Pattern recognition is to perceive a pattern, extract the relevant information, understand the content of the information and make decision automatically by machine or computer.
- Example 1: Is this apple or pear?

After perceiving the images, you may decide image 1 is pear and 2 is apple with 100% certainty (probability 1), based on color/shape?

- Example 2: Is this apple or pear?


After perceiving the images, you must first extract the right part from image and then make decision.

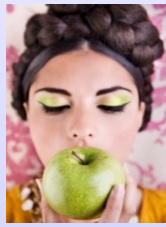
- Example 3: Is this apple or pear?

After perceiving the images, you may figure out:

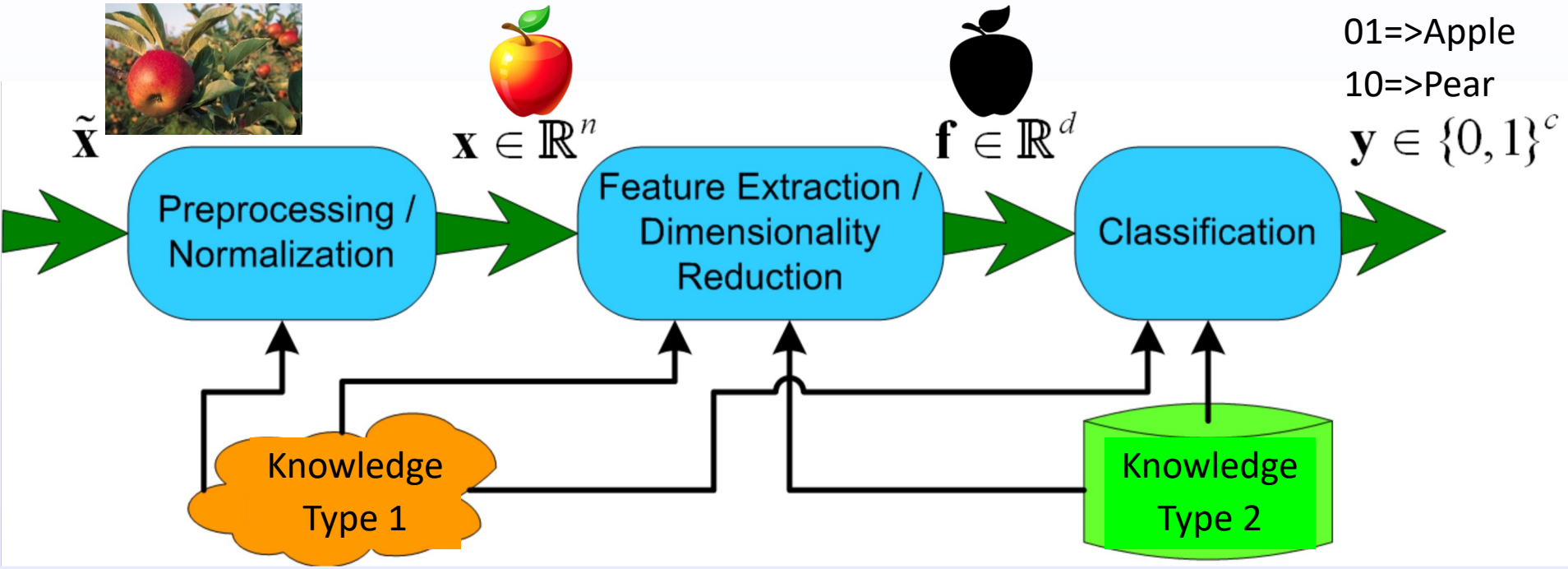
the probability of apple is 0.9, 0.4, 0.6 and 0.7.

So the probability of pear is 0.1, 0.6, 0.4 and 0.3. How do you make decision?



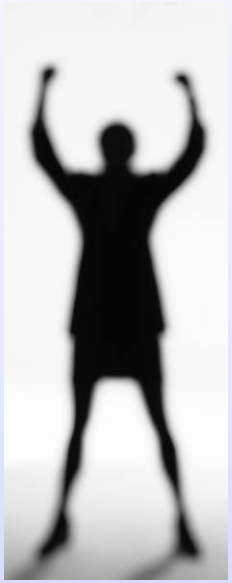
8 Understand Pattern Recognition & Decision Theory



- A pattern recognition system diagram
- Pattern recognition, especially image recognition, is very easy for human brain but very difficult for machine or computer.
- What tasks are very difficult for human brain but very easy for machine or computer?

8 Understand Pattern Recognition & Decision Theory

- How the Pattern Recognition works?
- A simple example 1: Somebody approaches you in the street who is totally disguised. You only know he/she is of height 1.72 meter. You need decide/judge if he/she a male or a female?
- What is your decision?
- if you are not silly you will judge it is a male .
- How do you work out that this person is a male?
- No female is of height of 1.72m? No.
- Or, are all male people of height 1.72m? No
- Or, are most male people are of 1.72m? No
- Is your decision definitely correct?



8 Understand Pattern Recognition & Decision Theory

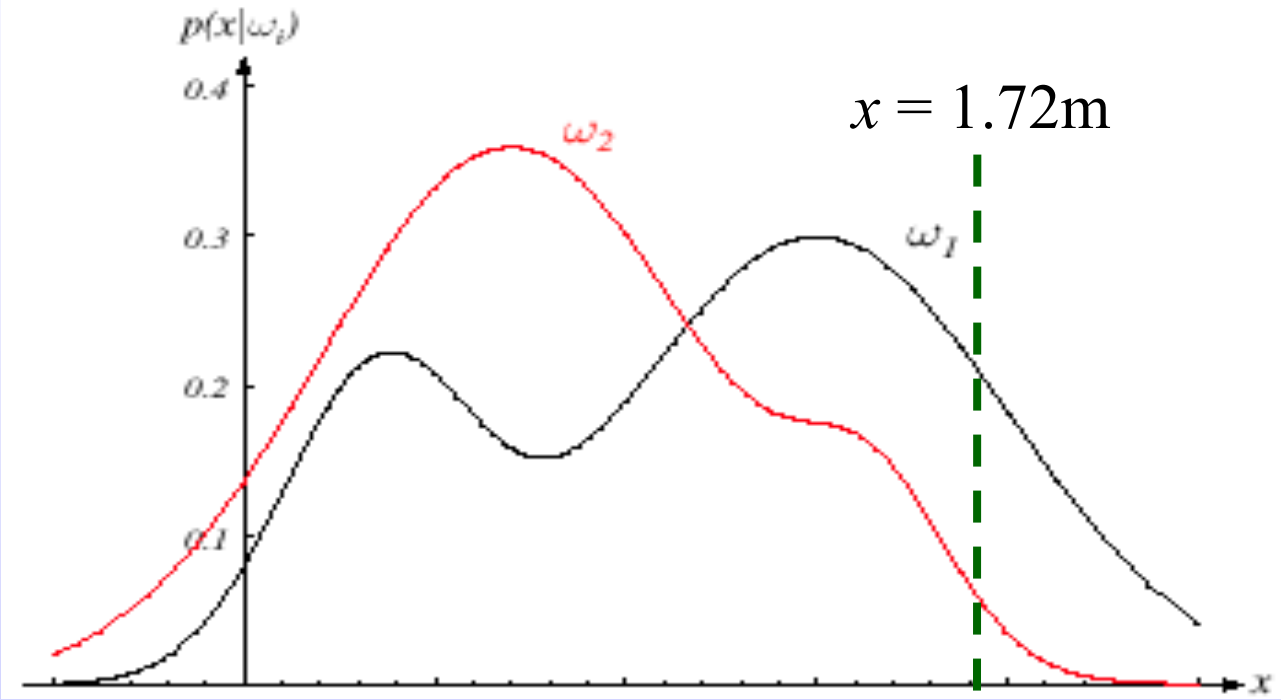
- You made a reasonable decision based on your knowledge that, for example, 30% of males are of height around 1.72m and 10% of females are of height around 1.72m.
- What is this knowledge based on which you make a reasonable decision?
- This knowledge in mathematical terms is that the probability of the height of male population is 0.3 around 1.72m, though it is less than 0.5 but the probability of the height of female population is 0.1 around 1.72m, which is much smaller than 0.3.
- Using mathematical formula, let x denote the height, ω_1 denote the class male and ω_2 denote the class female. This knowledge can be represented by $p(x|\omega_1)=0.3$ and $p(x|\omega_2)=0.1$ for $x=1.72\text{m}$.
- So you make a right decision of ω_1 because $p(x|\omega_1)>p(x|\omega_2)$

8 Understand Pattern Recognition & Decision Theory

- $p(x|\omega_1), p(x|\omega_2)$ are called class-conditional probability distribution function (discrete x) or density (continuous x). It is clear:

$$\sum_{x=0}^{3} p(x|\omega_1) = 1, \text{ and } \sum_{x=0}^{3} p(x|\omega_2) = 1 \text{ for discrete } x$$

$$\int_0^3 p(x|\omega_1)\, dx = 1, \text{ and } \int_0^3 p(x|\omega_2)\, dx = 1 \text{ for continuous } x$$



8 Understand Pattern Recognition & Decision Theory

- Is this definitely a wise decision? ω_1: because $p(x|\omega_1) > p(x|\omega_2)$
- Yes if you meet this person in the street. Why?
- Because in the world the human population has 50% males and 50% females. Mathematically, this means the prior probability:

$$p(\omega_1) = p(\omega_2) = 0.5, \qquad \text{with } p(\omega_1) + p(\omega_2) = 1$$

- Now, example 2: suppose you meet this person in the campus of a nurse school, where 95% of the people in the campus are female and 5% of the people are male. This case means

$$p(\omega_1) = 0.05, \qquad p(\omega_2) = 0.95, \qquad \text{so } p(\omega_1) \neq p(\omega_2)$$

- Will you in this case still judge this person is a male base on

$$\omega_1: \text{because } p(x|\omega_1) > p(x|\omega_2) \text{ at } x = 1.72\ ?$$

- Most likely not if you are smart. You may judge it is a female.
- Why?

8 Understand Pattern Recognition & Decision Theory

- Will you judge this person is a female just base on

$$p(\omega_1) = 0.05, \qquad p(\omega_2) = 0.95, \qquad \text{so } p(\omega_2) > p(\omega_1)\,?$$

- No. it means you judge it as a female for whoever you will seen.
- The right way for the decision is to compare the probability of male and the female in this scenario after you get the information of height 1.72m.
- Mathematically, this probability is denoted by $p(\omega_1|x)$ and $p(\omega_2|x)$. They are called a posterior probability. It means the probability of a class ω_i after knowing the data value x in a particular scenario.
- Therefore, the right decision is

$$\text{Decide } \omega_k\text{: if } p(\omega_k|x) > p(\omega_i|x), i \neq k$$

$$\text{or more general: } \omega_k = \arg\max_{\omega_i} [p(\omega_i|x)]$$

- not ω_1: if $p(x|\omega_1) > p(x|\omega_2)$ nor ω_1: if $p(\omega_1) > p(\omega_2)$ ×

8 Understand Pattern Recognition & Decision Theory

- As you decide ω_k, base on your observed value x, the probability that you make a correct decision is hence $p(\omega_k|x)$. Therefore, the probability that you make a wrong decision or error is:

$$p(e_k|x) = 1 - p(\omega_k|x)$$

- If your decision is ω_k, because $p(\omega_k|x)$ is maximum, consequently, the probability of the decision error will be minimum.
- Therefore, this decision rule is optimal in the sense of minimizing the probability of the decision error.
- This decision rule is called the maximum a posterior (MAP) rule,

$$\text{Decide: } \omega_k = \arg\max_{\omega_i} [p(\omega_i|x)] = \arg\min_{\omega_i} [p(e_i|x)]$$

- It minimizes the probability of the decision error.

8 Understand Pattern Recognition & Decision Theory

- More generally, If there are c possible classes, obviously

$$\sum_{i=1}^{c} p(\omega_i|x) = 1, \qquad \text{for } \forall x$$

- Therefore

$$p(e_k|x) = 1 - p(\omega_k|x) = \sum_{\substack{i=1 \\ i \neq k}}^{c} p(\omega_i|x)$$

- To compute $p(\omega_k|x)$, we just need a very basic formula in the probability theory. I trust all of you know and have good understand of the formula of computing joint probability

$$p(x, \omega_i) = p(x)p(\omega_i|x) = p(\omega_i)p(x|\omega_i)$$

- So we have: $p(\omega_i|x) = \dfrac{p(\omega_i)p(x|\omega_i)}{p(x)}$

$$p(A, B) = p(A)p(B|A) = p(B)p(A|B)$$

- In addition: $p(x) = \sum_{i=1}^{c} p(x|\omega_i)p(\omega_i)$

8 Understand Pattern Recognition & Decision Theory

- In the example 1, $p(x|\omega_1)$=0.3 and $p(x|\omega_2)$=0.1 for x=1.72m and $p(\omega_1) = p(\omega_2) = 0.5$
- So we have $p(x) = \sum_{i=1}^{c} p(x|\omega_i)p(\omega_i) = 0.3\times0.5 + 0.1\times0.5 = 0.2$
- and

$$p(\omega_1|x) = \frac{p(\omega_1)p(x|\omega_1)}{p(x)} = \frac{0.5\times0.3}{0.2} = 0.75$$

$$p(\omega_2|x) = \frac{p(\omega_2)p(x|\omega_2)}{p(x)} = \frac{0.5\times0.1}{0.2} = 0.25$$

- In the example 2, $p(\omega_1) = 0.05, p(\omega_2) = 0.95$
- So we have $p(x) = 0.3\times0.05 + 0.1\times0.95 = 0.11$

$$p(\omega_1|x) = \frac{0.05\times0.3}{0.11} = 0.136, \qquad p(\omega_2|x) = \frac{0.95\times0.1}{0.11} = 0.864$$

- We have mathematically proven that we should decide a person with height 1.72m in case 1 as man and in the case 2 as woman.

8 Understand Pattern Recognition & Decision Theory

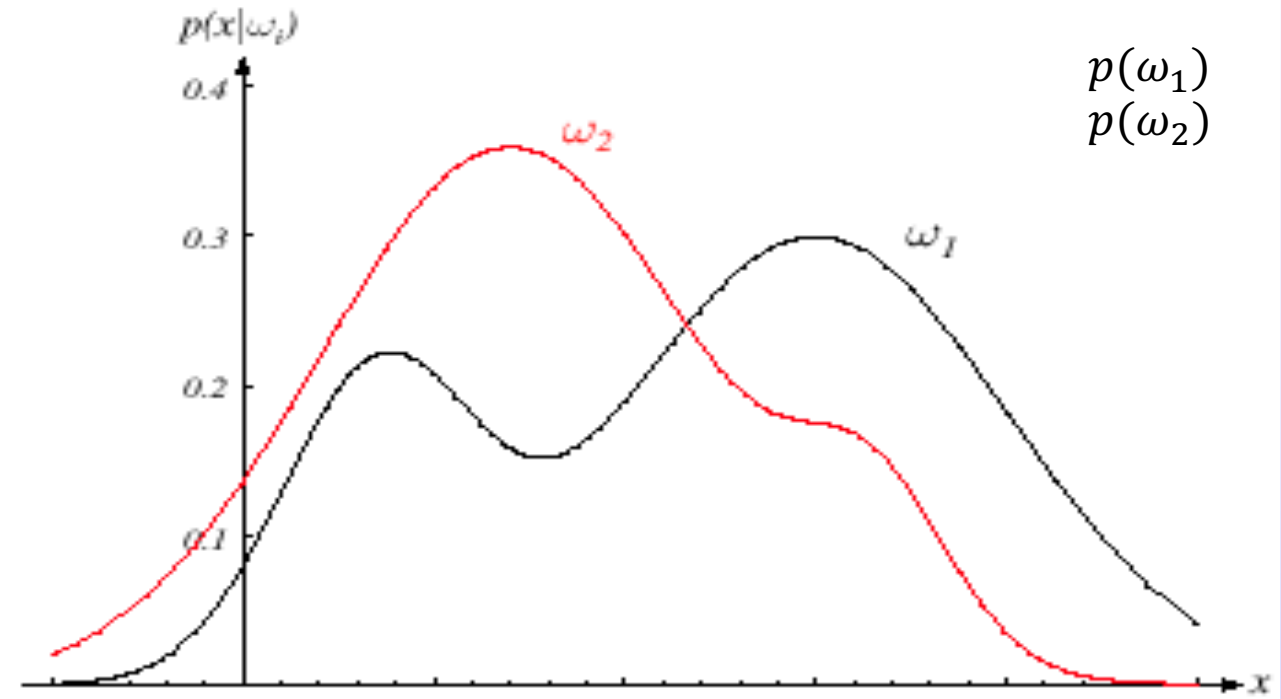
- Of cause, this decision could be correct or wrong. The probability of the wrong decision in case 1 is

$$p(e_1|x) = 1 - p(\omega_1|x) = 0.25$$

- And in case 2 is $p(e_2|x) = 1 - p(\omega_2|x) = 0.136$
- These two decisions are optimal as they minimize the decision error.
- What is the decision error if you make the opposite decision?
- The above are 2 simple pattern recognition examples with one perceiving variable and at a specific value, x=1.72m. Obviously, these two tasks can be very easily completed fully automatically by machine or computer. However, if you produce a machine to automatically do this task, you should design your machine to fully automatically recognize the person's gender for any value of x.

8 Understand Pattern Recognition & Decision Theory

- To fully automatically recognize a person's gender for all value of x, you need the prior probability $p(\omega_i)$ of all classes and the class conditional probability $p(x|\omega_i)$, of all possible value of x.



- The system will automatically classify any received value of x into male (class 1) or female (class 2).

8 Understand Pattern Recognition & Decision Theory

- The decision rule will be the same as before:

$$\text{Decide: } \omega_k = \arg\max_{\omega_i} [p(\omega_i|x)]$$

- But how good is the system? Or what is the performance of the system? Or how to evaluate your designed system?

- We have understand the formula to compute the error probability for a specific value of x.

$$p(e_k|x) = 1 - p(\omega_k|x) = \sum_{\substack{i=1 \\ i \neq k}}^{c} p(\omega_i|x)$$

- It is not difficult to understand that the performance of a pattern recognition system can be measured by the average of $p(e_k|x)$ over all possible value of x.

8 Understand Pattern Recognition & Decision Theory

- How to average of $p(e_k|x)$ over all possible value of x?

$$p(e)? = \frac{1}{n_x}\sum_{x=0}^{3} p(e_k|x) \text{ for discrete } x$$

$$p(e)? = \frac{1}{3}\int_0^3 p(e_k|x)dx, \qquad \text{for continuous } x$$

- The above is not a proper way because x is a random variable with different probability of occurrence at different x value.
- The right way for a random variable should be:

$$p(e) = \sum_{x=-\infty}^{\infty} p(e_k|x)\,p(x) \text{ for discrete } x$$

$$p(e) = \int_{-\infty}^{\infty} p(e_k|x)p(x)dx, \qquad \text{for continuous } x$$

8 Understand Pattern Recognition & Decision Theory

- Let's take the continuous x for further working:

$$p(e) = \int_{-\infty}^{\infty} p(e_k|x)p(x)dx = \int_{-\infty}^{\infty} [1 - p(\omega_k|x)]p(x)dx$$

$$= \int_{-\infty}^{\infty} [1 - \frac{p(\omega_k)p(x|\omega_k)}{p(x)}]p(x)dx = 1 - \int_{-\infty}^{\infty} p(\omega_k)p(x|\omega_k)dx$$

- Note that for different region of x, the pattern recognition system has different decision ω_k. Pattern recognition is to partition the whole space of x into c decision regions $\mathcal{R}_i$. Therefore,

$$p(e) = 1 - \sum_{i=1}^{c} \int_{\Re_i} p(\omega_i)p(x|\omega_i)dx = 1 - \sum_{i=1}^{c} p(\omega_i) \int_{\Re_i} p(x|\omega_i)dx$$

- Obviously, the probability of the correct decision is

$$p(correct) = 1 - p(e) = \sum_{i=1}^{c} p(\omega_i) \int_{\Re_i} p(x|\omega_i)dx$$

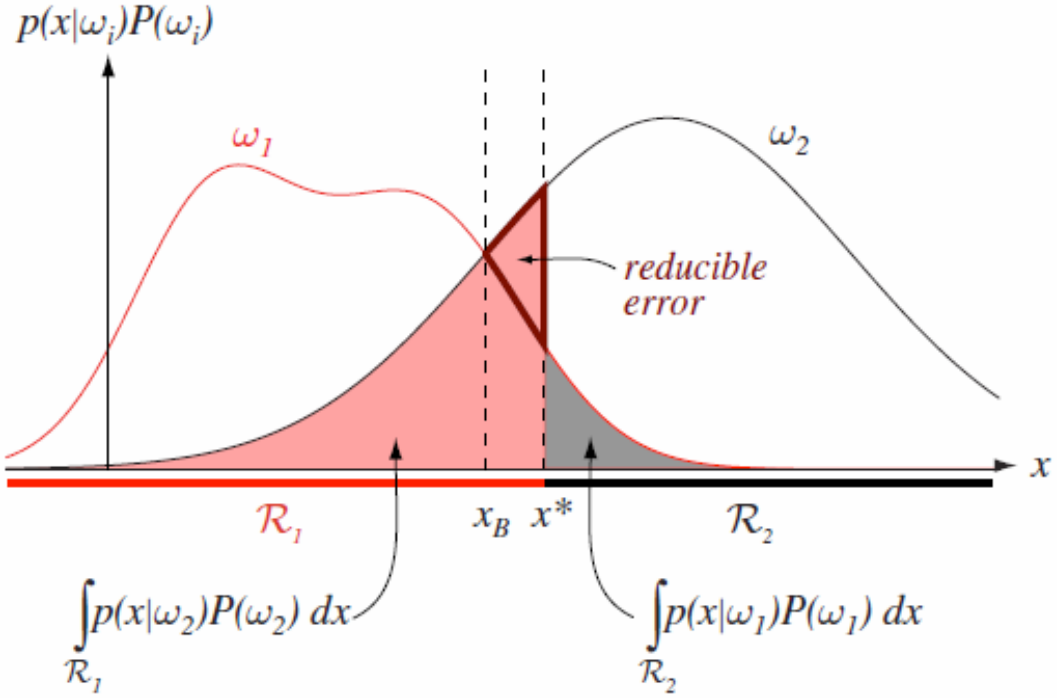


FIGURE 2.17. Components of the probability of error for equal priors and (nonoptimal) decision point x^*. The pink area corresponds to the probability of errors for deciding ω_1 when the state of nature is in fact ω_2; the gray area represents the converse, as given in Eq. 70. If the decision boundary is instead at the point of equal posterior probabilities, x_B, then this reducible error is eliminated and the total shaded area is the minimum possible; this is the Bayes decision and gives the Bayes error rate. From: Richard O.

- For the 2-class problem:

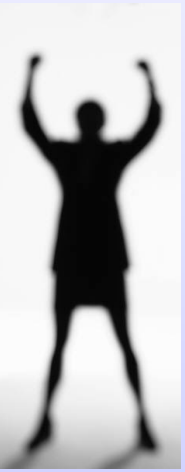
$$p(e) = 1 - p(\omega_1)\int_{\Re_1} p(x|\omega_1)dx - p(\omega_2)\int_{\Re_2} p(x|\omega_2)dx$$

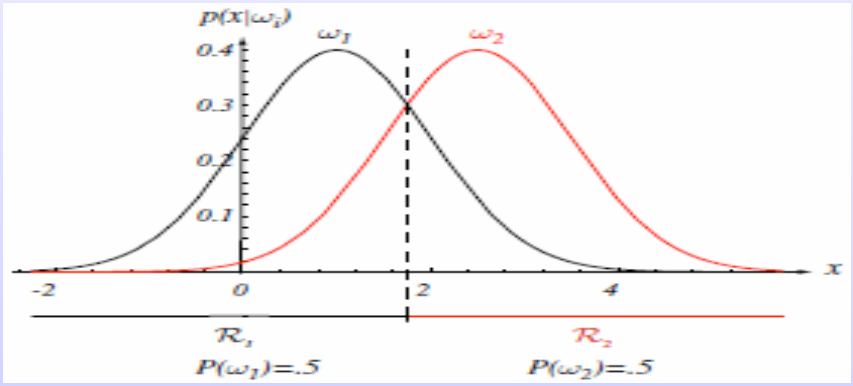
$$= p(\omega_1)\int_{\Re_2} p(x|\omega_1)dx + p(\omega_2)\int_{\Re_1} p(x|\omega_2)dx$$

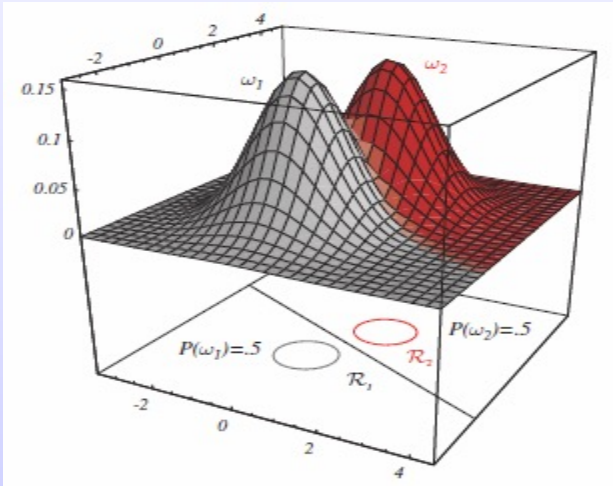
8 Understand Pattern Recognition & Decision Theory

- Now if we not only know the person's height x_1, but also the length of the hair x_2, it is not difficult to image that we will make better decision, i.e. have lower error probability. The recognition principle will be the same as before as long as now you use a vector $\mathbf{x} = [x_1, x_2]^T$

  to replace the scalar x before. The probability distribution is now two dimensional. A scalar threshold to separate the two classes in 1D case becomes a curve in the plane spanned by $\mathbf{x}$.

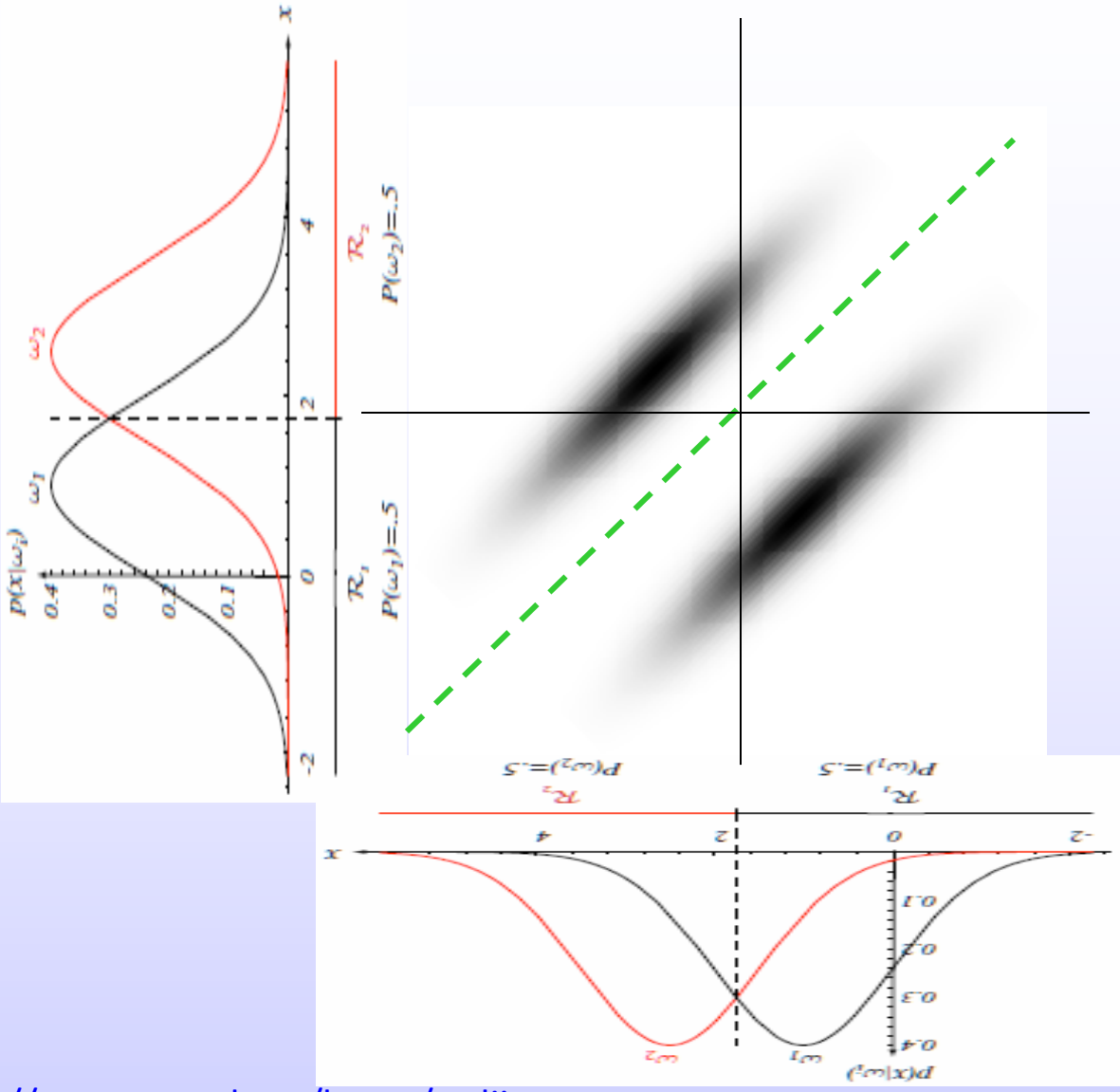






8 Understand Pattern Recognition & Decision Theory

- The increase of the information or data or feature dimension in general increases the probability of the correct decision or reduce the probability of the decision error.



8 Understand Pattern Recognition & Decision Theory

- Now I trust you have thoroughly understand the optimal decision rule. In fact, it is a very intuitive idea.
- However, to develop a simple idea into high techniques to solve complex problem, we need formulate the idea using neat, tidy and decent mathematics. Now let's formulate the simple and intuitive ideas again in another way.
- If you have to classify an object into one ω_k of the c classes ω_i before receiving any information of this particular object, how do you do?
- It is straightforward to choose the class with highest probability. Mathematically it is: $\text{Decide: } \omega_k = \arg\max_{\omega_i} [p(\omega_i)]$
- The probability of making mistake

$$p(e_k) = 1 - p(\omega_k) = \sum_{\substack{i=1 \\ i \neq k}}^{c} p(\omega_i)$$

is then minimum.

8 Understand Pattern Recognition & Decision Theory

- Now if you need classify an object into one ω_k of the c classes ω_i after receiving information (in a column vector $\mathbf{x}$) of this particular object, it is the same straightforward to choose the class with highest probability. Mathematically it is now:

$$\text{Decide: } \omega_k = \arg\max_{\omega_i} [p(\omega_i|\mathbf{x})]$$

- This decision rule is called the maximum a posterior (MAP) rule
- The probability of making mistake

$$p(e_k|\mathbf{x}) = 1 - p(\omega_k|\mathbf{x}) = \sum_{\substack{i=1 \\ i \neq k}}^{c} p(\omega_i|\mathbf{x})$$

  is then minimum.

- Allowing the use of more than one feature merely requires replacing the scalar x by the feature vector $\mathbf{x}$, where $\mathbf{x}$ is a point in a d-dimensional Euclidean space $\mathbb{R}^d$, called the feature space.

8 Understand Pattern Recognition & Decision Theory

- Note that $\because \sum_{i=1}^{c} p(\omega_i|\mathbf{x}) = 1 \quad \therefore p(\omega_k|\mathbf{x}) = 1 - \sum_{\substack{i=1 \\ i\neq k}}^{c} p(\omega_i|\mathbf{x})$
- Therefore, the decision rule can also be expressed as

$$\text{Decide: } \omega_k = \arg\min_{\omega_i} [\sum_{\substack{j=1 \\ j\neq i}}^{c} p(\omega_j|\mathbf{x})] = \arg\min_{\omega_i} [p(e_i|\mathbf{x})]$$

- It is therefore pretty clear that this decision rule minimizes the probability of the mistake or error. So it is also called the minimum error rate classifier.
- We see that the error is the sum of c-1 terms.

$$p(e_k|\mathbf{x}) = 1 - p(\omega_k|\mathbf{x}) = \sum_{\substack{i=1 \\ i\neq k}}^{c} p(\omega_i|\mathbf{x})$$

- It is the sum of the probabilities of all other classes.

8 Understand Pattern Recognition & Decision Theory

- In some application, wrongly classifying one class may not be at the same cost or same risk as wrongly classifying another class .
- Let's λ_{ki} denote the cost or risk of wrongly classifying the object of class ω_i into class ω_k. The cost or risk of the decision ω_k is then

$$R_k(\mathbf{x}) = \sum_{\substack{i=1 \\ i \neq k}}^{c} \lambda_{ki} p(\omega_i|\mathbf{x})$$

- Therefore the decision rule that minimizes the risk or cost becomes

$$\text{Decide: } \omega_k = \arg\min_{\omega_i} [R_i(\mathbf{x})] = \arg\min_{\omega_i} [\sum_{\substack{j=1 \\ j \neq i}}^{c} \lambda_{ij} p(\omega_j|\mathbf{x})]$$

- This is the famous Bayesian decision rule. Cost functions allow us to treat situations where some kinds of classification mistakes are more costly than others, although we often discuss the simplest case where all errors are equally costly.

8 Understand Pattern Recognition & Decision Theory

- In some applications, the correct decision for different class may have different cost$\lambda_{kk} \neq 0$. By including the cost of correct decision, the cost or risk of a decision is then

$$R_k(\mathbf{x}) = \sum_{i=1}^{c} \lambda_{ki} p(\omega_i|\mathbf{x})$$

- Therefore, the famous Bayesian decision rule is generalized as:

$$\begin{aligned}
\text{Decide } \omega_k &= \arg\min_{\omega_i} [R_i(\mathbf{x})] \\
&= \arg\min_{\omega_i} [\sum_{j=1}^{c} \lambda_{ij} p(\omega_j|\mathbf{x})] \\
&= \arg\min_{\omega_i} [\sum_{j=1}^{c} \lambda_{ij} p(\omega_j) p(\mathbf{x}|\omega_j) \\
&= \arg\min_{\omega_i} [\sum_{j=1} \lambda_{ij} p(\omega_j) p(\mathbf{x}|\omega_j)]
\end{aligned}$$

8 Understand Pattern Recognition & Decision Theory

$$R_k(\mathbf{x}) = \sum_{i=1}^{c} \lambda_{ki} p(\omega_i|\mathbf{x})$$

is the risk or cost for the decision ω_k at a specific value of $\mathbf{x}$, called the conditional risk.

- How good is a decision rule is evaluated by the average or overall cost or risk of a pattern recognition system

$$R = \int_{\Re_x} R_k(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \int_{\Re_x} \sum_{i=1}^{c} \lambda_{ki} p(\omega_i|\mathbf{x})\, p(\mathbf{x}) d\mathbf{x}$$

same as error

i: other class
k: considering class

$$= \int_{\Re_x} \sum_{i=1}^{c} \lambda_{ki} p(\omega_i) p(\mathbf{x}|\omega_i)\, d\mathbf{x} = \sum_{k=1}^{c} \int_{\Re_{xk}} \sum_{i=1}^{c} \lambda_{ki} p(\omega_i) p(\mathbf{x}|\omega_i)\, d\mathbf{x}$$

- This overall cost is minimized by the Bayesian decision rule. It delivers the best performance that can be achieved.

8 Understand Pattern Recognition & Decision Theory

- Two-category classification:
  - We have two possible classes : ω_1, ω_2.
  - We also have two actions:
    - α_1 corresponds to deciding that the true class is ω_1;
    - α_2 corresponds to deciding that the true class is ω_2.
  - Let $\lambda_{ij} = \lambda(\alpha_i|\omega_j)$, we have the conditional risks as

$$R_1(\mathbf{x}) = \lambda_{11}P(\omega_1|\mathbf{x}) + \lambda_{12}P(\omega_2|\mathbf{x})$$
$$R_2(\mathbf{x}) = \lambda_{21}P(\omega_1|\mathbf{x}) + \lambda_{22}P(\omega_2|\mathbf{x})$$

  - The Baves decision rule : decide that the true class is ω_1 if $R_1(\mathbf{x}) < R_2(\mathbf{x})$ and ω_2 otherwise.

8 Understand Pattern Recognition & Decision Theory

- From the decision rule, we obtain the following

$$(\lambda_{21} - \lambda_{11})P(\omega_1|\mathrm{x}) > (\lambda_{12} - \lambda_{22})P(\omega_2|\mathrm{x}).$$

- Normally $\lambda_{21} - \lambda_{11}$ and $\lambda_{12} - \lambda_{22}$ are positive because the loss is greater when making a mistake.
- We can also replace the posterior probability by the product of likelihood and prior , and drop the evidence term.
- Then we can write the Bayes decision rule as: Decide ω_1 if

$$\frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11})} \frac{P(\omega_2)}{P(\omega_1)}.$$

If the above likelihood ratio is greater than ratio of loss weighted priors, we take action α_1 (decide ω_1). Otherwise take action α_2 (decide ω_2)

This is the **Likelihood Ratio Test (LRT) .**

8 Understand Pattern Recognition & Decision Theory

- We can define the loss to be zero for correct decision and one for wrong decision for simplicity.

$$\lambda_{ij} = \begin{matrix} 0 & \text{if} & i = j \\ 1 & \text{if} & i \neq j \end{matrix}$$

- In a multi-category setting, we can write the loss in the matrix form, whose elemnets are λ_{ij},

$$\begin{bmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & 0 & 1 & \dots & 1 \\ \vdots & & \ddots & & \vdots \\ 1 & 1 & \dots & & 0 \end{bmatrix}$$

➢ With this 0/1 loss function, the Bayesian decision rule becomes MAP rule that minimizes the error rate.

8 Understand Pattern Recognition & Decision Theory

- Numerical Example 1:

In a study of two types of objects, it is found that the sizes of objects can be modeled as two Gaussian (normal) distributions. In general, there is about the same number of type A objects as there is for type B objects. For type A objects, its class conditional density is $\mathcal{N}(2,1)$ and for type B, its class conditional density is $\mathcal{N}(8,1)$. Find the decision rule that you can use to discriminate the two types of objects using the Likelihood Ratio Test if there is the same penalty for making all wrong decisions.

How would the decision rule be affected if there are twice more objects of type A than the objects of type B?

8 Understand Pattern Recognition & Decision Theory

Solution:

2 types of objects: $\{\omega_1$ for type A, ω_2 for type B$\}$; let size be x,

Type A: $p(x|\omega_1) \sim \mathcal{N}(2,1) \Rightarrow p(x|\omega_1) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2}(x-2)^2]$

Type B: $p(x|\omega_2) \sim \mathcal{N}(8,1) \Rightarrow p(x|\omega_2) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2}(x-8)^2]$

From the question, we know $P(\omega_2) = P(\omega_1)$.

Let penalty for making mistake be k, then $\lambda_{11} = \lambda_{22} = 0$, $\lambda_{12} = \lambda_{21} = k$.

Likelihood Ratio Test:

$$\frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{(\lambda_{12}-\lambda_{22})}{(\lambda_{21}-\lambda_{11})} \frac{P(\omega_2)}{P(\omega_1)} \Rightarrow \frac{\exp[-\frac{1}{2}(x-2)^2]}{\exp[-\frac{1}{2}(x-8)^2]} > 1$$

$$\begin{aligned} \text{take natural log, } -\frac{1}{2}[(x-2)^2 - (x-8)^2] &> 0 \\ \text{change sign, remove } \frac{1}{2}\text{, we get} (x-2)^2 - (x-8)^2 &< 0 \\ 12x - 60 &< 0 \\ \text{Therefore, } x &< 5 \end{aligned}$$

The rule: Decide ω_1 if $x < 5$, otherwise decide ω_2

8 Understand Pattern Recognition & Decision Theory

Solution:

If there are twice more objects of type A than objects of type B out there,

$$\frac{P(\omega_1)}{P(\omega_2)} = 2 \Rightarrow P(\omega_1) = 2P(\omega_2) \Rightarrow \frac{P(\omega_2)}{P(\omega_1)} = \frac{1}{2}$$

Following the previous solution, we now have
Likelihood Ratio Test:

$$\frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11})} \frac{P(\omega_2)}{P(\omega_1)} = \frac{1}{2} \Rightarrow \frac{\exp[-\frac{1}{2}(x-2)^2]}{\exp[-\frac{1}{2}(x-8)^2]} > \frac{1}{2}$$

take natural log, $-\frac{1}{2}[(x-2)^2 - (x-8)^2] > \ln\frac{1}{2}$

change sign, $\times 2$, we get $(x-2)^2 - (x-8)^2 < -2\ln\frac{1}{2}$

$$12x - 60 < -2\ln\frac{1}{2}$$

Therefore, $x < \frac{60 - 2\ln(1/2)}{12} = 5.16$

The rule: Decide ω_1 if $x < 5.16$, otherwise decide ω_2

The decision boundary is shifted towards class ω_2.

8 Understand Pattern Recognition & Decision Theory

- Numerical Example 2

A set of patterns is to be classified into 2 classes based on a scalar feature. We know that the conditional density functions of the feature for the pattern classes can be represented as two Gaussians, $\mathcal{N}(2,3)$ and $\mathcal{N}(4,1)$. The patterns from the two classes are equal likely to be seen. However, if a pattern from class 1 being wrongly classified into class 2, the cost associated is $\sqrt{3}$ while the cost for a pattern from class 2 being wrongly classified into class 1 is 1 . Using the Likelihood Ratio Test, find the decision rule that minimizes the probability of classification error.

8 Understand Pattern Recognition & Decision Theory

Solution:

We have 2 classes: $\{\omega_1$ for class 1, ω_2 for class 2$\}$; let the feature be x,

Class 1: $p(x|\omega_1) \sim \mathcal{N}(2,3) \Rightarrow p(x|\omega_1) = \frac{1}{\sqrt{2\pi}\sqrt{3}} \exp[-\frac{1}{2}\frac{(x-2)^2}{3}]$

Class 2: $p(x|\omega_2) \sim \mathcal{N}(4,1) \Rightarrow p(x|\omega_2) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2}(x-4)^2]$

From the question, we know $P(\omega_2) = P(\omega_1)$.

Also, $\lambda_{11} = \lambda_{22} = 0$, $\lambda_{12} = 1$ and $\lambda_{21} = \sqrt{3}$.

Likelihood Ratio Test:

$$\frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{(\lambda_{12}-\lambda_{22})}{(\lambda_{21}-\lambda_{11})}\frac{P(\omega_2)}{P(\omega_1)} \Rightarrow \frac{\frac{1}{\sqrt{2\pi}\sqrt{3}} \exp[-\frac{1}{2}\frac{(x-2)^2}{3}]}{\frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2}(x-4)^2]} > \frac{1}{\sqrt{3}}$$

Canceling out $\frac{1}{\sqrt{2\pi}}$, $\times\sqrt{3}$, we have : $\frac{\exp[-\frac{1}{2}\frac{(x-2)^2}{3}]}{\exp[-\frac{1}{2}(x-4)^2]} > 1$

take natural log, $-\frac{1}{2}\left[\frac{(x-2)^2}{3} - (x-4)^2\right] > 0$

$\times 6$, we get : $-(x-2)^2 + 3(x-4)^2 > 0$

expand and divide by 2 : $x^2 - 10x + 22 > 0$

8 Understand Pattern Recognition & Decision Theory

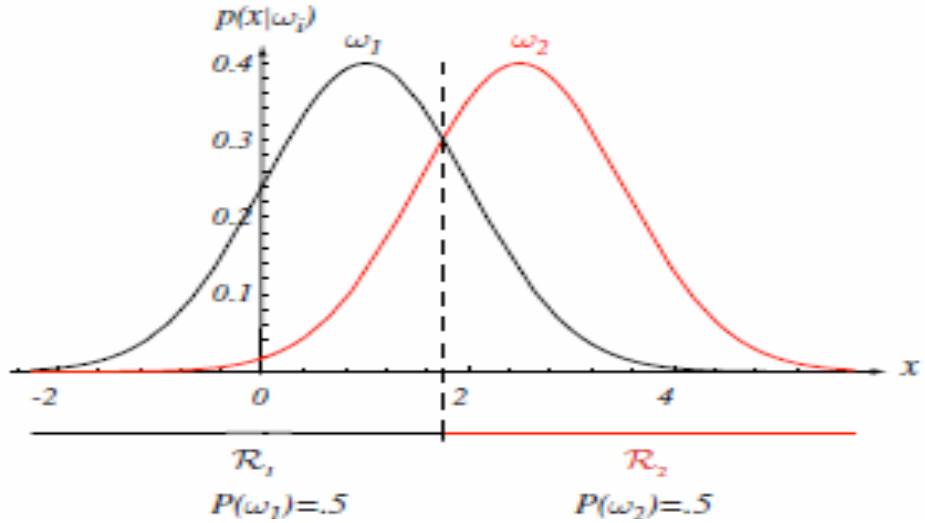
- Numerical Example 2

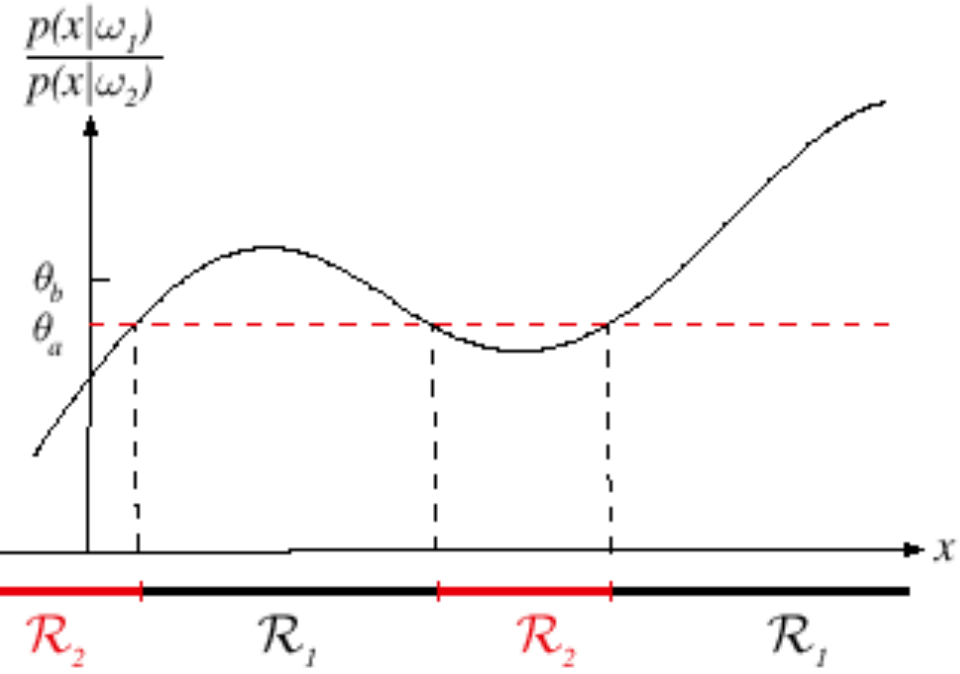
Solution (continued):
Solve the quadratic equation:
Let $f(x) = x^2 - 10x + 22$.

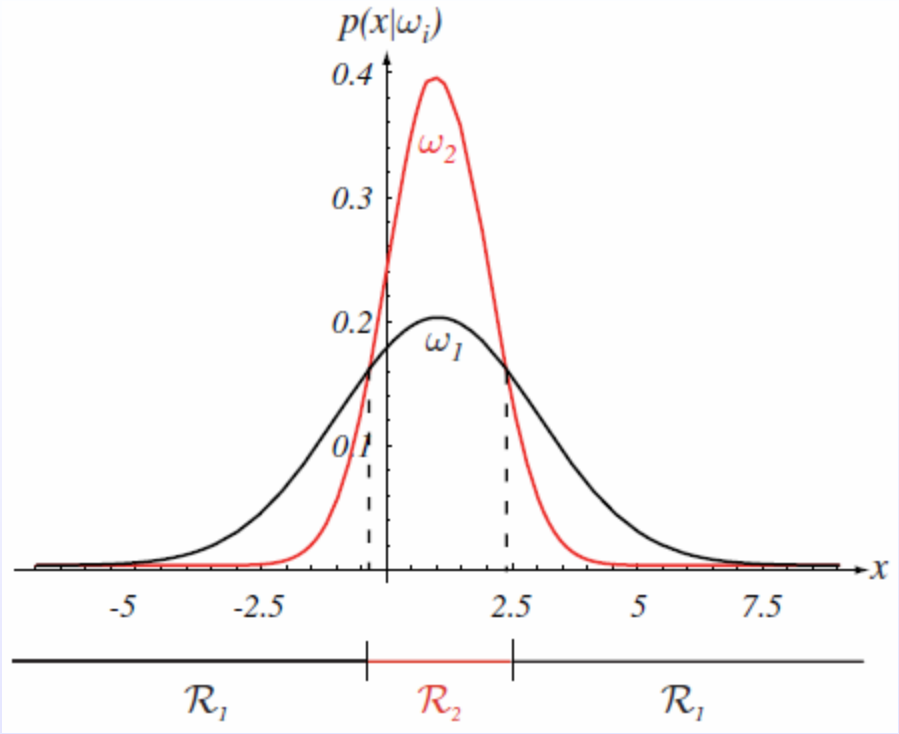
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \Rightarrow x = \frac{10 \pm \sqrt{10^2 - 4(22)}}{2} = 3.27, 6.73$$

We have $f'(x) = 2x - 10$.
Setting $f'(x) = 0$, we have $x = 5$.
We know that $f(5) < 0$.
Hence, by testing the regions, we find that
$f(x) < 0$ for $3.27 < x < 6.73$, and $f(x) > 0$ for $x < 3.27$ and $x > 6.73$.
Therefore, we have the following rule:
Decide ω_1 if $x < 3.27$ or $x > 6.73$, otherwise decide ω_2

8 Understand Pattern Recognition & Decision Theory







- Connected and disconnected decision regions

9 Statistical Estimation & Machine Learning

Outline

- Nonparametric approach to estimate the probability distribution density
  - Parzen-window approach
  - k–nearest-neighbor kNN rule
- Parametric approach to estimate the probability distribution density
  - Maximum likelihood (ML) estimation
  - Multivariate Gaussian probability distribution density

9 Statistical Estimation & Machine Learning

- We understand that the best decision or optimal classification is:

$$\text{Decide } \omega_k = \arg\min_{\omega_i} [p(e_i|\mathbf{x})] = \arg\min_{\omega_i} [1 - p(\omega_i|\mathbf{x})]$$
$$= \arg\max_{\omega_i} [p(\omega_i|\mathbf{x})] = \arg\max_{\omega_i} [p(\omega_i)p(\mathbf{x}|\omega_i)]$$

- To design an automatic pattern recognition system, we need know the probability distribution/density function (PDF) for all values of $\mathbf{x}$ so that the system can make decision for any value of received data $\mathbf{x}$. In practice, the PDF is often unknown and can only be estimated by collected examples/samples $\{\mathbf{x}_i\}=[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ for the design of the pattern recognition system.
- Determining some rules or some deterministic values on a random variable $\mathbf{x}$ based on a set of training samples $D=\{\mathbf{x}_i\}$ is the task of statistical estimation or machine learning.

9 Statistical Estimation & Machine Learning

- The probability distribution/density function (PDF) is the complete information about a random variable.
- It is difficult to estimate the posterior probability $p(\omega_k|\mathbf{x})$ function directly. So we learn the prior probability $p(\omega_k)$ and the class-conditional probability function $p(\mathbf{x}|\omega_k)$.
- The c prior probabilities can be easily estimated by

$$\hat{p}(\omega_k) = n_k/n$$

where n_k and n are the number of training samples of class ω_k and the total number of training samples.

- As the estimation method is the same for all classes, we simplifying the notation of $p(\mathbf{x}|\omega_k)$ to $p(\mathbf{x})$ and suppose we have n samples $\{\mathbf{x}_i\}=[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ drawn independently and identically distributed (i.i.d.) according to the probability law $p(\mathbf{x})$.

9 Statistical Estimation & Machine Learning

- According to the definition of probability density function, the probability P that $\mathbf{x}$ falls in a region R is:

$$P(\mathbf{x}) = \int_{R_\mathbf{x}} p(\mathbf{t})d\mathbf{t} \approx p(\mathbf{x})V$$

- where V is the volume of the region $R_\mathbf{x}$.
- If out of all n training samples of this class, there are k samples fall in the region R, the estimate of P is then.

$$\hat{P}(\mathbf{x}) = \frac{k}{n}$$

- Therefore, the estimate of the PDF $p(\mathbf{x})$ is

$$\hat{p}(\mathbf{x}) = \frac{k}{nV}$$

- This is called nonparametric approach to the estimation of the probability density function because no assumption of the model is applied.

9 Statistical Estimation & Machine Learning

- The procedure to estimate the PDF based on n training samples is: Given a value of $\mathbf{x}$, select a region/cell of size (volume) V centered at $\mathbf{x}$, count the number of samples k fall in the region/cell. The probability density $p(\mathbf{x})$ at $\mathbf{x}$ is then estimated as :

$$\hat{p}(\mathbf{x}) = \frac{k}{nV}$$

- Obviously, different shape and size of the region/cell lead to different estimate of PDF.
- In general, $\mathbf{x}$ is multiple dimensional $\mathbf{x}=[x_1, x_2, \ldots, x_d]$. If we choose a d-dimensional hypercube of the side length h as the region, a sample $\mathbf{x}_i=[x_{i1}, x_{i2}, \ldots, x_{id}]$ will fall into the hypercube if

$$\frac{|x_j - x_{ij}|}{h} < \frac{1}{2} \quad \text{for } \forall j = 1, 2, \ldots, d$$

9 Statistical Estimation & Machine Learning

representation of counting data points fall into region

- Therefore, we can express the number of samples falling into the cell k mathematically as

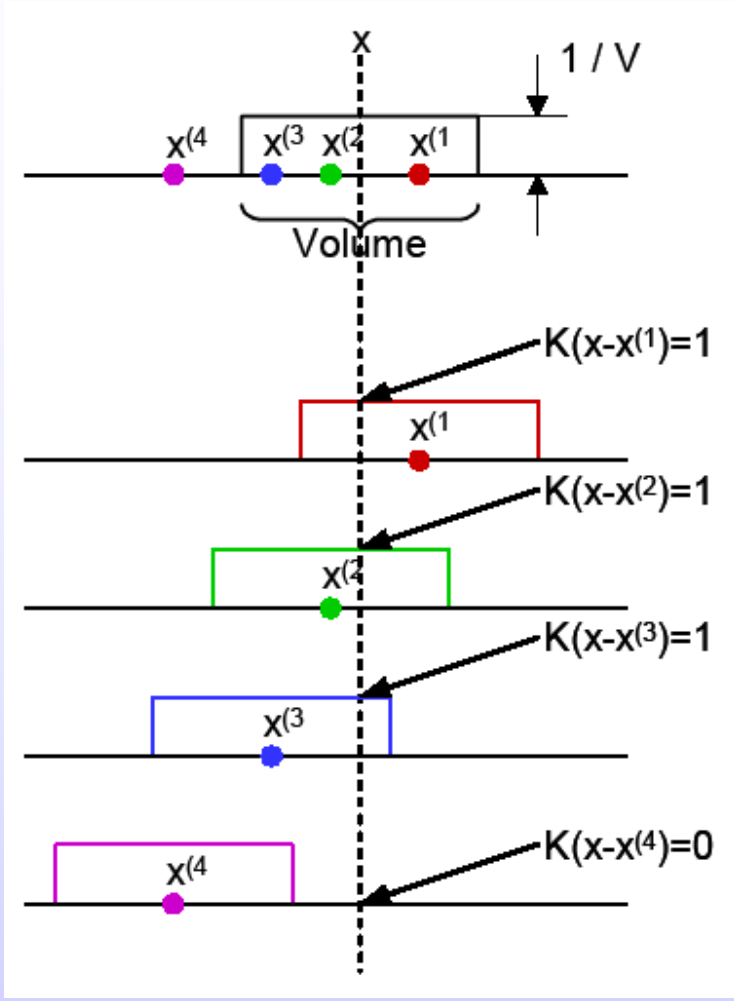
$$k = \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)$$

- Where the kernel function called Parzen-window is defined as

$$K(\mathbf{u}) = \begin{cases} 1 & \text{if } |u_j| < 1/2, \forall j = 1, 2, \ldots, d \\ 0 & \text{otherwise} \end{cases}$$

- The probability density $p(\mathbf{x})$ at $\mathbf{x}$ is then estimated as :

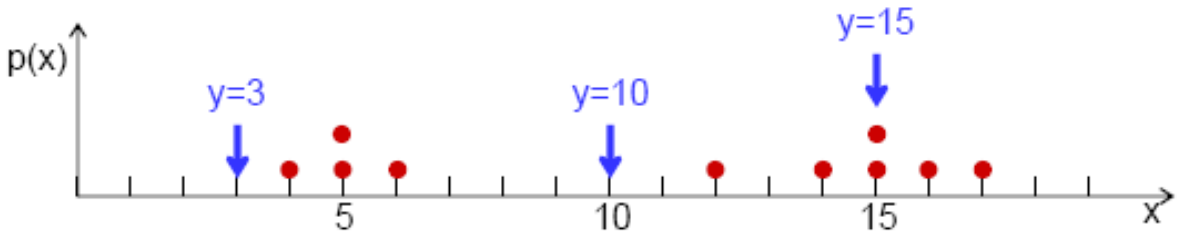
$$\hat{p}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)$$



9 Statistical Estimation & Machine Learning

- **Given the dataset below, use Parzen windows to estimate the density p(x) at y=3,10,15. Use a bandwidth of h=4**
  - $X = \{x^{(1}, x^{(2},\ldots x^{(N}\} = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$
- **Solution**
  - Let's first draw the dataset to get an idea of what numerical results we should expect

Example:



  - Let's now estimate p(y=3):

$$p_{KDE}(y=3)=\frac{1}{Nh^D}\sum_{n=1}^{N}K\left(\frac{y-x^{(n}}{h}\right)=\frac{1}{10\times4^1}\left[K\left(\underbrace{\frac{3-4}{4}}_{-1/4}\right)+K\left(\underbrace{\frac{3-5}{4}}_{-1/2}\right)+K\left(\underbrace{\frac{3-5}{4}}_{-1/2}\right)+K\left(\underbrace{\frac{3-6}{4}}_{-1}\right)+\ldots+K\left(\underbrace{\frac{3-17}{4}}_{-13/4}\right)\right]=$$

$$=\frac{1}{10\times4^1}[1+0+0+0+0+0+0+0+0+0]=\frac{1}{10\times4}=0.025$$

  - Similarly

$$p_{KDE}(y=10)=\frac{1}{10\times4^1}[0+0+0+0+0+0+0+0+0+0]=\frac{0}{10\times4}=0$$

$$p_{KDE}(y=15)=\frac{1}{10\times4^1}[0+0+0+0+0+1+1+1+1+0]=\frac{4}{10\times4}=0.1$$

Note here:

$$K(\mathbf{u})=\begin{cases}1 \text{ if } |u_j|<1/2\\ 0 \text{ otherwise}\end{cases}$$

9 Statistical Estimation & Machine Learning

- The rectangular kernel function produces unsmoothed PDF due to the unsmooth rectangular kernel function. In fact, we can choose any function as the kernel so long as:

$$K(\mathbf{u}) \geq 0, \quad \int_{-\infty}^{\infty} K(\mathbf{u}) d\mathbf{u} = 1 \text{ i.e. } \int_{-\infty}^{\infty} \frac{1}{h^d} K(\frac{\mathbf{x}-\mathbf{x}_i}{h}) d\mathbf{x} = 1$$

$$\because \int_{-\infty}^{\infty} K(\mathbf{u}) d\mathbf{u} = \int_{-\infty}^{\infty} K(\frac{\mathbf{x}-\mathbf{x}_i}{h}) d\frac{\mathbf{x}-\mathbf{x}_i}{h} = \int_{-\infty}^{\infty} \frac{1}{h^d} K(\frac{\mathbf{x}-\mathbf{x}_i}{h}) d\mathbf{x}$$

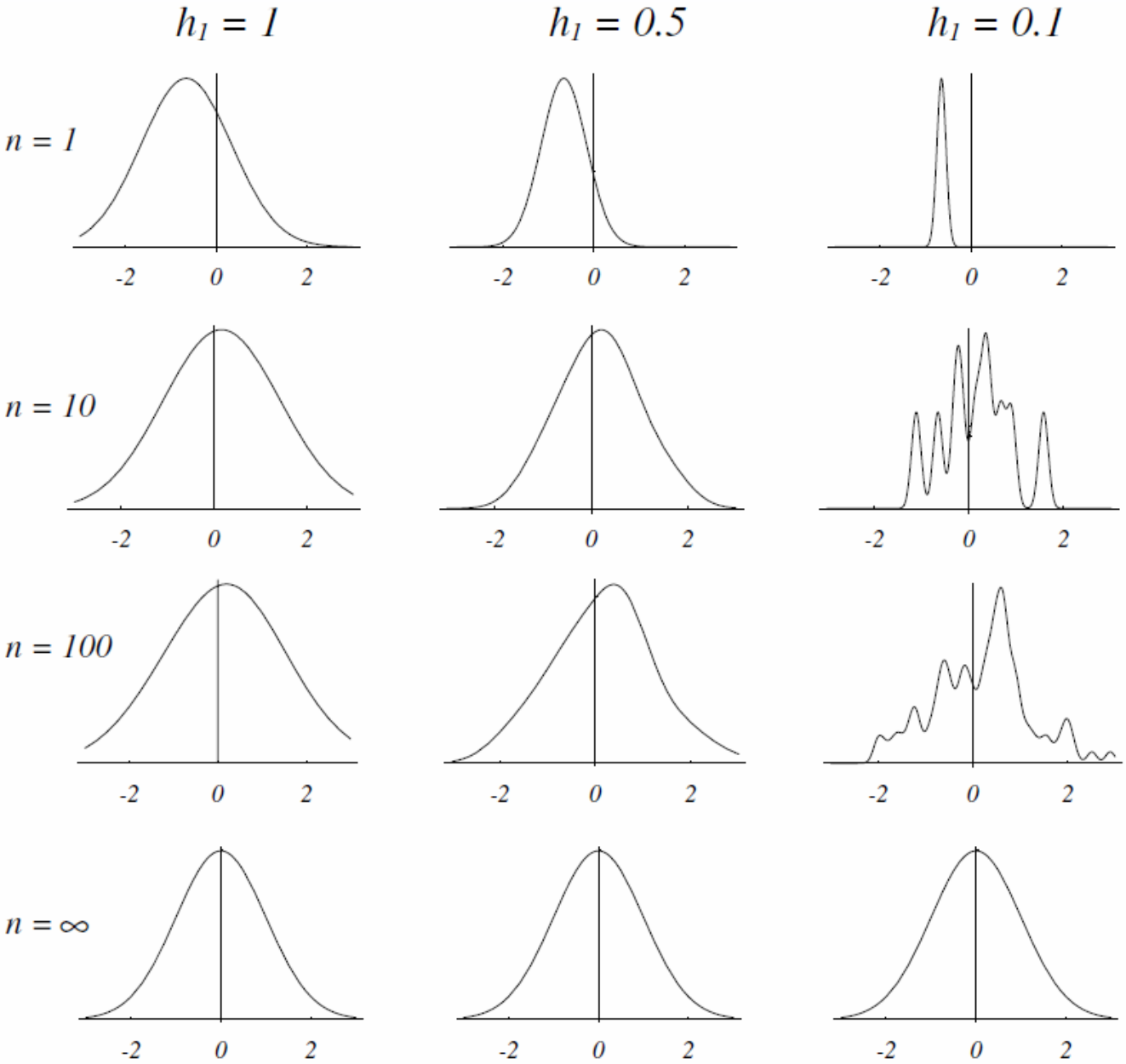
- Therefore, any DPF function can be served as the kernel function. This approach is called Parzen-window approach, it in fact interpolates the discrete points $\{\mathbf{x}_i\}=[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ into a continuous function PDF $p(\mathbf{x})$.

9 Statistical Estimation & Machine Learning

1-D example.

$$K(\mathbf{u}) = N(0, \mathbf{I}) =$$

$$\frac{1}{(2\pi)^{d/2}} \exp\left[-\frac{1}{2}\mathbf{u}^T\mathbf{u}\right]$$

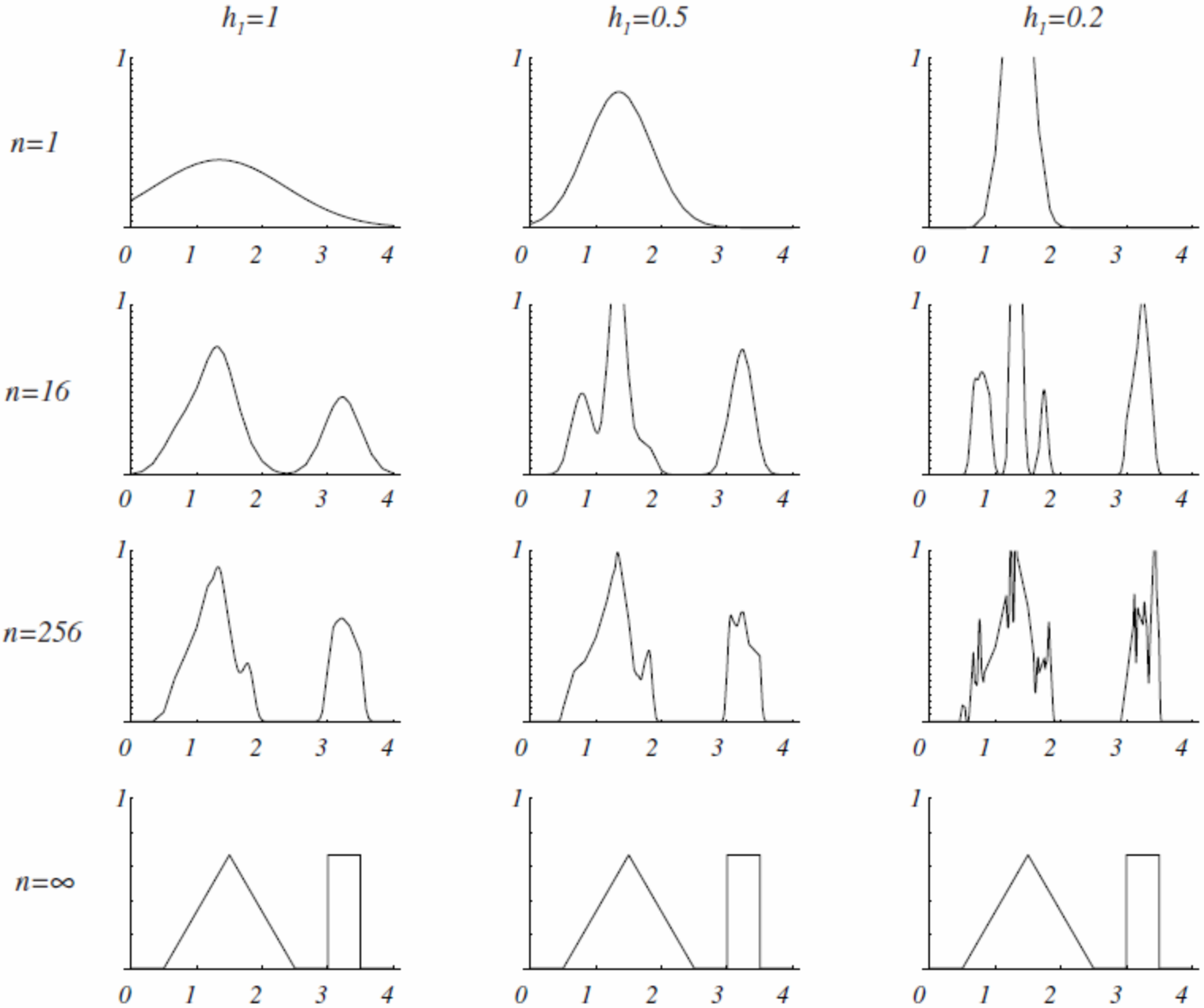


exdjiang@ntu.edu.sg

9 Statistical Estimation & Machine Learning

another 1-D example.

bimodal distribution



exdjiang@ntu.edu

9 Statistical Estimation & Machine Learning

- A single 2-D training sample with Gaussian kernel function of different width h.

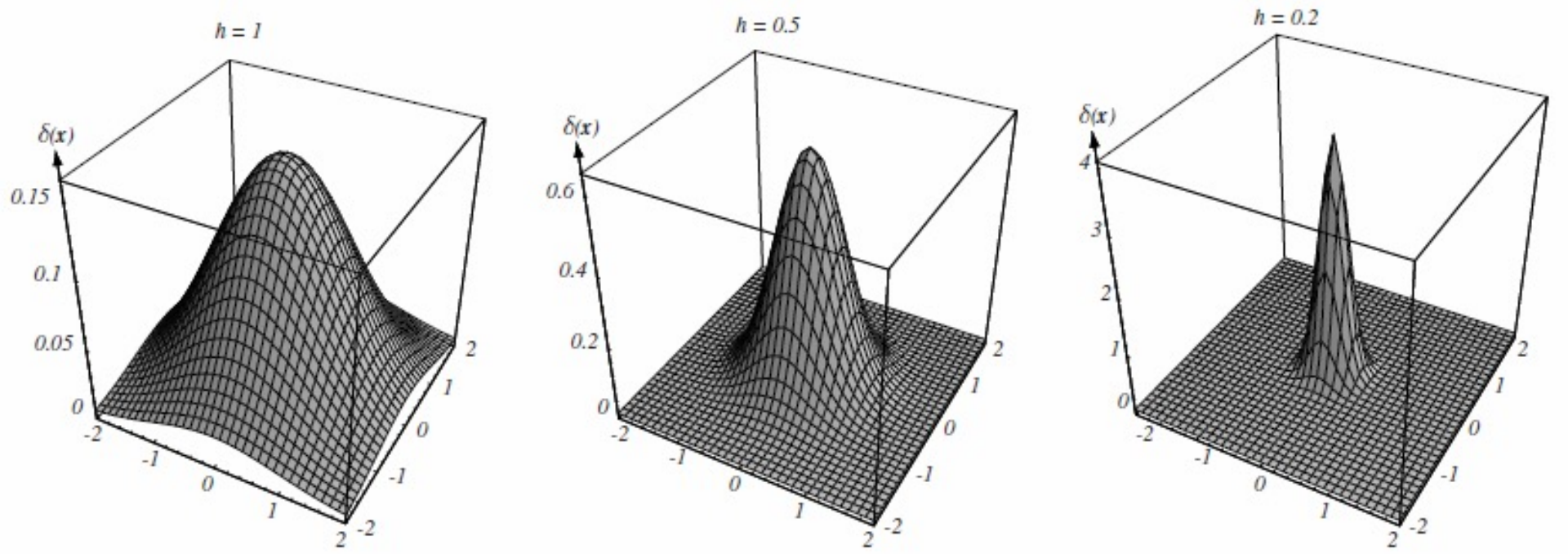


FIGURE 4.3. Examples of two-dimensional circularly symmetric normal Parzen windows for three different values of *h*. Note that because the $\delta(\mathbf{x})$ are normalized, different

9 Statistical Estimation & Machine Learning

- 5 2-D training samples with Gaussian kernel function of different width h.

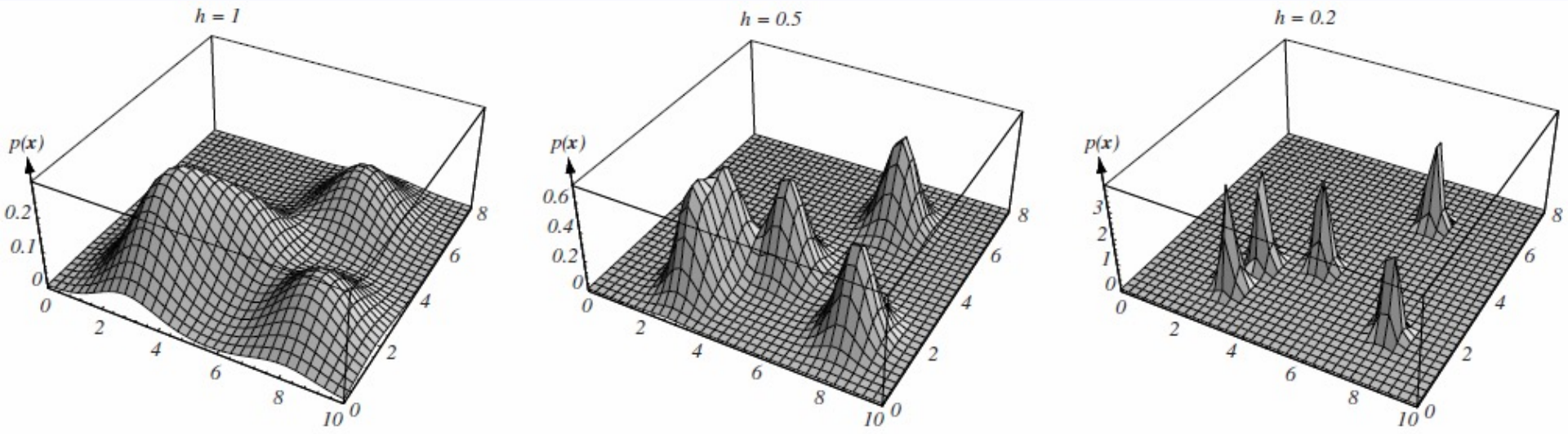


FIGURE 4.4. Three Parzen-window density estimates based on the same set of five samples, using the window functions in Fig. 4.3. As before, the vertical axes have been scaled to show the structure of each distribution.

- The probabilistic neural networks and the RBF neural networks are almost equivalent to the Parzen-window approach.

9 Statistical Estimation & Machine Learning

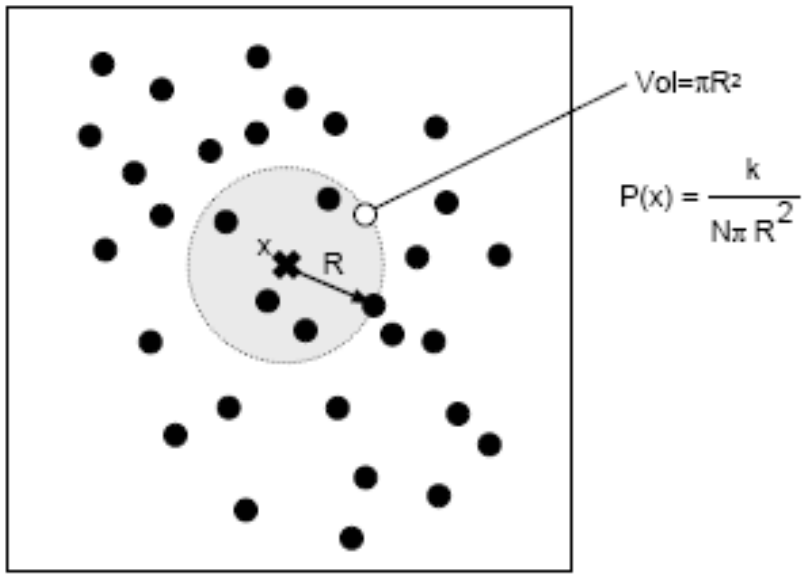
- To estimate the PDF by $$\hat{p}(\mathbf{x}) = \frac{k}{nV}$$

  the Parzen-window approach selects a region/cell of fixed size (volume) V centered at $\mathbf{x}$ and counts the number of samples k fall in the region/cell for the estimation of PDF.

- We can also select the fixed number of samples k and compute the size/volume that just encloses k samples to estimate the PDF by $$\hat{p}(\mathbf{x}) = \frac{k}{nV}$$

- This approach is called

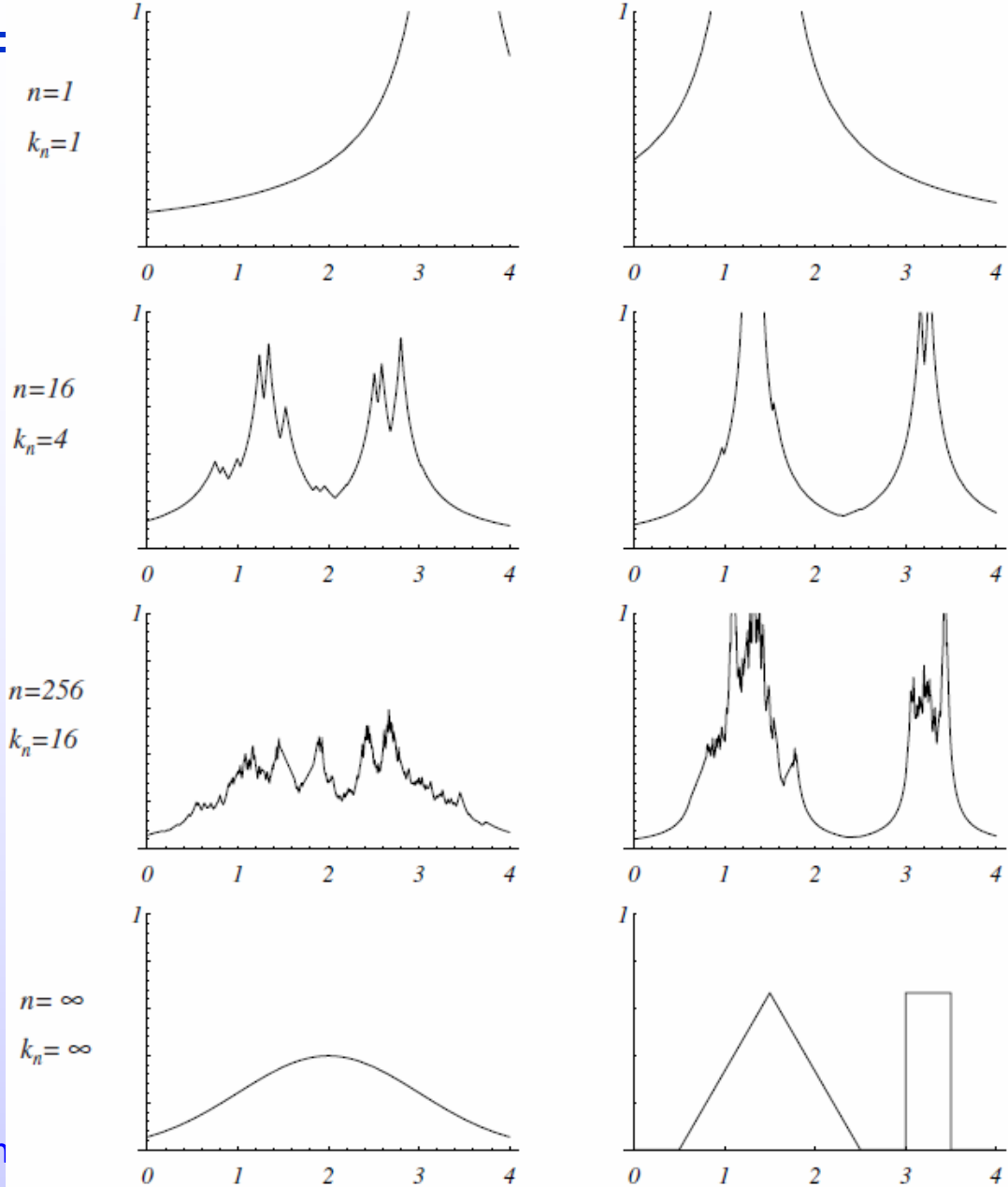
  k–nearest-neighbor kNN estimation.



9 Statistical Estimation & Machine Learning

- Several k-nearest-neighbor estimates of two uni-dimensional densities:

$$\hat{p}(\mathbf{x}) = \frac{k}{nV(k)}$$



9 Statistical Estimation & Machine Learning

$$\hat{p}(\mathbf{x}) = \frac{k}{nV(k)}$$

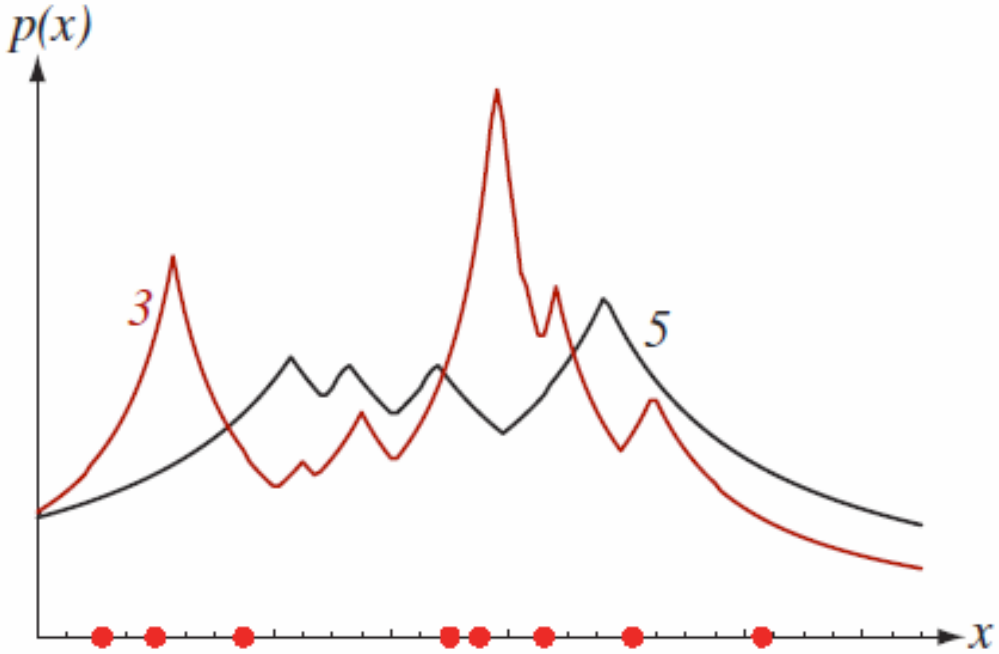


FIGURE 4.10. Eight points in one dimension and the k-nearest-neighbor density estimates, for $k = 3$ and 5. Note especially that the discontinuities in the slopes in the estimates generally lie *away* from the positions of the prototype points. From: Richard

9 Statistical Estimation & Machine Learning

$$\hat{p}(\mathbf{x}) = \frac{k}{nV(k)}$$

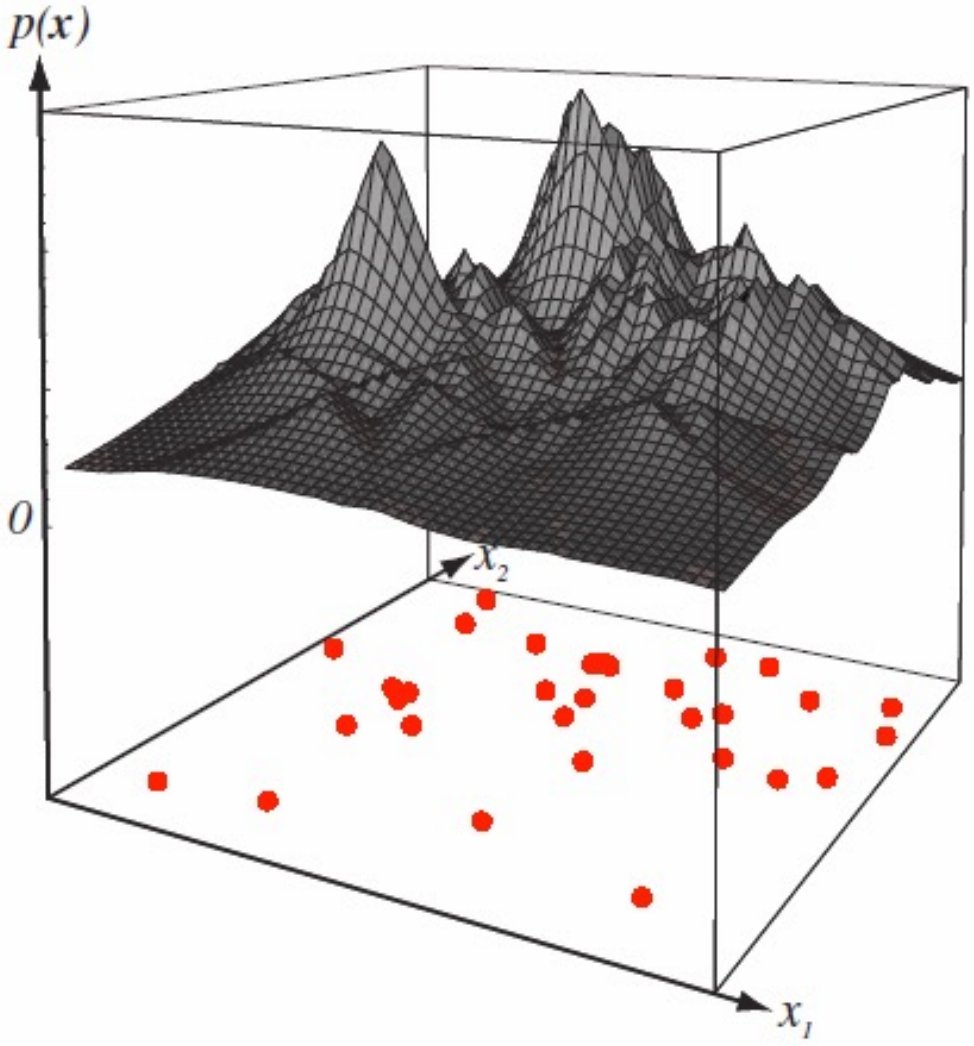


FIGURE 4.11. The k-nearest-neighbor estimate of a two-dimensional density for $k = 5$.

9 Statistical Estimation & Machine Learning

- The KNN technique can also be used for estimation of a-posteriori probabilities $P(\omega_i|\mathbf{x})$ from a set of n labeled samples. (Compare to PNNs and Parzen window estimates.)
- Suppose that we place a cell of volume V around $\mathbf{x}$ and capture k samples, k_i of which turn out to be labeled ω_i.
- An estimate for the joint probability is $p_n(\mathbf{x}, \omega_i) = \frac{k_i}{nV}$. $\hat{p}(\mathbf{x}) = \frac{k}{nV(k)}$
- Hence, we can estimate $P(\omega_i|\mathbf{x})$ by

$$\hat{p}(\omega_i|\mathbf{x}) = \hat{P}(\omega_i)\hat{p}(\mathbf{x}|\omega_i)/\hat{p}(\mathbf{x})$$

$$= \frac{n_i}{n}\frac{k_i}{n_i V(k)}\frac{nV(k)}{k} = \frac{k_i}{k}$$

$$\frac{p_n(\mathbf{x}, \omega_i)}{\sum_{j=1} p_n(\mathbf{x}, \omega_j)} = \frac{k_i}{k}.$$

$$\hat{P}(\omega_i) = \frac{n_i}{n}$$

$$\hat{p}(\mathbf{x}|\omega_i) = \frac{k_i}{n_i V(k)}$$

9 Statistical Estimation & Machine Learning

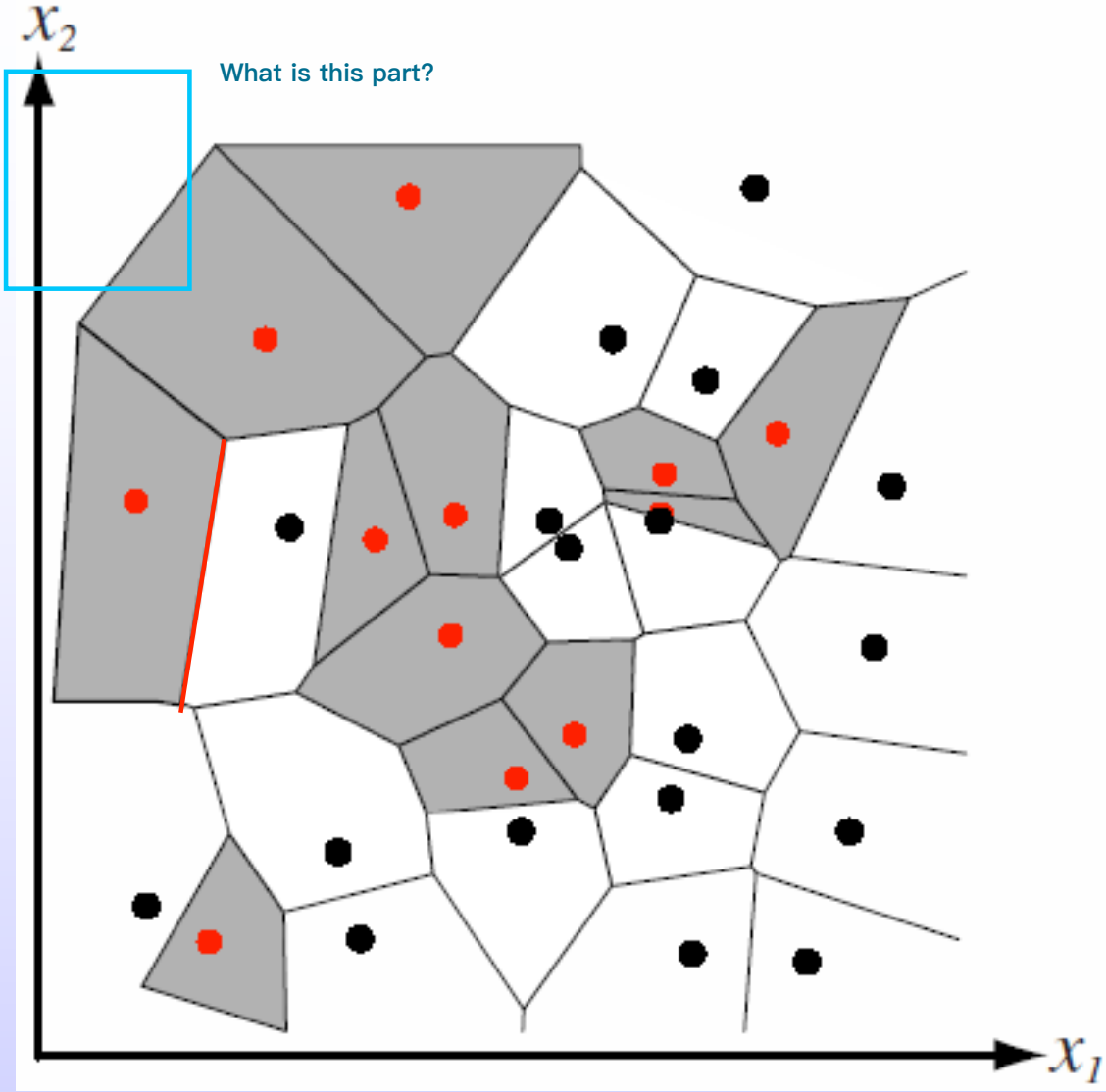
- Denote by $\mathcal{D}^n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ a set of labeled **prototypes** or training samples.
- Let $\mathbf{x}^*$ be the prototype nearest to $\mathbf{x}$.
- Then **the nearest-neighbor (NN) rule** for classifying $\mathbf{x}$ is to assign it the label associated with $\mathbf{x}^*$.
- More formally if we have a set of labeled training samples $\{(\mathbf{x}_1, \theta_1), \ldots, (\mathbf{x}_n, \theta_n)\}$, where each θ_i is one of the labels $\omega_1, \ldots, \omega_c$, then the NN decision rule is

$$\alpha_{nn}(\mathbf{x}) = \theta_k : k = \arg\min_i ||\mathbf{x} - \mathbf{x}_i||.$$

9 Statistical Estimation & Machine Learning

- Classification boundary of 1st-NN classifier, also called NN classifier.

Decision Boundary, k=1 (only one point):
Same distance between 2 points



9 Statistical Estimation & Machine Learning

- The error rate of NN classifier P for very large number of training samples are bounded as

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

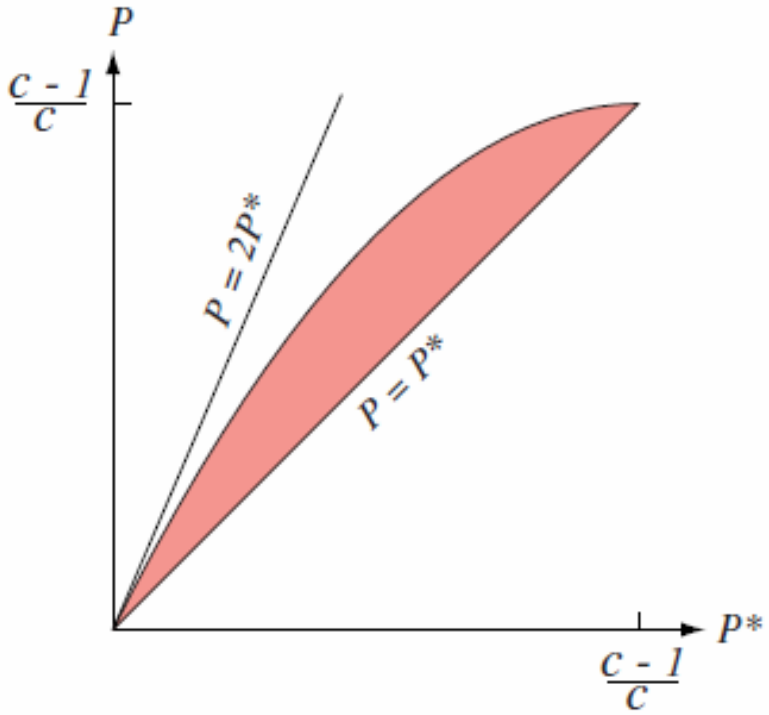


FIGURE 4.14. Bounds on the nearest-neighbor error rate P in a c-category problem given infinite training data, where P^* is the Bayes error (Eq. 52). At low error rates, the nearest-neighbor error rate is bounded above by twice the Bayes rate. From: Richard O.

9 Statistical Estimation & Machine Learning

- Generalization of the NN rule.
- The k_n nearest neighbors rule: Given a set of training samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and a test point $\mathbf{x}$, find k training points closest to $\mathbf{x}$, $\mathbf{x}_1^*, \ldots, \mathbf{x}_k^*$. Collect the labels associated $\theta_1^*, \ldots, \theta_k^*$ and classify $\mathbf{x}$ to the class which has the greatest number of representatives in $\theta_1^*, \ldots, \theta_k^*$.

vote

- In other words, the classification is performed by taking the majority vote among k nearest neighbors of $\mathbf{x}$.

9 Statistical Estimation & Machine Learning

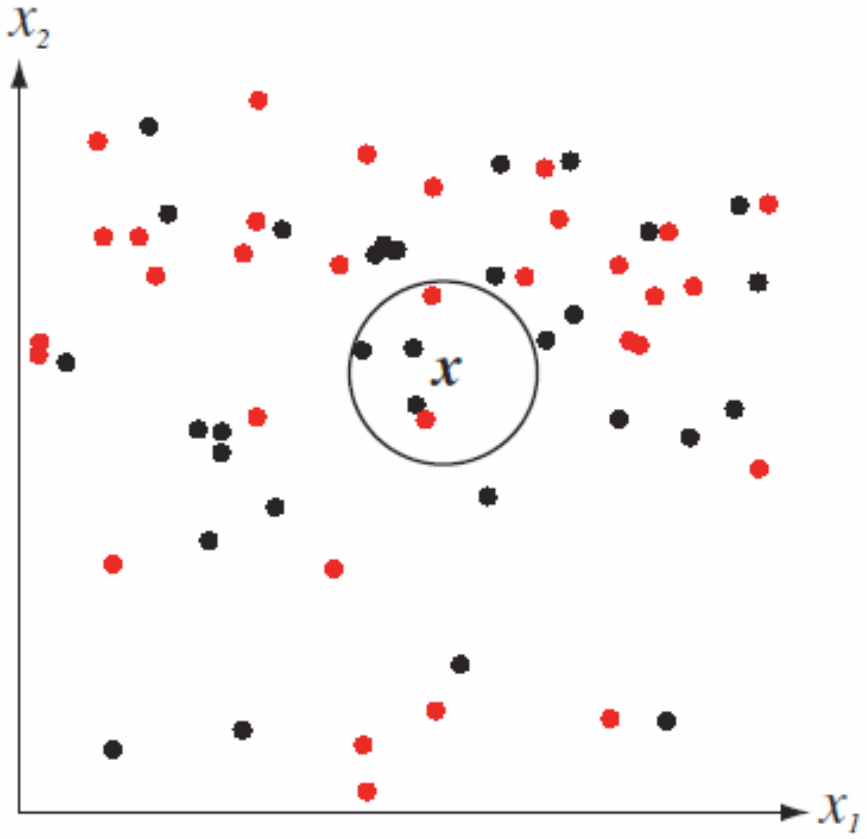


FIGURE 4.15. The k-nearest-neighbor query starts at the test point $\mathbf{x}$ and grows a spherical region until it encloses k training samples, and it labels the test point by a majority vote of these samples. In this $k = 5$ case, the test point $\mathbf{x}$ would be labeled the category of the black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

9 Statistical Estimation & Machine Learning

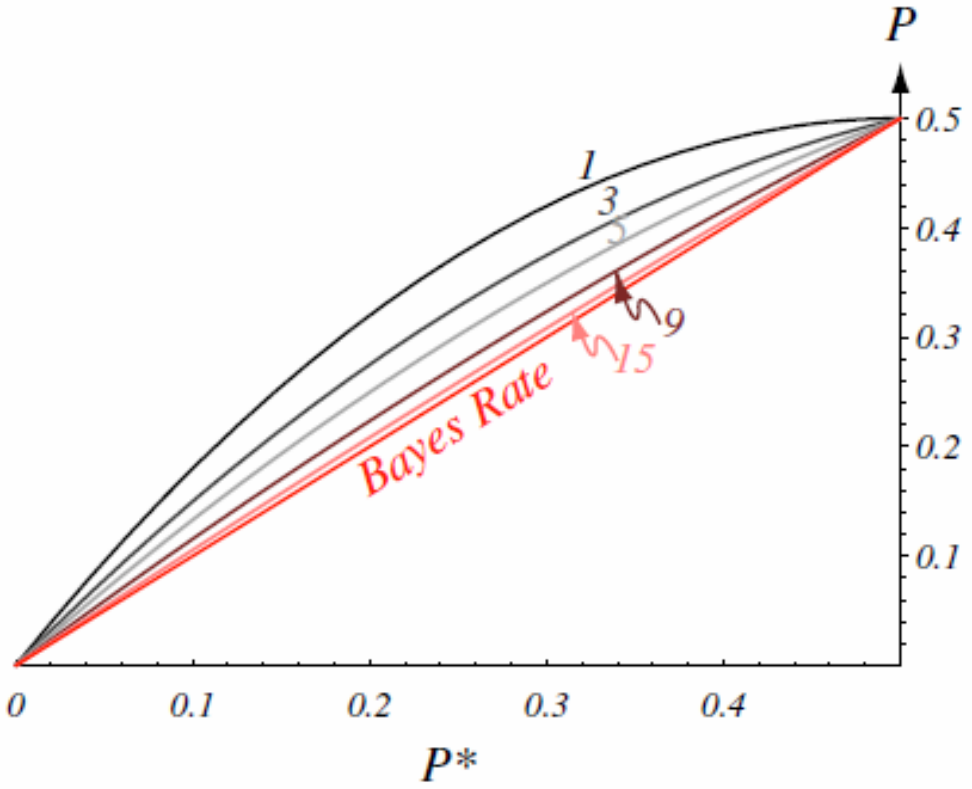


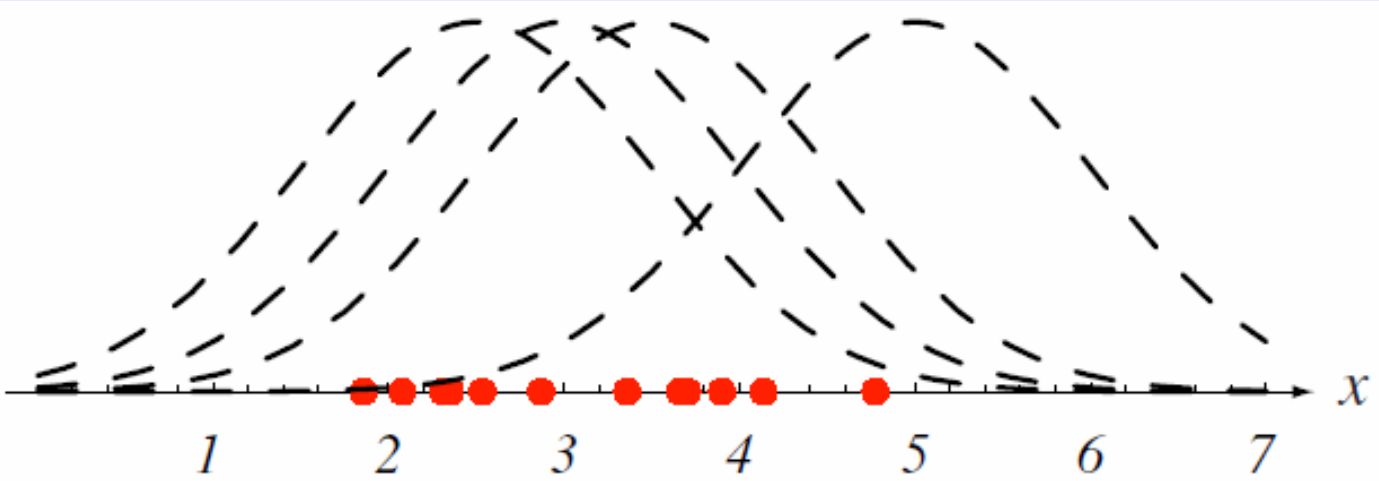
FIGURE 4.16. The error rate for the k-nearest-neighbor rule for a two-category problem is bounded by $C_k(P^*)$ in Eq. 54. Each curve is labeled by k; when $k = \infty$, the estimated probabilities match the true probabilities and thus the error rate is equal to the Bayes rate, that is, $P = P^*$. From: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

9 Statistical Estimation & Machine Learning

- We see that the learned conditional PDF from training samples could greatly deviate from the true PDF of the population, especially in case of small number of training samples.
- If our general knowledge about the problem permits us to model the conditional PDF, i.e. using a mathematical analytical function to represent the PDF with unknown parameter. The severity of these problems can be reduced significantly. Here we parameterize the conditional PDF, which is called parametric method.
- Suppose, for example, we can reasonably assume that $p(\mathbf{x}|\omega_k)$ is a Gaussian density with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. Although we do not know their values, this knowledge simplifies the problem from estimating an unknown function $p(\mathbf{x}|\omega_k)$ to estimating the unknown parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ only.

9 Statistical Estimation & Machine Learning

- Now we will use a set of training samples $D=\{\mathbf{x}_i\}=[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ drawn independently from the probability density $p(\mathbf{x}|\boldsymbol{\theta})$ to estimate the unknown parameter vector $\boldsymbol{\theta}$.
- Lets see an example to generate idea how to estimate the parameter of a given probability density $p(\mathbf{x}|\boldsymbol{\theta})$ reasonably based on the training data D.
- The graph shows several training points in one dimension, known or assumed to be drawn from a Gaussian of a particular variance, but unknown mean. Four PDF with 4 different means are shown in dashed lines.
- Which PDF you should choose?



exdjiang@ntu.edu.sg

9 Statistical Estimation & Machine Learning

- Now we formulate our idea mathematically.
- Obviously, the probability that a sample $\mathbf{x}_k$ occurs is $p(\mathbf{x}_k|\boldsymbol{\theta})$. As all samples in the training set are independently collected (occur), the probability that all samples occur is

$$p(D|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta})$$

- Intuitively, we should select the parameter so that the probability density $p(\mathbf{x}|\boldsymbol{\theta})$ best supports the actually observed training samples, i.e. to make the probability of all training data occur $p(D|\boldsymbol{\theta})$ maximal. Note that $p(D|\boldsymbol{\theta})$ is called the likelihood of $\boldsymbol{\theta}$ with respect to the set of samples D. Thus, this method is called the maximum likelihood (ML) estimation.

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} p(D|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta})$$

9 Statistical Estimation & Machine Learning

- It is often not easy to get an analytical solution of

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} p(D|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta})$$

due to the multiplication of the functions of $\boldsymbol{\theta}$ and $p(\mathbf{x}_k|\boldsymbol{\theta})$ is often nonlinear function of $\boldsymbol{\theta}$.

- Since the logarithm is monotonically increasing, maximizing the logarithm of a function also maximizes the function itself. The logarithm has nice property that converts the multiplication into summation and simplifies the exponential function.
- Thus, we maximize the log-likelihood instead of maximizing the likelihood

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ln p(D|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{k=1}^{n} \ln p\left(\mathbf{x}_k|\boldsymbol{\theta}\right)$$

9 Statistical Estimation & Machine Learning

- The solution can be found by the standard methods of differential calculus: Solving the equation that the gradient is zero.

$$\nabla_{\boldsymbol{\theta}} \ln p\left(D|\boldsymbol{\theta}\right) = 0, \qquad \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}} \ln p\left(\mathbf{x}_k|\boldsymbol{\theta}\right) = 0$$

- If the number of parameters to be estimated is q, then $\boldsymbol{\theta}$ is a q-component vector $\boldsymbol{\theta}=(\theta_1, \theta_2, \ldots, \theta_q)^T$. The gradient is a vector that contains partial differentiation against all components of $\boldsymbol{\theta}$.

$$\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \triangleq \begin{pmatrix} \dfrac{\partial f(\boldsymbol{\theta})}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial f(\boldsymbol{\theta})}{\partial \theta_q} \end{pmatrix}$$

9 Statistical Estimation & Machine Learning

- To see how maximum likelihood methods results apply to a specific case, suppose that the samples are drawn from a multivariate Gaussian population with unknown mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

$$p(\mathbf{x}|\boldsymbol{\theta}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

- The log-likelihood of a single sample is

$$\ln p\,(\mathbf{x}_k|\boldsymbol{\theta}) = -\frac{1}{2}\ln[\,(2\pi)^d|\boldsymbol{\Sigma}|] - \frac{1}{2}(\mathbf{x}_k-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k-\boldsymbol{\mu})$$

- Consider first the univariate case with $\boldsymbol{\theta}=(\theta_1, \theta_2)^T=(\mu, \sigma^2)^T$.

$$p(x|\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

9 Statistical Estimation & Machine Learning

- Here the log-likelihood of a single sample is simplified as

$$\ln p\,(x_k|\boldsymbol{\theta}) = -\frac{1}{2}\ln[\,2\pi\sigma^2] - \frac{1}{2\sigma^2}(x_k-\mu)^2$$

- Its derivative is

$$\nabla_{\boldsymbol{\theta}} \ln p\,(x_k|\boldsymbol{\theta}) = \begin{pmatrix} \frac{1}{\sigma^2}(x_k-\mu) \\ -\frac{1}{2\sigma^2} + \frac{(x_k-\mu)^2}{2\sigma^4} \end{pmatrix}$$

对theta ^2 求导

9 Statistical Estimation & Machine Learning

- Applying ML

$$\nabla_{\boldsymbol{\theta}} \ln p\,(D|\boldsymbol{\theta}) = \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}} \ln p\,(\mathbf{x}_k|\boldsymbol{\theta}) = 0$$

- We have

$$\sum_{k=1}^{n} \frac{1}{\sigma^2}(x_k - \mu) = 0$$

$$-\sum_{k=1}^{n} \frac{1}{2\sigma^2} + \sum_{k=1}^{n} \frac{(x_k - \mu)^2}{2\sigma^4} = 0$$

- Solve these two equations we have the following maximum likelihood estimates

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n} (x_k - \hat{\mu})^2$$

9 Statistical Estimation & Machine Learning

- While the analysis of the multivariate case is basically very similar, considerably more manipulations are involved. The result is that the maximum likelihood estimates for mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ of multivariate Gaussian PDF

$$p(\mathbf{x}|\boldsymbol{\theta}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

are given by

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

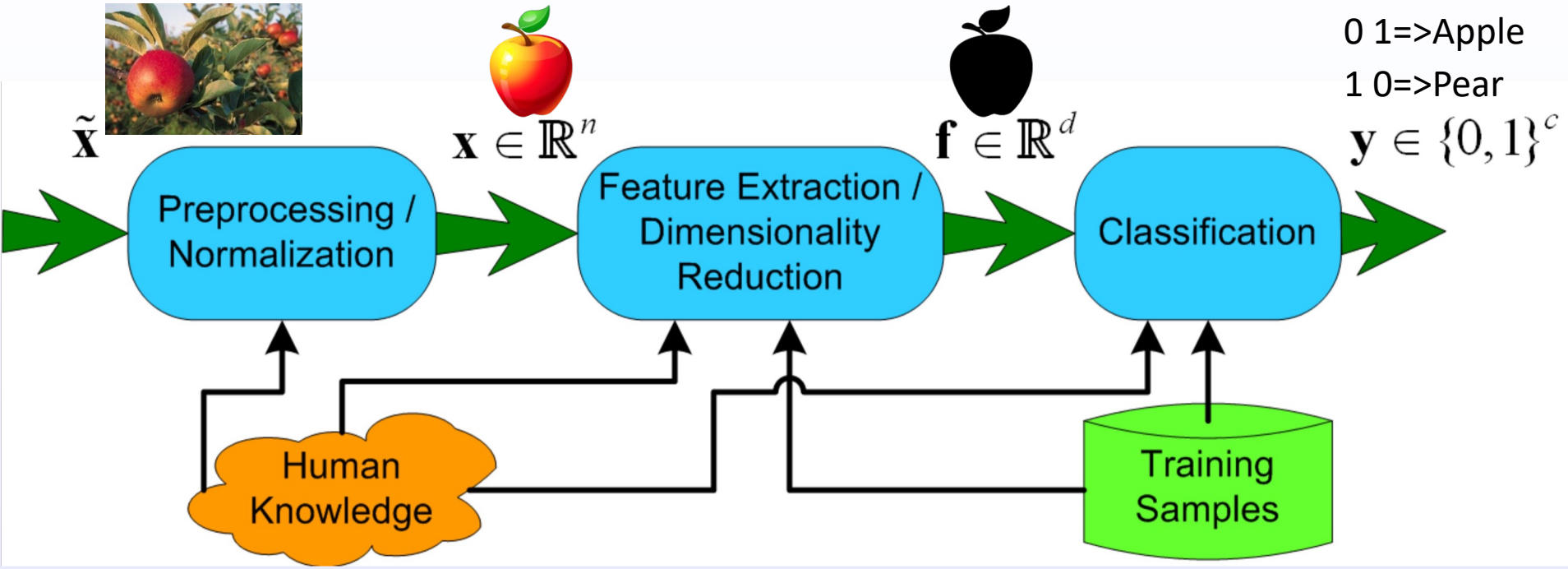
$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k-\hat{\boldsymbol{\mu}})(\mathbf{x}_k-\hat{\boldsymbol{\mu}})^T$$

12 Feature Extraction/Dimension Reduction v. Machine Learning

Outline:

- Restudy the Functionalities of pattern recognition modules
- Approaches to the feature extraction and dimensionality reduction
- Machine learning approaches:
  - Eigen-decomposition
  - PCA: the principal component analysis
  - LDA: the linear discriminant analysis

12 Feature Extraction/Dimension Reduction v. Machine Learning



$\tilde{\mathbf{x}} \in \mathbb{R}^{w \times h}$, often $w \times h > n \gg d \gg c$ $\quad \mathbf{y} \in \{1,2,\ldots,c\}$ or $\mathbf{y} \in \{0,1\}^c$

- Pattern recognition is a series of processes that reduce the dimensionality and the variation of samples for the same class and keep their discrimination for different classes.

Dimensionality reduction and Discriminative information extraction

12 Feature Extraction/Dimension Reduction v. Machine Learning

- Pattern recognition in general is to classify an observed data into one of the classes.
- The observed data often have multiple components and the data of a same class are in general not exact same and have some variation. We thus represent the observed data by a random vector.
- The probability distribution fully characterizes a random vector.
- In most cases, the probability distribution is unknown.
- Pattern Recognition via Machine Learning is to estimate the probability distribution directly or indirectly from the known data/samples.

12 Feature Extraction/Dimension Reduction v. Machine Learning

- In some applications such as visual object detection and recognition, bioinformatics and data mining, high data dimensionality imposes great burdens on the robust and accurate recognition due to insufficient knowledge about the data population and limited number of training samples.
- Dimensionality reduction thus becomes a separate and maybe the most critical module of such recognition systems.
- Extracting the discriminative and reliable features or dimensions is always the key step for image recognition and computer vision.

- Feature extraction and dimensionality reduction can be based on
  - Human expert knowledge
  - Image local structures
  - Image global structure
  - Machine learning from training database

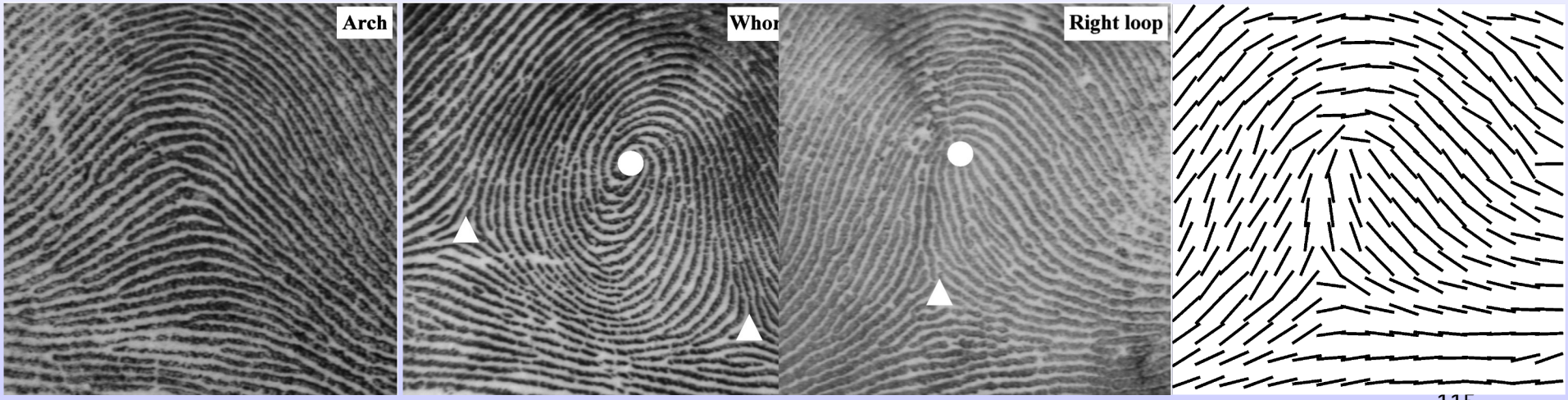
12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Human Knowledge:

- Fingerprint classification: for example, arch, whorl and loop
- Local orientations are intrinsic features for such task.(Human knowledge)
- Orientation field consisting of fingerprint local orientations represented by short lines.

X.D. Jiang, Extracting Image Orientation Feature by Using Integration Operator, *Pattern Recognition*, vol. 40, no. 2, pp. 705-717, February 2007.

X.D. Jiang, On Orientation and Anisotropy Estimation for Online Fingerprint Authentication, *IEEE Transactions on Signal Processing*, vol. 53, no. 10, pp. 4038- 4049, October 2005.



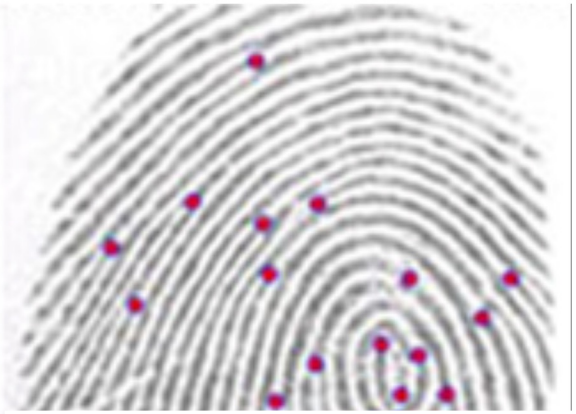
12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Human Knowledge:

Fingerprint identification: Human experts teach us, the reliable and discriminative features are minutia points: where ridge terminates or bifurcates.

X.D. Jiang, W. Yau and W. Ser, "Detecting the Fingerprint Minutiae by Adaptive Tracing the Gray Level Ridge," *Pattern Recognition*, vol. 34, no. 5, pp. 999-1013, May 2001.

X.D. Jiang, M. Liu and A. Kot, "Fingerprint Retrieval for Identification," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 4, pp. 532-542, December 2006.



Good knowledge about the features doesn't mean the computer can reliably extract them due to poor image quality and noise.

exdjiang@ntu.edu.sg http://www.ntu.edu.sg/home/exdjiang

12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Image Local Structures:

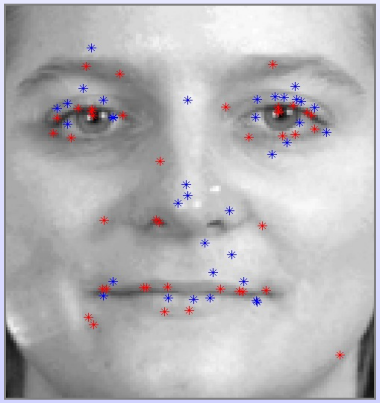
Corners and blobs called key points or interesting points have locations, scales and shapes.

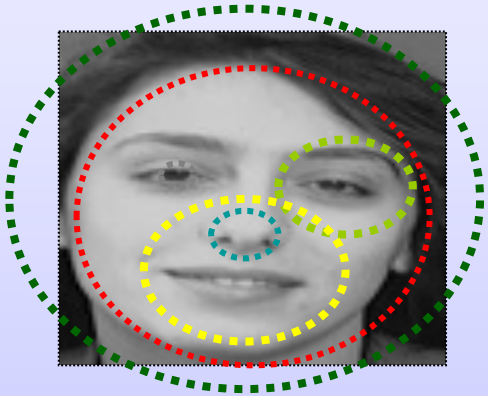
C. Geng and X.D. Jiang, "Face Recognition Based on the Multi-scale Local Image Structures," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2565-2575, October-November 2011.

C. Geng and X.D. Jiang, "Fully Automatic Face Recognition Framework based on Local and Global Features," *Machine Vision and Applications*, vol. 24, no. 3, pp. 537-549, April 2013.

Z. Miao and X.D. Jiang, "Interest Point Detection Using Rank Order LoG Filter," *Pattern Recognition*, vol. 46, no. 11, pp. 2890-2901, November 2013.

Z. Miao, X.D. Jiang and K. Yap, "Contrast Invariant Interest Point Detection by Zero-Norm LoG Filter," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 331 - 342, January 2016.





12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Image Global Structures:

Transform image $f(x,y)$ to feature $g(u,v)$

$$g(u,v) = T\{f(x,y)\}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(u,v,x,y)f(x,y)dxdy \quad \text{linear transform}$$

$$\Rightarrow \sum_{1}^{h}\sum_{1}^{w} w(u,v,x,y)f(x,y) \quad \text{for digital image}$$

$$\Rightarrow \sum_{1}^{h}\sum_{1}^{w} e^{-2\pi j(ux+vy)}f(x,y) \quad \text{Fourier transform}$$

$$\Rightarrow \sum_{1}^{h}\sum_{1}^{w} x^u y^v f(x,y) \quad \text{moments computing}$$

$$\Rightarrow \mathbf{f} = \mathbf{W}^T\mathbf{x} \quad \text{vector-matrix representation}$$

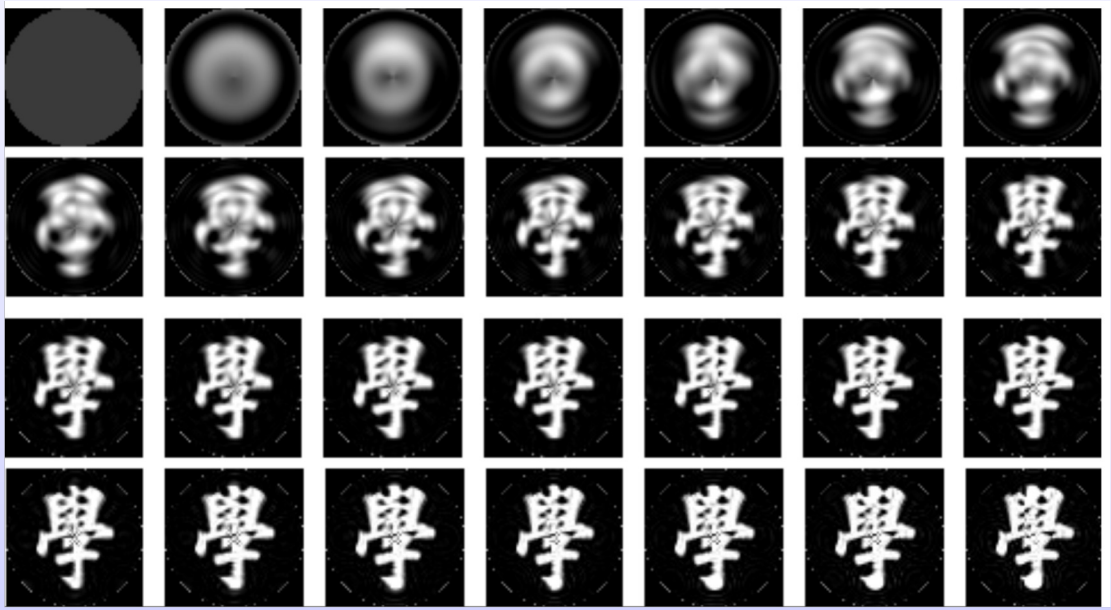
12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Image Global Structures:

Polar Complex Exponential Transform (PCET)

$$g(u,v) = \frac{1}{\pi}\int_0^{2\pi}\int_0^1 e^{-j(2\pi u r^2 + v\theta)} f(r,\theta)\, dr\, d\theta$$

Shows good property in the image reconstruction.



P. Yap, X.D. Jiang and A. Kot, "Two Dimensional Polar Harmonic Transforms for Invariant Image Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1259-1270, July 2010.

12 Feature Extraction/Dimension Reduction v. Machine Learning

Feature Extraction Based on Image Global Structures: Histogram of gradients and histogram of LBP

J. Ren, X.D. Jiang and J. Yuan, "LBP Encoding Schemes Jointly Utilizing the Information of Current Bit and Other LBP Bits," *IEEE Signal Processing letters*, vol. 22, no. 12, pp. 2373 - 2377, Dec. 2015.

J. Ren, X.D. Jiang and J. Yuan, "A Chi-Squared-Transformed Subspace of LBP Histogram for Visual Recognition," *IEEE Trans. Image Processing*, vol. 24, no. 6, pp. 1893-1904, June, 2015.

J. Ren, X.D. Jiang and J. Yuan, "Learning LBP Structure by Maximizing the Conditional Mutual Information," *Pattern Recognition*, vol. 48, no. 10, pp. 3180 - 3190, Oct. 2015.

J. Ren, X.D. Jiang, J. Yuan and W. Gang, "Optimizing LBP Structure for Visual Recognition Using Binary Quadratic Programming," *IEEE Signal Processing letters*, vol. 21, no. 11, pp. 1346-1350, Nov. 2014.

A. Satpathy, X.D. Jiang and H. Eng, "LBP Based Edge-Texture Features for Object Recognition," *IEEE Trans. Image Processing*, vol. 23, no. 5, pp. 1953-1964, May, 2014.

A. Satpathy, X.D. Jiang and H. Eng, "Human Detection by Quadratic Classification on Subspace of Extended Histogram of Gradients," *IEEE Trans. Image Processing*, vol. 23, no. 1, pp. 287-297, Jan, 2014.

J. Ren, X.D. Jiang and J. Yuan, "Noise-Resistant Local Binary Pattern with an Embedded Error-Correction Mechanism," *IEEE Trans. Image Processing*, vol. 22, no. 10, pp. 4049-4060, Oct, 2013.

12 Feature Extraction/Dimension Reduction v. Machine Learning

Problems:

- Poor or even no human knowledge about the effective features.
- Image patterns are so complex that it is impossible to incorporate all different variations into the deterministic computer program.

$$[300\times200] \Rightarrow \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{60000} \end{pmatrix} \Rightarrow \mathbf{f} = \begin{bmatrix} f \\ \vdots \\ f_{60} \end{bmatrix} = \mathbf{\Phi}^T\mathbf{x}$$

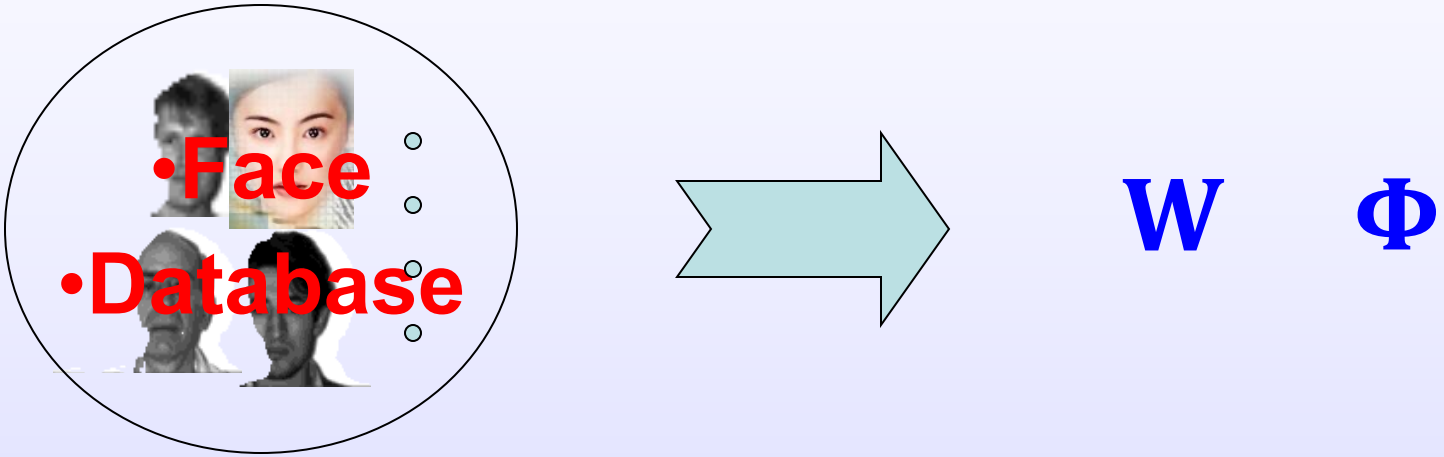
Feature Scaling,
Dimension Reduction
Feature Extraction

Solution: Take all pixels of the whole image as initial features and derive the effective features based on machine learning.

A real application example: J. Ren, X.D. Jiang and J. Yuan, "A Complete and Fully Automated Face Verification System on Mobile Devices," Pattern Recognition, vol. 46, no. 1, pp. 45-56 January 2013.

12 Feature Extraction/Dimension Reduction v. Machine Learning

The transform parameter W or Φ that weight the features and reduce the feature dimensionality is determined by machine (computer) learning (training) from an image database.



- Why is the transform necessary?
- what principles are used to derive W ?
- What are the potential problems?
- How to alleviate these problems?

12 FE/DR v. ML --PCA: Principal Component Analysis

Given a data set of q n-dimensional training samples

$$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q$$

Each sample represented by a column vector is a point in an n-dimensional space.

How to use one point, $\mathbf{x}_0$ to best represent all training data?

i.e. $$\varepsilon^2 = \sum_{i=1}^{q} ||\mathbf{x}_0 - \mathbf{x}_i||^2 = \sum_{i=1}^{q} (\mathbf{x}_0 - \mathbf{x}_i)^T (\mathbf{x}_0 - \mathbf{x}_i) \Rightarrow \text{minimum}$$

- It is very easy to prove that the solution is the sample mean

$$\mathbf{x}_0 = \boldsymbol{\mu} = \frac{1}{q} \sum_{i=1}^{q} \mathbf{x}_i$$

Just for symbolic simplicity, we centralize training samples and define a training data matrix by

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \boldsymbol{\mu}, \qquad \mathbf{X} = [\tilde{\mathbf{x}}_1 \quad \tilde{\mathbf{x}}_2 \quad \ldots \quad \tilde{\mathbf{x}}_q]$$

12 FE/DR v. ML --PCA: Principal Component Analysis

- Now we want to just use one dimension $\phi,\ ||\phi||^2 = \phi^T\phi = 1$ to best represent all samples, $\tilde{\mathbf{x}}_i$.
- The *n*-dimensional data $\tilde{\mathbf{x}}_i$ are reduced to one-dimensional data.

$$a_i = \phi^T \tilde{\mathbf{x}}_i$$

- The best dimension ϕ makes reconstruction error minimum.

$$\varepsilon^2 = \sum_{i=1}^{q} ||\tilde{\mathbf{x}}_i - a_i\phi||^2 \Rightarrow \text{minimum}$$

- It is easy to have

$$\varepsilon^2 = \sum_{i=1}^{q} ||\tilde{\mathbf{x}}_i - a_i\phi||^2 = \sum_{i=1}^{q} (\tilde{\mathbf{x}}_i - a_i\phi)^T(\tilde{\mathbf{x}}_i - a_i\phi)$$

$$= \sum_{i=1}^{q} ||\tilde{\mathbf{x}}_i||^2 - \sum_{i=1}^{q} {a_i}^2 \quad (a_i\phi^T\tilde{\mathbf{x}}_i = {a_i}^2,\ a_i\phi^T a_i\phi = {a_i}^2\phi^T\phi = {a_i}^2)$$

$$= \sum_{i=1}^{q} ||\tilde{\mathbf{x}}_i||^2 - \sum_{i=1}^{q} \phi^T\tilde{\mathbf{x}}_i{\tilde{\mathbf{x}}_i}^T\phi = \sum_{i=1}^{q} ||\tilde{\mathbf{x}}_i||^2 - \phi^T\left(\sum_{i=1}^{q} \tilde{\mathbf{x}}_i{\tilde{\mathbf{x}}_i}^T\right)\phi$$

12 FE/DR v. ML --PCA: Principal Component Analysis

- The sample covariance matrix of all training data,

$$\mathbf{S}^t = \frac{1}{q}\sum_{i=1}^{q}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T = \frac{1}{q}\sum_{i=1}^{q}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i{}^T$$

  is also called the total scatter matrix in the literature.

- To minimize $\varepsilon^2 = \sum_{i=1}^{q}||\tilde{\mathbf{x}}_i||^2 - q\phi^T\mathbf{S}^t\phi$

  is to maximize $\phi^T\mathbf{S}^t\phi$ with the constraint of $||\phi||^2 = \phi^T\phi = 1$

- Use Lagrange optimization method

$$f(\phi,\lambda) = \phi^T\mathbf{S}^t\phi - \lambda(\phi^T\phi - 1) \Rightarrow \max imum$$

$$\frac{\partial f}{\partial \phi} = 2\mathbf{S}^t\phi - 2\lambda\phi = 0$$

$$\mathbf{S}^t\phi = \lambda\phi$$

https://www.youtube.com/watch?v=PFDu9oVAE-g

- The solution is eigenvalue and eigenvector of matrix $\mathbf{S}^t$

12 FE/DR v. ML --PCA: Principal Component Analysis

- We see that if ϕ is the eigenvector of the covariance matrix $\mathbf{S}^t$, $\phi^T \mathbf{S}^t \phi$ is maximum and the reconstruction error is minimum.

- The variance of the data projected onto the dimension spanned by the eigenvector is maximum.

$$\phi^T \mathbf{S}^t \phi = \frac{1}{q} \sum_{i=1}^{q} \phi^T (\mathbf{x}_i - \boldsymbol{\mu}) \left[\phi^T (\mathbf{x}_i - \boldsymbol{\mu})\right]^T = \frac{1}{q} \sum_{i=1}^{q} {a_i}^2$$

- Eigenvalue of a covariance matrix is the variance of the data projected onto the eigenvetor.

$$\because \mathbf{S}^t \phi = \lambda \phi \quad \therefore \phi^T \mathbf{S}^t \phi = \phi^T \lambda \phi = \lambda \phi^T \phi = \lambda$$

12 FE/DR v. ML --PCA: Principal Component Analysis

- Reducing the n-dimensional data $\mathbf{x}_i$ into lower m-dimensional data $\mathbf{y}_i$ by

$$\mathbf{y}_i = \mathbf{\Phi}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

where

$$\mathbf{\Phi} = [\phi_1\ \phi_2 \ldots \phi_m], \qquad m < n$$

consists of m eigenvectors corresponding to the m largest eigenvalues of the total scatter matrix $\mathbf{S}^t$.

- We can reconstruct the original n-dimensional data $\mathbf{x}_i$ using the lower m-dimensional data $\mathbf{y}_i$ using $\hat{\mathbf{x}}_i = \mathbf{\Phi}\mathbf{y}_i + \boldsymbol{\mu}$
- The reconstruction error is minumum.

$$\varepsilon^2 = \sum_{i=1}^{q} ||\mathbf{x}_i - \hat{\mathbf{x}}_i||^2 \Rightarrow \text{minimum}$$

12 FE/DR v. ML --PCA: Principal Component Analysis

- Therefore, the famous principal component analysis PCA transforms the *n*-dimensional *X* into *m*-dimensional *Y* by:

$$\mathbf{y} = \mathbf{\Phi}^T(\mathbf{x} - \boldsymbol{\mu})$$

where $\mathbf{\Phi} = [\phi_1\ \phi_2 \ldots \phi_m], \qquad m < n$

consists of m eigenvectors corresponding to the m largest eigenvalues of the total scatter matrix $\mathbf{S}^t$.

The reconstruction error is: $\Delta = \mathbf{x} - \hat{\mathbf{x}} = \mathbf{x} - \mathbf{\Phi}\mathbf{y} - \boldsymbol{\mu}$

and the mean squared reconstruction error

$$E[||\Delta||^2] = E[\Delta^T\Delta] = \sum_{k=m+1}^{n} \lambda_k$$

被忽略的维度（主成分）有分量

where λ_k are eigenvalues sorted in descending order.

12 FE/DR v. ML --PCA: Principal Component Analysis

- Note that:

$$\mathbf{S}^t = \frac{1}{q}\sum_{i=1}^{q}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T = \frac{1}{q}\sum_{i=1}^{q}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i{}^T = \frac{1}{q}\mathbf{X}\mathbf{X}^T$$

- The rank of the total scatter matrix $\mathbf{S}^t$ is min(q-1,n) at most. Therefore, it has q-1 nonzero eigenvalues at most.
- Thus, PCA with q-1 dimensional $\mathbf{y}$ will fully represent the original n-diemsional data $\mathbf{x}$, q-1≤n, because the reconstruction error is zero.

选择q-1个特征

$$E[||\Delta||^2] = \sum_{k=q}^{n}\lambda_k = 0$$

q: number of training samples
n: sample dimention(variable)

- Obviously, the scatter matrix or covariance matrix is a *nxn* symmetry matrix.

$$\mathbf{S}^t = (\mathbf{S}^t)^T$$

12 FE/DR v. ML -- Eigen-decomposition

- Eigenvalue and eigenvector of a nxn matrix Σ is defined mathematically by:

$$\boldsymbol{\Sigma}\phi_i = \lambda_i\phi_i, \qquad i = 1, 2, \ldots, n$$

- If $\boldsymbol{\Sigma}$ is a symmetry matrix, eigenvectors corresponding to the distinct eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ are orthogonal. Take the unit length of $\phi_1, \phi_2, \ldots, \phi_n$

$$\phi_i^T \phi_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

- Prove it ! (orthogonal+unit length is called orthonormal)

12 FE/DR v. ML -- Eigen-decomposition

We have $\boldsymbol{\Sigma}\phi_k = \lambda_k\phi_k$ and $\boldsymbol{\Sigma}\phi_j = \lambda_j\phi_j, \quad k \neq j$

$\therefore \quad \phi_j{}^T(\boldsymbol{\Sigma}\phi_k) = \phi_j{}^T(\lambda_k\phi_k)$ and $(\boldsymbol{\Sigma}\phi_j)^T\phi_k = (\lambda_j\phi_j)^T\phi_k$

$\therefore \quad \phi_j{}^T\boldsymbol{\Sigma}\phi_k = \lambda_k\phi_j{}^T\phi_k$ and $\phi_j\boldsymbol{\Sigma}^T\phi_k = \lambda_j\phi_j{}^T\phi_k$

$\because \boldsymbol{\Sigma}^T = \boldsymbol{\Sigma}$, we have $\lambda_k\phi_j{}^T\phi_k = \lambda_j\phi_j{}^T\phi_k$

$\therefore (\lambda_k - \lambda_j)\phi_j{}^T\phi_k = 0$

$\because (\lambda_k - \lambda_j) \neq 0$

$\therefore \phi_j{}^T\phi_k = 0$

12 FE/DR v. ML -- Eigen-decomposition

- Let $\mathbf{\Phi}$ be the orthonormal matrix formed by the eigenvectors

$$\mathbf{\Phi} = [\phi_1 \, \phi_2 \, \ldots \, \phi_n]$$

Obviously: $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{I}$ $\quad \therefore \mathbf{\Phi}^{-1} = \mathbf{\Phi}^T, \quad \therefore \mathbf{\Phi}\mathbf{\Phi}^T = \mathbf{I}$

- Let $\mathbf{\Lambda}$ be a diagonal matrix:

$$\mathbf{\Lambda} = diag(\lambda_1 \, \lambda_2 \, \ldots \, \lambda_n) = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

- From $\quad \mathbf{\Sigma}\varphi_i = \lambda_i\varphi_i, \qquad i = 1, 2, \ldots, n$

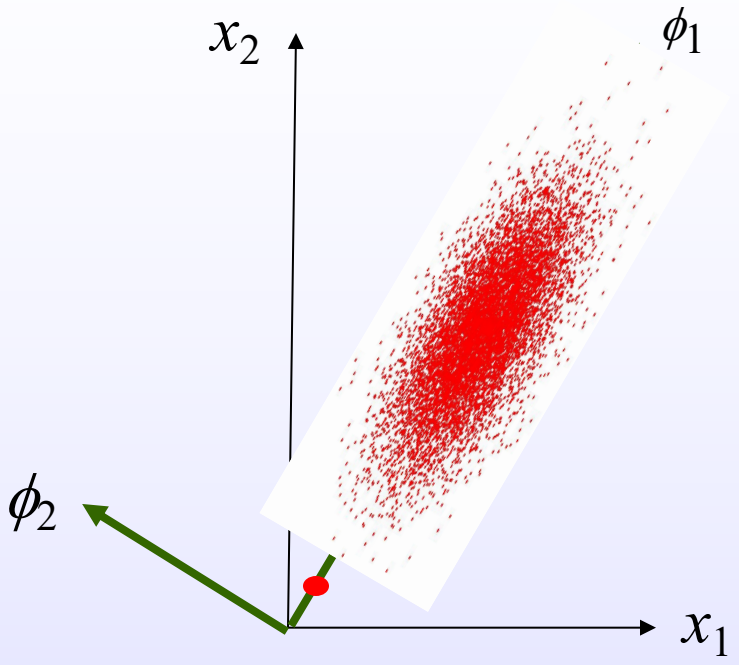
- We have $\quad \mathbf{\Sigma\Phi} = \mathbf{\Phi\Lambda} \qquad \therefore \mathbf{\Sigma} = \mathbf{\Phi\Lambda\Phi}^T$, or $\mathbf{\Lambda} = \mathbf{\Phi}^T\mathbf{\Sigma\Phi}$

12 FE/DR v. ML – understand Eigen-decomposition

$$\boldsymbol{\Sigma}_x = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}$$

$$= [\phi_1 \quad \phi_2] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} [\phi_1 \quad \phi_2]^T$$

$$= \boldsymbol{\Phi \Lambda \Phi}^T$$



$$\boldsymbol{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \boldsymbol{\Phi}^T \boldsymbol{\Sigma}_x \boldsymbol{\Phi}$$

$$= \boldsymbol{\Phi}^T \mathbf{X}\mathbf{X}^T \boldsymbol{\Phi} = \boldsymbol{\Phi}^T \mathbf{X} (\boldsymbol{\Phi}^T \mathbf{X})^T = \mathbf{Y}\mathbf{Y}^T = \boldsymbol{\Sigma}_y = \boldsymbol{\Lambda}$$

where: $\mathbf{Y} = \boldsymbol{\Phi}^T \mathbf{X} = \mathbf{W}^T \mathbf{X}$

Eigen Faces

Input Face







Mean MEF1 MEF2 MEF3 MEF4 MEF5 MEFm

w0 w1 w2 w3 w4 w5 wm

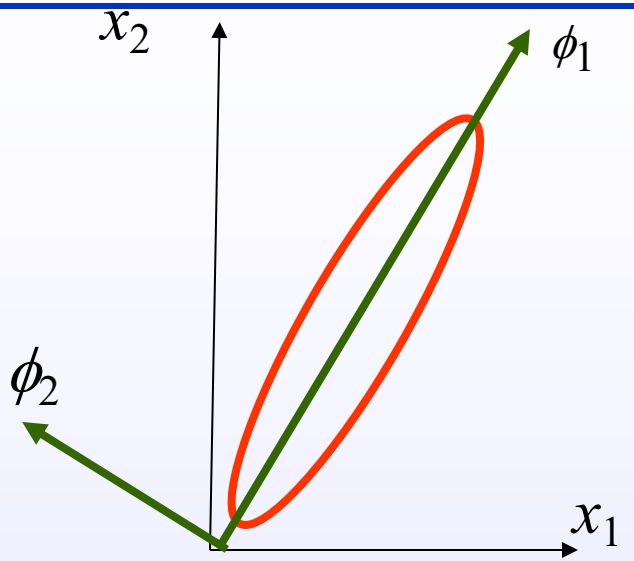
Reconstructed Face (Weighted sum)

12 FE/DR v. ML --PCA: Principal Component Analysis

$$PCA\text{: } \max trace[\mathbf{\Phi}^T \mathbf{S}^t \mathbf{\Phi}] = \sum_{k=1}^{m} \lambda_k^t$$

$\mathbf{\Phi}$: $[n \times m]$, e.g. $=[60000 \times 80]$

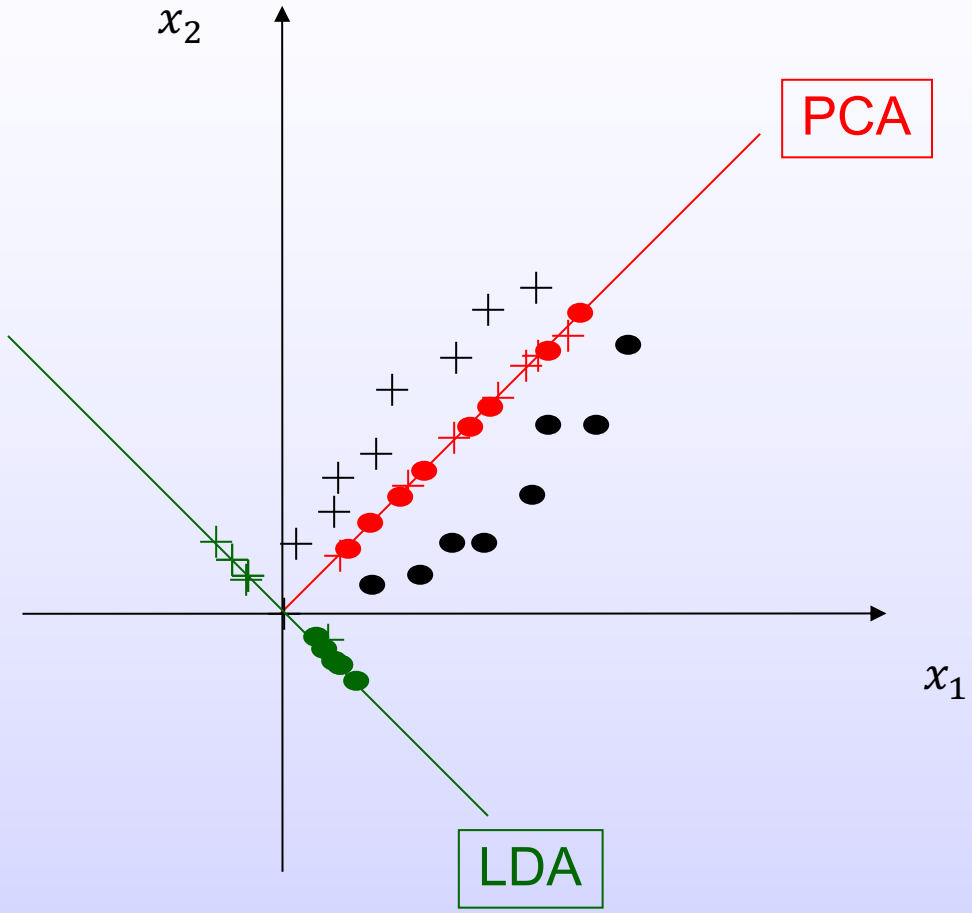
$$\mathbf{y} = \mathbf{\Phi}^T(\mathbf{x} - \boldsymbol{\mu})$$



- PCA is an unsupervised method that minimizes the reconstruction error or maximize the variation (information carried by the data) in the reduced sub-space. This dimension reduction will reduce the computational complexity of the subsequent processing.
- However, any information lost may reduces the accuracy of the subsequent processing. Further more, the lost information, though less important for data representation, could be critical for discriminating the data class membership.

12 FE/DR v. ML --LDA: Linear Discriminant Analysis

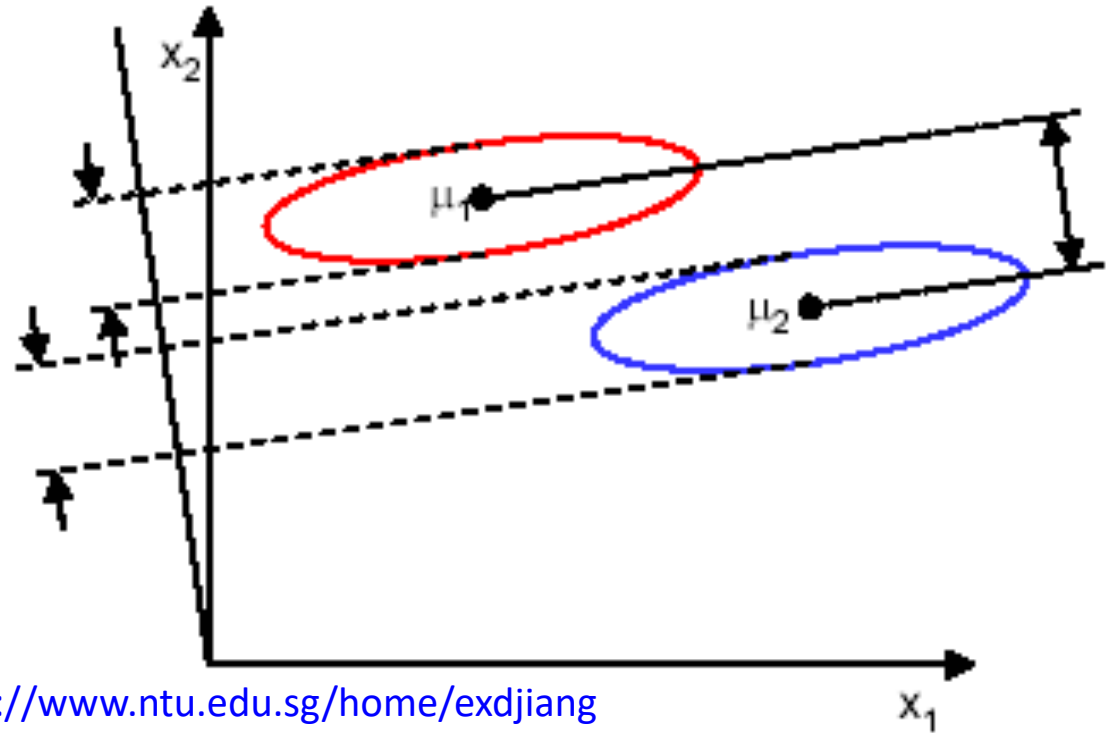
- Problem of PCA in classification?



12 FE/DR v. ML --LDA: Linear Discriminant Analysis

- Desired? properties to determine the projection vector for classification :
  - Maximize separation between projected class means
  - Minimize projected within-class scatter (variance)

$$\mathbf{y} = \mathbf{\Phi}^T(\mathbf{x} - \boldsymbol{\mu})$$



exdjiang@ntu.edu.sg

12 FE/DR v. ML --LDA: Linear Discriminant Analysis

- Given q n-dimensional training samples of c classes

$$\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x}_q$$

- The number of training samples of class ω_j is q_j, j = 1, 2,… c.
- The covariance matrix of class ω_j is computed as

$$\boldsymbol{\Sigma}_j = \frac{1}{q_j}\sum_{X_i \in \omega_j} (\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^T, \text{ where } \boldsymbol{\mu}_j = \frac{1}{q_j}\sum_{X_i \in \omega_j} \mathbf{x}_i$$

- The within class scatter matrix of the c classes is defined as

$$\mathbf{S}^w = \sum_{j=1}^{c} \frac{q_j}{q} \boldsymbol{\Sigma}_j$$

- The between class scatter matrix of c classes is defined as

$$\mathbf{S}^b = \sum_{j=1}^{c} \frac{q_j}{q} (\boldsymbol{\mu}_j - \boldsymbol{\mu})(\boldsymbol{\mu}_j - \boldsymbol{\mu})^T, \text{ obviously: } \boldsymbol{\mu} = \frac{1}{q}\sum_{i=1}^{q} \mathbf{x}_i = \sum_{j=1}^{c} \frac{q_j}{q} \boldsymbol{\mu}_j$$

12 FE/DR v. ML --LDA: Linear Discriminant Analysis

- Instead of

$$PCA\text{: } \max trace[\mathbf{\Phi}^T \mathbf{S}^t \mathbf{\Phi}] = \sum_{k=1}^{m} \lambda_k^t$$

$$LDA\text{: } \min trace[\mathbf{\Phi}^T \mathbf{S}^w \mathbf{\Phi}] \ \& \ \max trace[\mathbf{\Phi}^T \mathbf{S}^b \mathbf{\Phi}]$$

$$\Rightarrow LDA\text{: } \max trace\left[\mathbf{\Phi}^T \mathbf{S}^{w^{-1}} \mathbf{S}^b \mathbf{\Phi}\right] = \sum_{k=1}^{m} \lambda_k^{b/w}$$

- Note that:

$trace(\mathbf{S}^t) \Rightarrow$ total variation amount of all samples
$trace(\mathbf{S}^w) \Rightarrow$ total variation amount of samples within classes
$trace(\mathbf{S}^b) \Rightarrow$ total variation amount of samples between classes

- It is easy to prove: $\mathbf{S}^t = \mathbf{S}^w + \mathbf{S}^b$

12 FE/DR v. ML --LDA: Linear Discriminant Analysis

- As pattern recognition is not to represent the original data, but to discriminate the class membership of the data, obviously, LDA extracts much more discriminative features than PCA. Therefore, the vast majority of researchers prefer LDA to PCA for feature extraction and dimensionality reduction for pattern recognition.

$$LDA:\ \max t\,race\left[\mathbf{\Phi}^T\mathbf{S}^{w^{-1}}\mathbf{S}^b\mathbf{\Phi}\right]=\sum_{k=1}^{m}\lambda_k^{b/w}$$

- The rank of $\mathbf{S}^w$ is $min(q\text{-}c,n)$ at most. For many applications $q\text{-}c<<n$. So, we have problem because $\mathbf{S}^w$ is singular and its inverse does not exist.
- Thousands of research papers proposed various LDA variants trying to solve this problem!