图像处理与识别

——Part 9 图像识别(I)

主讲：张磊

Definition

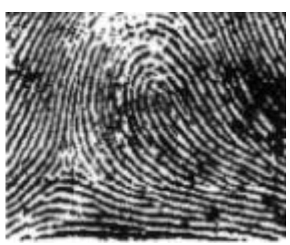
- What is statistical recognition?

What is pattern? 模式是被人类感知和理解的实体，无确切定义。

There are a number of patterns.

Some examples:


Palmprint


Fingerprint

QR code

License plate no.


Face

Definition

- What is statistical recognition?

What is recognition? 识别是指模式（patterns）被人类判定（discriminate）为某种熟悉或已知的类别。

Two types of recognition:

Classification(分类):

将某种未知（unknown）的模式赋予已知(known)的类（label）。

Clustering(聚类):

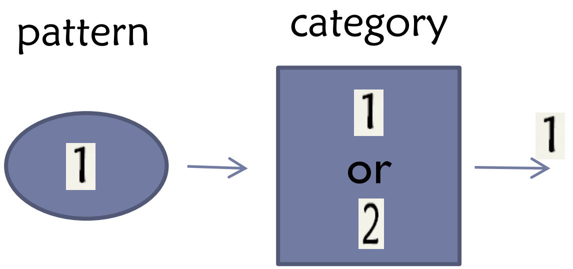
将未知（unknown）类别的一组模式，根据某种相似性度量，对相似模式进行聚集。

Definition

- Examples

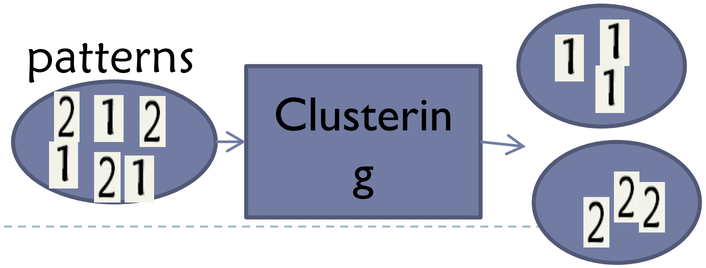
Classification(分类):

将某种未知（unknown）的模式赋予已知(known)的类（label）。



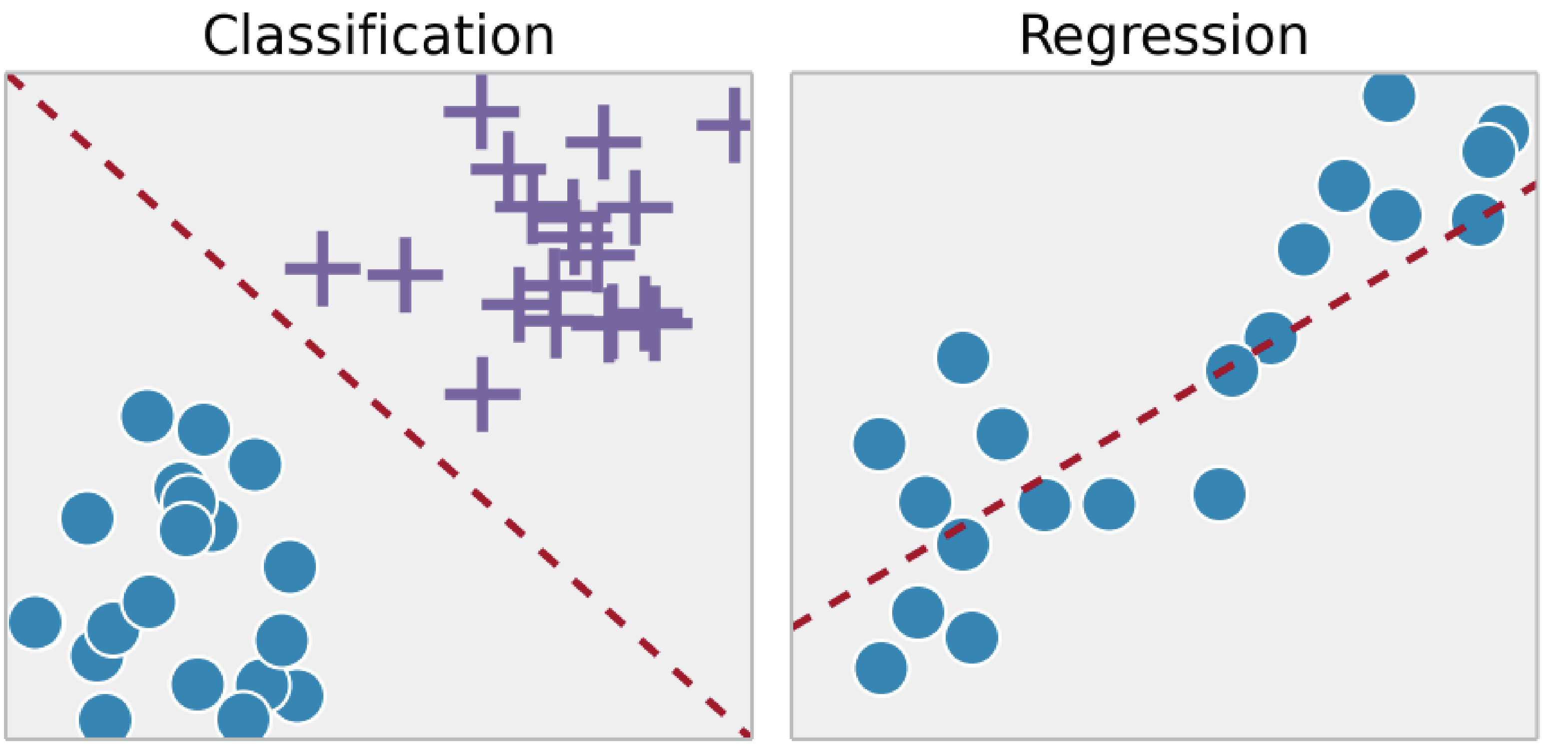
Clustering(聚类):

将未知（unknown）类别的一组模式，根据某种相似性度量，对相似模式进行聚集。



Definition

- Classification vs. Regression

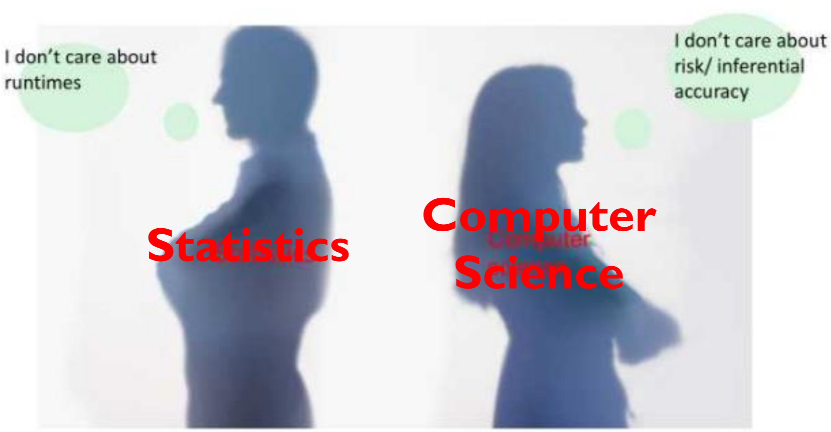


Definition

- What is statistical recognition?

统计识别（或模式识别）是指通过处理和分析某种事物或现象带来的信息，从而实现对事物或现象进行**解析**、**识别**、**分类**的过程。

An important part in Artificial Intelligence (AI)!



Definition

- What is statistical recognition?

从实现上，统计识别（或模式识别）的过程是机器对已知或未知事物进行感知（perceive）、分析（analysis）和识别（recognition）的过程。

It also gives intelligence (but not wisdom) to machine (AI)!

机器智能具有模仿人的某种行为的能力，但不同于智慧（自我推理）。

Definition

- What is statistical recognition?

感知（perceive）：Information observation and acquisition from others (新的事物)

分析（analysis）：Information processing and intelligent discrimination

识别（recognition）：Make a decision on the observed information for classification (label prediction)

统计识别是从大量先验数据中，寻找可以分类的模型或统计规律，通过智能算法实现新样例的归类。

Definition

- why statistical recognition?

For humans, recognition is an easy task:

Recognize a handwritten digit, recognize a face, discriminate an object/image,

Discriminate an odorant, spoken recognition,···

However, for machines, recognition is never easy.

Therefore, machine recognition like humans is challenging.

Definition

- Basic terms of statistical recognition?

样例（samples or instances）：

Patterns that we would like to classify (分类对象).

特征（feature）：

Attributes that characterize the properties of samples (表征样本的属性).

训练集（training）：

The data used for learning the classification model parameters (hypothesis，假设获取)

Definition

- Basic terms of statistical recognition?

测试集（testing）：

The data for evaluating the obtained hypothesis（假设检验）.

模型（model）：

Mathematical descriptions for induction and inference (用于归纳和推理的数学描述).

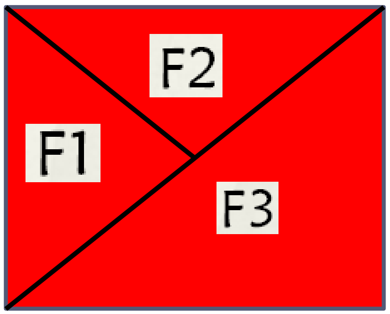
特征空间（feature space）：

The space spanned by features(特征张成的空间)

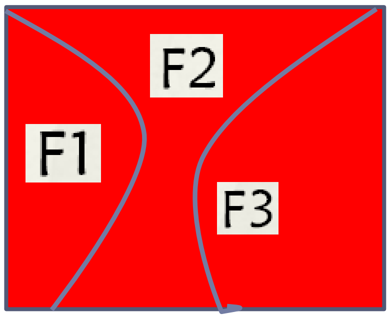
Definition

- Basic terms of statistical recognition?

决策边界（decision boundary）：Three categories F1,F2,F3



Linear decision



quadratic decision

决策边界即为某个线性、二次或非线性的决策函数表示

Definition

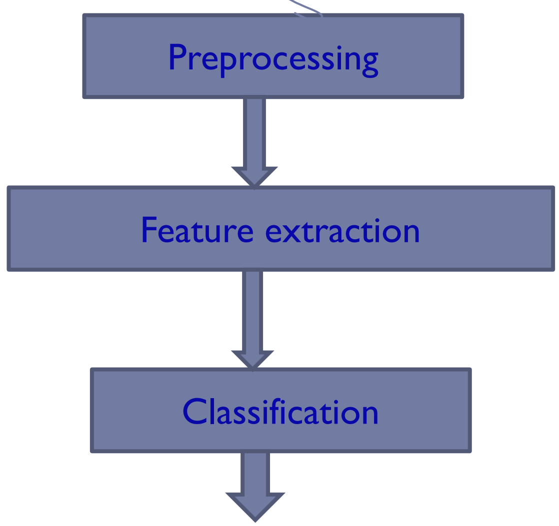
- How do statistical recognition?

For Example:

Fish species recognition of see bas and salmon

(鲈鱼和三文鱼)

Three basic steps for recognition



鲈鱼 or三文鱼?

Definition

Step I: Preprocessing (预处理)

Goal: Preprocess the image captured by the camera, such that subsequent operations could be simplified without losing relevant information

Routine image processing → 常规图像处理

- Adjust the level of illumination
- Denoising
- Enhance the level of contrast

segmentation → 图像分割

- Isolate different fishes from one another
- Isolate fishes from the background

……

Definition

Step II: Feature Extraction (特征抽取)

Goal: Extract features (with good distinguishing ability) from the preprocessed image to be used for subsequent classification

Sea bass is usually longer than a salmon → "length" could be a good candidate for features

特征属性

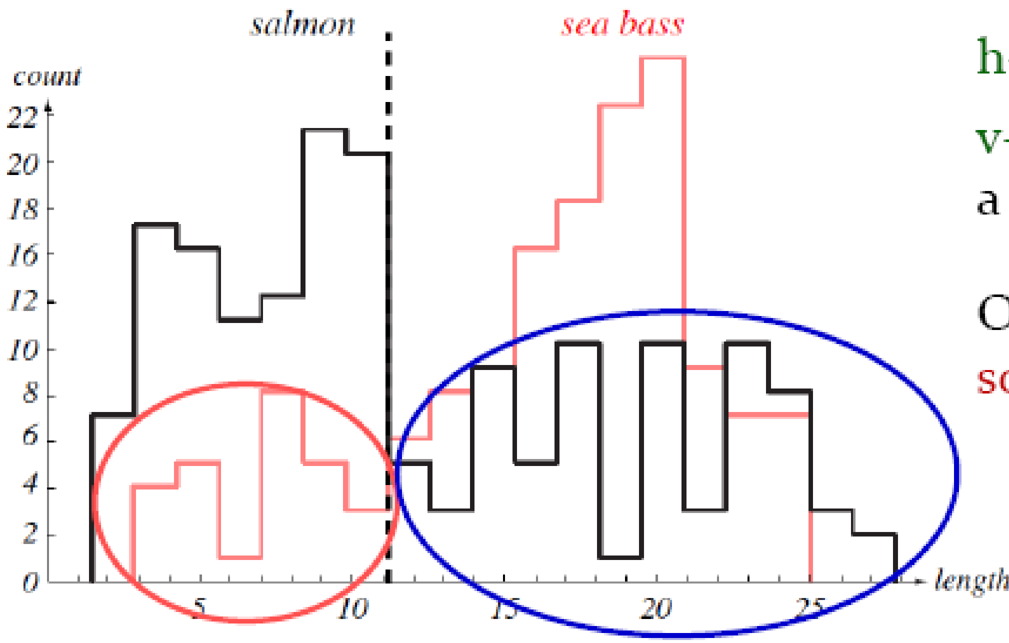
Sea bass is usually brighter than a salmon → "lightness" of *fish scales* could be another good candidate for features

……

Definition

Step III: Classification (分类)

Goal: To distinguish different types of objects (in this case, *sea bass* vs. *salmon*) based on the extracted features



histogram for *length*

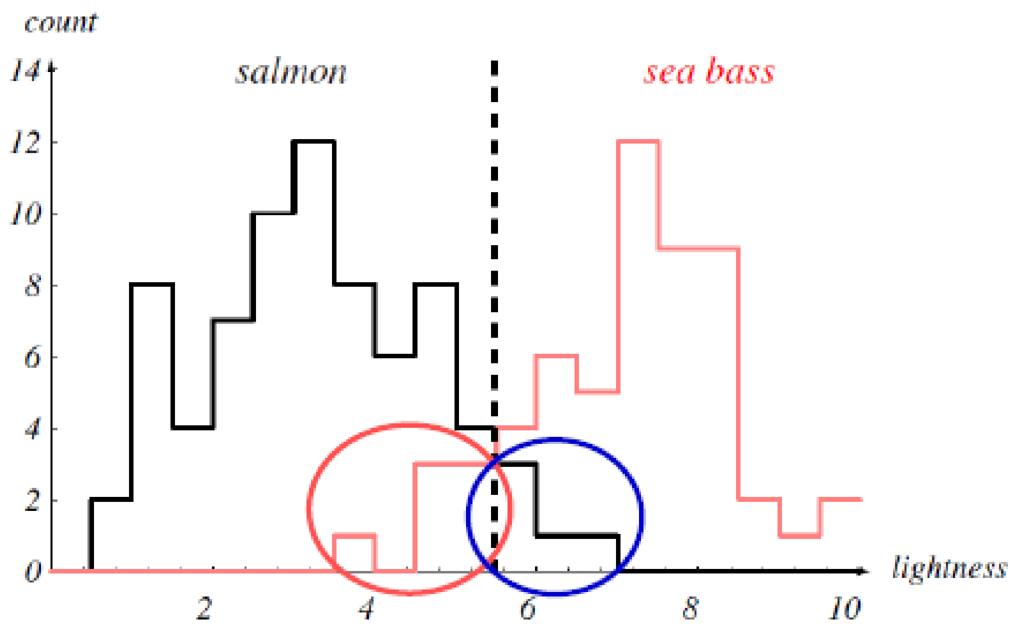
h-axis: length of fish

v-axis: number of fishes with a certain length

On average, sea bass is somewhat longer than salmon

Too much overlaps → poor separation with the length feature

Definition



histogram for *lightness*

h-axis: lightness of fish scales

v-axis: number of fishes with a certain lightness

On average, sea bass is much brighter than salmon

Less overlaps → better separation with the lightness feature, but still a bit unsatisfactory

What if no other single feature yields better performance? → Use more features at the same time!

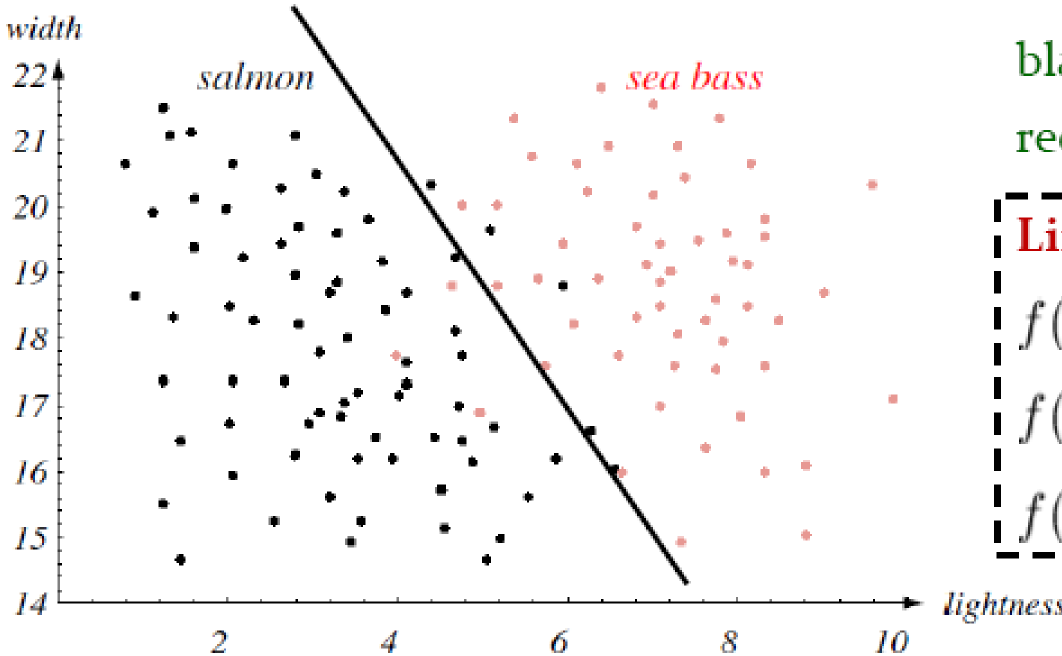
Definition

Using two features simultaneously

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

x_1 : fish width

x_2 : fish lightness



scatter plot for the feature vectors

black dots: salmon samples

red dots: sea bass samples

Linear decision boundary:

$$f(x_1, x_2) = a \cdot x_1 + b \cdot x_2 + c$$

$$f(x_1, x_2) > 0 \implies \text{sea bass}$$

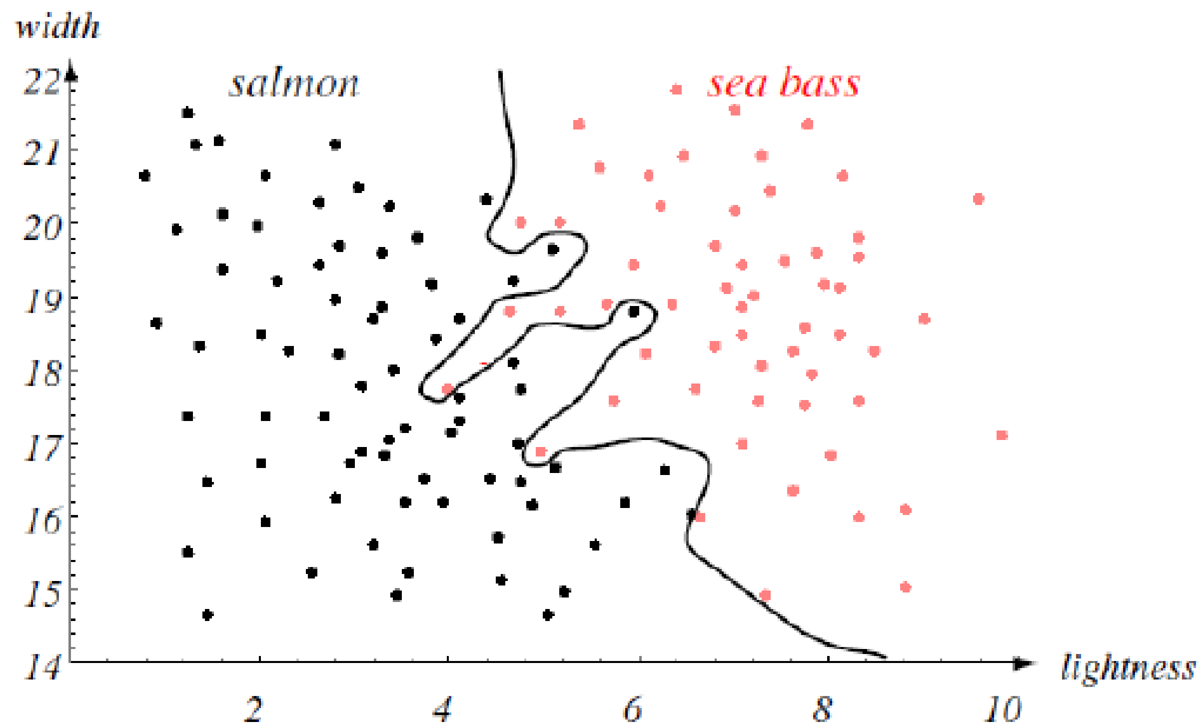
$$f(x_1, x_2) \leq 0 \implies \text{salmon}$$

Much better than single feature

Definition

Linear decision boundary:

Complex decision boundary



All the training samples (i.e. known patterns) have been separated perfectly

Can we truly feel satisfied?

Definition

Generalization

[泛化能力/推广能力]

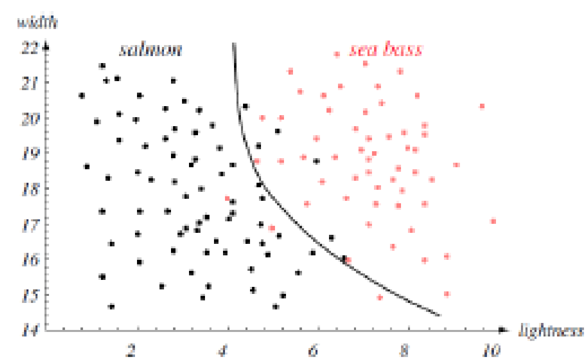
The ultimate goal!

The central aim of designing a classifier is to make correct decisions when presented with *novel* (*unseen/test*) patterns, not on training patterns whose labels are already known

e.g. it's useless to get 100% accuracy when answering homework questions while get low accuracy when answering exam questions

Performance on the training set → Tradeoff

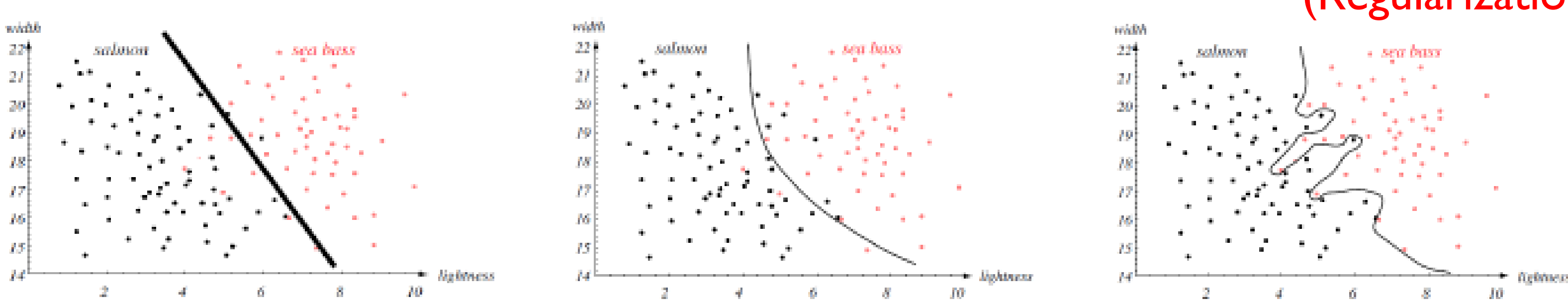
Simplicity of the classifier → Tradeoff



Definition

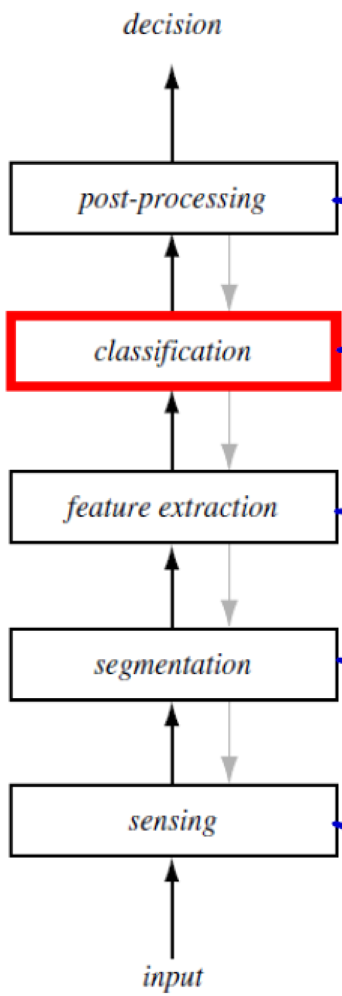
- We can get perfect classification performance on the training data by choosing complex models
  - Complex models are tuned to the particular training samples, rather than the characteristics of the true model
- Models overly complex than necessary lead to overfitting
  - Good performance on the training data, but poor performance on novel data
- How can we find principled ways to obtain best complexity?

(Regularization)



Pattern Recognition System

In addition to the usual "bottom-up" flow of data, some systems also employ feedback from higher levels back down to lower levels (gray arrows)



A post processor decide on the appropriate action based on the classification

A classifier uses extracted features to assign the sensed object to a category

A feature extractor measures object properties that are useful for classification

A segmentor isolates sensed objects from the background or from other object

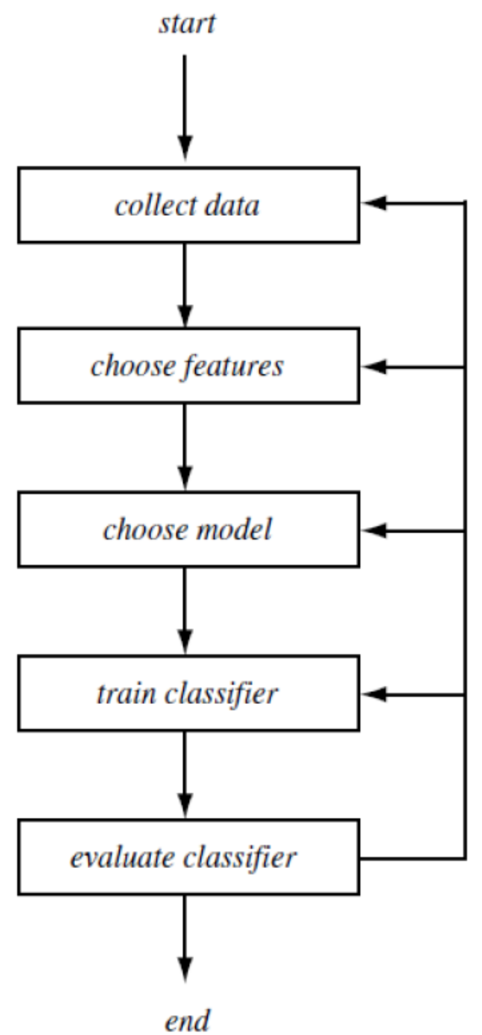
A sensor converts physical inputs (e.g. images, sounds) into digital signal data

Pattern Recognition System

The design of a PR system usually entails a number of different activities, such as *data collection, feature choice, model choice, classifier training, classifier evaluation.*

- Data collection accounts for a large part of the cost of developing a PR system
- Feature choice and model choice are highly domain-dependent, where *prior knowledge* (先验知识) plays very important role

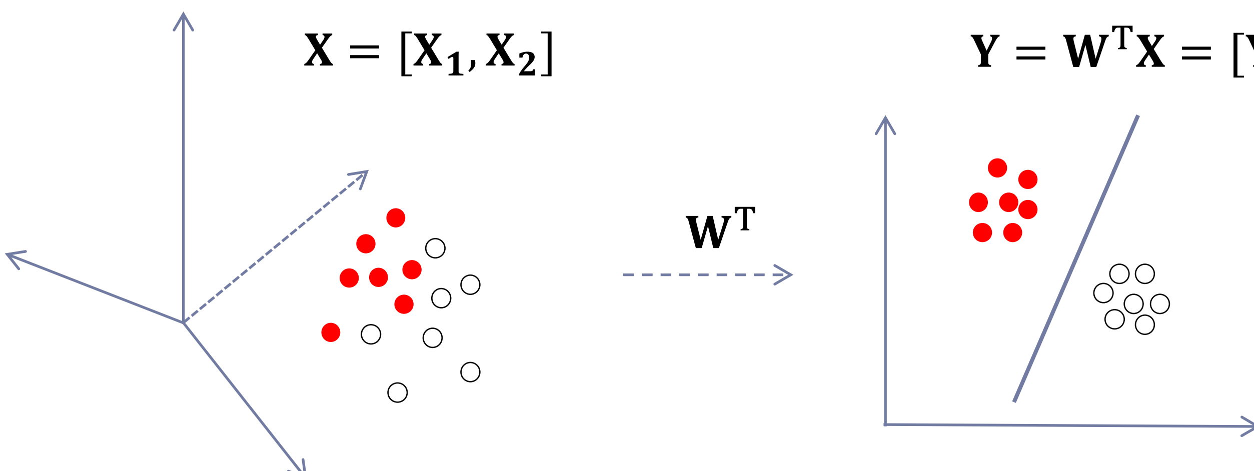
  e.g.: lightness might be a good feature for distinguishing sea bass and salmon; linear model might be preferred than nonlinear ones
- Various activities may be repeated in order to obtain satisfactory results



A Toy Example: LDA (线性判别分析)

例：设有两类样本$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \Re^{D\times N}$,存在一个线性变换$\mathbf{W} \in \Re^{D\times d}$，使得经过线性变换后两类数据集线性可分。

任务：采用LDA模型计算$\mathbf{W} = \arg\max_{\mathbf{W}} \frac{\mathbf{W}^{\mathrm{T}}\mathbf{S}_{\mathbf{B}}\mathbf{W}}{\mathbf{W}^{\mathrm{T}}\mathbf{S}_{\mathbf{W}}\mathbf{W}}$；



(a) 高维空间　　(b) 低维空间

Decision based Recognition

- Decision function

Let $\mathbf{x} = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$ represent a pattern vector. For a problem of C classes, $w_1, w_2, \ldots, w_C$, the basic problem of decision theory is to find C decision functions $d_1(\mathbf{x})$, $d_2(\mathbf{x}), \cdots, d_C(\mathbf{x})$.

$$d_i(\mathbf{x}) > d_j(\mathbf{x}), \text{if } \mathbf{x} \text{ belongs to } w_i$$

The pattern class with respect to the maximum d(x) is the decision class of pattern **x**.

Decision based Recognition

- Decision function

Generally, we use a single function as the decision bound of two classes,

$$d_{ij}(\mathbf{x}) = d_i(\mathbf{x}) - d_j(\mathbf{x})$$

Decision process:

$$d_{ij}(\mathbf{x}) > 0, \text{if } \mathbf{x} \text{ belongs to } w_i$$
$$d_{ij}(\mathbf{x}) < 0, \text{if } \mathbf{x} \text{ belongs to } w_j$$

The decision boundary can be represented as

$$d_{ij}(\mathbf{x}) = 0$$

Template Matching

- Minimum distance classifier

Two Elements for Recognition based on Matching:

- Prototype pattern vector

$$\mathbf{m}_i = \frac{1}{N_i} \sum_{\mathbf{x} \in w_i} \mathbf{x}_i \, , i = 1,2, \dots , C$$

- Distance metric

$$D_i(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_i\|^2, i = 1,2, \dots , C$$

- Distance minimum criterion

$$w(x) = \underset{i}{\operatorname{argmin}} \, D_i(\mathbf{x}), i = 1,2, \dots , C$$

Template Matching

- What is the relation between minimum distance classifier and decision function?

Recall the distance

$$D_i(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_i\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x} - 2\mathbf{x}^{\mathrm{T}}\mathbf{m}_i + \mathbf{m}_i^{\mathrm{T}}\mathbf{m}_i$$

Select the minimum distance is equivalent to select the maximum value of the following decision function

$$d_i(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{m}_i - \frac{1}{2}\mathbf{m}_i^{\mathrm{T}}\mathbf{m}_i$$

The class with respect to the maximum $d_i(\mathbf{x})$ is the class of **x**.

Template Matching

- Decision bound of the minimum distance classifier

$$d_{ij}(\mathbf{x}) = d_i(\mathbf{x}) - d_j(\mathbf{x})$$
$$= \mathbf{x}^{\mathrm{T}}\mathbf{m}_i - \frac{1}{2}\mathbf{m}_i^{\mathrm{T}}\mathbf{m}_i - \left(\mathbf{x}^{\mathrm{T}}\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^{\mathrm{T}}\mathbf{m}_j\right)$$
$$= \mathbf{x}^{\mathrm{T}}(\mathbf{m}_i - \mathbf{m}_j) - \frac{1}{2}(\mathbf{m}_i - \mathbf{m}_j)^{\mathrm{T}}(\mathbf{m}_i + \mathbf{m}_j)^{\mathrm{T}}$$

The decision bound

Let $d_{ij}(\mathbf{x}) = 0$, i.e.

$$\mathbf{x}^{\mathrm{T}}(\mathbf{m}_i - \mathbf{m}_j) - \frac{1}{2}(\mathbf{m}_i - \mathbf{m}_j)^{\mathrm{T}}(\mathbf{m}_i + \mathbf{m}_j)^{\mathrm{T}} = 0$$

Template Matching

- Example of minimum distance classifier

There are two classes w_1, w_2. Their mean vector are

$$\mathbf{m}_1 = [4.3, 1.3]^{\mathrm{T}}, \mathbf{m}_2 = [1.5, 0.3]^{\mathrm{T}}$$

Given a pattern vector $\mathbf{x} = [x_1, x_2]^{\mathrm{T}}$, then how to determine its decision function and decision bound?

Decision function:

$$d_1(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{m}_1 - \frac{1}{2}\mathbf{m}_1^{\mathrm{T}}\mathbf{m}_1 = 4.3x_1 + 1.3x_2 - 10.1$$

$$d_2(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{m}_2 - \frac{1}{2}\mathbf{m}_2^{\mathrm{T}}\mathbf{m}_2 = 1.5x_1 + 0.3x_2 - 1.17$$

The decision bound

Let $d_{12}(\mathbf{x}) = 0$, i.e.

$$d_{12}(\mathbf{x}) = d_1(\mathbf{x}) - d_2(\mathbf{x})$$

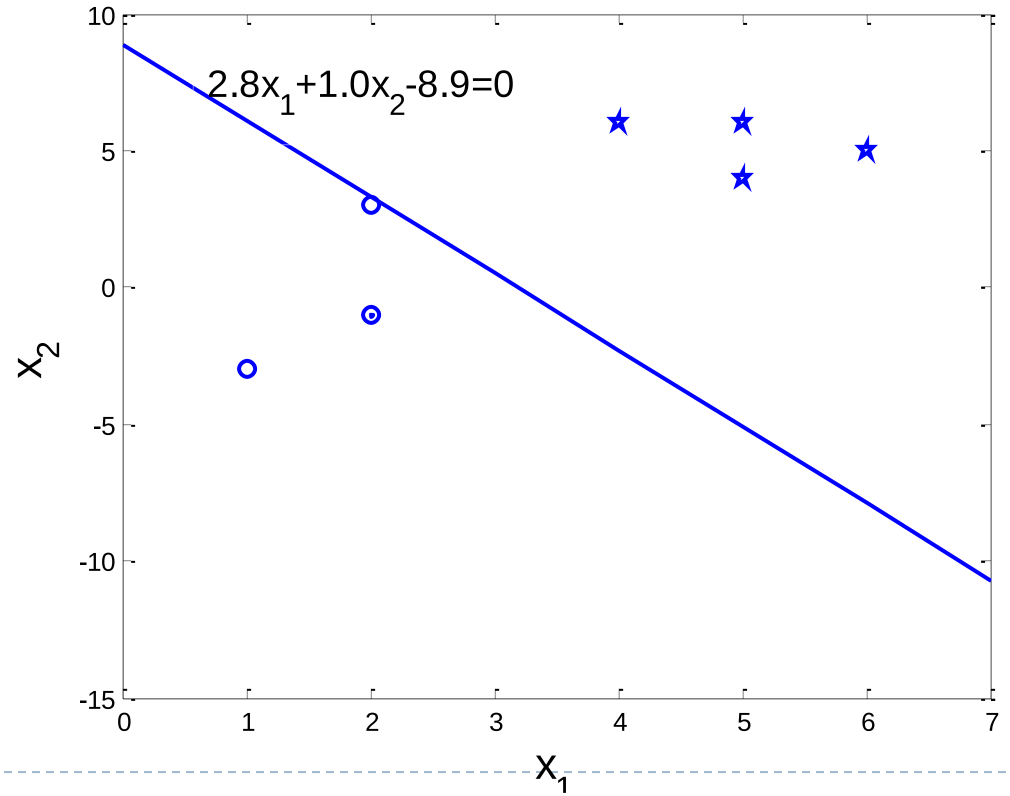
$$= \mathbf{x}^{\mathrm{T}}(\mathbf{m}_1 - \mathbf{m}_2) - \frac{1}{2}(\mathbf{m}_1 - \mathbf{m}_2)^{\mathrm{T}}(\mathbf{m}_1 + \mathbf{m}_2)^{\mathrm{T}}$$

$$= 2.8x_1 + 1.0x_2 - 8.9 = 0$$

Template Matching

- Example of minimum distance classifier

Given some patterns $\mathbf{x} = [2, -1]^T$, $\mathbf{x} = [1, -3]^T$, $\mathbf{x} = [5,6]^T$, $\mathbf{x} = [6,5]^T$, $\mathbf{x} = [5,4]^T$, classify them.



The Optimal Statistical Classifier

- Optimal statistical classifier

A recognition method based on probability.

Given a pattern x, the probability of x belongs to w_i is

$$p(w_i|\mathbf{x})$$

Loss: L_{ij}; if the pattern comes from w_i, but classified as w_j.

Average loss: a pattern x can be classified as any class of the total C classes. The average loss $r_j(x)$ that x is (wrongly) classified as w_j is represented as

$$r_j(x) = \sum_{k=1}^{C} L_{kj} p(w_k|\mathbf{x}), \textit{conditional average risk}$$

The Optimal Statistical Classifier

- Optimal statistical classifier

Bayesian formula

$$p(A|B) = \frac{p(A)p(B|A)}{p(B)}$$

There is

$$r_j(\mathbf{x}) = \sum_{k=1}^{C} L_{kj} p(w_k|\mathbf{x}) = \sum_{k=1}^{C} L_{kj} \frac{p(\mathbf{x}|w_k)P(w_k)}{p(\mathbf{x})}$$

$\forall w_k, k = 1, \dots, C, p(\mathbf{x})\ is\ the\ same$, therefore,

$$r_j(\mathbf{x}) = \sum_{k=1}^{C} L_{kj} p(\mathbf{x}|w_k)P(w_k)$$

The Optimal Statistical Classifier

- Optimal statistical classifier

$$r_j(\mathbf{x}) = \sum_{k=1}^{C} L_{kj} p(\mathbf{x}|w_k) P(w_k)$$

where $p(\mathbf{x}|w_k)$ is the p.d.f. of **x** from class w_k, $P(w_k)$ is the probability that class w_k happens.

$r_j(\mathbf{x})$ is the average loss that x is classified as w_j.

If there is

$$r_i(\mathbf{x}) = \sum_{k=1}^{C} L_{ki} p(\mathbf{x}|w_k) P(w_k) < r_j(\mathbf{x}) = \sum_{k=1}^{C} L_{kj} p(\mathbf{x}|w_k) P(w_k)$$

Then x belongs to w_i

The Optimal Statistical Classifier

- Optimal statistical classifier

Generally, the loss L_{ij} is imposed to be 0 if correctly classified, and 1, otherwise. (0-1损失). Then,

$$L_{ij} = 1 - \delta_{ij}$$

$$\delta_{ij} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}$$

$$r_j(\mathbf{x}) = \sum_{k=1}^{C} L_{kj} p(\mathbf{x}|w_k) P(w_k) = \sum_{k=1}^{C} (1 - \delta_{kj}) p(\mathbf{x}|w_k) P(w_k)$$
$$= P(\mathbf{x}) - p(\mathbf{x}|w_j) P(w_j)$$

We know that x belongs to w_i, if $r_i(\mathbf{x}) < r_j(\mathbf{x})$, there is

$$P(\mathbf{x}) - p(\mathbf{x}|w_i) P(w_i) < P(\mathbf{x}) - p(\mathbf{x}|w_j) P(w_j)$$

$$\Downarrow$$

$$p(\mathbf{x}|w_i) P(w_i) > p(\mathbf{x}|w_j) P(w_j)$$

The Optimal Statistical Classifier

The decision function of x

$$d_j(x) = p(\mathbf{x}|w_j)P(w_j)$$

For a Bayesian classifier, it computes the decision function.

if $d_i(x) > d_j(x)$, then **x** belongs to w_i

The p.d.f. $p(\mathbf{x}|w_j)$ of x belongs to w_j and the probability $P(w_j)$ should be known.

$$P(w_j) = \frac{1}{C}$$

$p(\mathbf{x}|w_j)$ is often supposed to be Gaussian function.

Bayesian classifier shows the minimum loss if the assumption is approaching the actual case.

The Optimal Statistical Classifier

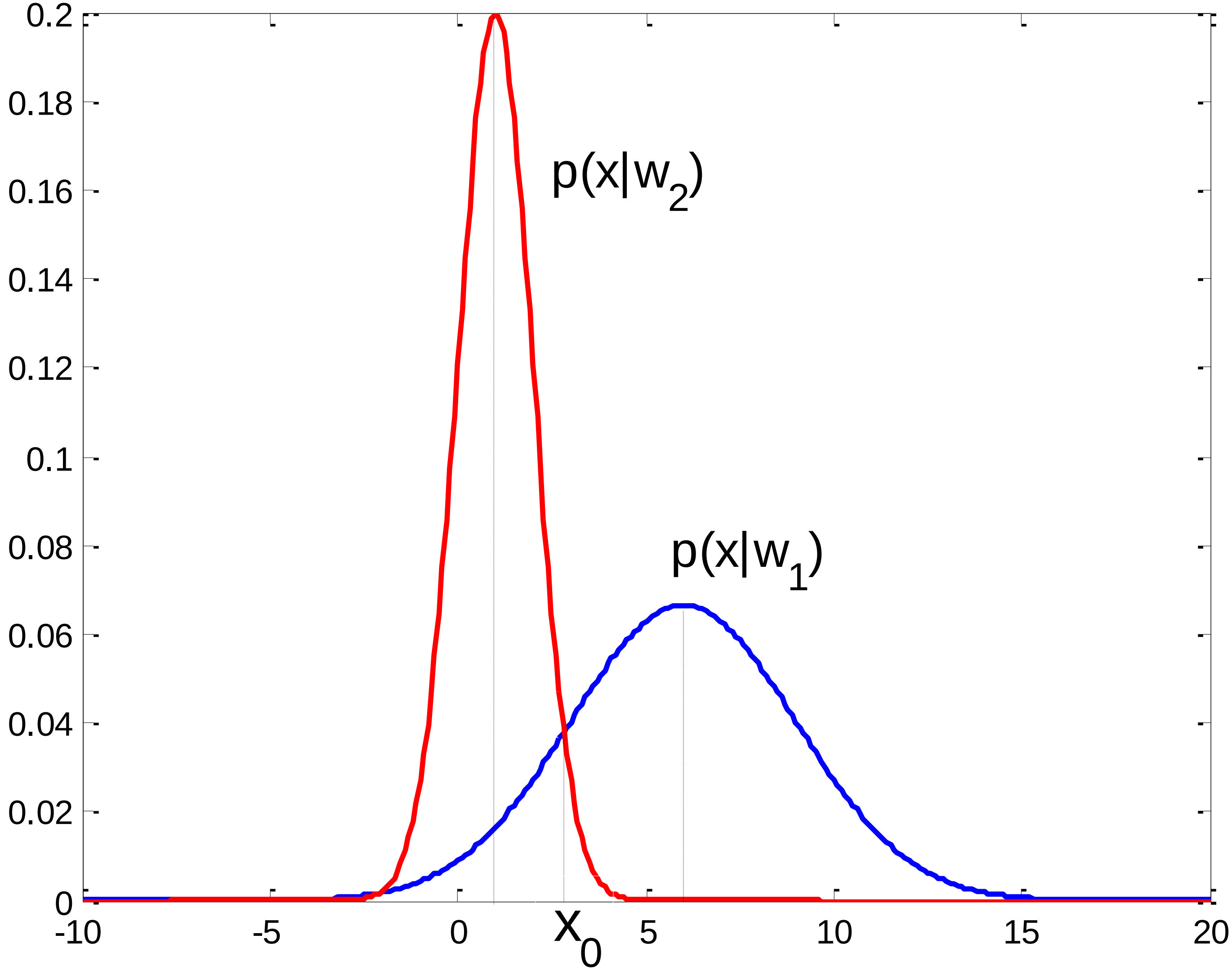
- Example: one dimension (Gaussian)

Suppose there are 2 classes (w_1, w_2) based on Gaussian distribution, with mean value m_1 and m_2, standard variance σ_1 and σ_2.

Bayesian decision function

$$d_j(x) = p(x|w_j)P(w_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x-m_j)^2}{2\sigma_j^2}} P(w_j), j = 1,2$$

The Optimal Statistical Classifier



The Optimal Statistical Classifier

- Example: N-dimension (Gaussian)

Suppose there are C classes $(w_1, w_2, \dots, w_c)$ based on Gaussian distribution, with mean vector

$$\mathbf{m}_j = E_j\{\mathbf{x}\} = \frac{1}{N_j}\sum_{\mathbf{x}\in w_j}\mathbf{x}$$

Covariance matrix

$$\mathbf{\Sigma}_j = E_j\{(\mathbf{x}-\mathbf{m}_j)(\mathbf{x}-\mathbf{m}_j)^{\mathrm{T}}\} = \frac{1}{N_j}\sum_{\mathbf{x}\in w_j}\mathbf{x}\mathbf{x}^{\mathrm{T}} - \mathbf{m}_j\mathbf{m}_j^{\mathrm{T}}$$

Bayesian decision function

$$d_j(\mathbf{x}) = p(\mathbf{x}|w_j)P(w_j) = \frac{1}{(2\pi)^{n/2}\left|\mathbf{\Sigma}_j\right|^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\mathbf{m}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\mathbf{m}_j)}P(w_j)$$

The Optimal Statistical Classifier

- Example: N-dimension (Gaussian)

Bayesian decision function

$$d_j(\mathbf{x}) = p(\mathbf{x}|w_j)P(w_j)$$

$$= \frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}_j|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{m}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\mathbf{m}_j)} P(w_j)$$

Due to the exponent form of Gaussian function

The natural logarithm is often used

$$\ln d_j(\mathbf{x}) = \ln p(\mathbf{x}|w_j)P(w_j) = \ln p(\mathbf{x}|w_j) + \ln P(w_j)$$

$$= -\frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|\mathbf{\Sigma}_j| - \frac{1}{2}\left[(\mathbf{x}-\mathbf{m}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\mathbf{m}_j)\right] + \ln P(w_j)$$

The Optimal Statistical Classifier

- Example: N-dimension (Gaussian)

Bayesian decision function

$$d_j(\mathbf{x}) = \ln P(w_j) - \frac{1}{2}\ln|\mathbf{\Sigma}_j| - \frac{1}{2}\left[(\mathbf{x}-\mathbf{m}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\mathbf{m}_j)\right]$$

If the covariance matrix $\mathbf{\Sigma}_j$ is the same for all classes,

$$d_j(\mathbf{x}) = \ln P(w_j) - \mathbf{x}^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}\mathbf{m}_j$$

If covariance matrix $\mathbf{\Sigma}_j = \mathbf{I}$, and $P(w_j) = \frac{1}{c}$,

$$d_j(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^{\mathrm{T}}\mathbf{m}_j$$

贝叶斯意义上的

minimum distance classifier

Linear Decision Function

For a n-dim problem, $\mathbf{x} = [x_1, x_2, \dots, x_n]^{\mathrm{T}}$, the linear decision function is

$$d(x) = \sum_{i=1}^{n} w_i x_i + b$$

where w_i is the weight imposed on x_i.

$d(\mathbf{x})$is the final output with a summation (activate).

$$d(x) > 0, \mathbf{x} \in +1$$
$$d(x) < 0, \mathbf{x} \in -1$$

The decision boundary is

$$d(x) = 0 \rightarrow w_1 x_1 + w_2 x_2 + w_3 x_3 + \cdots + w_n x_n + b = 0$$

Linear Decision Function

$$w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b = 0$$

is a hyperplane in n-dimensional pattern space.

The final b is called bias, which is proportional to the orthogonal distance between the origin O and the hyperplane.

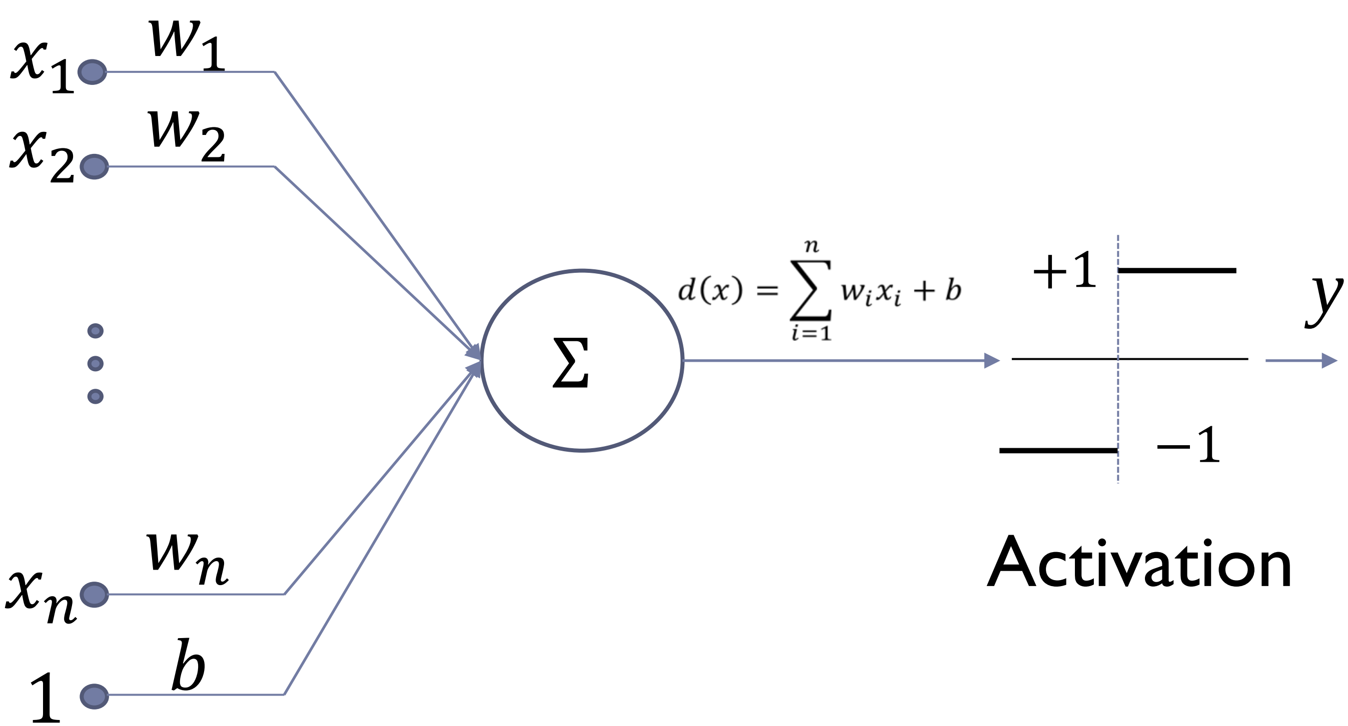
If $b = 0$, the hyperplane is through the origin

$$y = \begin{cases} 1, d(x) > 0 \\ -1, d(x) < 0 \end{cases} \infty y = sgn\big(d(x)\big)$$

$$y = \begin{cases} 1, \sum_{i=1}^{n} w_i x_i > -b \\ -1, \sum_{i=1}^{n} w_i x_i < -b \end{cases}$$

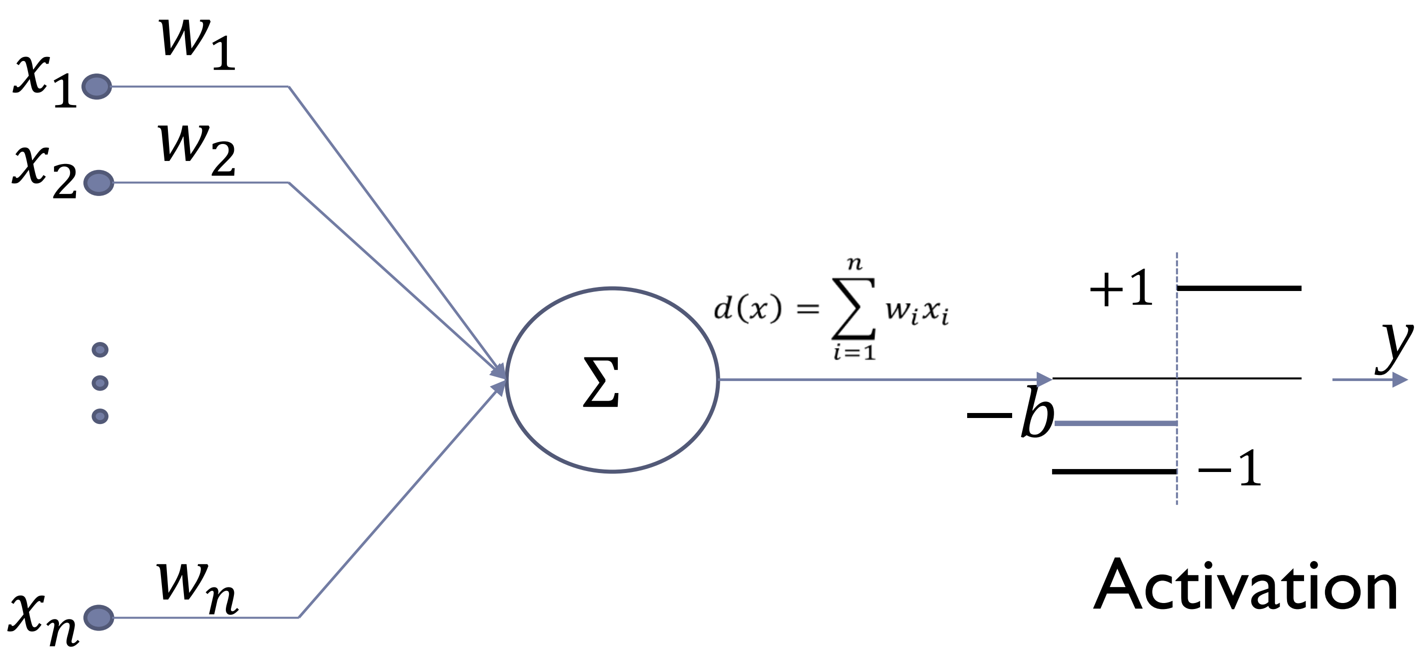
Linear Decision Function

- Perception



Linear Decision Function

- Perception



Linear Decision Function

- Perception
- Parameter Training

$$d(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i + b = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b \text{ (raw)}$$

By extending the pattern vector $\mathbf{x} = [x_1, x_2, \dots, x_n]^{\mathrm{T}}$ into $\mathbf{x} = [x_1, x_2, \dots, x_n, 1]^{\mathrm{T}}$, let $w_{n+1} = b$, there is

$$d(\mathbf{x}) = \sum_{i=1}^{n+1} w_i x_i = \mathbf{w}^{\mathrm{T}}\mathbf{x} \text{ (modified)}$$

where $\mathbf{w} = [w_1, w_2, \dots, w_n, w_{n+1}]^T$.

Linear Decision Function

- Perception

□ Training Algorithm

Let $\mathbf{w}(1)$ be the initial weight vector.

The *k*-th iteration:

- if **x**(k) belongs to "+1" class, but

$$\mathbf{w}(k)^{\mathrm{T}}\mathbf{x}(k) \le 0$$

update $\mathbf{w}(k+1) = \mathbf{w}(k) + \delta \cdot \mathbf{x}(k), \delta > 0$

- if x(k) belongs to "-1" class, but

$$\mathbf{w}(k)^{\mathrm{T}}\mathbf{x}(k) \ge 0$$

update $\mathbf{w}(k+1) = \mathbf{w}(k) - \delta \cdot \mathbf{x}(k), \delta > 0$

- else $\mathbf{w}(k+1) = \mathbf{w}(k)$, keep unchanged.

Linear Decision Function

- Perception

- Example

$$+1:\ x(1)=[0,0]^T, x(2)=[0,1]^T$$
$$-1:\ x(3)=[1,0]^T, x(4)=[1,1]^T$$

Extend by add "1":

$$+1:\ x(1)=[0,0,1]^T, x(2)=[0,1,1]^T$$
$$-1:\ x(3)=[1,0,1]^T, x(4)=[1,1,1]^T$$

Let $\delta=1$, $\mathbf{w}(1)=[0,0,0]^T$, update $\mathbf{w}$?

Training:

The 1st iteration:

$$\mathbf{w}(1)^{\mathrm{T}}\mathbf{x}(1)=[0,0,0]\begin{bmatrix}0\\0\\1\end{bmatrix}=0$$

$$\mathbf{w}(2)=\mathbf{w}(1)+1\cdot\mathbf{x}(1)=[0,0,1]^T$$

Linear Decision Function

- Perception

- Example

$$+1:\ x(1) = [0,0]^T, x(2) = [0,1]^T$$
$$-1:\ x(3) = [1,0]^T, x(4) = [1,1]^T$$

Extend by add “1”：

$$+1:\ x(1) = [0,0,1]^T, x(2) = [0,1,1]^T$$
$$-1:\ x(3) = [1,0,1]^T, x(4) = [1,1,1]^T$$

Let $\delta = 1$, $\mathbf{w}(1) = [0,0,0]^T$, update $\mathbf{w}$?

Training:

The 2nd iteration:

$$\mathbf{w}(2)^{\mathrm{T}}\mathbf{x}(2) = [0,0,1]\begin{bmatrix}0\\1\\1\end{bmatrix} = 1 > 0$$

$$\mathbf{w}(3) = \mathbf{w}(2) = [0,0,1]^T, \text{keep unchanged}$$

Linear Decision Function

- Perception
- Example

$$+1:\ x(1) = [0,0]^T, x(2) = [0,1]^T$$
$$-1:\ x(3) = [1,0]^T, x(4) = [1,1]^T$$

Extend by add "1":

$$+1:\ x(1) = [0,0,1]^T, x(2) = [0,1,1]^T$$
$$-1:\ x(3) = [1,0,1]^T, x(4) = [1,1,1]^T$$

Let $\delta = 1$, $\mathbf{w}(1) = [0,0,0]^T$, update $\mathbf{w}$?

Training:

The 3rd iteration:

$$\mathbf{w}(3)^{\mathrm{T}}\mathbf{x}(3) = [0,0,1]\begin{bmatrix}1\\0\\1\end{bmatrix} = 1 > 0$$

$$\mathbf{w}(4) = \mathbf{w}(3) - \mathbf{x}(3) = [-1\ 0\ 0]^{\mathrm{T}}$$

Linear Decision Function

- Perception
- Example

$$+1:\ x(1)=[0,0]^T, x(2)=[0,1]^T$$
$$-1:\ x(3)=[1,0]^T, x(4)=[1,1]^T$$

Extend by add "1" :

$$+1:\ x(1)=[0,0,1]^T, x(2)=[0,1,1]^T$$
$$-1:\ x(3)=[1,0,1]^T, x(4)=[1,1,1]^T$$

Let $\delta=1$, $\mathbf{w}(1)=[0,0,0]^T$, update $\mathbf{w}$?

Training:

The 4th iteration:

$$\mathbf{w}(4)^{\mathrm{T}}\mathbf{x}(4)=[-1,0,0]\begin{bmatrix}1\\1\\1\end{bmatrix}=-1<0$$

$$\mathbf{w}(5)=\mathbf{w}(4)=[-1\ 0\ 0]^{\mathrm{T}}$$

Linear Decision Function

- Perception

□ Example

$$+1:\ x(1) = [0,0]^T, x(2) = [0,1]^T$$
$$-1:\ x(3) = [1,0]^T, x(4) = [1,1]^T$$

Extend by add "1":

$$+1:\ x(1) = [0,0,1]^T, x(2) = [0,1,1]^T$$
$$-1:\ x(3) = [1,0,1]^T, x(4) = [1,1,1]^T$$

Let $\delta = 1$, $\mathbf{w}(1) = [0,0,0]^T$, update w?

Training:

The t iteration:

$$\mathbf{w}(t)^{\mathrm{T}}\mathbf{x}(t) =?$$

Linear Decision Function

- Perception

□ Widrow-Hoff or LMS Training Algorithm (最小均方)

The objective function

$$J(w) = \frac{1}{2}\left(r - \mathbf{w}^{\mathrm{T}}\mathbf{x}\right)^{2}$$

where r is the expected output, r=+1/-1

The objective is to find the minimum J(w), by using gradient descent.

The gradient of J(w) w.r.t. w is

$$\frac{\partial J(w)}{\partial w} = -\left(r - \mathbf{w}^{\mathrm{T}}\mathbf{x}\right)\mathbf{x}$$

Linear Decision Function

- Perception

□ Widrow-Hoff or LMS Training Algorithm

The weights adjustment

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \delta\left[\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}\right]_{\mathbf{w}=\mathbf{w}(k)}, \delta > 0$$

$$= \mathbf{w}(k) + \delta[r(k) - \mathbf{w}^T(k)\mathbf{x}(k)]\mathbf{x}(k)$$

For iteration, $\mathbf{w}(1)$ is randomly generated.

The increment (delta) of weights is

$$\Delta\mathbf{w} = \mathbf{w}(k+1) - \mathbf{w}(k) = \delta \cdot \mathbf{e}(k)\mathbf{x}(k)$$

where $\mathbf{e}(k) = r(k) - \mathbf{w}^T(k)\mathbf{x}(k)$

Linear Decision Function

- Perception

□ Widrow-Hoff or LMS Training Algorithm

When w(k) is updated into w(k+1), the delta of error

$$\Delta\mathbf{e} = e(k+1) - e(k)$$
$$= [r(k) - \mathbf{w}^T(k+1)\mathbf{x}(k)] - [r(k) - \mathbf{w}^T(k)\mathbf{x}(k)]$$
$$= -[\mathbf{w}^T(k+1) - \mathbf{w}^T(k)]\mathbf{x}(k) = -\Delta\mathbf{w}^{\mathrm{T}}\mathbf{x}(k)$$

Substitute $\Delta\mathbf{w} = \delta \cdot \mathbf{e}(k)\mathbf{x}(k)$ into $\Delta\mathbf{e}$,

$$\Delta\mathbf{e} = -\delta \cdot \mathbf{e}(k)\|\mathbf{x}(k)\|^2$$

δ is the learning rate, and important for convergence