贝叶斯判别 - 应用

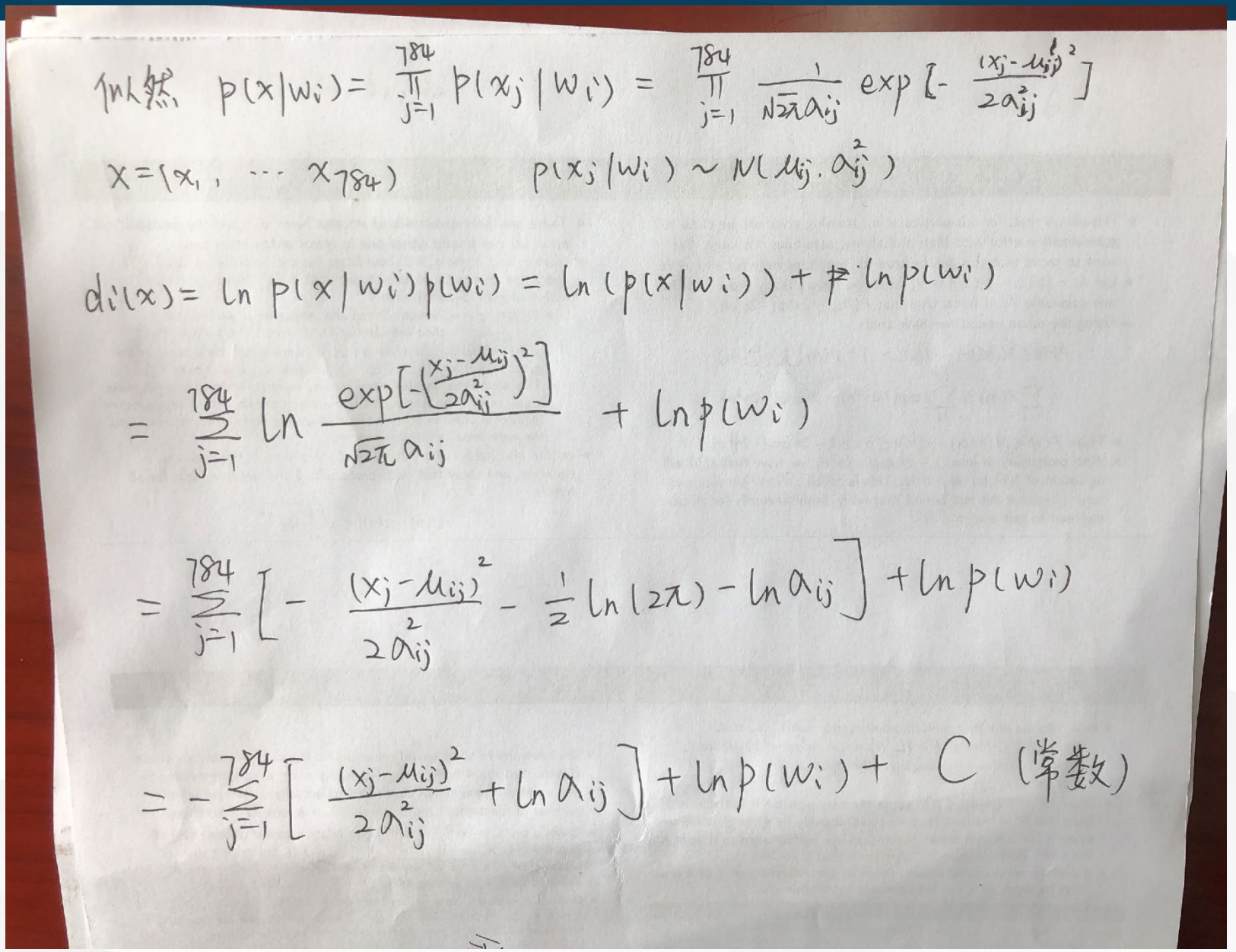
- 手写数字识别
- Mnist:

  每个数字为

  784（28*28）

  维的向量

$$p(y = j \mid x_1, x_2, \cdots, x_{784})$$


则 $p(x|w_i)=\prod_{j=1}^{784} p(x_j|w_i)=\prod_{j=1}^{784}\frac{1}{\sqrt{2\pi}\sigma_{ij}}\exp\left[-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}\right]$
$x=(x_1,\cdots x_{784})$
$p(x_j|w_i)\sim N(\mu_{ij},\sigma_{ij}^2)$
$d_i(x)=\ln p(x|w_i)p(w_i)=\ln(p(x|w_i))+\ln p(w_i)$
$=\sum_{j=1}^{784}\ln\frac{\exp\left[-\left(\frac{x_j-\mu_{ij}}{2\sigma_{ij}^2}\right)^2\right]}{\sqrt{2\pi}\sigma_{ij}}+\ln p(w_i)$
$=\sum_{j=1}^{784}\left[-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}-\frac{1}{2}\ln(2\pi)-\ln\sigma_{ij}\right]+\ln p(w_i)$
$=-\sum_{j=1}^{784}\left[\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}+\ln\sigma_{ij}\right]+\ln p(w_i)+C$ (常数)

```
digit_Number(4) = size(train3',2);
digit_Number(5) = size(train4',2);

total_N= sum(digit_Number);
prior_digit = digit_Number./total_N; %先验概率

train0 = double(train0)/256;
train1 = double(train1)/256;
train2 = double(train2)/256;
train3 = double(train3)/256;
train4 = double(train4)/256;


norm_para_m = [mean(train0);mean(train1);mean(train2);mean(train3); mean(train4)]'; %784*5的向量，代表了每类每个分量的均值
norm_para_v = [std(train0,1);std(train1,1);std(train2,1);std(train3,1);std(train4,1)]'+0.000001;%784*5的向量，代表了每类每个分量的标准差

test0 =double(test0)/256;
pro=[];
for i=1:size(test0,1)
    delta = repmat(test0(i,:)',1,5)- norm_para_m;
    delta =(delta.*delta)./(2*norm_para_v.*norm_para_v);

    pro(i,:) = -sum(log(norm_para_v)+delta)+log(prior_digit);
end
[ m,value idx]=max(pro');
accuracy_0 = sum(idx==1)/size(test0,1)
test1 =double(test1)/256;
pro=[];
for i=1:size(test1,1)
    delta = repmat(test1(i,:)',1,5)- norm_para_m;
    delta =(delta.*delta)./(2*norm_para_v.*norm_para_v);

    pro(i,:) = -sum(log(norm_para_v)+delta)+log(prior_digit);
end
[ m,value idx]=max(pro');
accuracy_1 = sum(idx==2)/size(test1,1)
```

结果

- accuracy_0 = 0.9724
- accuracy_1 = 0.9683
- accuracy_2 = 0.3469
- 0.2616 0.0426 0.3469 0.3275 0.0213
- accuracy_3 = 0.7663
- accuracy_4 = 0.8289

第三章 判别函数

第三章 判别函数

3.1 线性判别函数
3.2 广义线性判别函数
3.3 分段线性判别函数
3.4 模式空间和权空间
3.5 Fisher线性判别
3.6 感知器算法
3.7 采用感知器算法的多类模式的分类
3.8 可训练的确定性分类器的迭代算法
3.9 势函数法 — 一种确定性的非线性分类算法
3.10 决策树简介

3.1 线性判别函数

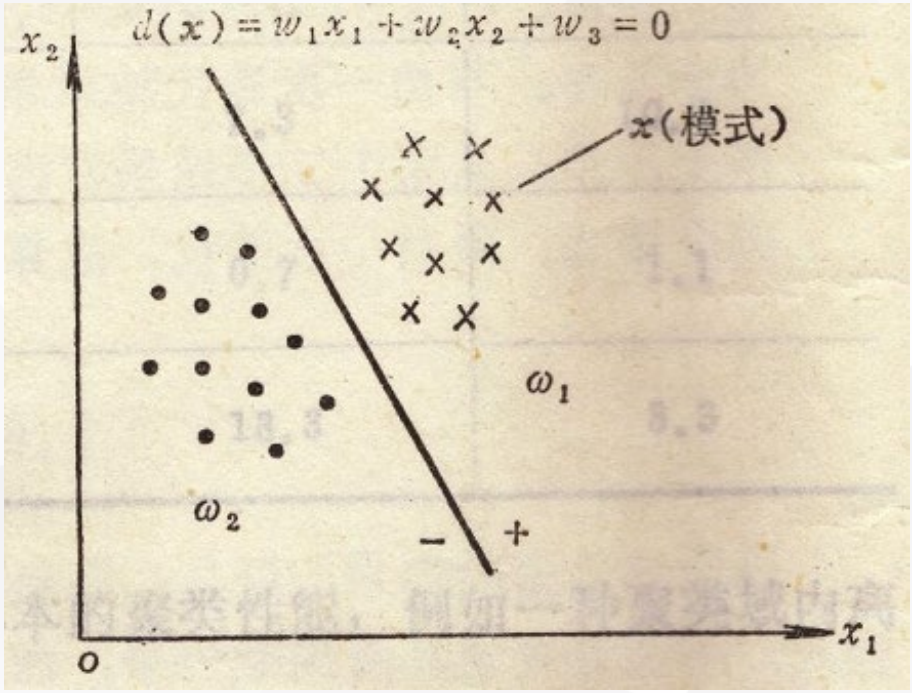
3.1.1 用判别函数分类的概念

- 模式识别系统的主要作用
  - 判别各个模式(也称样本)所属的类别
- 对一个两类问题的判别，就是将模式x划分成ω_1和ω_2两类。

3.1 线性判别函数

3.1.1 用判别函数分类的概念

- 描述：两类问题的判别函数



3.1 线性判别函数

3.1.1 用判别函数分类的概念

- 用判别函数进行模式分类依赖的两个因素

（1）判别函数的几何性质：线性的和非线性的函数。

线性的是一条直线；

非线性的可以是曲线、折线等；

线性判别函数建立起来比较简单（实际应用较多）；

非线性判别函数建立起来比较复杂。

（2）判别函数的系数：判别函数的形式确定后，主要就是确定判别函数的系数问题。

只要被研究的模式是可分的，就能用给定的模式样本集来确定判别函数的系数。

3.1.2 线性判别函数

- n维线性判别函数的一般形式
  - 权向量
  - 增广模式向量
  - 增广权向量
- 分类问题
  - 两类情况：判别函数$d(x)$
  - 多类情况：设模式可分成$\omega_1, \omega_2, \ldots, \omega_M$共$M$类，则有三种划分方法

多类情况1　　多类情况2　　多类情况3

3.1 线性判别函数

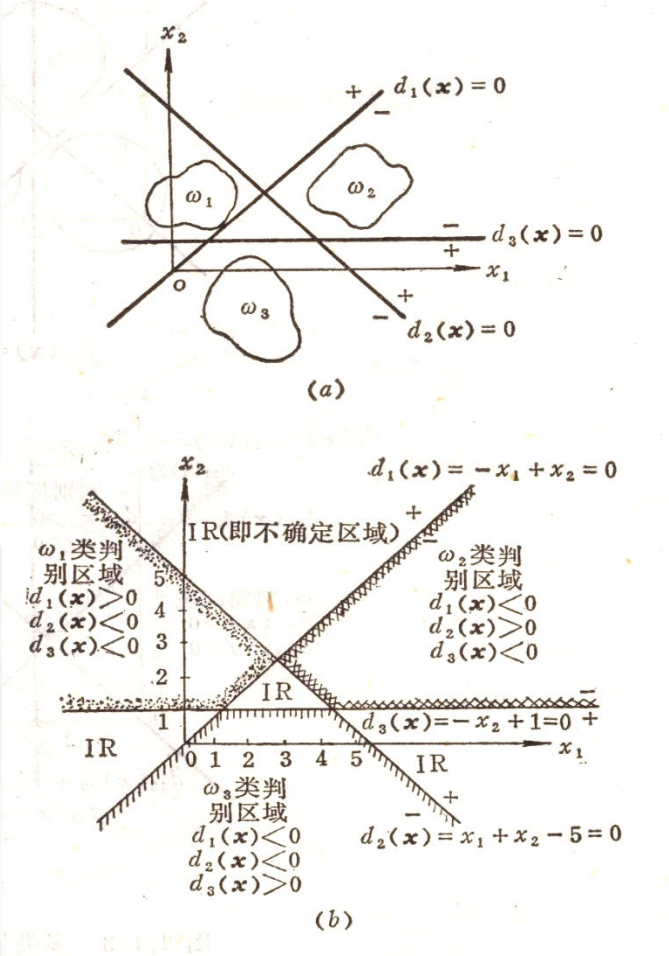
3.1.2 线性判别函数

- 分类问题
  - 多类情况1

    判别函数

    图例

    例子



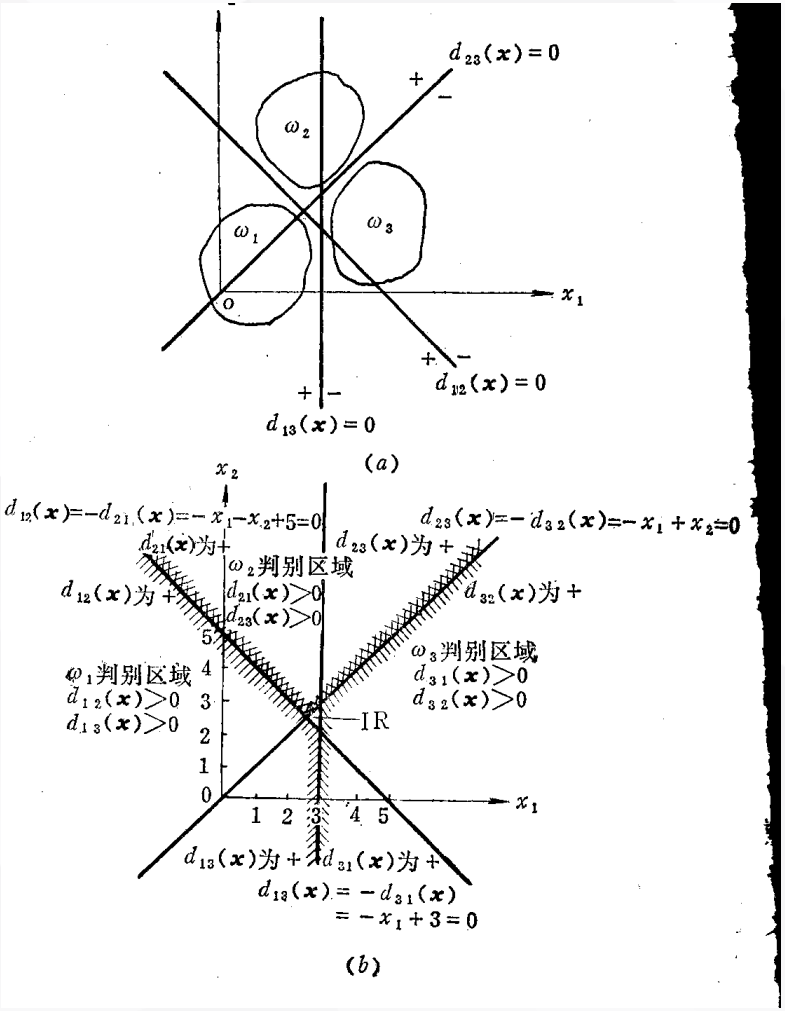
3.1.2 线性判别函数

- 分类问题
  - 多类情况2

    判别函数

    图例

    例子



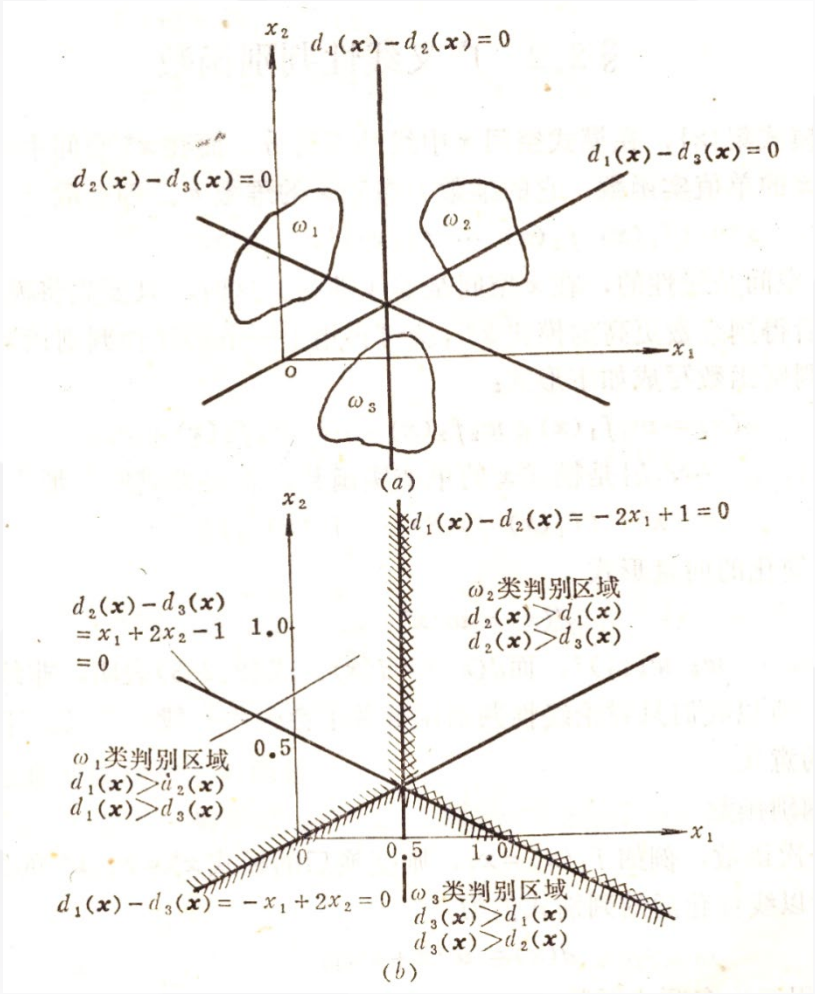
3.1.2 线性判别函数

- 分类问题
  - 多类情况3

    判别函数

    图例

    例子



3.1.2 线性判别函数

■小结：线性可分

•模式分类若可用任一个线性函数来划分，则这些模式就称为线性可分的，否则就是非线性可分的。

•一旦线性函数的系数w_k被确定，这些函数就可用作模式分类的基础。

3.1.2 线性判别函数

■多类情况1和多类情况2的比较

- 对于*M*类模式的分类，多类情况1需要*M*个判别函数，而多类情况2需要*M* *(*M* - *1*)/*2*个判别函数，当*M*较大时，后者需要更多的判别式（这是多类情况2的一个缺点）。
- 采用多类情况1时，每一个判别函数都要把一种类别的模式与其余*M*-*1*种类别的模式分开，而不是将一种类别的模式仅与另一种类别的模式分开。
- 由于一种模式的分布要比*M*-*1*种模式的分布更为聚集，因此多类情况2对模式是线性可分的可能性比多类情况1更大一些（这是多类情况2的一个优点）。

作业（1）

- 在一个10类的模式识别问题中，有3类单独满足多类情况1，其余的类别满足多类情况2。问该模式识别问题所需判别函数的最少数目是多少？

作业（2）

- 一个三类问题，其判别函数如下：

$d_1(\boldsymbol{x})=-x_1,\ d_2(\boldsymbol{x})=x_1+x_2-1,\ d_3(\boldsymbol{x})=x_1-x_2-1$

1. 设这些函数是在多类情况1条件下确定的，绘出其判别界面和每一个模式类别的区域。
2. 设为多类情况2，并使：$d_{12}(\boldsymbol{x})=d_1(\boldsymbol{x}),\ d_{13}(\boldsymbol{x})=d_2(\boldsymbol{x}),\ d_{23}(\boldsymbol{x})=d_3(\boldsymbol{x})$。绘出其判别界面和多类情况2的区域。
3. 设$d_1(\boldsymbol{x}),\ d_2(\boldsymbol{x})$和$d_3(\boldsymbol{x})$是在多类情况3的条件下确定的，绘出其判别界面和每类的区域。

3.2 广义线性判别函数

- 出发点
  - 线性判别函数简单，容易实现；
  - 非线性判别函数复杂，不容易实现；
  - 若能将非线性判别函数转换为线性判别函数，则有利于模式分类的实现。

3.2 广义线性判别函数

- 基本思想

  设有一个训练用的模式集$\{x\}$，在模式空间x中线性不可分，但在模式空间x^*中线性可分，其中x^*的各个分量是x的单值实函数，x^*的维数k高于x的维数n，即若取

  $$x^* = (f_1(x), f_2(x), \ldots, f_k(x)),\ k>n$$

  则分类界面在x^*中是线性的，在x中是非线性的，此时只要将模式x进行非线性变换，使之变换后得到维数更高的模式x^*，就可以用线性判别函数来进行分类。

- 描述

3.2 广义线性判别函数

■广义线性判别函数的意义

- 线性的判别函数
- $f_i(x)$选用二次多项式函数

  x是二维的情况

  x是n维的情况

- $f_i(x)$选用r次多项式函数，x是n维的情况

  例子

  $d(x)$的总项数 $N_w = C_{n+r}^{r} = \frac{(n+r)!}{r!n!}$

■ 说明

- $d(x)$的项数随r和n的增加会迅速增大，即使原来模式x的维数不高，若采用次数r较高的多项式来变换，也会使变换后的模式x*的维数很高，给分类带来很大困难。
- 实际情况可只取r=2，或只选多项式的一部分，例如r=2时只取二次项，略去一次项，以减少x*的维数。

证明

- 对于n维x向量，若用r次多项式，$d(x)$的权系数的总项数为：

$$N_w = C_{n+r}^{r} = \frac{(n+r)!}{r!n!}$$

证明

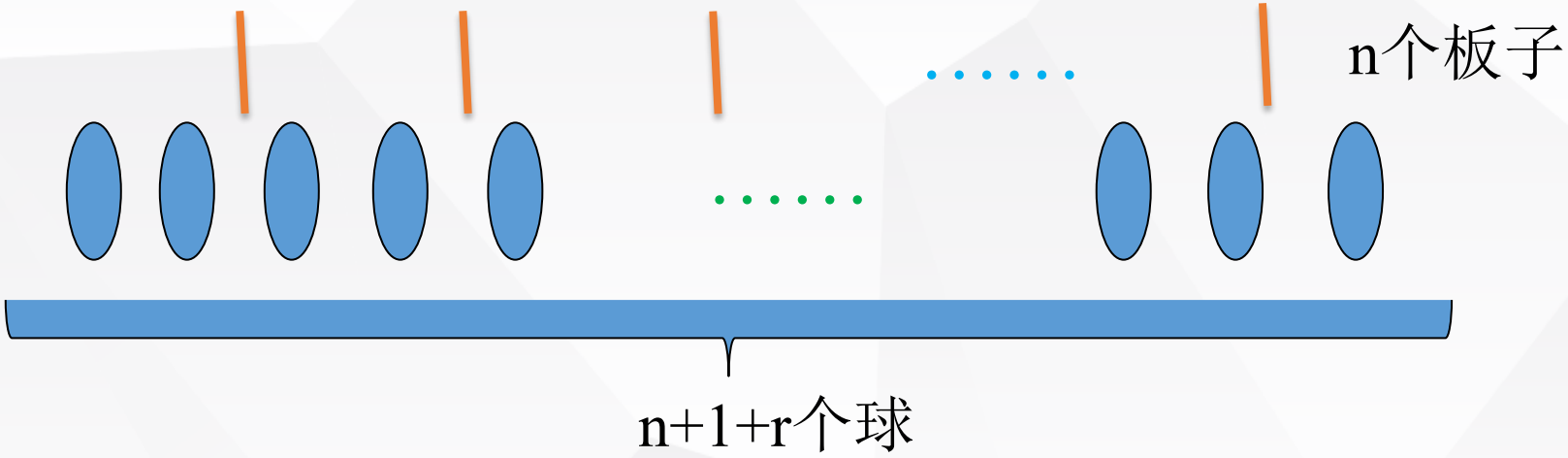
判别函数各项的一般化表达：$x_1^{r_1} x_2^{r_2} \cdots x_n^{r_n} 1^{r_{n+1}}$

其中r_i>=0的整数，且 $\sum_{i=1}^{n+1} r_i = r$

令$y_i = 1 + r_i$,则 为$y_i > 0$, $\sum_{i=1}^{n+1} y_i = r + n + 1$

证明

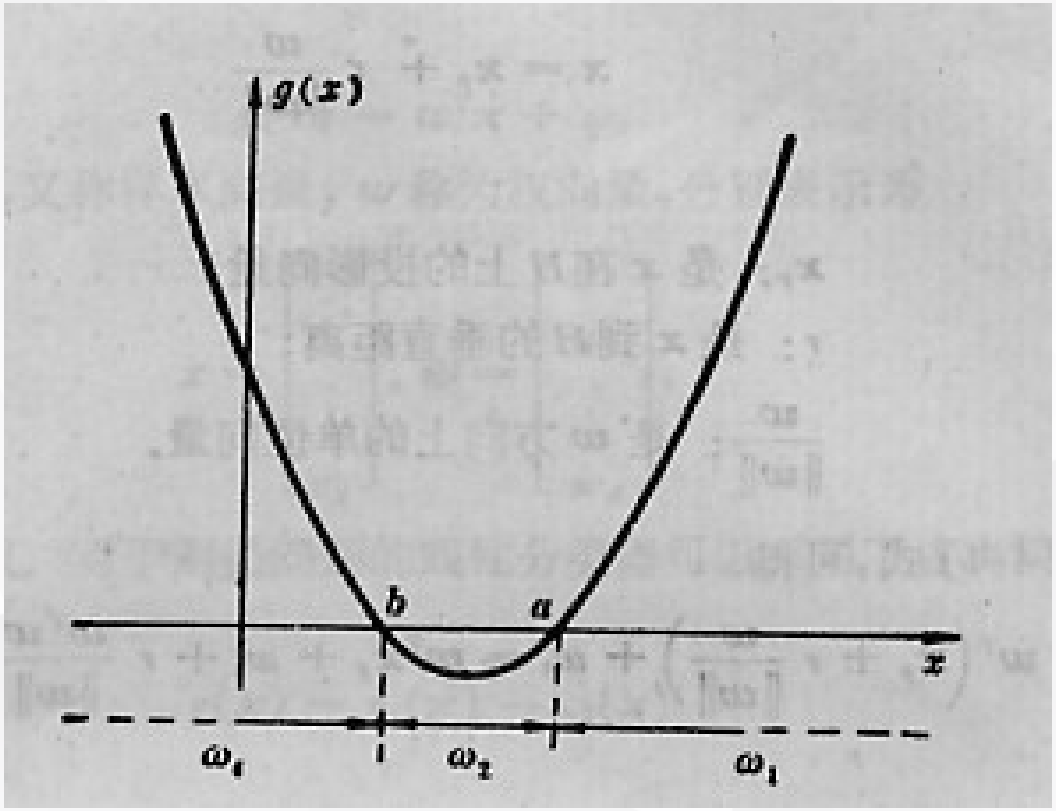
多少种方法



$$N_w = C_{n+r}^{r} = \frac{(n+r)!}{r!n!}$$

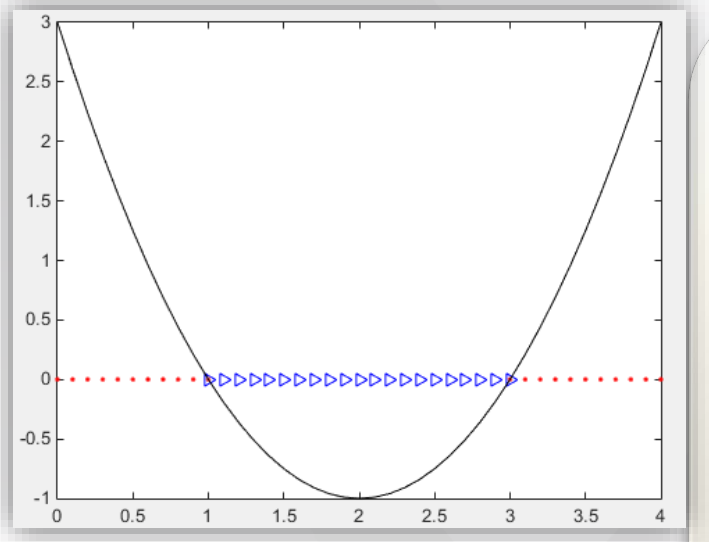
3.2 广义线性判别函数

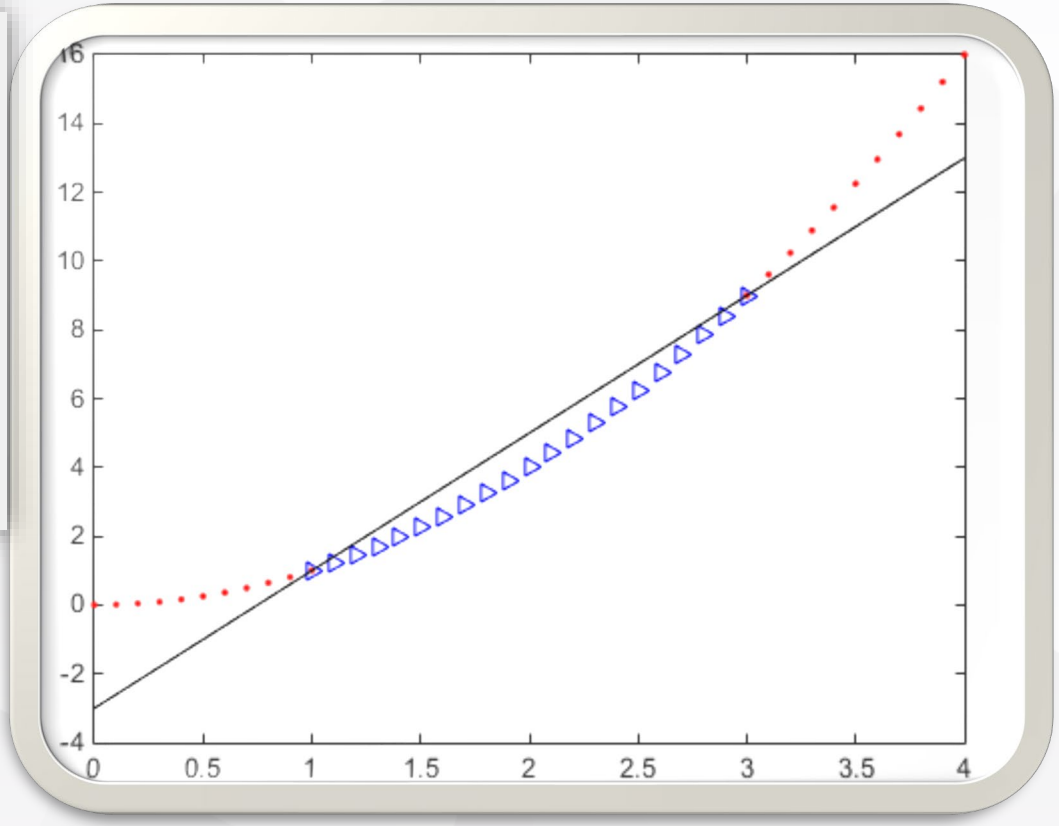
- [例子：一维样本空间 -〉二维样本空间]



3.2 广义线性判别函数

特征空间变换之后





作业

- 两类模式，每类包括5个3维不同的模式向量，且良好分布。如果它们是线性可分的，问权向量至少需要几个系数分量？假如要建立二次的多项式判别函数，又至少需要几个系数分量？（设模式的良好分布不因模式变化而改变。）

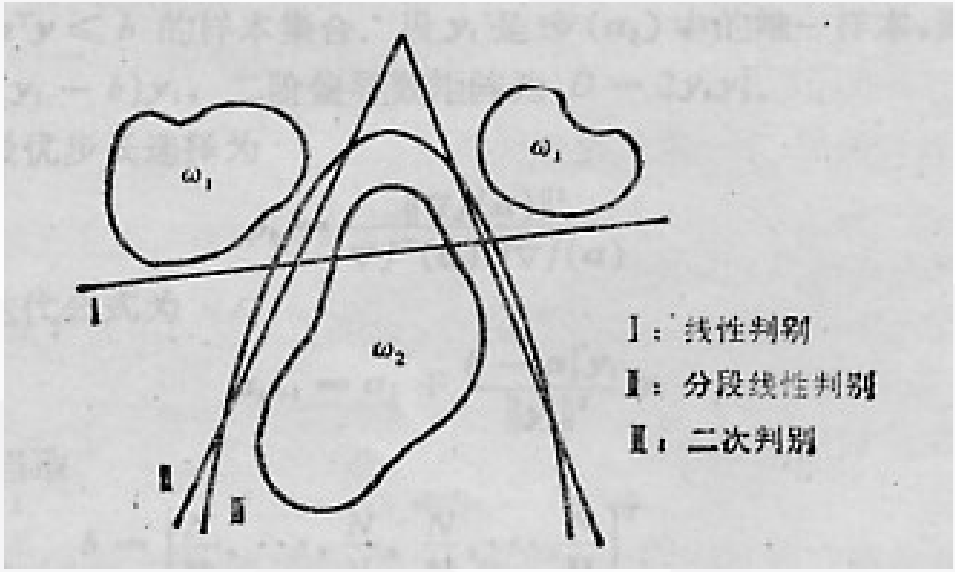
3.3 分段线性判别函数

出发点

- 线性判别函数在进行分类决策时是最简单有效的，但在实际应用中，常常会出现不能用线性判别函数直接进行分类的情况。
- 采用广义线性判别函数的概念，可以通过增加维数来得到线性判别，但维数的大量增加会使在低维空间里在解析和计算上行得通的方法在高维空间遇到困难，增加计算的复杂性。
- 引入分段线性判别函数的判别过程，它比一般的线性判别函数的错误率小，但又比非线性判别函数简单。

3.3 分段线性判别函数

- 图例：用判别函数分类
  - 可用一个二次判别函数来分类
  - 也可用一个分段线性判别函数来逼近这个二次曲线

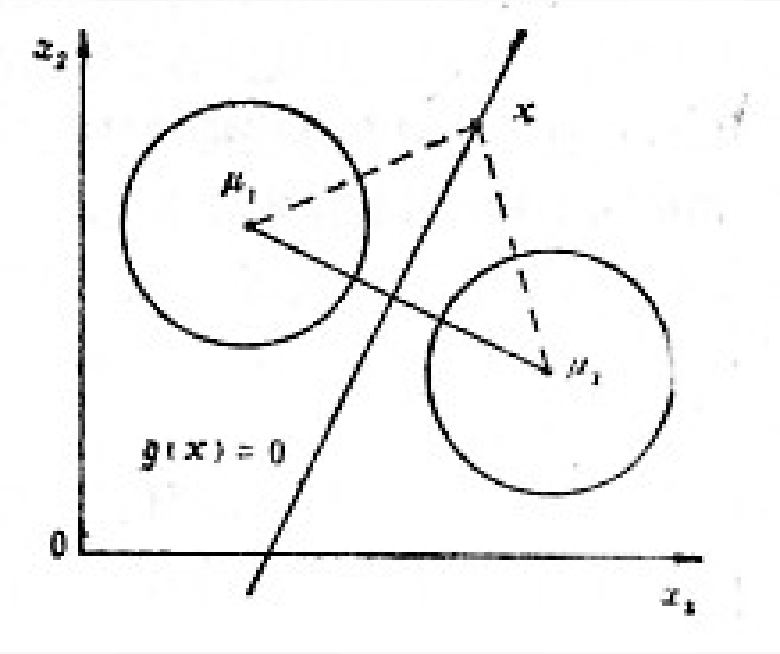


3.3 分段线性判别函数

■分段线性判别函数的设计

- •采用最小距离分类的方法
  - •最小距离分类

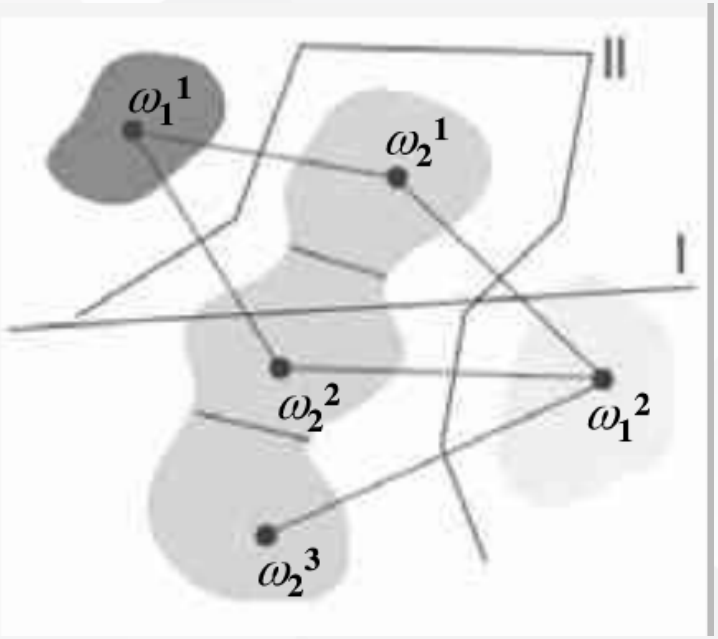
- •这种方法只有在种类别密集地分布在其均值附近时才有效



3.3 分段线性判别函数

■图例：分段线性分类设计

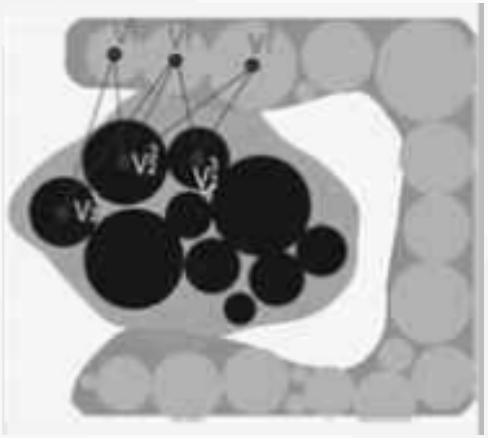
对于各类交错分布的情况，若再用每类一个均值代表点产生最小距离分类器，就会产生很明显的错误率。在这种情况下，可以运用聚类方法将一些类分解成若干个子类，再用最小距离分类。



I：线性距离判别
II：分段线性距离判别

3.3 分段线性判别函数

- 寻找交遇区—找到互为最小距离的原型对，组成“交遇区”。
- 用局部训练模式产生分段线性判别函数并迭代优化决策面。
- 撤走已分类正确的样本，从剩下的样本集合中，寻找交遇区，产生分段线性判别函数。



3.4 模式空间和权空间

- 分类描述
- 模式空间
  - 对一个线性方程$w_1x_1+w_2x_2+w_3x_3=0$，它在三维空间(x_1, x_2, x_3)中是一个平面方程式，$\boldsymbol{w}=(w_1\ w_2\ w_3)^T$是方程的系数。
  - 把$\boldsymbol{w}$向量作为该平面的法线向量，则该线性方程决定的平面通过原点且与$\boldsymbol{w}$垂直。

3.4 模式空间和权空间

■模式空间

•若$\boldsymbol{x}$是二维的增广向量，此时$x_3=1$，则在非增广的模式空间中即为$\{x_1, x_2\}$二维坐标，判别函数是下列联立方程的解、负两侧，$\boldsymbol{w}$ 离开直线的一侧为正，$\boldsymbol{w}$ 射向直线的一侧为负。

$$w_1x_1+w_2x_2+w_3=0$$

$$x_3=1$$

即为这两个平面相交的直线AB

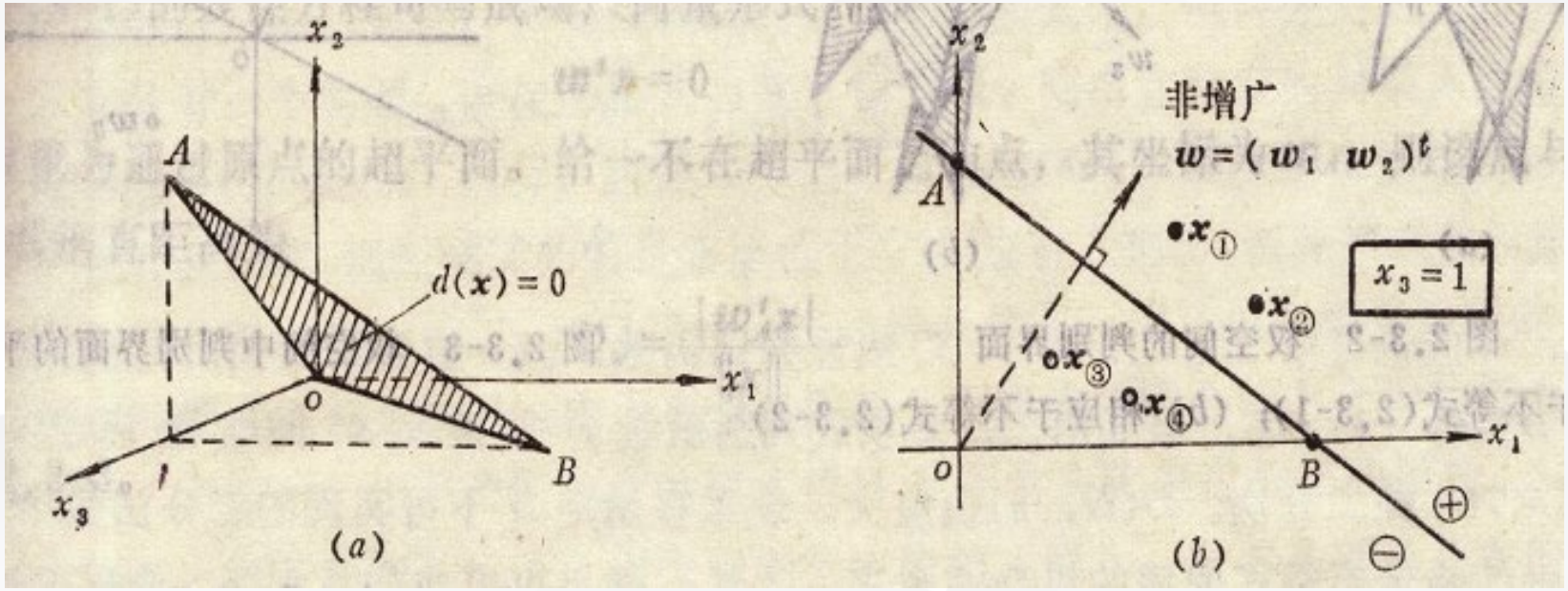
•此时，$\boldsymbol{w}=(w_1\ w_2)^T$为非增广的权向量，它与直线AB垂直；AB将平面分为正

3.4 模式空间和权空间

- 模式空间

(a) 增广向量决定的平面

(b) 非增广向量决定的直线



3.4 模式空间和权空间

■权空间

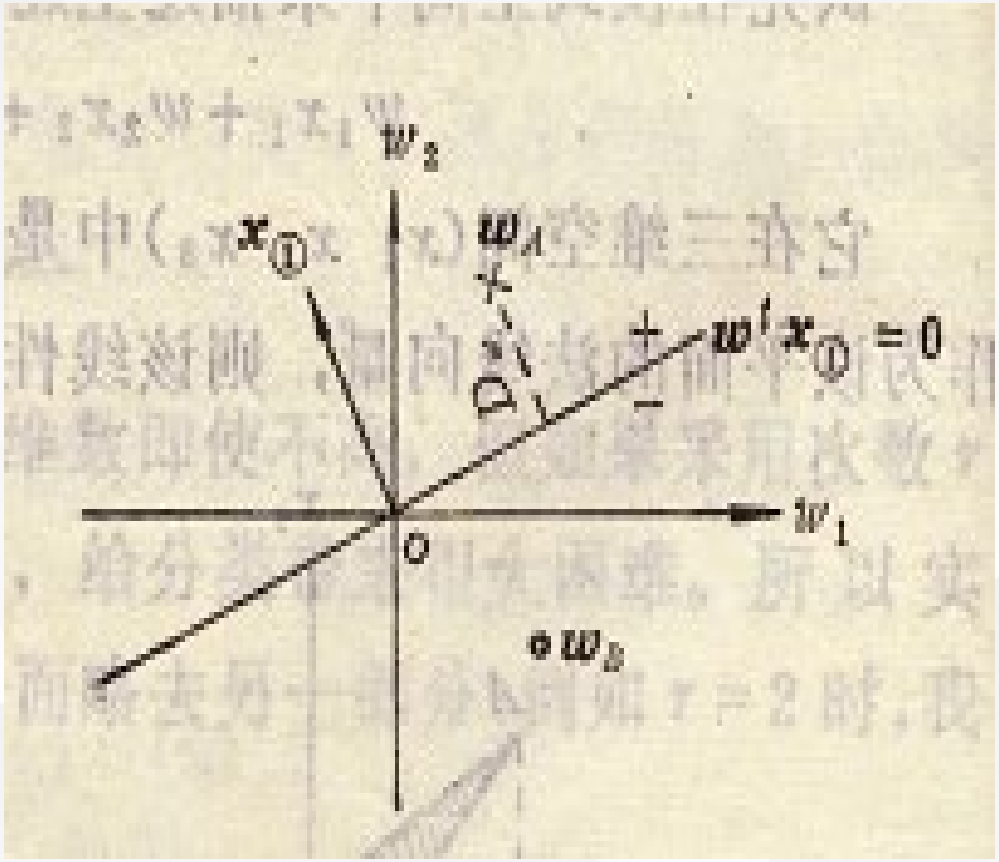
•若将方程$x_1w_1+x_2w_2+w_3=0$绘在权向量$\boldsymbol{w}=(w_1\ w_2\ w_3)^T$的三维空间中，则$\boldsymbol{x}=(x_1\ x_2\ 1)^T$为方程的系数。

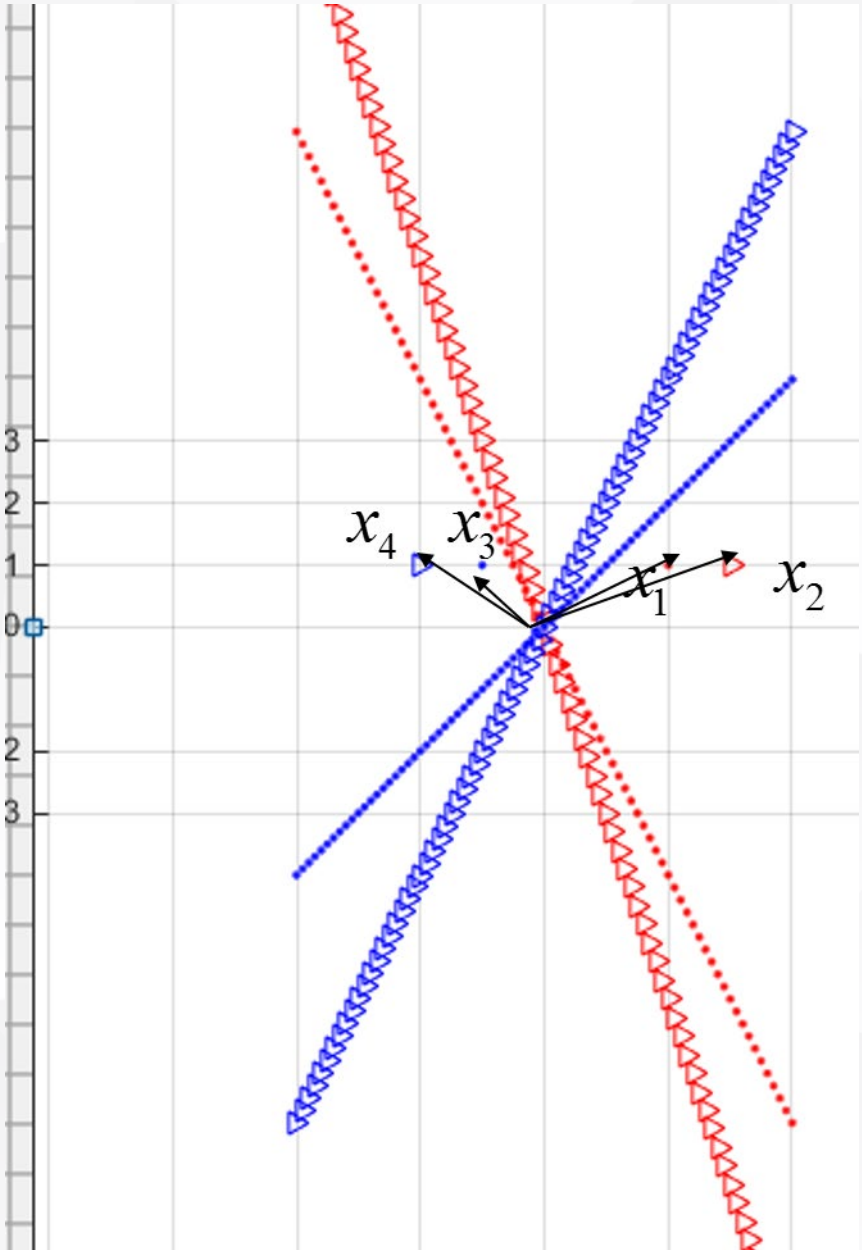
•若以$\boldsymbol{x}$向量作为法线向量，则该线性方程所决定的平面为通过原点且与法线向量垂直的平面，它同样将权空间划分为正、负两边。

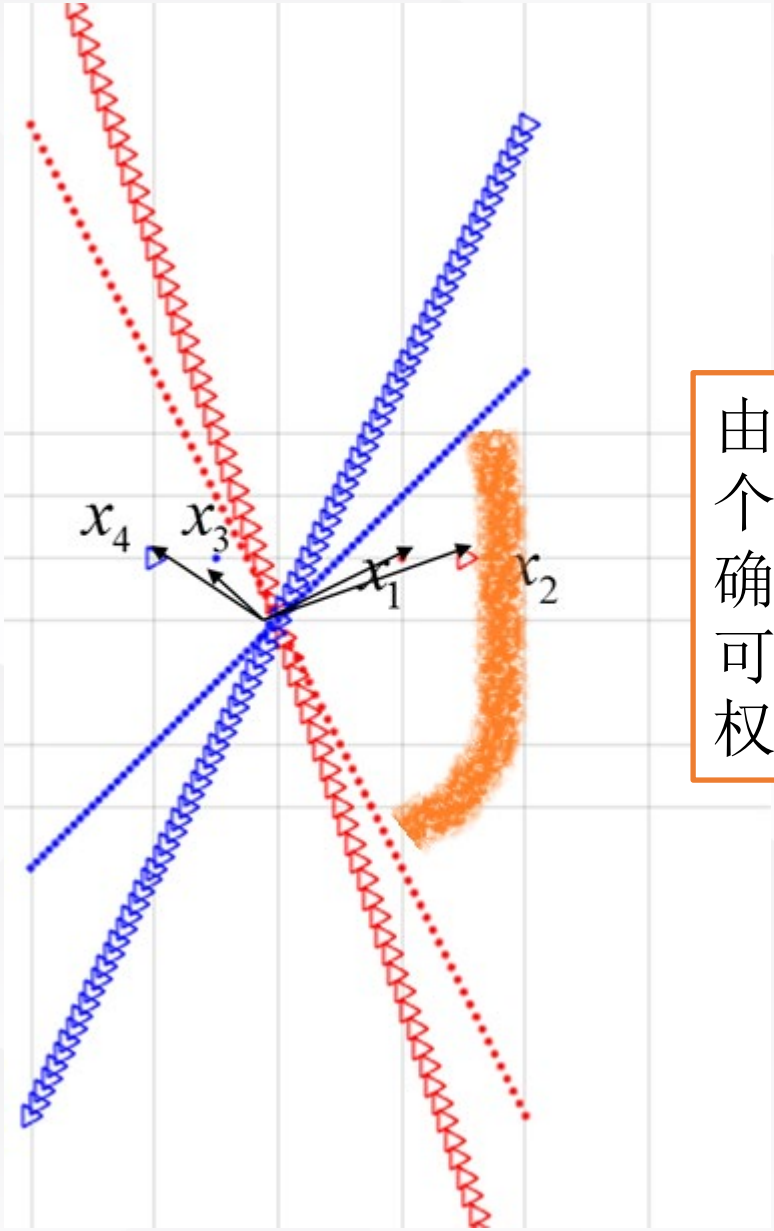
•在系数$\boldsymbol{x}$不变的条件下，若$\boldsymbol{w}$值落在法线向量离开平面的一边，则$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}>0$，若$\boldsymbol{w}$值落在法线向量射向平面的一边，则$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}<0$。

3.4 模式空间和权空间

- 权空间中判别界面的平面示意图







由这四个样本确定的可行的权空间

3.5 Fisher线性判别

出发点

- 应用统计方法解决模式识别问题时，一再碰到的问题之一就是维数问题。
- 在低维空间里解析上或计算上行得通的方法，在高维空间里往往行不通。
- 因此，降低维数有时就会成为处理实际问题的关键。

3.5 Fisher线性判别

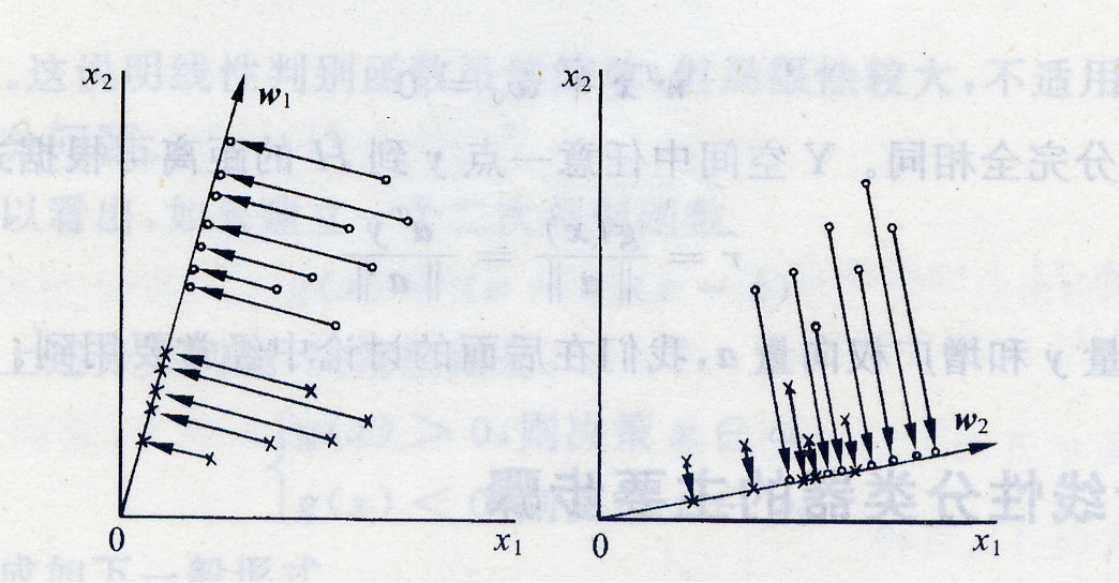
问题描述

- 考虑把d维空间的样本投影到一条直线上，形成一维空间，即把维数压缩到一维。
- 然而，即使样本在d维空间里形成若干紧凑的互相分得开的集群，当把它们投影到一条直线上时，也可能会是几类样本混在一起而变得无法识别。
- 但是，在一般情况下，总可以找到某个方向，使在这个方向的直线上，样本的投影能分得开。

3.5 Fisher线性判别

■问题描述

•问题：如何根据实际情况找到一条最好的、最易于分类的投影线，这就是Fisher判别方法所要解决的基本问题。



3.5 Fisher线性判别

- 从d维空间到一维空间的一般数学变换方法

  •假设有一集合Γ包含N个d维样本$\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N$，其中$N_1$个属于$\omega_1$类的样本记为子集$\Gamma_1$，$N_2$个属于$\omega_2$类的样本记为子集$\Gamma_2$。若对$\boldsymbol{x}^n$的分量做线性组合可得标量：

$$y_n = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^n,\ n=1,2,\ldots,N$$

  •这样便得到N个一维样本y_n组成的集合，并可分为两个子集Γ_1'和Γ_2'。

3.5 Fisher线性判别

- 从d维空间到一维空间的一般数学变换方法

  •实际上，$|w|$的值是无关紧要的，它仅是y_n乘上一个比例因子，重要的是选择w的方向。w的方向不同，将使样本投影后的可分离程度不同，从而直接影响分类效果。

  •因此，上述寻找**最佳**投影方向的问题，在数学上就是寻找最好的变换向量w^*的问题。

3.5 Fisher线性判别

- Fisher准则函数的定义
  - 几个必要的基本参量
  - 我们希望投影后，在一维Y空间中各类样本尽可能分得开些，即希望两类均值之差越大越好，同时希望各类样本内部尽量密集，即希望类内离散度越小越好。
  - Fisher准则函数定义
  - **最佳**变换向量w^*的求取

3.5 Fisher线性判别

■基于最佳变换向量w^*的投影

•w^*是使Fisher准则函数$J_F(w)$取极大值时的解，也就是d维X空间到一维Y空间的最佳投影方向。有了w^*，就可以把d维样本x投影到一维，这实际上是多维空间到一维空间的一种映射，这个一维空间的方向w^*相对于Fisher准则函数$J_F(w)$是最好的。

•利用Fisher准则，就可以将d维分类问题转化为一维分类问题，然后，只要确定一个阈值T，将投影点y_n与T相比较，即可进行分类判别。

$$w^* = S_w^{-1}(m_1 - m_2)$$

3.5 Fisher线性判别

Fisher线性判别与贝叶斯判别的关系

- 两类问题且其类模式都是正态分布的情况，且$C_1=C_2=C$时

$$d_1(\boldsymbol{x})-d_2(\boldsymbol{x})=lnP(w_1)-lnP(w_2)+(\boldsymbol{m}_1-\boldsymbol{m}_2)^T\boldsymbol{C}^{-1}\boldsymbol{x}-\frac{1}{2}\boldsymbol{m}_1^T\boldsymbol{C}^{-1}\boldsymbol{m}_1+\frac{1}{2}\boldsymbol{m}_2^T\boldsymbol{C}^{-1}\boldsymbol{m}_2=0$$

- Fisher线性判别的最佳变换向量

$$\boldsymbol{w}^*=\boldsymbol{S}_w^{-1}(\boldsymbol{m}_1-\boldsymbol{m}_2)$$

3.5 Fisher线性判别

- 多类情形——前面是针对只有两个类的情况，假设类别变成多个了，那么要怎么改变，才能保证投影后类别能够分离呢？
- 我们之前讨论的是如何将d维降到一维，现在类别多了，一维可能已经不能满足要求。假设我们有M个类别，需要K维向量（或者叫做基向量）来做投影。

3.5 Fisher线性判别

■多类情形

- K个投影向量 $\boldsymbol{W}=[w_1,w_2,\cdots,w_K]$,样本点投影后结果为

$$y=[y_1,y_2,\cdots,y_K]^T$$

$$y=\boldsymbol{W}^T\boldsymbol{x}$$

- 类间散度矩阵与两类情形略有不同：原来度量的是两个均值点的散列情况，现在度量的是每类均值点与样本中心的散列情况

$$\boldsymbol{S}_b=(\boldsymbol{m}_1-\boldsymbol{m}_2)(\boldsymbol{m}_1-\boldsymbol{m}_2)^T$$

$$\boldsymbol{S}_b=\sum_{j=1}^{C}N_j(\boldsymbol{m}_j-\boldsymbol{m})(\boldsymbol{m}_j-\boldsymbol{m})^T$$

秩最大为多少？

- 与两类形情一样推导

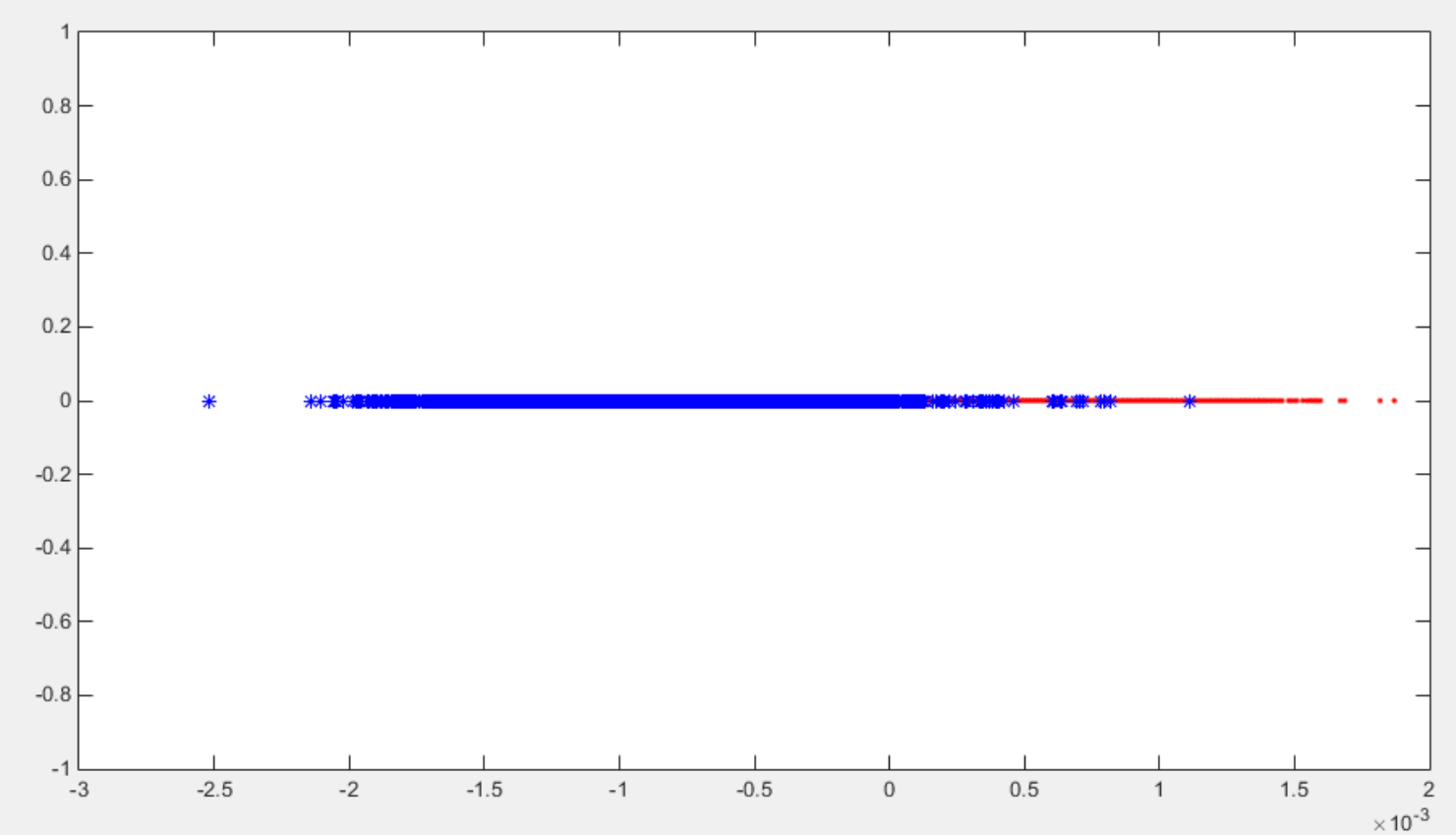
$$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b\boldsymbol{W}^*=\lambda\boldsymbol{W}^*$$

K最大为多少？

$$\boldsymbol{S}_i=\sum_{\boldsymbol{x}\in\Gamma_i}(\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^T,i=1,2,\cdots C,\quad \boldsymbol{S}_w=\sum_{j=1}^{C}\boldsymbol{S}_j$$

例子

- 红色的点代表数字 2，蓝色的点代表数字 3



3.6 感知器算法

出发点

- 一旦判别函数的形式确定下来，不管它是线性的还是非线性的，剩下的问题就是如何确定它的系数。
- 在模式识别中，系数确定的一个主要方法就是通过对已知样本的训练和学习来得到。
- 感知器算法就是通过训练样本模式的迭代和学习，产生线性（或广义线性）可分的模式判别函数。

3.6 感知器算法

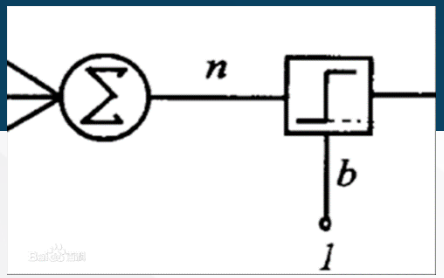
■基本思想

•采用感知器算法(Perception Approach)能通过对训练模式样本集的“学习”得到判别函数的系数。

■说明

•这里采用的算法不需要对各类别中模式的统计性质做任何假设，因此称为确定性的方法。

3.6 感知器算法



■背景

- “感知器”一词出自于20世纪50年代中期到60年代中期人们对一种分类学习机模型的称呼，它是属于有关动物和机器学习的仿生学领域中的问题。
- 当时的一些研究者认为感知器是一种学习机的强有力模型，后来发现估计过高了，但发展感知器的一些相关概念仍然沿用下来。
- 感知器作为人工神经网络中最基本的单元，有多个输入和一个输出组成。

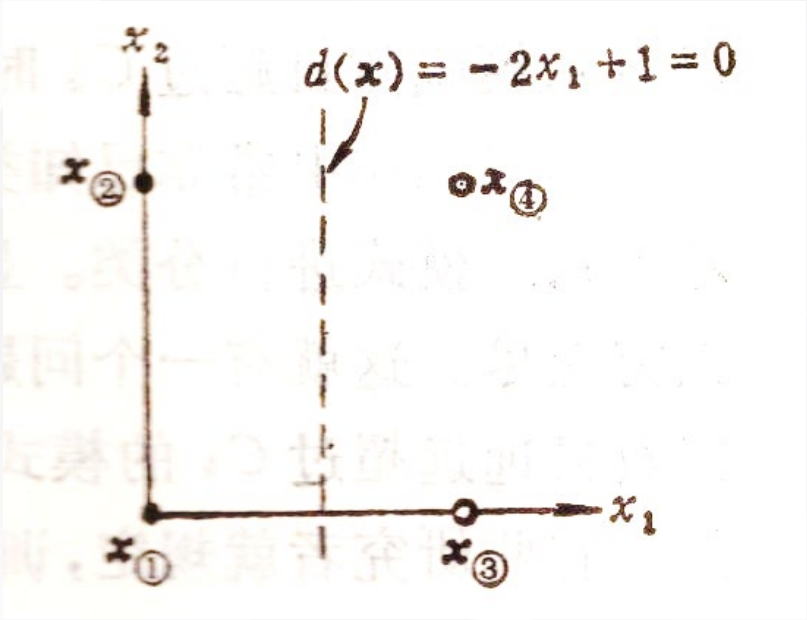
3.6 感知器算法

$$J(w)=\sum_{x\in\tilde{X}}-w^{T}x,\ \tilde{X}\text{被错分的样本集}$$

- 感知器的训练算法
- 感知器算法实质上是一种赏罚过程
  - 对正确分类的模式则“赏”，实际上是“不罚”，即权向量不变。
  - 对错误分类的模式则“罚”，使$w(k)$加上一个正比于x_k的分量。
  - 当用全部模式样本训练过一轮以后，只要有一个模式是判别错误的，则需要进行下一轮迭代，即用全部模式样本再训练一次。
  - 如此不断反复直到全部模式样本进行训练都能得到正确的分类结果为止。

3.6 感知器算法

- [例子]



- 感知器算法的收敛性
  - 只要模式类别是线性可分的，就可以在有限的迭代步数里求出权向量。

证明

- 只要模式类别是线性可分的，感知器算法就可以在有限的迭代步数里求出权向量。

思路：如果第k+1次迭代生成的权矢量比第k次迭代生成的权矢量更接近解矢量，则收敛。

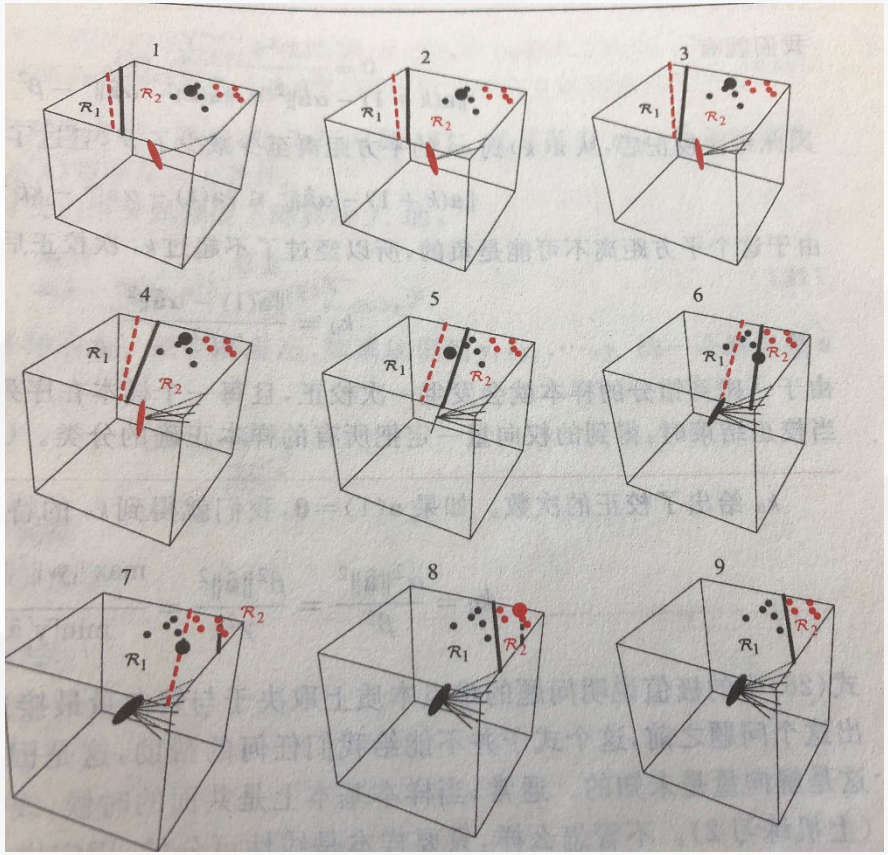
由于模式类别是线性可分的，所以存在 $w^*, (w^*)^T x>0$

证明

$$\|w(k+1)-\alpha w^*\|^2 < \|w(k)-\alpha w^*\|^2$$

虽然这并不总是成立，但是只要计算了足够长的步数，上述式子将成立。

黑色和红色分别属于两类样本。当前这轮迭代中，被错分的样本为大黑点。虚线为原来的判别面，实线为更新后的判别面。



证明

$$\boldsymbol{w}(k+1)-\alpha\boldsymbol{w}^{*}=\boldsymbol{w}(k)-\alpha\boldsymbol{w}^{*}+\boldsymbol{x}_k$$

$$\left\|\boldsymbol{w}(k+1)-\alpha\boldsymbol{w}^{*}\right\|^2=\left\|\boldsymbol{w}(k)-\alpha\boldsymbol{w}^{*}\right\|^2+2(\boldsymbol{w}(k)-\alpha\boldsymbol{w}^{*})^T\boldsymbol{x}_k+\left\|\boldsymbol{x}_k\right\|^2$$

$$\leq\left\|\boldsymbol{w}(k)-\alpha\boldsymbol{w}^{*}\right\|^2-2\alpha\boldsymbol{w}^{*T}\boldsymbol{x}_k+\left\|\boldsymbol{x}_k\right\|^2$$

由于$\boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}_{\mathrm{k}}>0$,当$\alpha$足够大时，第2项将对第3项起支配作用。特别是，如果设 $\beta^2=\max_k\left\|\boldsymbol{x}_k\right\|^2,\gamma=\min_k[\boldsymbol{w}^{*T}\boldsymbol{x}_k]>0$

证明

$$\|\boldsymbol{w}(k+1)-\alpha\boldsymbol{w}^*\|^2 < \|\boldsymbol{w}(k)-\alpha\boldsymbol{w}^*\|^2 - 2\alpha\gamma + \beta^2$$

如果选 $\alpha = \frac{\beta^2}{\gamma}$，则有 $\|\boldsymbol{w}(k+1)-\alpha\boldsymbol{w}^*\|^2 < \|\boldsymbol{w}(k)-\alpha\boldsymbol{w}^*\|^2 - \beta^2$

- 因此每次校正后，权系数到$\boldsymbol{w}^*$的平方距离至少减少 β^2 了过了k步后，

  $$\|\mathbf{w}(k+1) - \alpha\mathbf{w}^*\|^2 < \|\mathbf{w}(1) - \alpha\mathbf{w}^*\|^2 - \mathrm{k}\beta^2$$

- 由于距离不能为负，所以经过了不超过 $k_0 = \frac{\|\boldsymbol{w}(1)-\alpha\boldsymbol{w}^*\|^2}{\beta^2}$ 次后，权系数的校正将终止。

作业及编程

- 用感知器算法求下列模式分类的解向量 w:

  ω_1: $\{(0\ 0\ 0)^T, (1\ 0\ 0)^T, (1\ 0\ 1)^T, (1\ 1\ 0)^T\}$

  ω_2: $\{(0\ 0\ 1)^T, (0\ 1\ 1)^T, (0\ 1\ 0)^T, (1\ 1\ 1)^T\}$

- 编写求解上述问题的感知器算法程序。

3.7 采用感知器算法的多类模式的分类

- 采用3.1的多类情况3，将感知器算法推广到多类模式。
- 感知器算法判别函数的推导
- [例子]

3.7 采用感知器算法的多类模式的分类

■讨论

•这里的分类算法都是通过模式样本来确定判别函数的系数，但一个分类器的判断性能最终要受并未用于训练的那些未知样本来检验。

•要使一个分类器设计完善，必须采用有代表性的训练数据，它能够合理反映模式数据的整体。

3.7 采用感知器算法的多类模式的分类

■讨论

•要获得一个判别性能好的线性分类器，究竟需要多少训练样本?

直观上是越多越好，但实际上能收集到的样本数目会受到客观条件的限制；

过多的训练样本在训练阶段会使计算机需要较长的运算时间；

一般来说，合适的样本数目可如下估计：

若k是模式的维数，令$C=2(k+1)$，则通常选用的训练样本数目约为C的10~20倍。

作业

- 用多类感知器算法求下列模式的判别函数：

  ω_1: $(-1\ -1)^T$

  ω_2: $(0\ 0)^T$

  ω_3: $(1\ 1)^T$

思考

- 感知器算法存在许多解？
  - 初值的选择
  - 迭代过程中误分类点的选择顺序

选做

- 编写求解上述问题的感知器算法程序，求下列模式分类的解向量w:

  ω_1: $\{(0\ 0\ 0)^T, (1\ 0\ 0)^T, (1\ 0\ 1)^T, (1\ 1\ 0)^T\}$

  ω_2: $\{(0\ 0\ 1)^T, (0\ 1\ 1)^T, (0\ 1\ 0)^T, (1\ 1\ 1)^T\}$

  - 尝试不同的初始值
  - 尝试不同的迭代顺序

3.8 可训练的确定性分类器的迭代算法

3.8.1 梯度法

定义

- 梯度是一个向量，它的最重要性质就是指出了函数 f 在其自变量增加时最大增长率的方向。
- 负梯度指出 f 的最陡下降方向
- 利用这个性质，可以设计一个迭代方案来寻找函数的最小值。

3.8 可训练的确定性分类器的迭代算法

3.8.1 梯度法

■采用梯度法求解的基本思想

•对感知器算法

$$w(k+1)=\begin{cases} w(k) & ,\ \text{if } w^T(k)x_k>0 \\ w(k)+Cx_k & \text{if } w^T(k)x_k\le 0 \end{cases}$$

式中的$w(k)$、x_k随迭代次数k而变，是变量。

•定义一个对错误分类敏感的准则函数$J(w,x)$。先任选一个初始权向量$w(1)$，计算准则函数J的梯度，然后从$w(1)$出发，在最陡方向（梯度方向）上移动某一距离得到下一个权向量$w(2)$。

•从$w(k)$导出$w(k+1)$的一般关系式

3.8 可训练的确定性分类器的迭代算法

3.8.1 梯度法

■讨论

•若正确地选择了准则函数$J(w,x)$，则当权向量w是一个解时，J达到极小值（J的梯度为零）。由于权向量是按J的梯度值减小，因此这种方法称为梯度法（最速下降法）。

•为了使权向量能较快地收敛于一个使函数J极小的解，C值的选择是很重要的。

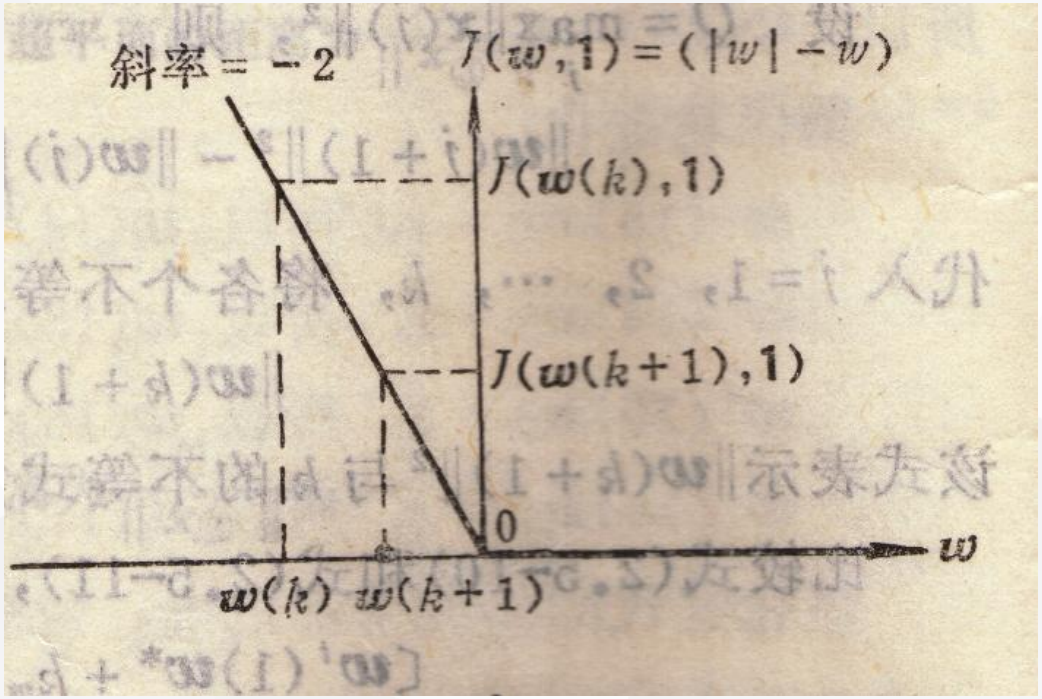
若C值太小，则收敛太慢；

若C值太大，则搜索可能过头，引起发散。

3.8 可训练的确定性分类器的迭代算法

3.8.1 梯度法

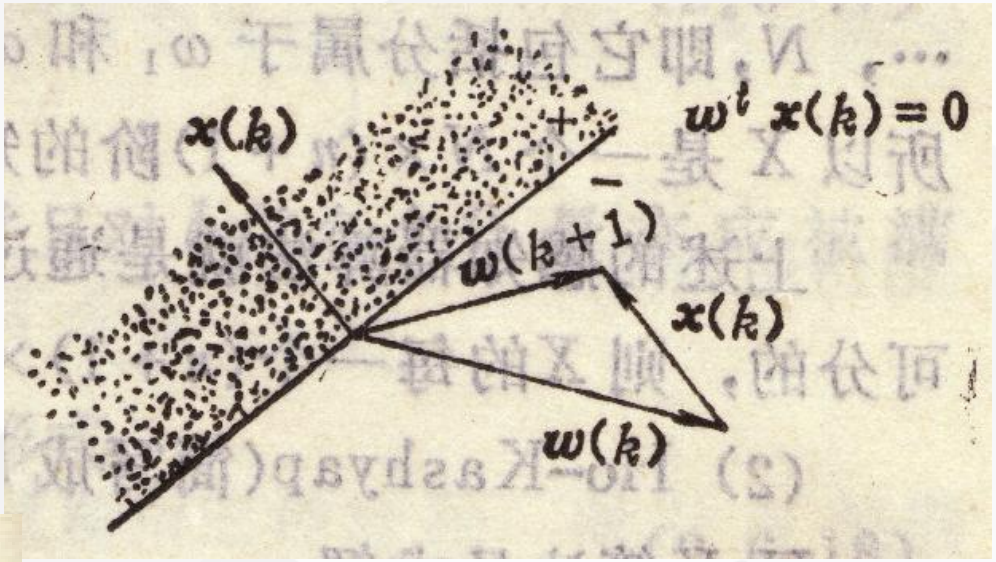
- [例子]

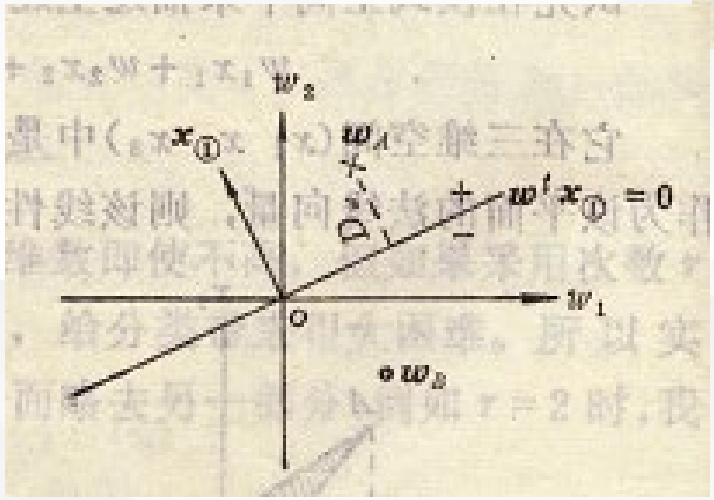


3.8 可训练的确定性分类器的迭代算法

3.8.2 固定增量的逐次调整算法

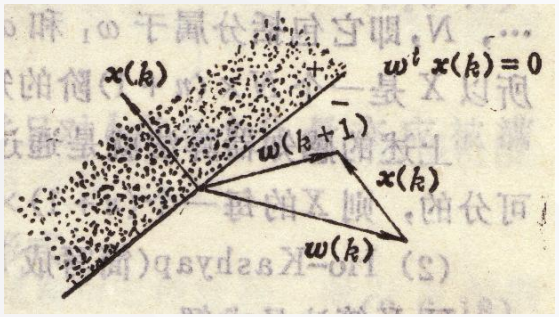
- 描述





3.8 可训练的确定性分类器的迭代算法

3.8.2 固定增量的逐次调整算法



■过程说明：

- 设已由前一步迭代得到$w(k)$的值。
- 读入模式样本x_k，判别$w^{\mathrm{T}}(k)x_k$是否大于0。在示意图中，x_k界定的判别界面为$w^{\mathrm{T}}(k)x_k=0$。当$w(k)$在判别界面的负区域时，$w^{\mathrm{T}}(k)x_k<0$。
- 校正：$w(k+1)= w(k)+ x_k$，这里取C=1。
- 校正后，$w(k+1)$向量比$w(k)$向量更接近于模式x_k所决定的正区域。

3.8.2 固定增量的逐次调整算法

■讨论

- 若模式是线性可分的，选择合适的准则函数$J(w,x)$，算法就能给出解。
- 若模式不是线性可分的，算法的结果就会来回摆动，得不到收敛。

作业

- 采用梯度法和准则函数

$$J(w,x,b)=\frac{1}{8\|x\|^2}\left[\left(w^T x-b\right)-\left|w^T x-b\right|\right]^2$$

式中实数$b>0$，试导出两类模式的分类算法。

3.8 可训练的确定性分类器的迭代算法

3.8.3 最小平方误差(LMSE)算法

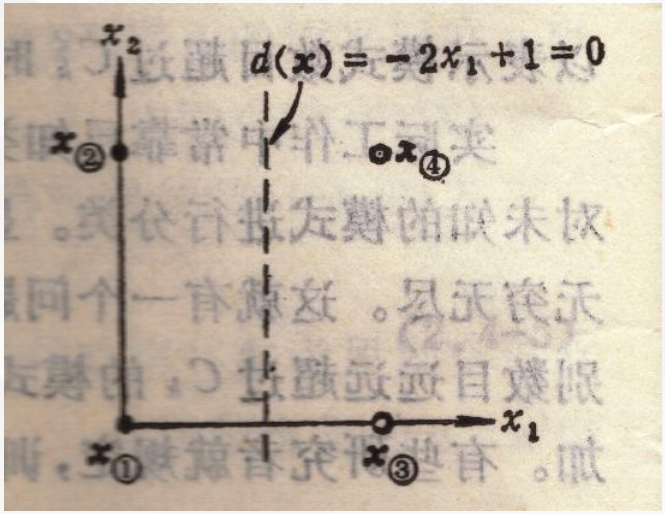
■出发点

- 感知器算法只是当被分模式可用一个特定的判别界面分开时才收敛，在不可分情况下，只要计算程序不终止，它就始终不收敛。
- 即使在模式可分的情况下，也很难事先算出达到收敛时所需要的迭代次数。
- 这样，在模式分类过程中，有时候会出现一次又一次迭代却不见收敛的情况，白白浪费时间。
- 为此需要知道：发生迟迟不见收敛的情况时，到底是由于收敛速度过慢造成的呢，还是由于所给的训练样本集不是线性可分造成的呢?
- 最小平方误差(LMSE)算法，除了对可分模式是收敛的以外，对于类别不可分的情况也能指出来。

3.8 可训练的确定性分类器的迭代算法

3.8.3 最小平方误差(LMSE)算法

- 分类器的不等式方程
- Ho-Kashyap(H-K)算法
- 模式类别可分性的判别
- [例1：有解情况]
- [例2：无解情况]

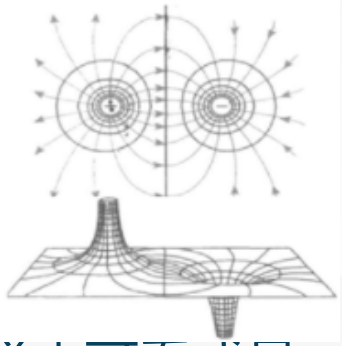


3.8.3 最小平方误差(LMSE)算法

■小结

- 固定增量算法：实现相对简单，可直接引伸到多类模式的分类情况，但未提供模式线性可分的测试特征；
- LMSE算法：相对复杂，需要对$X^{T}X$求逆（维数高时求逆比较困难），但对两类情况，提供了线性可分的测试特征。

3.9 势函数法 — 一种确定性的非线性分类方法



■目的

- 用势函数的概念来确定判别函数和划分类别界面。

■基本思想

- 假设要划分属于两种类别ω_1和ω_2的模式样本，这些样本可看成是分布在n维模式空间中的点$\boldsymbol{x}_k$。
- 把属于ω_1的点比拟为某种能源点，在点上，电位达到峰值。
- 随着与该点距离的增大，电位分布迅速减小，即把样本$\boldsymbol{x}_k$附近空间$\boldsymbol{x}$点上的电位分布，看成是一个势函数$K(\boldsymbol{x}, \boldsymbol{x}_k)$。
- 对于属于ω_1的样本集群，其附近空间会形成一个“高地”，这些样本点所处的位置就是“山头”。
- 同理，用电位的几何分布来看待属于ω_2的模式样本，在其附近空间就形成“凹地”。
- 只要在两类电位分布之间选择合适的等高线，就可以认为是模式分类的判别函数。

3.9 势函数法 — 一种确定性的非线性分类方法

3.9.1 判别函数的产生

- 模式分类的判别函数可由分布在模式空间中的许多样本向量$\{\boldsymbol{x}_k, k=1,2,\ldots$且 $x_k \in (\omega_1 \cup \omega_2)\}$的势函数产生。
- 任意一个样本所产生的势函数以$K(\boldsymbol{x}, \boldsymbol{x}_k)$表征，则判别函数$d(\boldsymbol{x})$可由势函数序列$K(\boldsymbol{x}, \boldsymbol{x}_1)$，$K(\boldsymbol{x}, \boldsymbol{x}_2)$，…来构成，序列中的这些势函数相应于在训练过程中输入机器的训练模式样本$\boldsymbol{x}_1$，$\boldsymbol{x}_2$，…。
- 在训练状态，模式样本逐个输入分类器，分类器就连续计算相应的势函数，在第k步迭代时的积累位势决定于在该步前所有的单独势函数的累加。
- 以$K(\boldsymbol{x})$表示积累位势函数，若加入的训练样本$\boldsymbol{x}_{k+1}$是错误分类，则积累函数需要修改，若是正确分类，则不变。

3.9 势函数法 — 一种确定性的非线性分类方法

3.9.1 判别函数的产生

- 逐步分析
- 从势函数可以看出，积累位势起着判别函数的作用
  - 当x_{k+1}属于ω_1时，$K_k(x_{k+1})>0$；当x_{k+1}属于ω_2时，$K_k(x_{k+1})<0$，则积累位势不做任何修改就可用作判别函数。
- 由于一个模式样本的错误分类可造成积累位势在训练时的变化，因此势函数算法提供了确定ω_1和ω_2两类判别函数的迭代过程。
- 判别函数表达式
  - 取$d(x)=K(x)$，则有：$d_{k+1}(x)=d_k(x)+r_{k+1}K(x, x_{k+1})$

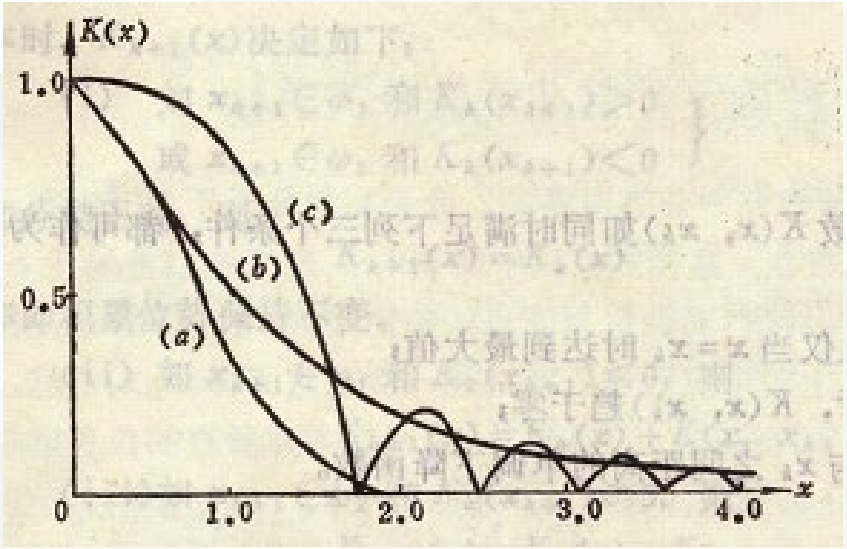
3.9.2 势函数的选择

- 选择势函数的条件：一般来说，若两个n维向量x和x_k的函数$K(x, x_k)$同时满足下列三个条件，则可作为势函数。
  - $K(x, x_k)=K(x_k, x)$，并且当且仅当$x=x_k$时达到最大值；
  - 当向量x与x_k的距离趋于无穷时，$K(x, x_k)$趋于零；
  - $K(x, x_k)$是光滑函数，且是x与x_k之间距离的单调下降函数。

3.9 势函数法 — 一种确定性的非线性分类方法

3.9.2 势函数的选择

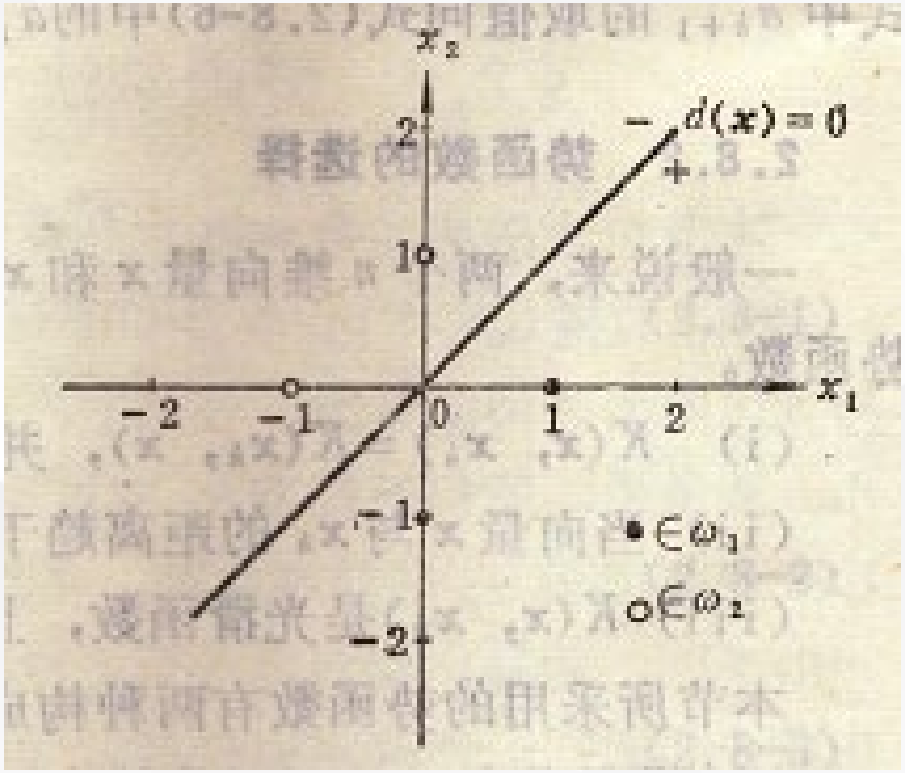
- 构成势函数的两种方式
  - 第一类势函数
  - 第二类势函数



3.9 势函数法 — 一种确定性的非线性分类方法

3.9.2 势函数的选择

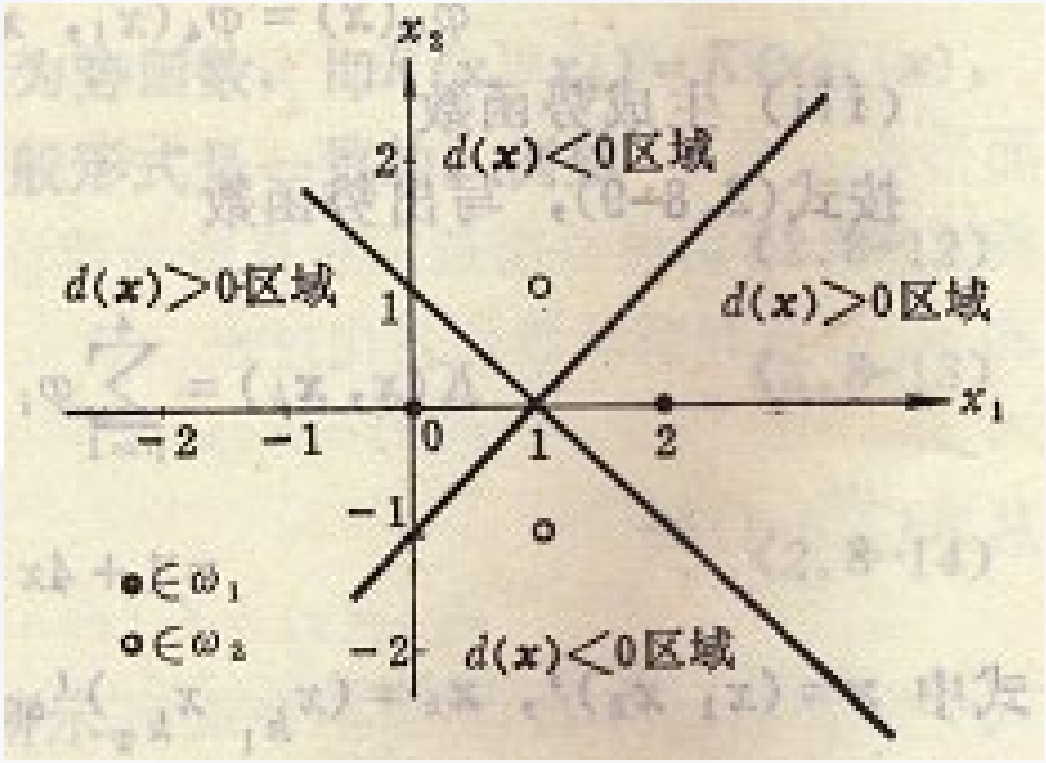
- [例1]



3.9 势函数法 — 一种确定性的非线性分类方法

3.9.2 势函数的选择

- [例2]



3.9.2 势函数的选择

■讨论

- 用第二类势函数，当训练样本维数和数目都较高时，需要计算和存储的指数项较多。
- 正因为势函数由许多新项组成，因此有很强的分类能力。

作业（1）

- 用二次埃尔米特多项式的势函数算法求解以下模式的分类问题

  ω_1: $\{(0\ 1)^T, (0\ -1)^T\}$

  ω_2: $\{(1\ 0)^T, (-1\ 0)^T\}$

作业（2）

- 用下列势函数

$$K(x, x_k) = e^{-\alpha \|x - x_k\|^2}$$

求解以下模式的分类问题

ω_1: $\{(0\ 1)^T, (0\ -1)^T\}$

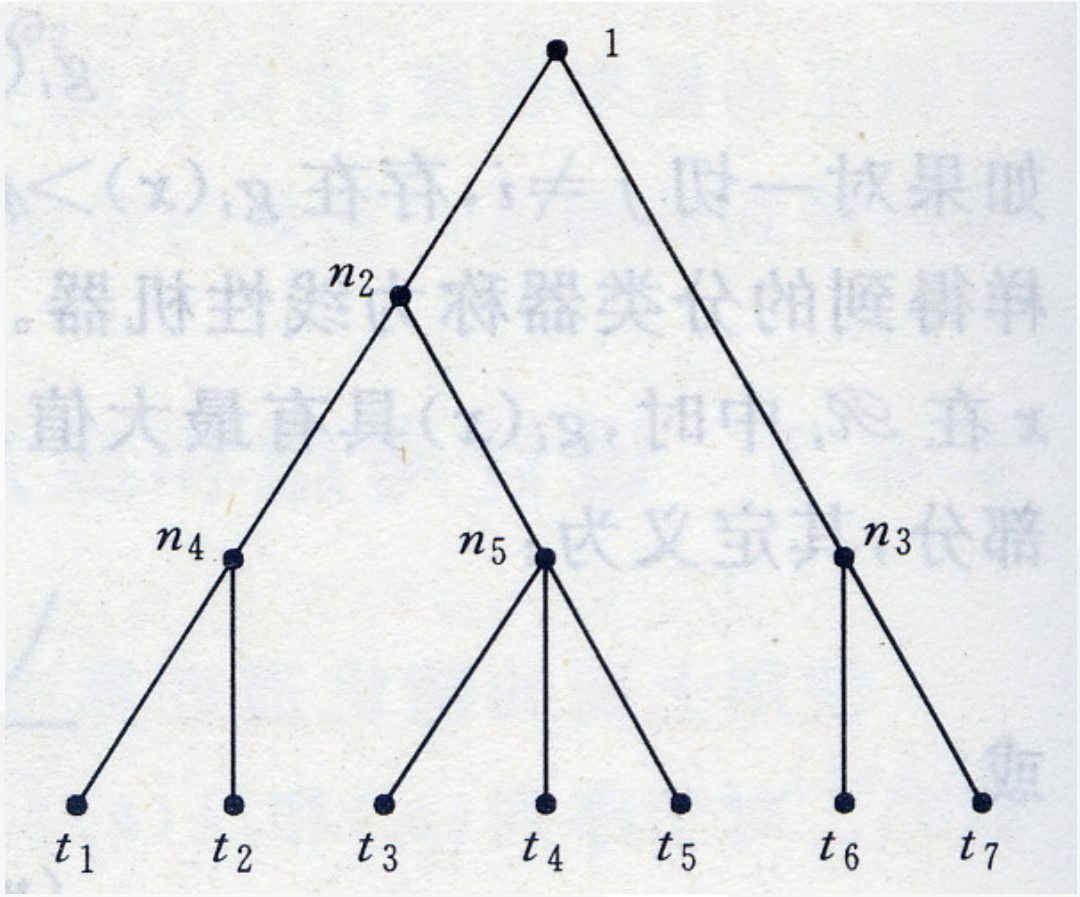
ω_2: $\{(1\ 0)^T, (-1\ 0)^T\}$

3.10 决策树简介

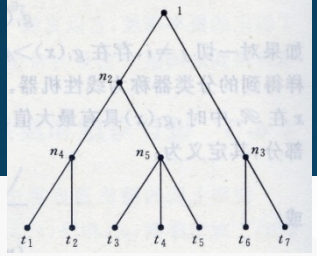
- 决策树，或称多级分类器，是模式识别中进行分类的一种有效方法，对于多类或多峰分布问题，这种方法尤为方便。
- 利用树分类器可以把一个复杂的多类别分类问题，转化为若干个简单的分类问题来解决。
- 它不是企图用一种算法、一个决策规则去把多个类别一次分开，而是采用分级的形式，使分类问题逐步得到解决。

3.10 决策树简介

■决策树示意图

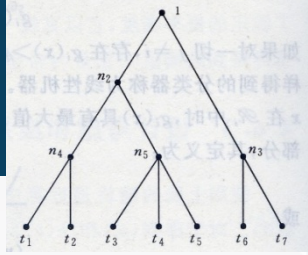


3.10 决策树简介

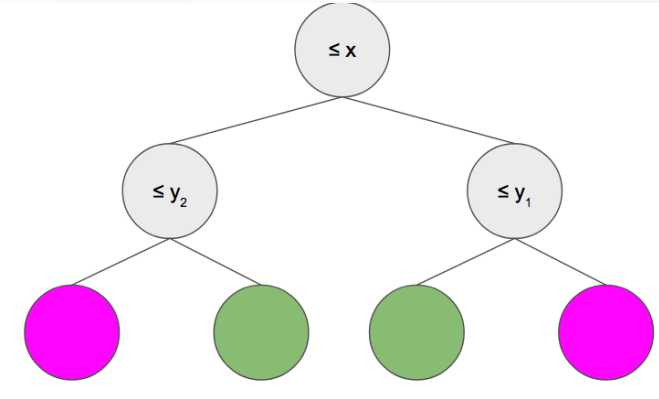


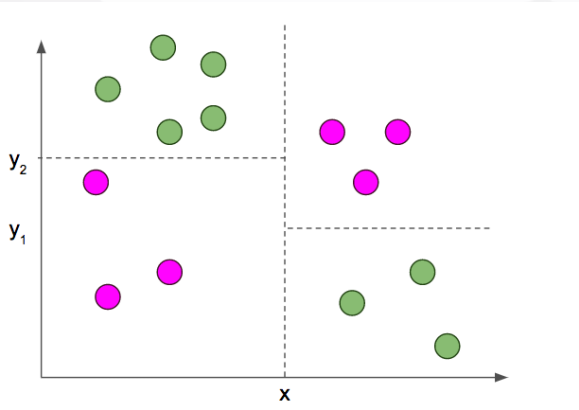
- 一般来讲，一个决策树由一个根节点n_1，一组非终止节点n_i和一些终止节点t_j组成，可对t_j标以各种类别标签，有时不同的终止节点上可以出现相同的类别标签。
- 如果用T表示决策树，则一个决策树T对应于特征空间的一种划分，它把特征空间分成若干个区域，在每个区域中，某类的样本占优势，因此可以标出该类样本的类别标签。

3.10 决策树简介

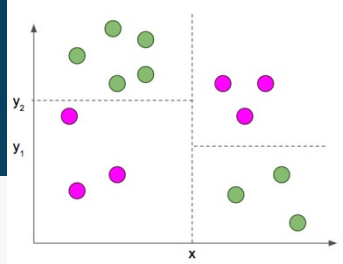


- 决策树的一种简单形式是二叉树，它是指除叶结点外，树的每个节点仅分为两个分支，即每个非终止节点n_i都有且仅有两个子节点n_{il}和n_{ir}。
- 二叉树结构分类器可以把一个复杂的多类别分类问题转化为多级多个两类问题来解决，在每个非终止节点n_i都把样本集分成左右两个子集。





3.10 决策树简介

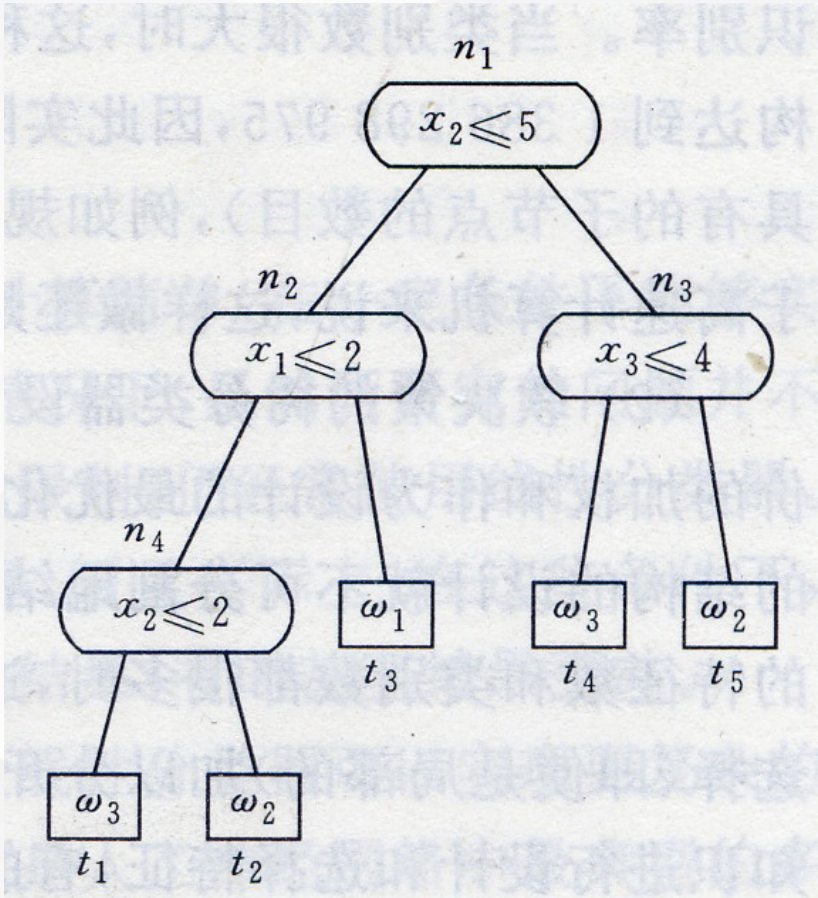


- 分成的每一部分仍然可能包含多个类别的样本，可以把每一部分再分成两个子集，如此下去，直至分成的每一部分只包含同一类别的样本，或某一类样本占优势为止。
- 二叉树结构分类器概念简单、直观、便于解释，而且在各个节点上可以选择不同的特征和采用不同的决策规则，因此设计方法灵活多样，便于利用先验知识来获得一个较好的分类器。

3.10 决策树简介

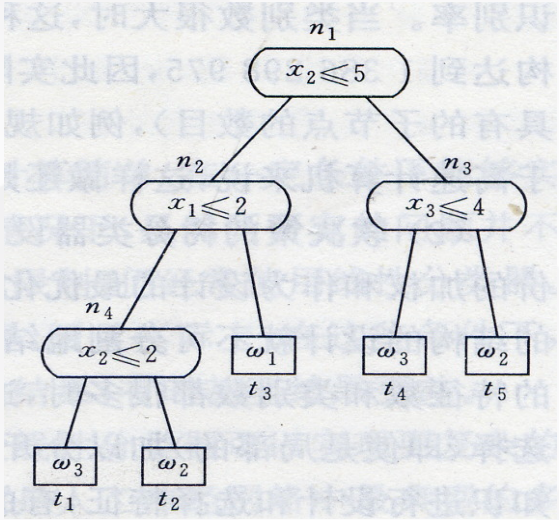
■一个二叉决策树的例子,模式样本

$$\boldsymbol{x} = (x_1, x_2, x_3)^{\mathrm{T}}$$



3.10 决策树简介

■一个二叉决策树的例子



•在此例中，每个节点只选择一个特征，并给出相应的决策阈值。

•对于一个未知样本x，只要从根节点到叶结点，顺序把x的某个特征观测值与相应的阈值相比较，就可做出决策，把x分到相应的分支，最后分到合适的类别中去。如$x=(1,3,6)^T$

3.10 决策树简介

- 在设计一个决策树时，主要应解决以下几个问题：
  - 选择一个合适的树结构，即合理安排树的节点和分支；
  - 确定在每个非终止节点上要使用的特征；
  - 在每个非终止节点上选择合适的决策规则。
- 上述三个问题解决了，决策树的设计也就完成了。二叉树的设计也不例外。

3.10 决策树简介

- 把一个多类别分类问题转化为两类问题的形式是多种多样的，因此，对应的二叉树的结构也是各不相同的。通常的目的是要找一个最优的决策树。
- 一个性能良好的决策树结构应该具有小的错误率和低的决策代价。
- 但是由于很难把错误率的解析表达式和树的结构联系起来，而且在每个节点上所采用的决策规则也仅仅是在该节点上所采用的特征观测值的函数，因此，即使每个节点上的性能都达到最优，也不能说整个决策树的性能达到最优。

3.10 决策树简介

- 在实际问题中，人们往往提出其它一些优化准则，例如极小化整个树的节点数目，或从根节点到叶结点的最大路经长度，或从根节点到叶结点的平均路经长度等，然后采用动态规划的方法，力争设计出能满足某种准则的“最优”决策树。