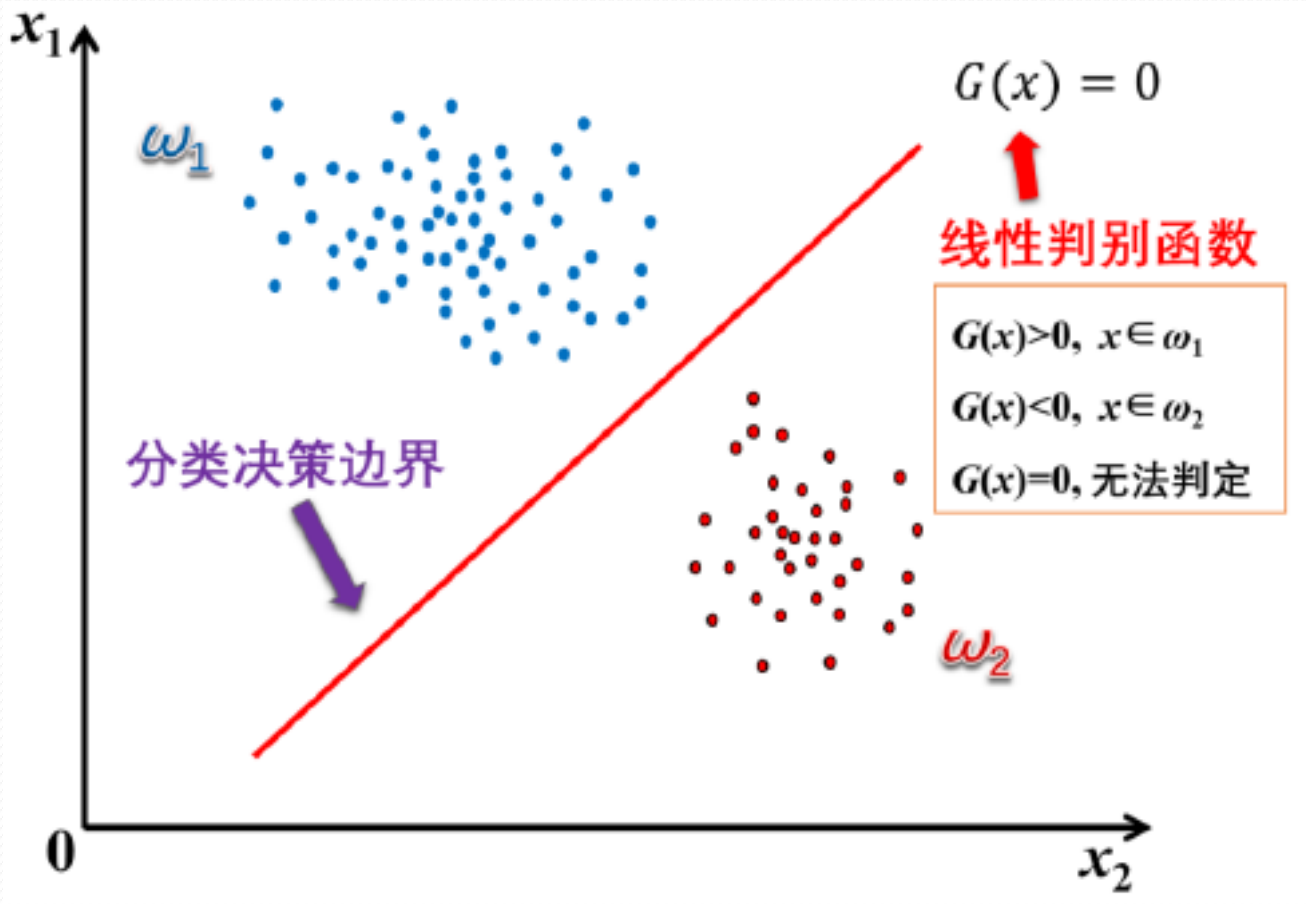
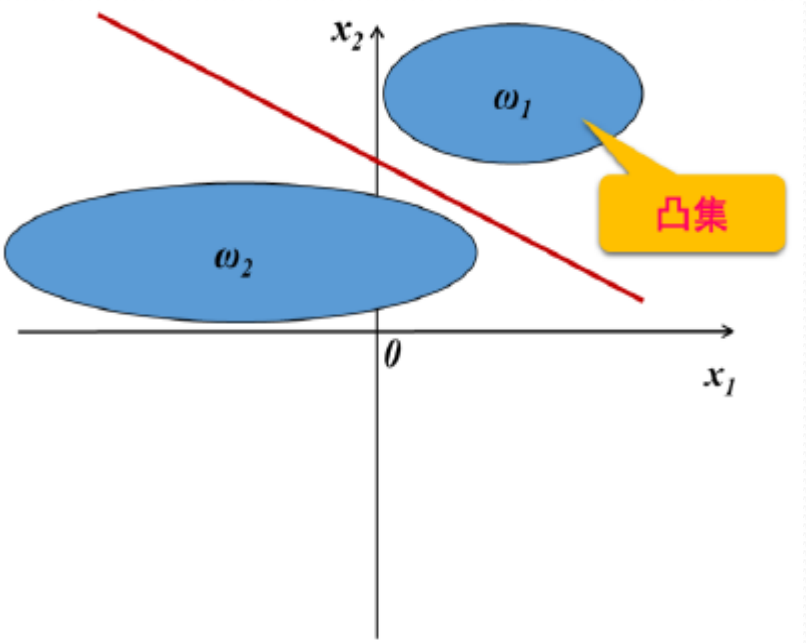
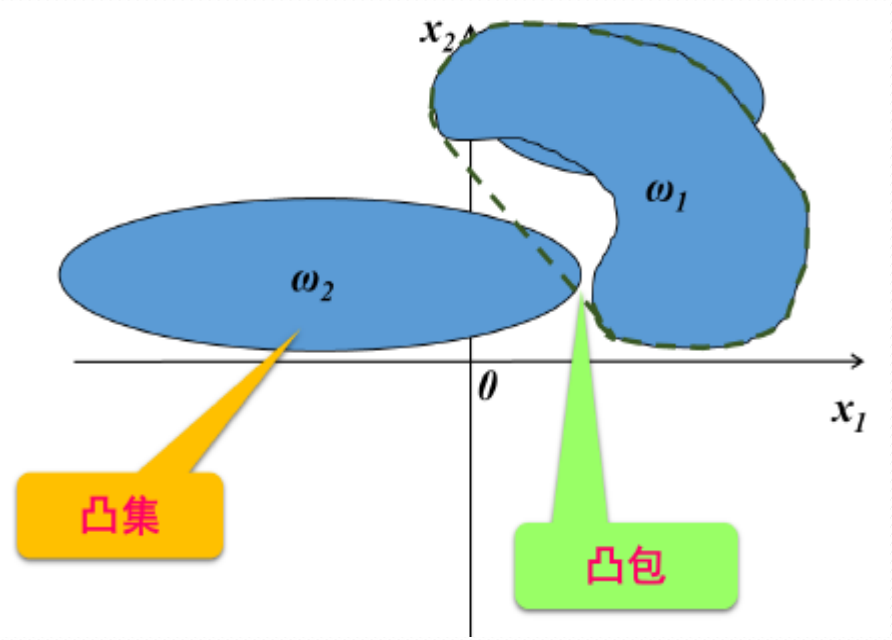
线性判据

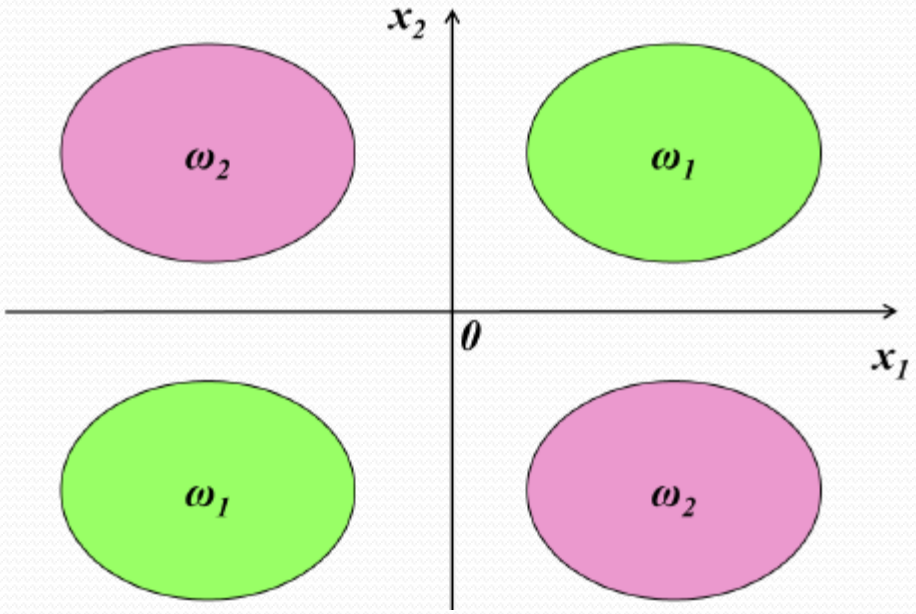

x_1
$G(x) = 0$
ω_1
线性判别函数
$G(x)>0,\ x\in\omega_1$
$G(x)<0,\ x\in\omega_2$
$G(x)=0$, 无法判定
分类决策边界
ω_2
0
x_2



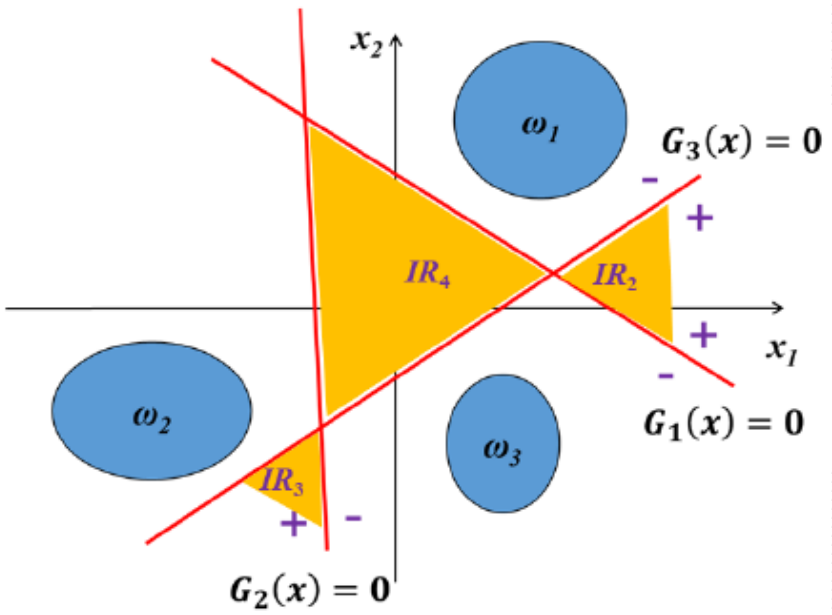
- 如果有一个样本集，它的各个类别样本的分布区域相交，那么肯定是线性不可分的；如果各个类别样本的分布区域互不相交，并且都是凸集，那么它一定是线性可分的。



- 如果虽然各个类别样本的分布区域不相交，但是有的区域是凹集，我们还无法直接判定样本集是否线性可分。



- 如果同一类别样本的分布区域是由不连通的子区域组成的，也会带来线性不可分的问题。
- 典型案例：**异或**问题。

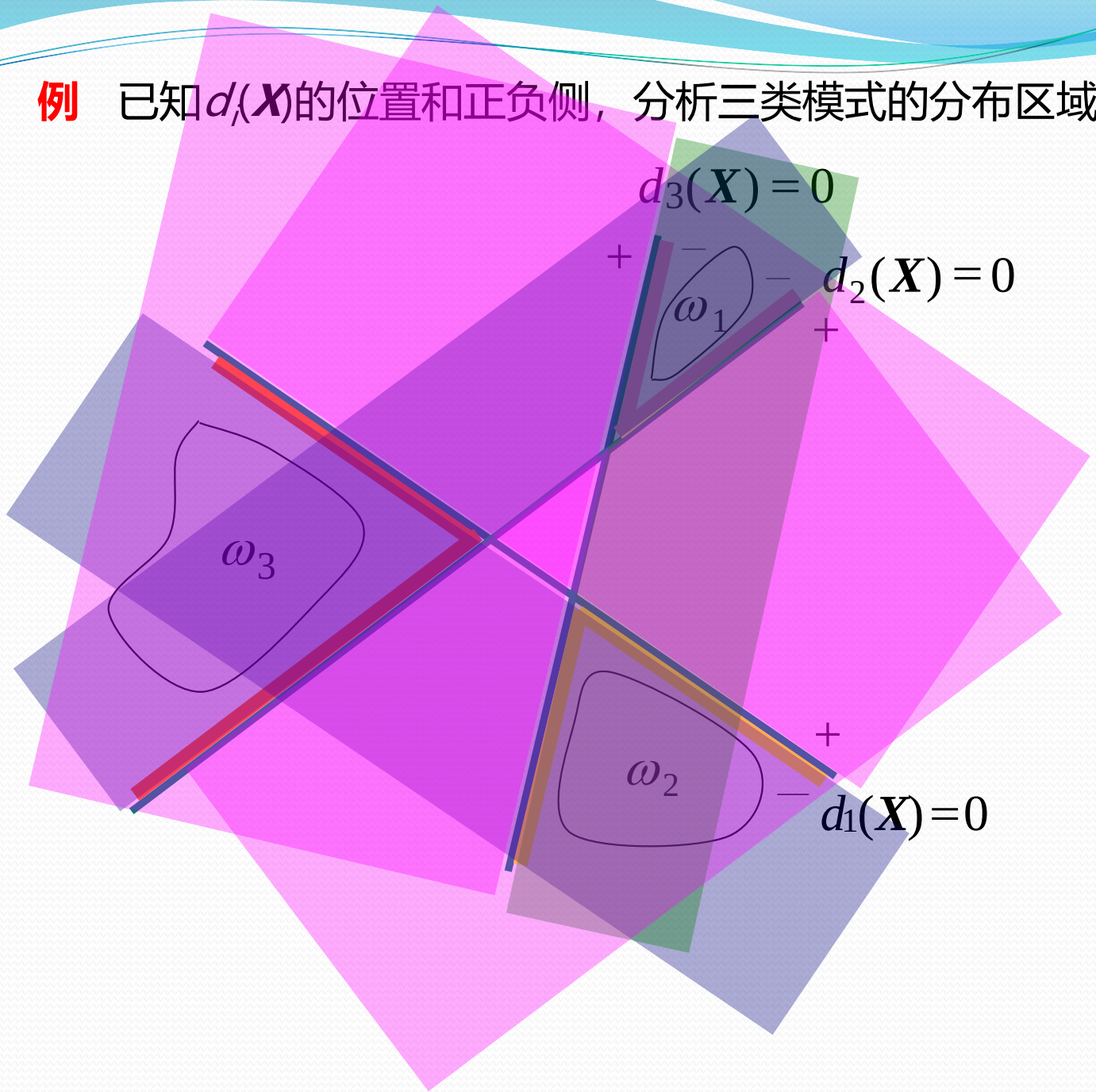


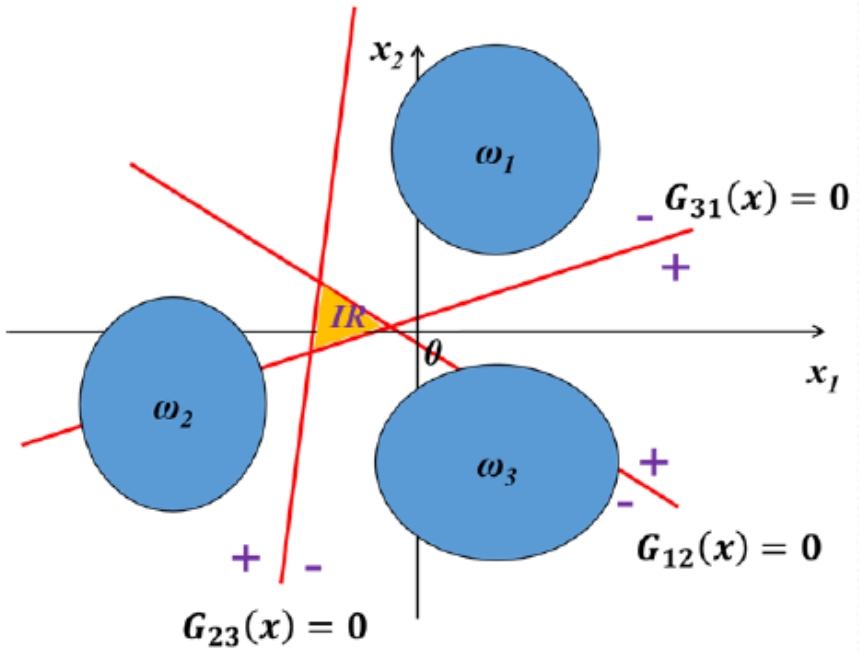
$G_1(x)>0$	√	X	X	√	√	X	X
$G_2(x)>0$	X	√	X	√	X	√	X
$G_3(x)>0$	X	X	√	X	√	√	X
分类决策	$x\in\omega_1$	$x\in\omega_2$	$x\in\omega_3$	IR_1	IR_2	IR_3	IR_4

- 有判别函数G1(x)，当G1(x)>0时，样本属于w1类，当G1(x)<0时，样本不属于w1类。就是说，G1(x)=0这条线性分类决策边界，可以划分对于w1类的正样本和负样本。

- 对于w2类和w3类，也同样有对应的线性判别函数G2(x)和G3(x)。
- 但该方法带来的不可识别区域很多，整体分类器的性能并不好。

例 已知$d_i(\boldsymbol{X})$的位置和正负侧，分析三类模式的分布区域。

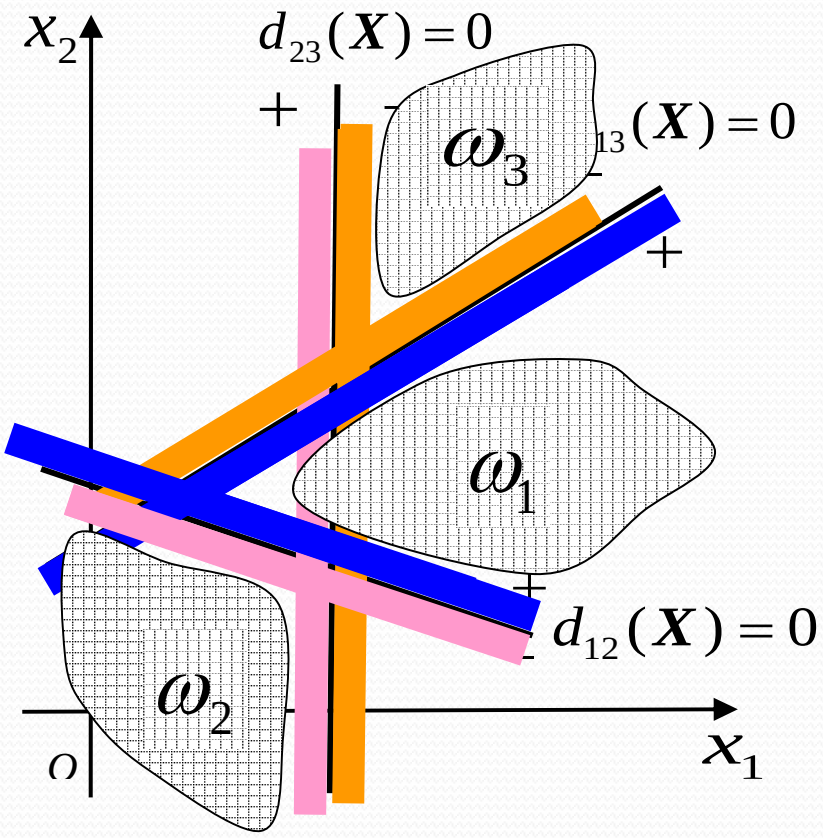


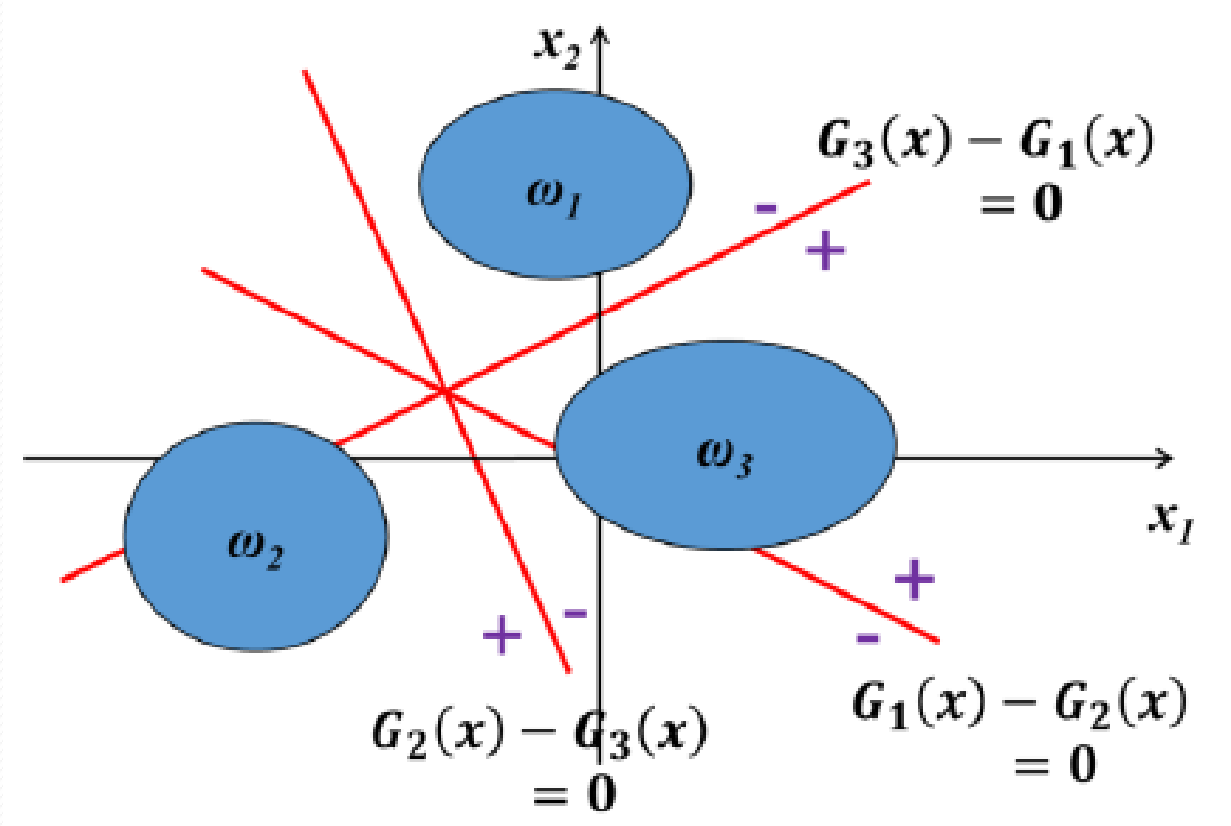


$G_{12}(x)>0$	√	X	-	X
$G_{23}(x)>0$	-	√	X	X
$G_{31}(x)>0$	X	-	√	X
分类决策	$x\in\omega_1$	$x\in\omega_2$	$x\in\omega_3$	*IR*

- $G_{12}(x)>0$，$G_{31}(x)<0$时，我们可以唯一地判定样本属于w1类。此时与w1类无关的两两判别函数$G_{23}(x)$的值不会影响我们的分类决策结果。
- 两两可分的多分类线性判别，不可识别区域比绝对可分的情况大大减少。
- k个类别需要 k(k-1)/2 个判别函数。

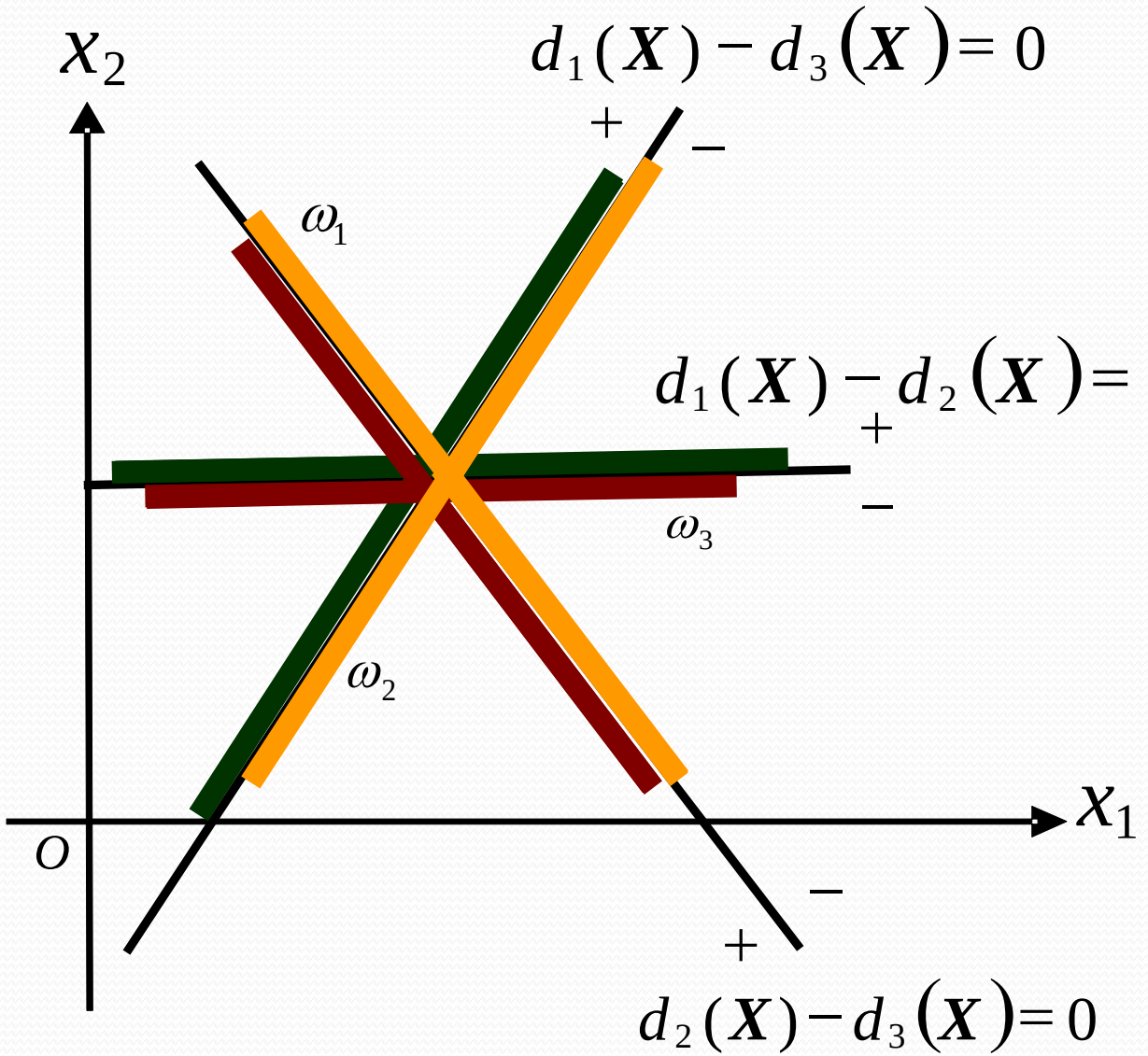
例 已知 $d_{ij}(\boldsymbol{X})$ 的位置和正负侧，分析三类模式的分布区域 。





在“最大值可分”中，样本集中的每个类别对应有一个判别函数，而一个样本将被划分到取值最大的那个判别函数所对应的类别中。

例 已知判决界面的位置和正负侧，分析三类模式的分布区域。



例 一个三类模式（M=3）分类器，其判决函数为：

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

试判断$\boldsymbol{X}_0=[1,1]^T$属于哪一类，且分别给出三类的判决界面。

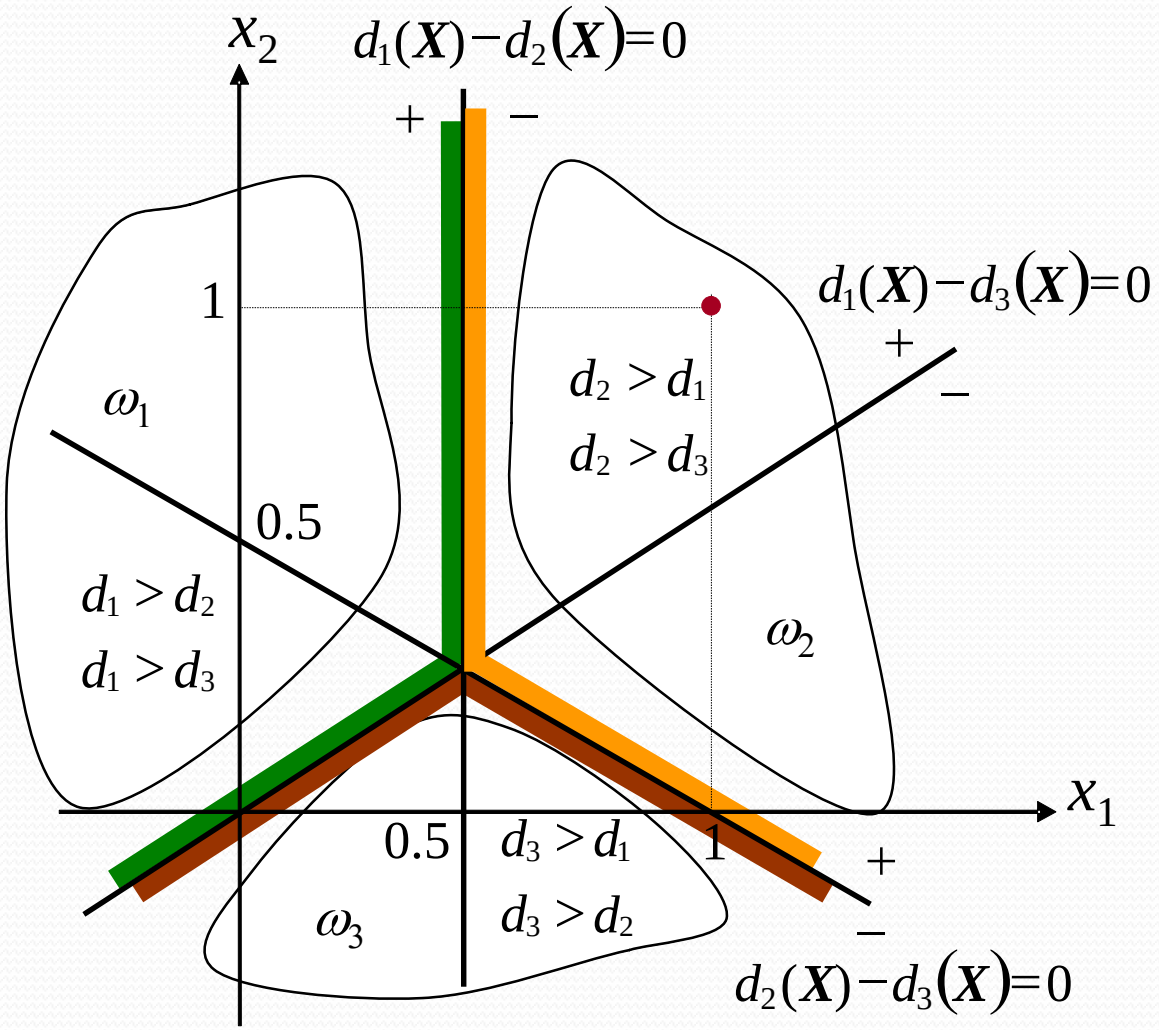
解：①

$$\left.\begin{aligned} d_1(\boldsymbol{X}_0) &= -1+1=0 \\ d_2(\boldsymbol{X}_0) &= 1+1-1=1 \\ d_3(\boldsymbol{X}_0) &= -1 \end{aligned}\right\} \Rightarrow \left.\begin{aligned} d_2(\boldsymbol{X}_0) > d_1(\boldsymbol{X}_0) \\ d_2(\boldsymbol{X}_0) > d_3(\boldsymbol{X}_0) \end{aligned}\right\} \Rightarrow \boldsymbol{X}_0 \in \omega_2$$

② ω_1 类的判决函数：

$$d_1(\boldsymbol{X})-d_2(\boldsymbol{X})=-2x_1+1=0$$

$$d_1(\boldsymbol{X})-d_3(\boldsymbol{X})=-x_1+2x_2=0$$



ω_2 类的判决函数：

$$d_2(\boldsymbol{X})-d_1(\boldsymbol{X})$$

$$=2x_1-1=0$$

$$d_2(\boldsymbol{X})-d_3(\boldsymbol{X})$$

$$=x_1+2x_2-1=0$$

ω_3 类的判决函数：

$$d_{31}(\boldsymbol{X})=-d_{13}(\boldsymbol{X})$$

$$d_{32}(\boldsymbol{X})=-d_{23}(\boldsymbol{X})$$

判决界面如图所示。

线性判据：概述

- 判别模型：

Given $\{\underline{x}|\underline{x}\epsilon C_i, i = 1...k\}$

find the discriminant (decision function) $d(\underline{x}, C_i)$ such that

$$d(\underline{x}, C_i) > d(\underline{x}, C_j) \; \forall \; \underline{x}\epsilon C_i, i \neq j$$

- 线性判据(linear discriminants)：如果两个类之间的边界可以由一个线性函数表达，则该线性函数就是这两个类的线性判据。

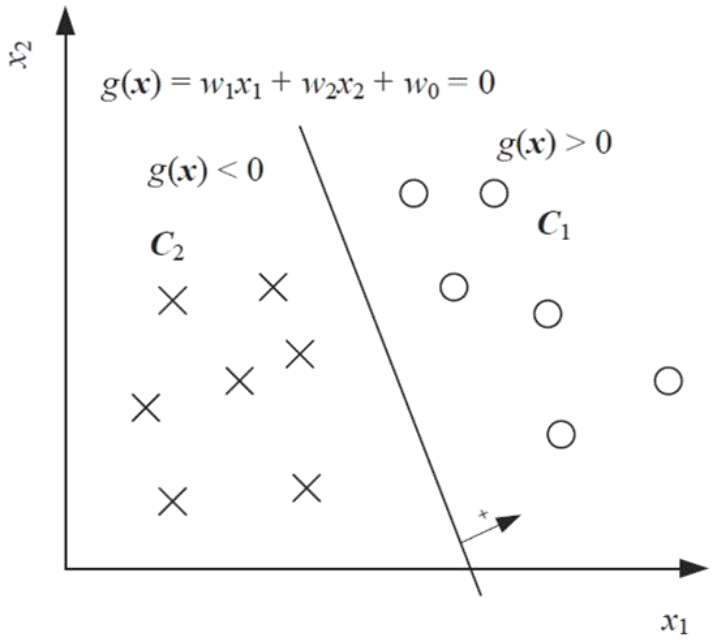
$$d(\underline{x}) = \underline{w}^T \underline{x} + w_0$$

  - 学习过程：已知labeled训练样本，估计线性判据函数参数ω, ω_0 。
  - 分类过程：将测试模式x带入线性判据函数，判断函数值。
    - 对于二分类情况的判别：

$$d(\underline{x}) \begin{matrix} C_1 \\ \gtrless \\ C_2 \end{matrix} 0$$

  - 注意：线性判据不仅限于二类分类，也可以用于多类分类。

- 线性判据的几何示意：



- 线性判据的学习方法：
  - 1）Trial and error：
    - 枚举所有的可能性
    - Guess with learning：猜测与学习结合
  - 2）优化一个目标函数

- 线性判据的优势：
  - 计算量少：在学习和分类过程中，线性判据方法都比基于概率分布的生成模型方法的计算量少。
  - 适用于训练模式较少的情况。此时很难准确估计概率分布。

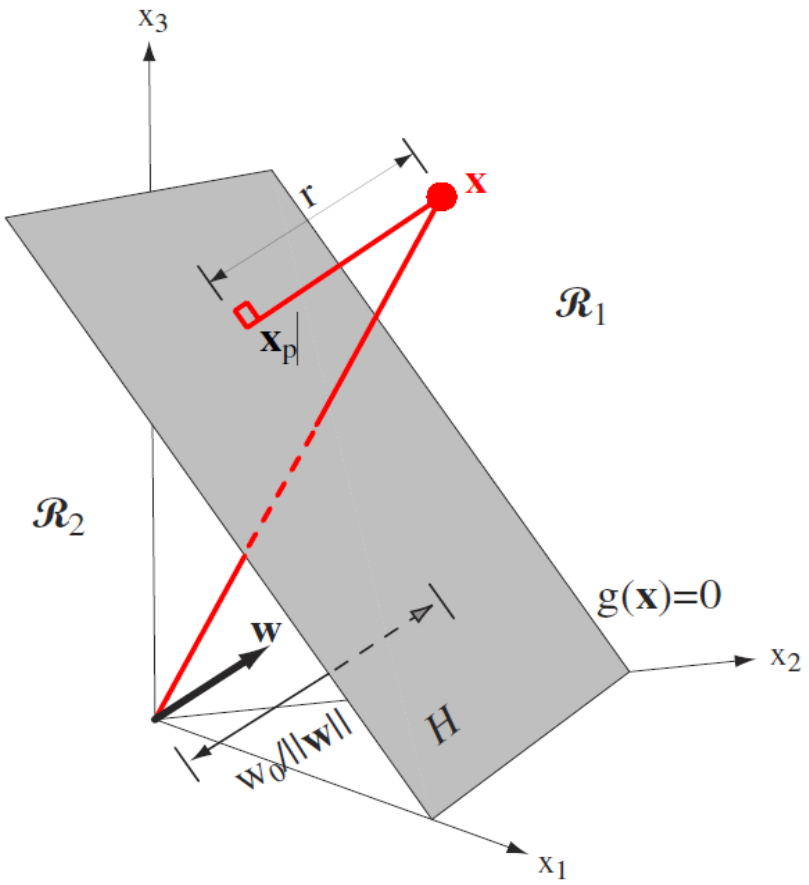
线性判据：几何解释

- w垂直于分类边界上的任何向量：在决策（分类）边界上找两个点x_1和x_2 可得

$$g(\mathbf{x}_1)=0, g(\mathbf{x}_2)=0 \Rightarrow \mathbf{w}^T\mathbf{x}_1+w_0=\mathbf{w}^T\mathbf{x}_2+w_0 \Rightarrow \mathbf{w}^T(\mathbf{x}_1-\mathbf{x}_1)=0$$

- 可见，$\boldsymbol{w}$垂直于决策边界上的任何向量，即$\boldsymbol{w}$垂直于决策边界H，是H的法向量。

- $\boldsymbol{w}$的作用：决定了决策边界H的方向。



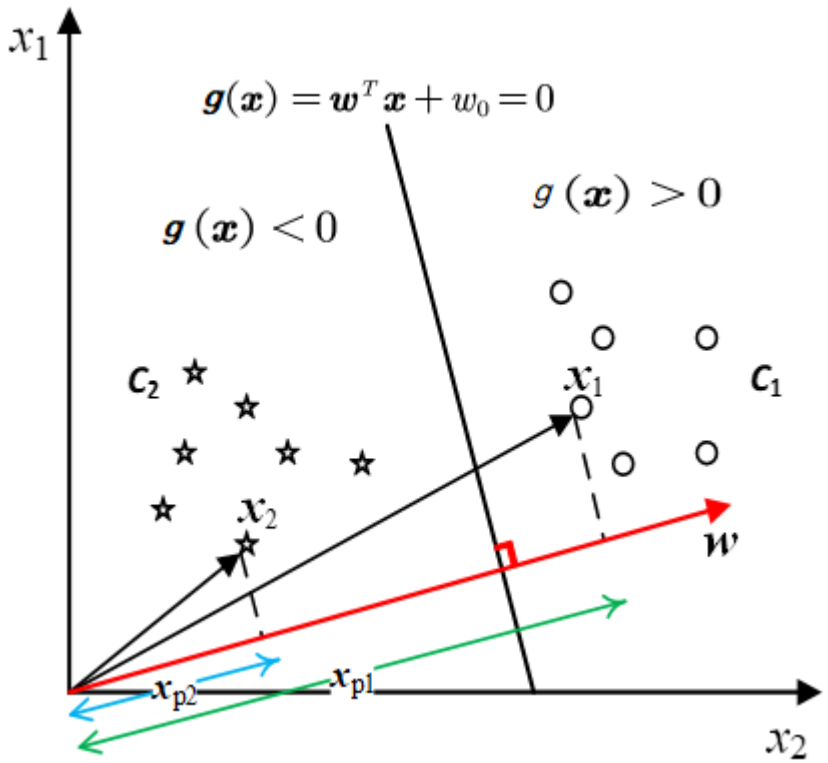
线性判据：几何解释

- 根据点积和投影的关系：$\boldsymbol{w}^T\boldsymbol{x} = x_p||\boldsymbol{w}||$
- 给定任意一对C_1和C_2类样本$\boldsymbol{x}_1$和$\boldsymbol{x}_2$。
- 则可以看出：$\boldsymbol{x}_1$在$\boldsymbol{w}$上的投影总是大于$\boldsymbol{x}_2$在$\boldsymbol{w}$上的投影：

$$\mathbf{w}^T\boldsymbol{x}_1 > \mathbf{w}^T\boldsymbol{x}_2,$$

$$\forall \boldsymbol{x}_1 \in C_1, \boldsymbol{x}_2 \in C_2$$

- 需要找到一个合适的w_0，使得：
$g(\boldsymbol{x}_1) > 0, \quad g(\boldsymbol{x}_2) < 0$



- w_0决定了决策边界的偏移量，使其能够满足两个类输出值分别为正负。

线性判据：几何解释

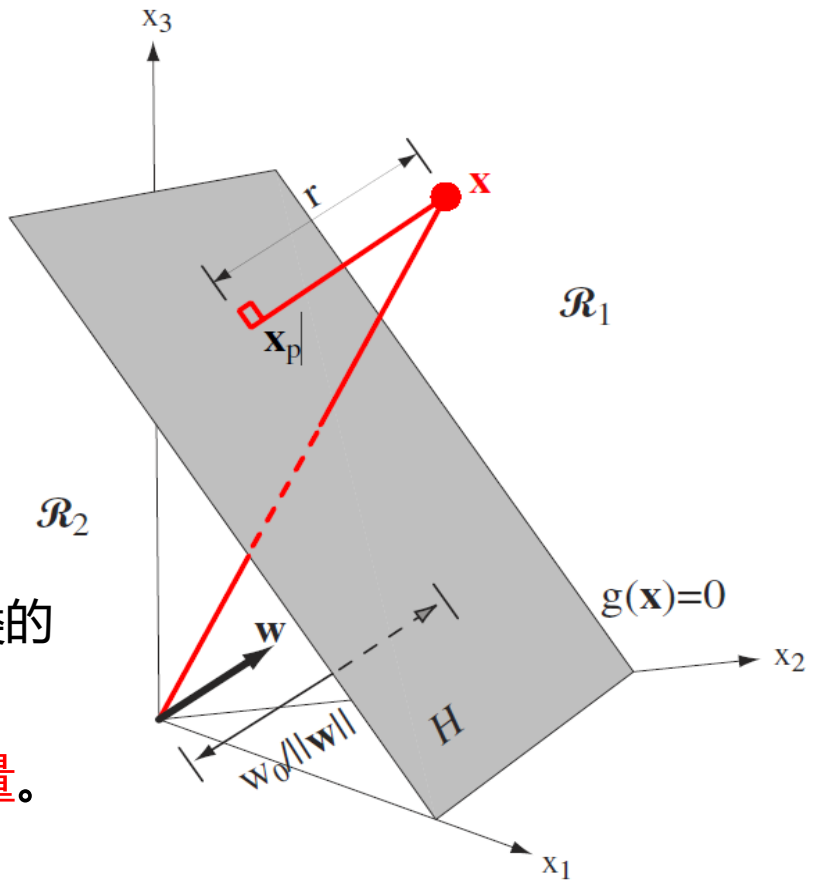
- 给定x_p表示任意样本x投影到决策边界的点， r表示从x到x_p（即到决策边界）的距离（有正负），则x可以重新表达为：

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

$$\Rightarrow \mathbf{w}^T\mathbf{x} + w_0 = \mathbf{w}^T\mathbf{x}_p + w_0 + r\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|}$$

$$\Rightarrow g(\mathbf{x}) = r\|\mathbf{w}\|$$

$$\Rightarrow r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|}$$



- 距离r的绝对值可以作为confidence score：值越大，这个点属于正类或者负类的程度越大。
- g($\boldsymbol{x}$)是样本$\boldsymbol{x}$到决策面H的代数距离度量。

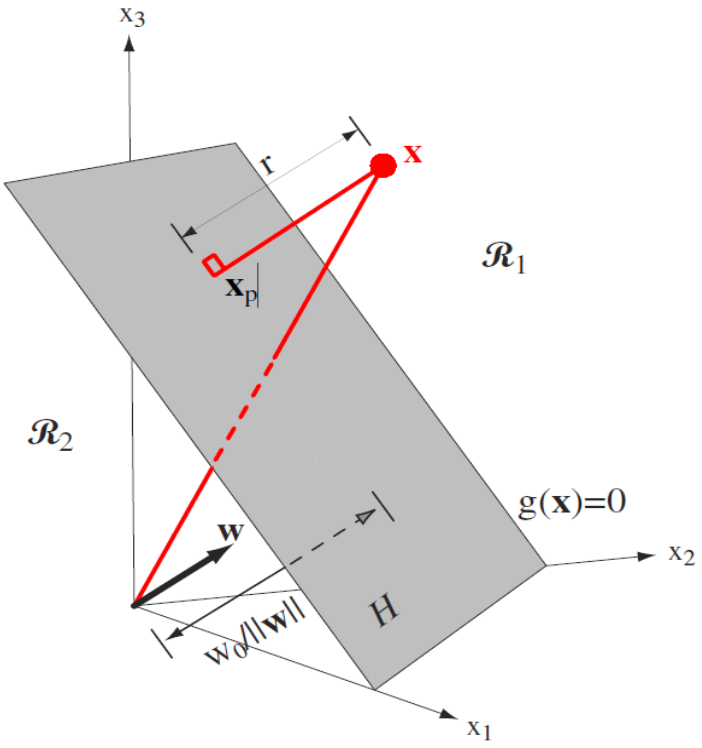
- 如果r_0 表示从x_p（分类边界）到坐标原点$x_0 = 0$的距离，则r_0等于：

$$\mathbf{x}_0 = \mathbf{x}_p + \frac{g(\mathbf{x}_0)\mathbf{w}}{\|\mathbf{w}\|^2}$$

$$\Rightarrow \mathbf{0} = \mathbf{x}_p + \frac{w_0\mathbf{w}}{\|\mathbf{w}\|^2}$$

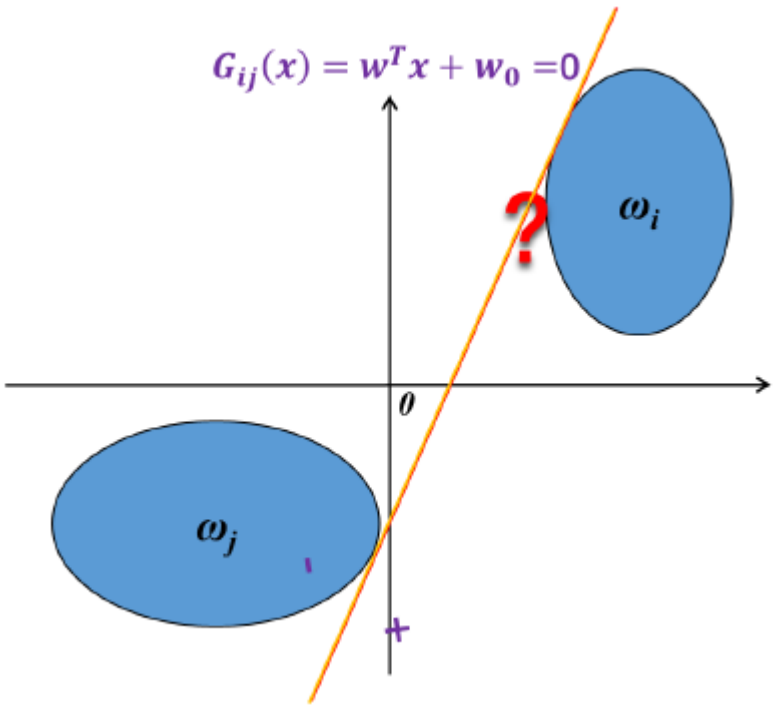
$$\Rightarrow \|\mathbf{x}_p\| = \frac{w_0\|\mathbf{w}\|}{\|\mathbf{w}\|^2}$$

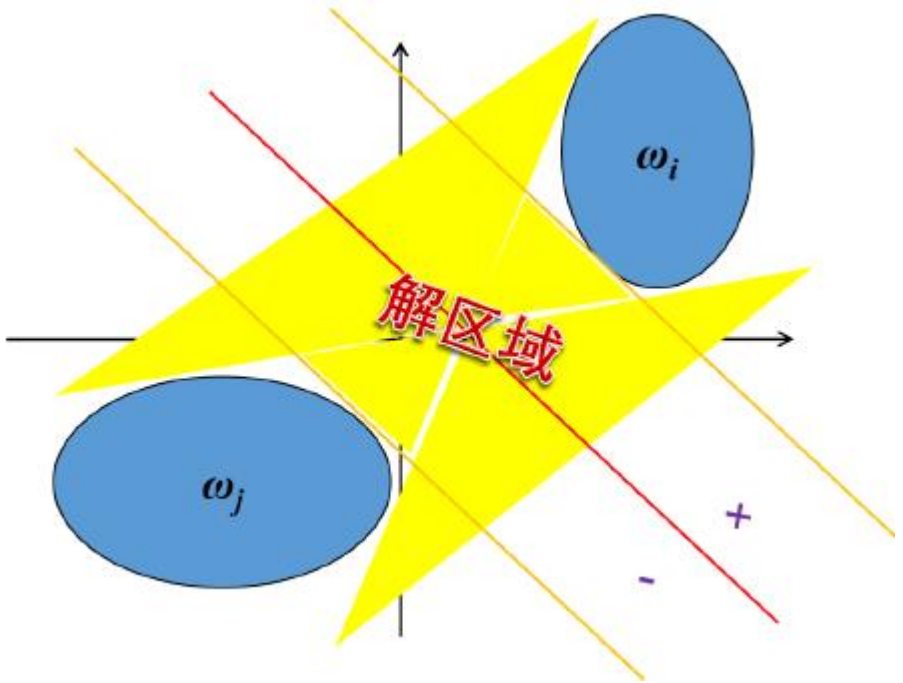
$$\Rightarrow r_0 = \frac{w_0}{\|\mathbf{w}\|}$$



- 可见，w_0决定分类边界相对于坐标原点的位置，w决定分类边界的方向。

线性判据的学习：解区域



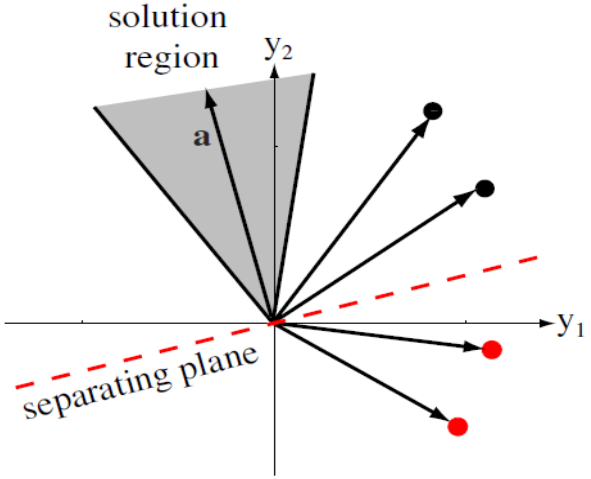


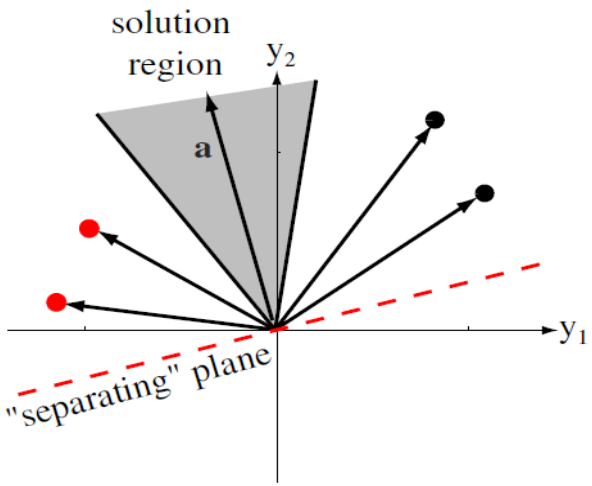
线性判据的学习：概述

- 线性判据的参数不是只有一个解，学习算法就是要从求解区域中找到一个最优解。参数的每个可能解相当于是参数空间的一个点（向量）。
- 解域(solution region)：在参数空间内，参数的所有可能解所处的范围。
- 如果是两类（正负类）分类，将负类的输出值取反，则得到：

$$g(\mathbf{y}) = \mathbf{a}^T\mathbf{y} > 0, \forall y \in C_1 \bigcup C_2$$

- 每个训练样本y_i在参数空间定义了一个通过原点、且垂直于y_i的超平面。
- 给定N个训练样本，参数向量a的解域位于N个超平面正半部分的交集。





Four training samples (black for C_1, red for C_2)

线性判据的学习：目标函数

- 线性判据的学习过程：
  - 线性判据的参数不是只有一个解，学习算法就是要从求解区域中找到一个最优解。
  - 设计目标函数：目标函数反映如何实现有效决策（分类）的核心思想。
    - 常见的目标函数：训练误差、交叉熵……

$$\min_{\theta} \sum_{n=1}^{N} ||f(\boldsymbol{x}_n|\theta) - \boldsymbol{t}_n||_2^2$$

    - 加入正则项（约束条件），提高泛化能力

$$\min_{\theta} \sum_{n=1}^{N} ||f(\boldsymbol{x}_n|\theta) - \boldsymbol{t}_n||_2^2 + \lambda||\theta||$$

线性判据的学习：目标函数

- 求解目标函数：最小化/最大化目标函数
  - 解析求解：求关于训练参数的偏导（梯度），并设置偏导为0.
  - 梯度下降法：先猜测参数初始值，然后不断的根据当前梯度迭代更新参数。
  - 拉格朗日乘数法：约束优化问题（带约束条件）
  - 矩阵特征值/奇异值分解法。。。。。。

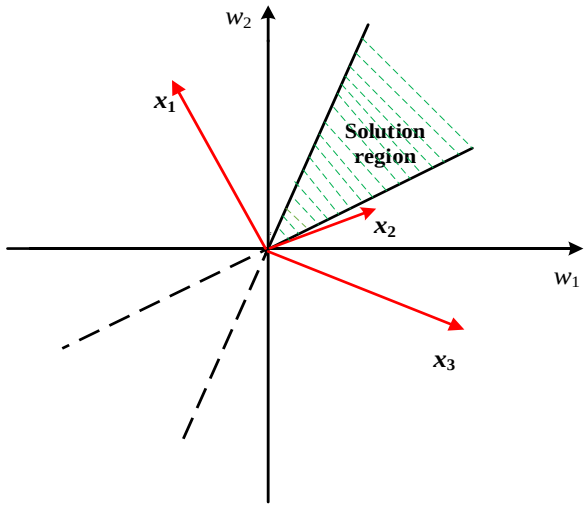
- 常见的线性判据学习算法：
  - 感知机、Fisher判据、Logistic判据、支持向量机

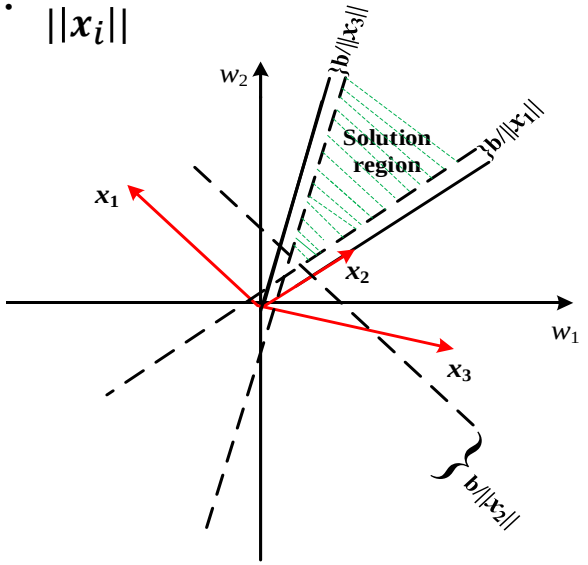
线性判据的学习：约束条件

- 此外，可以在学习算法中加入约束条件，提高泛化能力。
  - 例如，在两类线性判据中，加入边缘（margin）约束b，得到如下约束条件：

$$f(\boldsymbol{x}_n) = \boldsymbol{w}^T \boldsymbol{x}_n > b, \forall \boldsymbol{x}_n$$
$$\text{where } b > 0$$

  - 加入约束后，使得解域范围收缩。
  - 沿着每个样本向量$\boldsymbol{x}_i$方向的收缩距离为：$\frac{b}{||\boldsymbol{x}_i||}$





感知机算法

- 将线性判据函数表达为：$d(x) = w^T x + w_0$
- 线性判据为：$d(\underline{x}) \underset{C_2}{\overset{C_1}{\gtrless}} 0$
- 要根据labeled训练样本，来估计(学习)函数参数：w, w_0
- 感知机算法(perceptron algorithm)
  - 步骤1. 将两个参数合为一个参数，线性函数改写为：

    设$a = \begin{bmatrix} w \\ w_0 \end{bmatrix}$，$y = \begin{bmatrix} x \\ 1 \end{bmatrix}$，则$d(y) = a^T y$

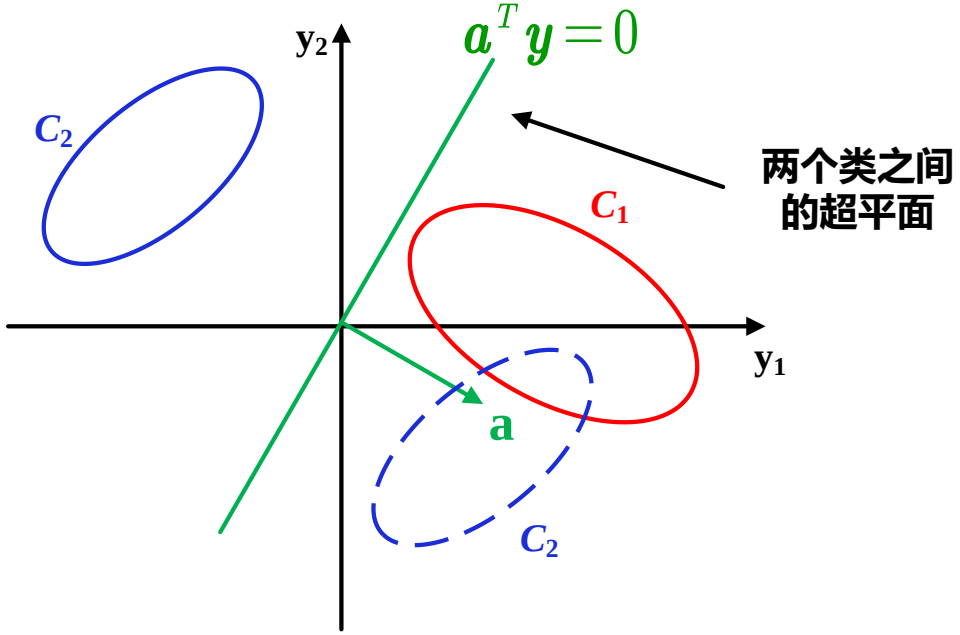
  - 步骤2. 将类2的训练样本全部取反：

    Let $y_i = -y_i$，$\forall i \in C_2$　　从而得到：$a^T y > 0$，$\forall y \in C_1 \bigcup C_2$

  - 步骤3. 用一个递归过程来找到a的解：Guess with learning

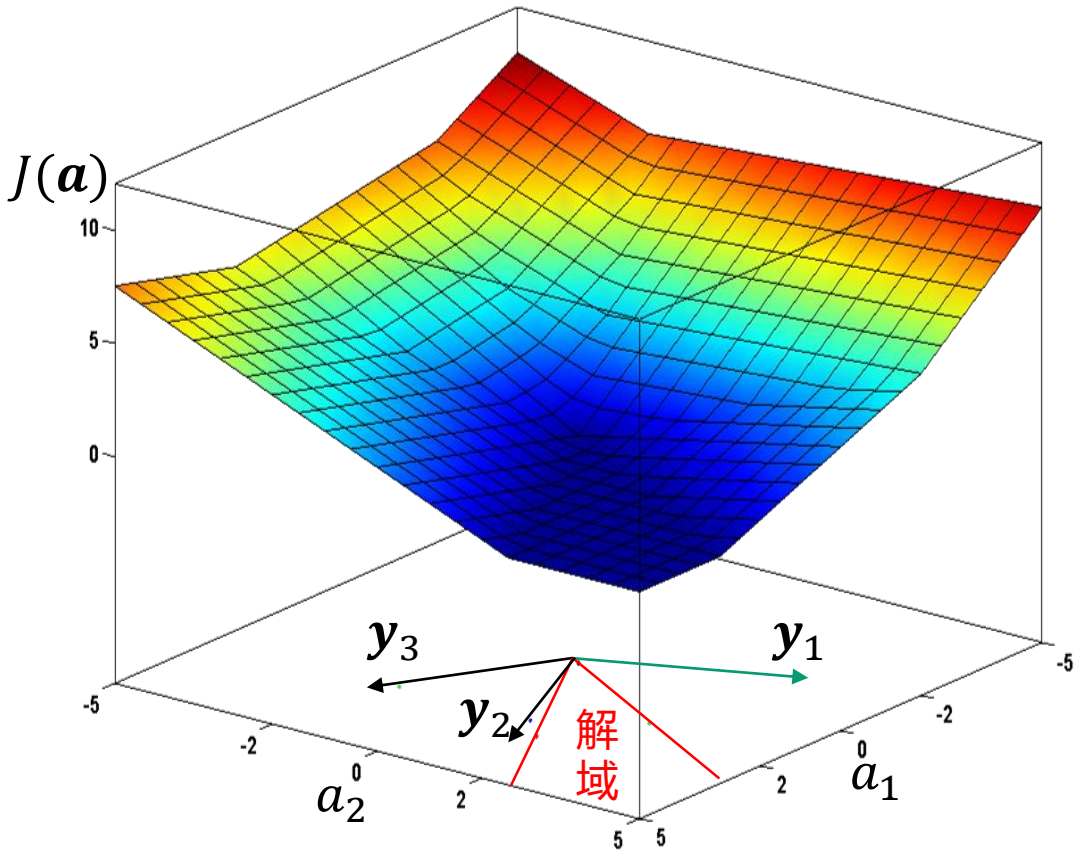
预处理的几何解释

- 在几何上，通过在特征空间上增加一个维度，使得决策边界可以通过原点（w_0项）。
- 翻转C_2类的样本：得到一个平面使得所有样本位于该平面同一侧。



感知机：解域

- 满足$\boldsymbol{a}^T\boldsymbol{y} > 0$的参数$\boldsymbol{a}$**不止**一个。如何从解域里找到最优解?



并行感知机

- 并行感知机算法：定义一个目标函数，通过优化该目标函数来求解a。
- 我们知道学习后的感知机应该满足：

$$g(\mathbf{y}) = \mathbf{a}^T\mathbf{y} > 0, \forall y \in C_1 \bigcup C_2$$ 意味着两个类的输出值都应为正数。

- 目标函数：被错误分类的训练样本输出值之和取反。（根据几何意义，输出值的绝对值越大，错误的程度越大。）

$$J(a) = -\sum_{y \in Y} a^T y, \quad Y = \left\{ y \mid a^T y \le 0 \right\}$$

- 取目标函数关于a的偏导：

$$\frac{\partial J(a)}{\partial a} = -\sum_{y \in Y} y$$

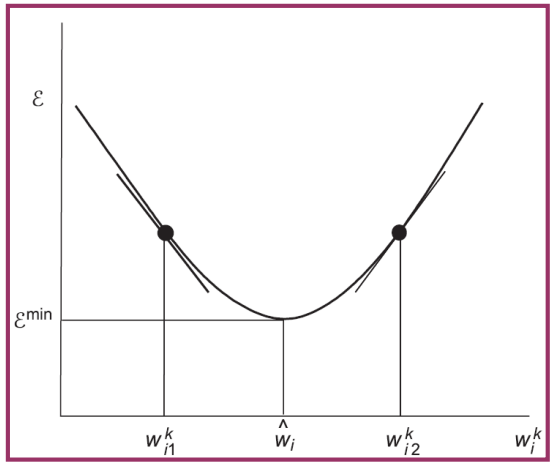
★该目标函数是关于a的一次线性函数。

- 训练误差的表现形式之一。
- 由于参数不一定最优，所以存在上述误差。
- 目标函数是关于参数a的函数。

- 偏导不含有a，所以不能简单的通过设置偏导为0来求解a。

并行感知机：梯度下降法

- 使用梯度下降法：使用当前梯度值迭代更新参数。
  - 注意：通常参数a是多维向量，梯度也是多维向量。梯度下降法的更新操作是每个维度各自独立进行的。
  - 更新的方向（正负）：在当前参数取值a_k下，每个维度的梯度**反**方向就是该维度往目标函数最小值收敛的最速下降方向（steepest descent）。
  - 更新的大小：每个维度的梯度幅值代表参数在该维度上的更新程度。
  - 通常加入步长(η_k)来调整更新的幅度。每次迭代可以用不同的步长。



并行感知机算法

- 使用最速下降(steepest descent)算法。

步长

$$a_{k+1} = a_k - \rho_k \left.\frac{\partial J(a)}{\partial a}\right|_{a=a_k}$$

在a_k点下降最快的方向

从一个任意的初始权向量出发，沿**准则函数值下降最快**的方向，也就是**负梯度方向**对权向量进行一步步修正，直到获得全局最优解为止。

- 如果设步长为1，可以得到：

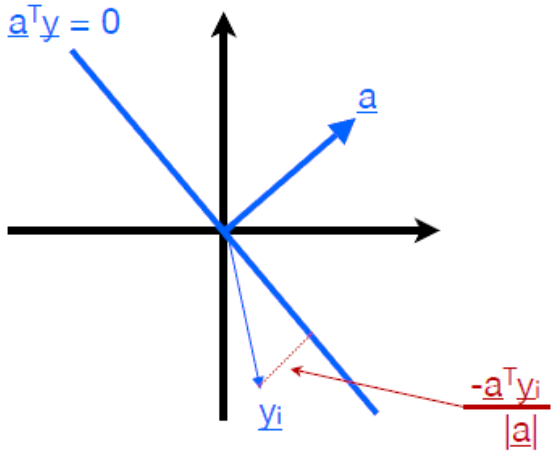
$$a_{k+1} = a_k + \sum_{Y(a_k)} y_i, \quad 其中：\ Y(a_k) = \{y \mid a_k^T y \le 0\}$$

- 并行感知机算法步骤如下：

$$a_0 = 0$$

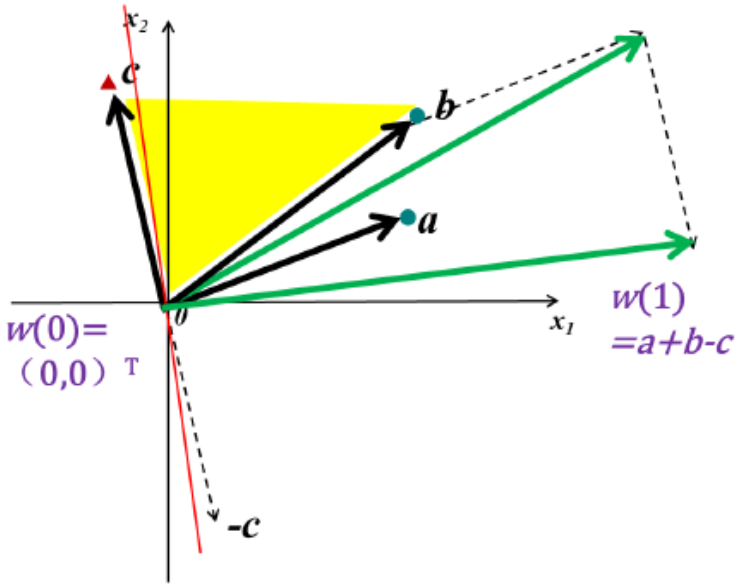
$$a_{k+1} = a_k + \sum_{Y(a_k)} y_i$$

- 并行感知机算法的优势：收敛的速度快。



并行感知机：梯度下降法算法流程

- 并行感知机算法流程：
  - 初始化参数：$a_0, \eta(\cdot), \theta$
  - 迭代更新：基于当前梯度更新参数a，然后基于新参数更新训练误差（同时也是更新了错误分类样本集合Y_k）。
  - 判断停止条件：所有训练样本的输出值都大于0，或者更新值小于阈值θ。



假设我们取初始权向量为0 向量，显然所有的样本都将被错分。若取调整步长为1，下一步的w(1)，就等于0 向量加上所有样本的向量和。

串行感知机

- 串行感知机：训练样本是一个个串行给出的，称作fix-increment感知机。
- 目标函数：如果当前样本是错误分类了，则以它的输出值取反作为目标函数。否则，目标函数是0.

$$J(\mathbf{a}) = \begin{cases} -\mathbf{a}^T\mathbf{y} & \text{if } \mathbf{a}^T\mathbf{y} < 0 \\ 0 & \text{otherwise} \end{cases}$$

- 最小化目标函数：取关于参数向量a的偏导

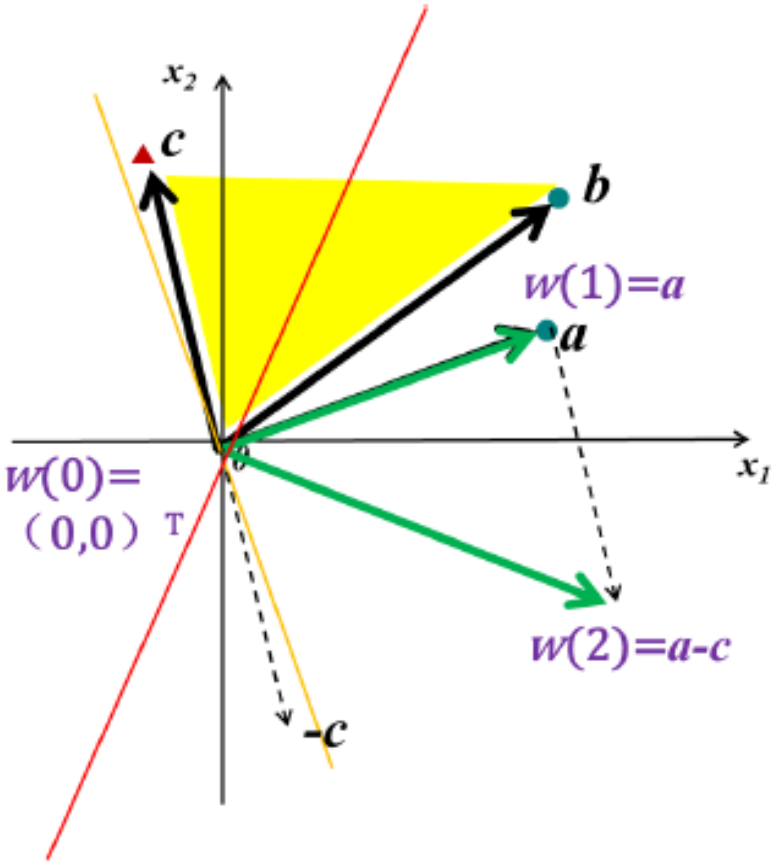
$$\frac{\partial J(\mathbf{a})}{\partial \mathbf{a}} = \begin{cases} -\mathbf{y} & \text{if } \mathbf{a}^T\mathbf{y} < 0 \\ 0 & \text{otherwise} \end{cases}$$

- 梯度下降法：给定当前训练样本y_k，就用当前梯度值更新一次参数。直至所有训练样本都被正确分类。

$$\mathbf{a}_{k+1} = \mathbf{a}_k - \eta_k \nabla \mathbf{J}(\mathbf{a}_k)$$

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \eta_k \mathbf{y}_k$$

串行感知机：参数更新过程图示



取初始权向量为0向量，依次处理a、b、c三个样本。

感知机算法是一种赏罚过程：

分类正确时，对权向量“赏”——即“不罚”，即权向量不变；
分类错误时，对权向量“罚”——即修改，向正确的方向转换。

（1）选择N个分属于ω_1和ω_2类的模式样本构成训练样本集$\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$，构成**增广向量形式**，并进行**规范化处理**。任取权向量初始值$\boldsymbol{W}(1)$，开始迭代。迭代次数$k=1$。

（2）用全部训练样本进行一轮迭代，计算$\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i$的值，并修正权向量。

分两种情况，更新权向量的值：

① 若$\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i \le 0$，　分类器对第$i$个模式做了错误分类，

权向量校正为：　$\boldsymbol{W}(k+1)=\boldsymbol{W}(k)+c\boldsymbol{X}_i$　c：正的校正增量。

② 若$\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i > 0$，　分类正确，权向量不变：

$$\boldsymbol{W}(k+1)=\boldsymbol{W}(k)$$

统一写为：

$$\boldsymbol{W}(k+1)=\begin{cases}\boldsymbol{W}(k) & 若\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i > 0\\ \boldsymbol{W}(k)+c\boldsymbol{X}_i & 若\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i \le 0\end{cases}$$

③ 分析分类结果：只要有一个错误分类，回到②，直至对所有样本正确分类。

收敛：经过算法的有限次迭代运算后，求出了一个使所有样本都能正确分类的W，则称算法是收敛的。

收敛条件：模式类别线性可分。

例 已知两类训练样本

$$\omega_1：\ \boldsymbol{X}_1=[0,0]^{\mathrm{T}} \qquad \boldsymbol{X}_2=[0,1]^{\mathrm{T}}$$

$$\omega_2：\ \boldsymbol{X}_3=[1,0]^{\mathrm{T}} \qquad \boldsymbol{X}_4=[1,1]^{\mathrm{T}}$$

用感知器算法求出将模式分为两类的权向量解和判别函数。

解：所有样本写成增广向量形式；

进行规范化处理，属于ω_2的样本乘以(－1)。

$$\boldsymbol{X}_1=[0,0,1]^{\mathrm{T}} \quad \boldsymbol{X}_2=[0,1,1]^{\mathrm{T}} \quad \boldsymbol{X}_3=[-1,0,-1]^{\mathrm{T}} \quad \boldsymbol{X}_4=[-1,-1,-1]^{\mathrm{T}}$$

任取$\boldsymbol{W}(1)=\mathbf{0}$，取$c=1$，迭代过程为：

第一轮：

$$\boldsymbol{W}^{\mathrm{T}}(1)\boldsymbol{X}_1=[0,0,0]\begin{bmatrix}0\\0\\1\end{bmatrix}=0,\le 0,\qquad 故\boldsymbol{W}(2)=\boldsymbol{W}(1)+\boldsymbol{X}_1=[0,0,1]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(2)\boldsymbol{X}_2=[0,0,1]\begin{bmatrix}0\\1\\1\end{bmatrix}=1,>0,\qquad 故\boldsymbol{W}(3)=\boldsymbol{W}(2)=[0,0,1]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(3)\boldsymbol{X}_3=[0,0,1]\begin{bmatrix}-1\\0\\-1\end{bmatrix}=-1,\le 0,\qquad 故\boldsymbol{W}(4)=\boldsymbol{W}(3)+\boldsymbol{X}_3=[-1,0,0]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(4)\boldsymbol{X}_4=[-1,0,0]\begin{bmatrix}-1\\-1\\-1\end{bmatrix}=1,>0,\qquad 故\boldsymbol{W}(5)=\boldsymbol{W}(4)=[-1,0,0]^{\mathrm{T}}$$

有两个$\boldsymbol{W}^{\mathrm{T}}(k)\boldsymbol{X}_i\le 0$的情况（错判），进行第二轮迭代。

第二轮：

$$\boldsymbol{W}^{\mathrm{T}}(5)\boldsymbol{X}_1=0\le 0,\quad 故\boldsymbol{W}(6)=\boldsymbol{W}(5)+\boldsymbol{X}_1=[-1,0,1]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(6)\boldsymbol{X}_2=1>0,\quad 故\boldsymbol{W}(7)=\boldsymbol{W}(6)=[-1,0,1]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(7)\boldsymbol{X}_3=0\le 0,\quad 故\boldsymbol{W}(8)=\boldsymbol{W}(7)+\boldsymbol{X}_3=[-2,0,0]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(8)\boldsymbol{X}_4=2>0,\quad 故\boldsymbol{W}(9)=\boldsymbol{W}(8)=[-2,0,0]^{\mathrm{T}}$$

第三轮：

$$\boldsymbol{W}^{\mathrm{T}}(9)\boldsymbol{X}_1=0\le 0,\quad 故\boldsymbol{W}(10)=\boldsymbol{W}(9)+\boldsymbol{X}_1=[-2,0,1]^{\mathrm{T}}$$

$$\boldsymbol{W}^{\mathrm{T}}(10)\boldsymbol{X}_2=1>0,\quad 故\boldsymbol{W}(11)=\boldsymbol{W}(10)$$

$$\boldsymbol{W}^{\mathrm{T}}(11)\boldsymbol{X}_3=1>0,\quad 故\boldsymbol{W}(12)=\boldsymbol{W}(11)$$

$$\boldsymbol{W}^{\mathrm{T}}(12)\boldsymbol{X}_4=1>0,\quad 故\boldsymbol{W}(13)=\boldsymbol{W}(12)$$

第四轮：

$$\boldsymbol{W}^{\mathrm{T}}(13)\boldsymbol{X}_1=1>0,\quad 故\boldsymbol{W}(14)=\boldsymbol{W}(13)$$

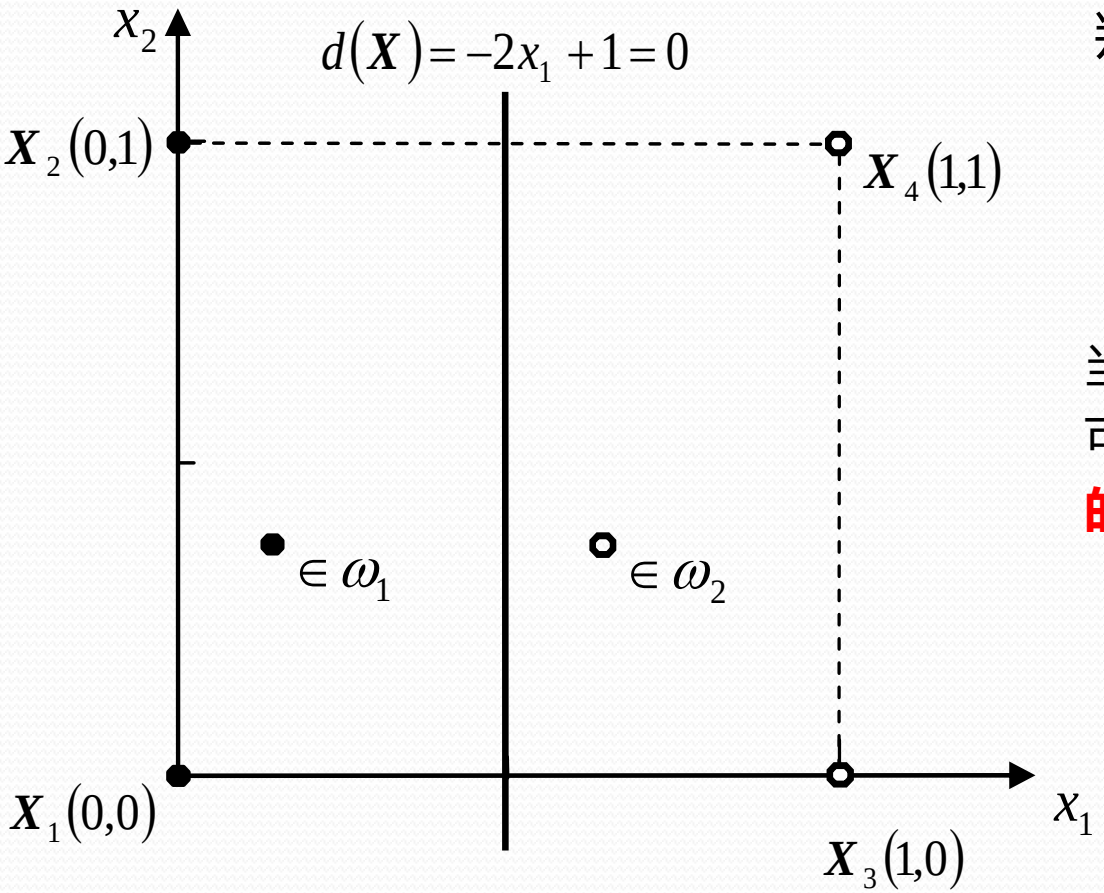
$$\boldsymbol{W}^{\mathrm{T}}(14)\boldsymbol{X}_2=1>0,\quad 故\boldsymbol{W}(15)=\boldsymbol{W}(14)$$

$$\boldsymbol{W}^{\mathrm{T}}(15)\boldsymbol{X}_3=1>0,\quad 故\boldsymbol{W}(16)=\boldsymbol{W}(15)$$

$$\boldsymbol{W}^{\mathrm{T}}(16)\boldsymbol{X}_4=1>0,\quad 故\boldsymbol{W}(17)=\boldsymbol{W}(16)$$

该轮迭代的分类结果全部正确，故解向量 $\boldsymbol{W}=[-2,0,1]^{\mathrm{T}}$

相应的判别函数为： $d(\boldsymbol{X})= -2x_1+1$



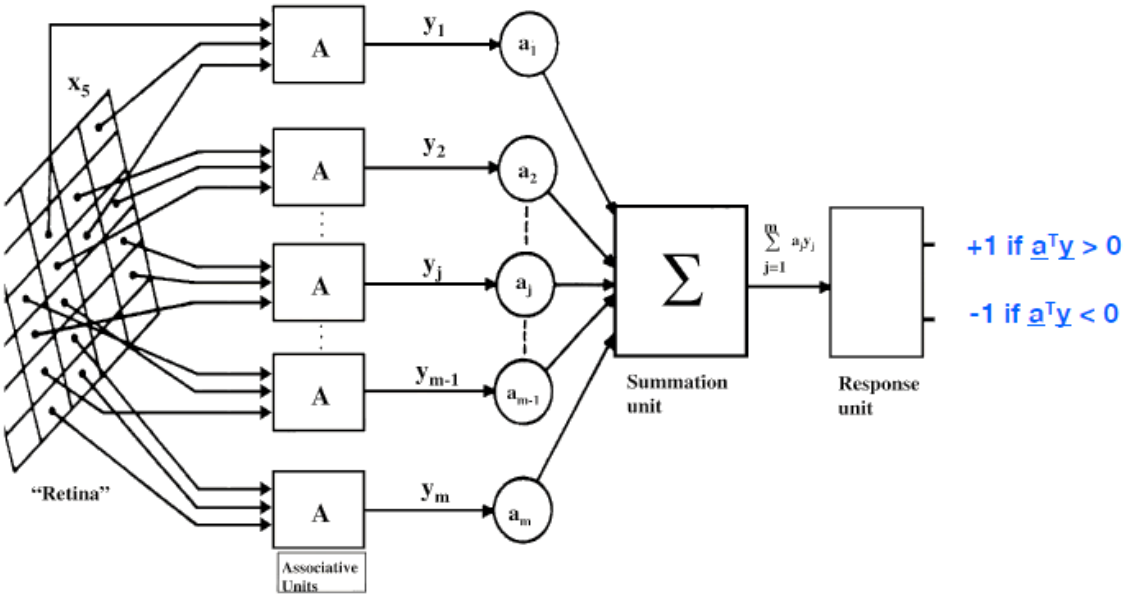
判别界面$d(\boldsymbol{X})=0$如图示。

当c、$\boldsymbol{W}(1)$取其他值时，结果可能不一样，所以**感知器算法的解不是单值的。**

感知机算法

- 感知机是由Frank Rosenblatt在1957年提出，是人工神经网络的创立者。
- 感知机是单个神经元的雏形。
- 当两个类是线性可分时，可以保证算法收敛性。

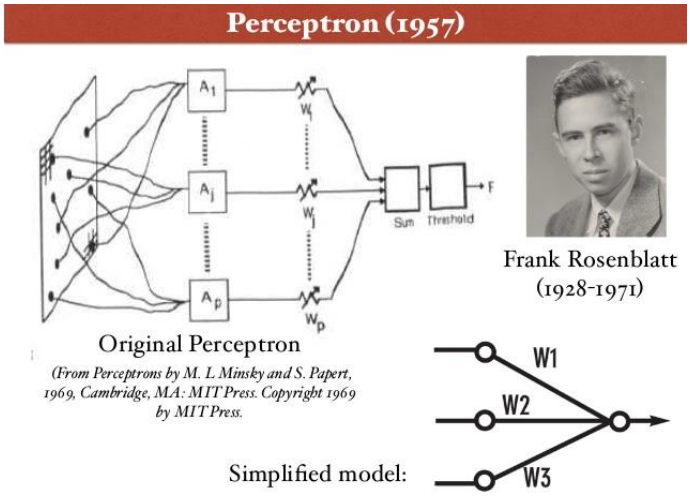
- Rosenblatt devised the perceptron as a model of learning in the brain.

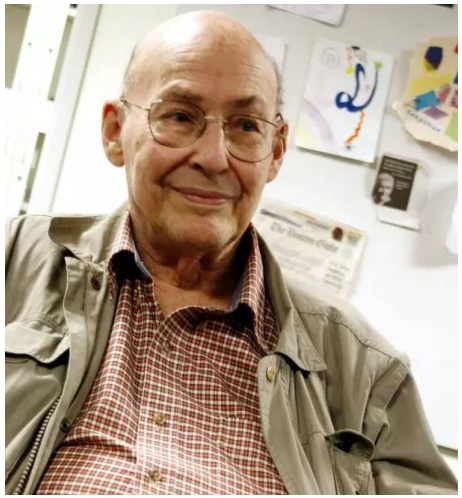


- Each time a pattern from class 1 falls on the 'retina' and produces an output +1, nothing happens. If the output is -1, then weights are updated $\underline{a}_{k+1} = \underline{a}_k + \underline{y}^k$

Frank Rosenblatt vs. Minsky

- Hebb在1949年出版的《行为的组织》中提出了其神经心理学理论。Hebb认为神经网络的学习过程最终是发生在神经元之间的突触部位，突触的联结强度随着突触前后神经元的活动而变化，变化的量与两个神经元的活性之和成正比
- 康奈尔航空实验室心理学家Frank Rosenblatt 受到这种思想的启发，提出了感知机，并认为其足以创造一个可以学会识别物体、能够识别出人并叫出他们的名字，立即把演讲内容翻译成另一种语言并写下来的机器。
- 但是1969年，Minsky 和Papert所著的《Perceptron》一书出版，该书从数学角度证明了关于单层感知器的计算具有根本的局限性，甚至连XOR这样的问题也不能解决，神经网络进入了萧条期。





感知机算法

- 感知机算法是首个采用误差反馈学习规则来是实现的机器学习算法。
- 它模拟人类学习的试错过程，不是希望计算机能够一次学会复杂的逻辑推理，而是通过一个形式确定但是参数不确定的模型来逐步逼近真实存在的客观规律，所以对后世产生了深远的影响。
- 一直到现在热门的深度学习，从根本上仍然是以有监督的误差反馈学习为核心，当然，其分类器的模型和学习信号的设计，又比感知器有了更多的进步。

1、 考虑一种情况，在类 ω_1 中包含两个特征向量，$[0,1]^T$。类 ω_2 中包含 $[1,0]^T$ 和 $[1,1]^T$ 两个向量。根据感知器算法，其中 $\rho=1$，$\omega(0)=[0.5,0.5]^T$，设计一个线性分离器来区分这两类。

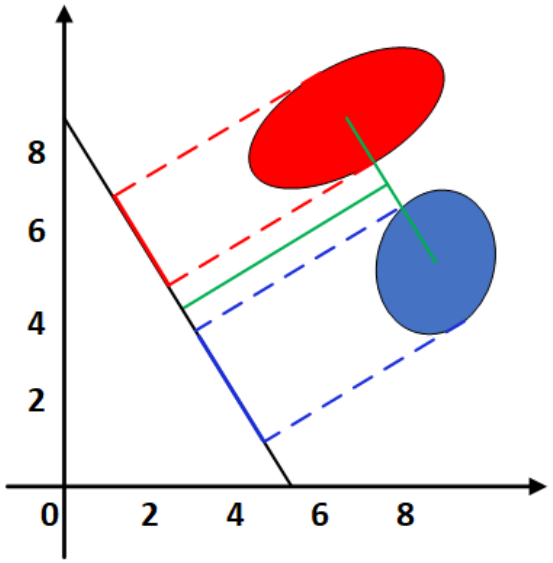
并行感知机：在每一步递推时将所有被错分的样本都找出来，再将其向量和用于修正权向量。

串行感知机：每次处理 1 个样本，如果发现分类错误，就用这一个被错分的样本来修正权向量。

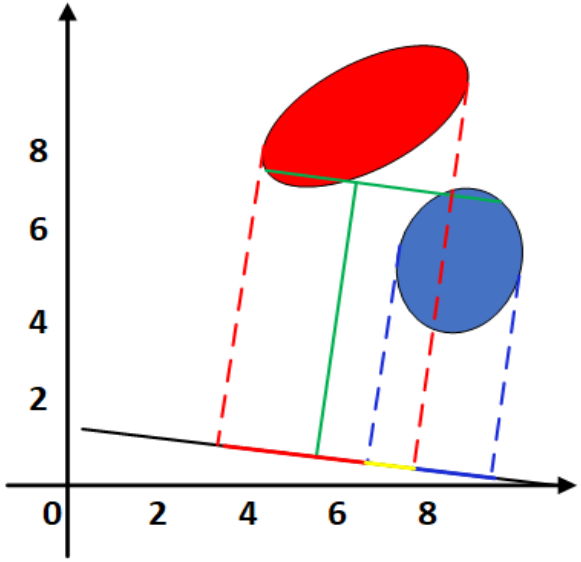
Fisher线性判据

- 线性判据的模型可以看做是把原空间各点x投影到新的一维空间y。

$$y = \boldsymbol{w}^T \boldsymbol{x} + w_0$$

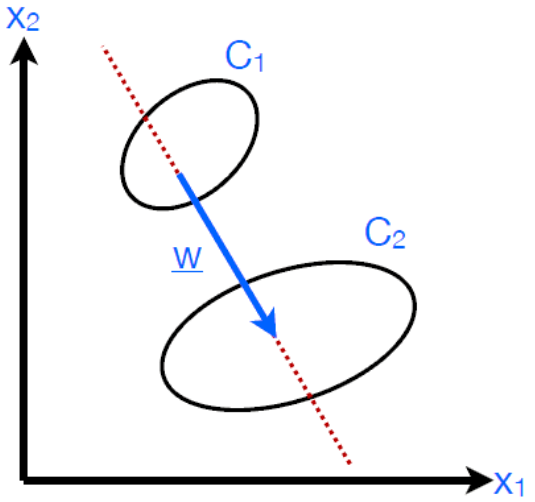


- w 的方向不同，对应不同的投影结果。
- 怎么样学习w，使得二类可以有效分开。

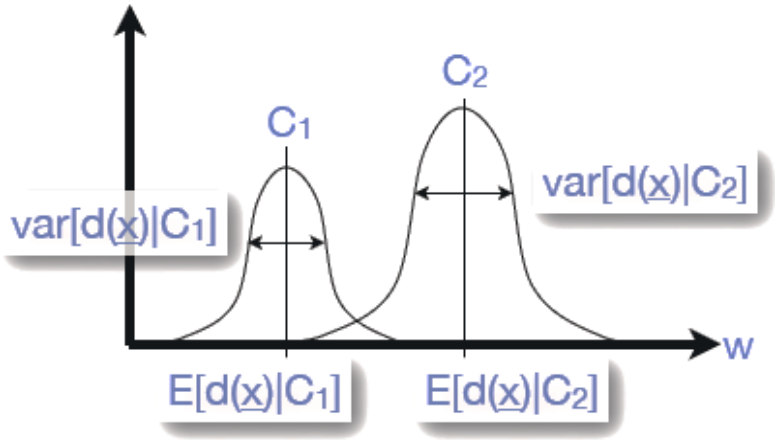


Fisher线性判据：目标函数

- 理想情况下，线性判据函数应该能够最大化类之间(inter-class)的变化，而同时最小化类内部(intra-class)的变化。
- 该目标函数不仅提供最佳分类，同时保证类的紧凑性。



- 类间样本的差异程度：用两类样本分布的均值之差度量。
- 类内样本的离散程度：用每类样本分布的协方差矩阵表征。



- 给定如下线性判据函数： $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$

- 目标函数定义如下：在投影轴w上

$$J(\mathbf{w}) = \frac{\|E[g(\mathbf{x})|C_1] - E[g(\mathbf{x})|C_2]\|_2^2}{N_1 var[g(\mathbf{x})|C_1] + N_2 var[g(\mathbf{x})|C_2]}$$

where $E[\cdot]$ denotes expectation, $var[\cdot]$ denotes covariance,
N_1 and N_2 denotes training sample numbers of C_1 and C_2.
Note that expectation and covariance are in $\mathbf{w}$ axis rather than feature space.

- 最大化该目标函数：
  - 首先计算目标函数中的每一项：

$$E[g(\mathbf{x})|C_1] = E[\mathbf{w}^T\mathbf{x} + w_0|C_1] = \mathbf{w}^T E[\mathbf{x}|C_1] + w_0 = \mathbf{w}^T\mathbf{m}_1 + w_0$$

$$E[g(\mathbf{x})|C_2] = E[\mathbf{w}^T\mathbf{x} + w_0|C_2] = \mathbf{w}^T E[\mathbf{x}|C_2] + w_0 = \mathbf{w}^T\mathbf{m}_2 + w_0$$

$$var[g(\mathbf{x})|C_1] = \frac{1}{N_1}\sum_{\mathbf{x}_i \in C_1}[\mathbf{w}^T\mathbf{x}_i + w_0]^2 = \mathbf{w}^T\mathbf{S}_1\mathbf{w}$$

$$var[g(\mathbf{x})|C_2] = \frac{1}{N_2}\sum_{\mathbf{x}_i \in C_2}[\mathbf{w}^T\mathbf{x}_i + w_0]^2 = \mathbf{w}^T\mathbf{S}_2\mathbf{w}$$

where $\mathbf{m}_1$ and $\mathbf{m}_2$ are sample means, $\mathbf{S}_1$ and $\mathbf{S}_2$ are sample covariances.

  - 带入目标函数，可以得到目标函数的新表达：

$$J(w) = \frac{w^T(m_1 - m_2)(m_1 - m_2)^T w}{w^T N_1 S_1 w + w^T N_2 S_2 w}$$

Fisher线性判据：求解

- 作如下定义：

类间散度(between class scatter)：$S_B = (m_1 - m_2)(m_1 - m_2)^T$

类内散度(within class scatter)：$S_W = N_1 S_1 + N_2 S_2$

- 目标函数写为：

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

在输入特征空间的 S_B 是一个矩阵，但是在投影轴w上是一个标量！

- 目标函数对w求偏导：
- 设偏导为0：

$$\frac{\partial J(w)}{\partial w} = \frac{2S_B w}{w^T S_W w} - 2\frac{w^T S_B w}{\left(w^T S_W w\right)^2} S_W w$$

J是特征值，w是特征向量。如果J取最大值，则w就是最大特征值对应的特征向量。

$$\because \frac{\partial J(w)}{\partial w} = 0 \Rightarrow S_B w = J(w) S_W w$$

$$A\underline{\phi} = \lambda\underline{\phi} \qquad S_W^{-1} S_B w = J(w) w$$

Fisher线性判据：求解

- 求解最优的w：不需求解特征值。

$$S_W^{-1} S_B W = J(W) W$$

$$\Rightarrow S_B W = J(W) S_W W$$

$$\Rightarrow (m_1 - m_2)(m_1 - m_2)^T W = J(W) S_W W$$

$$(\underline{m}_1 - \underline{m}_2)(\underline{m}_1 - \underline{m}_2)^T \underline{w} = J(\underline{w}) S_W \underline{w}$$

Ignore scalars, because all we need is the direction.

$$\Rightarrow \underline{w} = S_W^{-1}(\underline{m}_1 - \underline{m}_2)$$

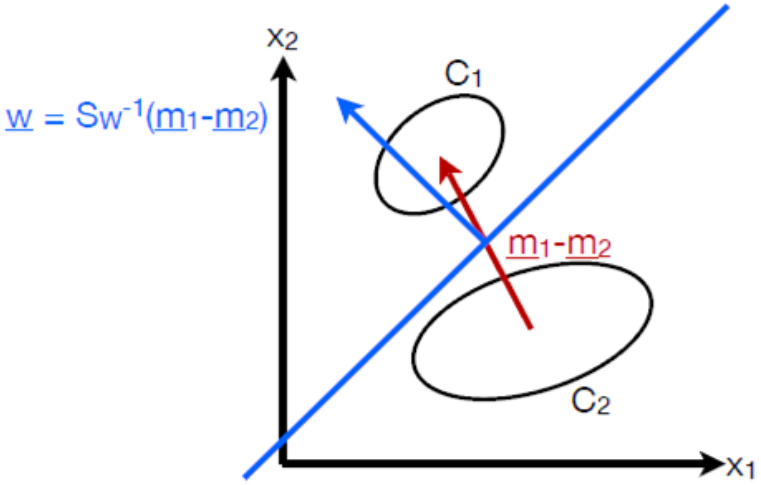
- 选取w_0： $w_0 = -w^T m$, m是所有样本的均值 （将所有样本的均值投影到坐标原点）

- 最终得到**Fisher**线性判据：

$$d_{FLD}(x) = w^T x + w_0 = (m_1 - m_2)^T S_W^{-1}(x - m)$$

Fisher线性判据：协方差S_w的作用

Geometrically, S_W^{-1} rotates the vector $(\underline{m_1}-\underline{m_2})$ between the means to take into account the shapes of the class distributions.



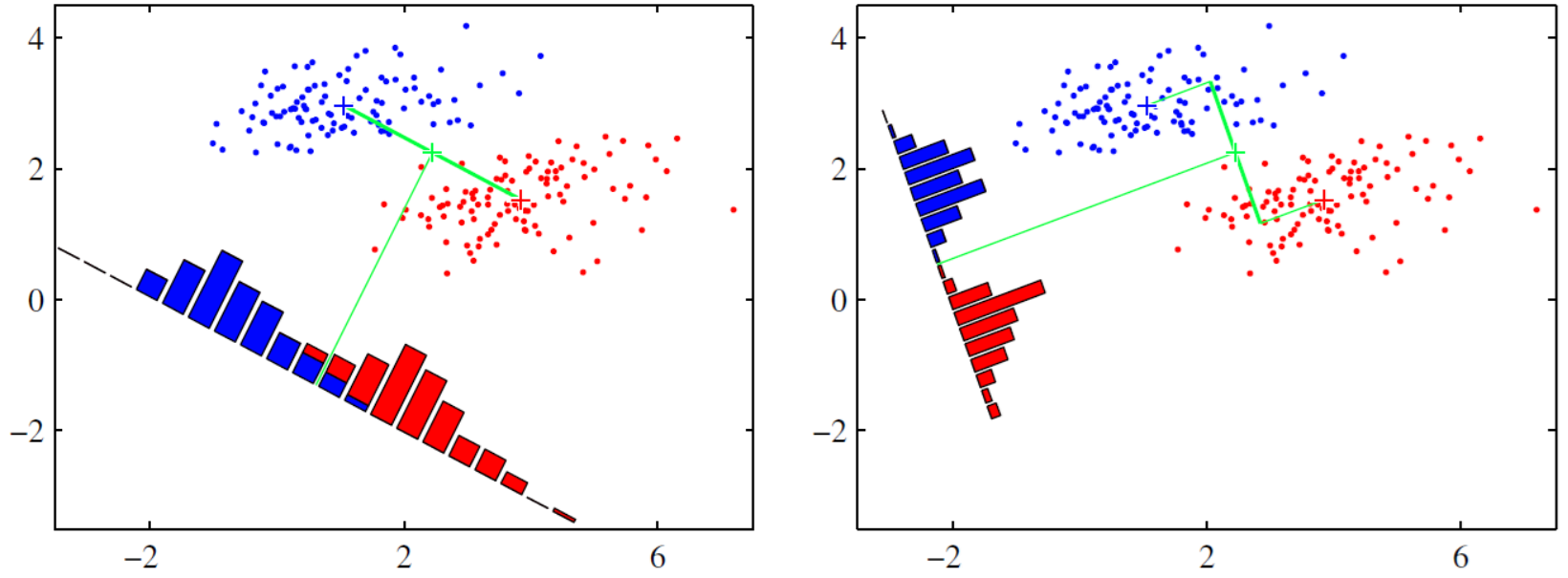
- 决策边界为过点m，斜率为$((m_1-m_2)\mathbf{S}_w^{-1}$的超平面。
- 没有过点$\frac{m_1+m_2}{2}$
- 协方差$\mathbf{S}_w$的作用：在几何上，$\mathbf{S}_w^{-1}$旋转向量(m_1-m_2)，以考虑类分布的形状。

Fisher线性判据：几何解释

- 根据两个向量点积的几何含义，线性判据是将样本x投影到w，并判断投影值是否大于$-w_0$ 实现分类。
- 所以，Fisher判据可以看做监督式学习的降维计算，降维的目的是提高分类性能。
- 所以，Fisher判据首先希望两个类的均值在w上的投影距离越大越好。

Fisher线性判据：几何解释

- 但是，原先在多维特征空间可分的两个类的样本，投影到一维轴w上后，会出现两类样本重叠。
- 这是由于两类样本分布形状（通过协方差矩阵非对角线体现）决定的。
- 所以，Fisher判据希望两个类的类内方差尽量的小，从而降低重叠度。



Fisher线性判据：多类分类

- 可以基于前述的三种多类分类思想实现。也可以重新设计针对多类的目标函数实现。
- 要实现多类（K个类）的分类，就需要多个线性判据。假设需要M个线性判据，即把训练样本分别投影到M条w轴上：$w^1, w^2, \ldots, w^M$。

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}, \text{ where } \mathbf{y} = [y_1, y_2, \ldots, y_M]^T, \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M]$$

- 新目标函数中的类间散度S_B在投影空间的定义为：

$$\mathbf{S}_B = \sum_{k=1}^{K} N_k(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T$$

$$\text{where } \mathbf{m}_k = \frac{1}{N_k}\sum_{n\in C_k}\mathbf{y}_n, \ \mathbf{m} = \frac{1}{N}\sum_{k=1}^{K} N_k\mathbf{m}_k$$

由于有多个投影轴，在投影空间的类间散度都是一个矩阵了！

- 新目标函数中的类内散度S_w在投影空间的定义为：

$$\mathbf{S}_W = \sum_{k=1}^{K}\sum_{n\in C_k}(\mathbf{y}_n - \mathbf{m}_k)(\mathbf{y}_n - \mathbf{m}_k)^T$$

- 新目标函数定义：最大化类间散度，同时最小化类内散度。

对W求偏导

$$J(\mathbf{w}) = \mathrm{Tr}(\mathbf{S}_W^{-1}\mathbf{S}_B) = \mathrm{Tr}\{(\mathbf{W}^T\mathbf{S}_W^x\mathbf{W})^{-1}(\mathbf{W}^T\mathbf{S}_B^x\mathbf{W})\}$$