模式识别和机器学习

主讲: 路永钢

E-mail: ylu@lzu.edu.cn

模式识别

第四章 分类和判别函数

内容

- 支持向量机 (SVM)

SVM的历史

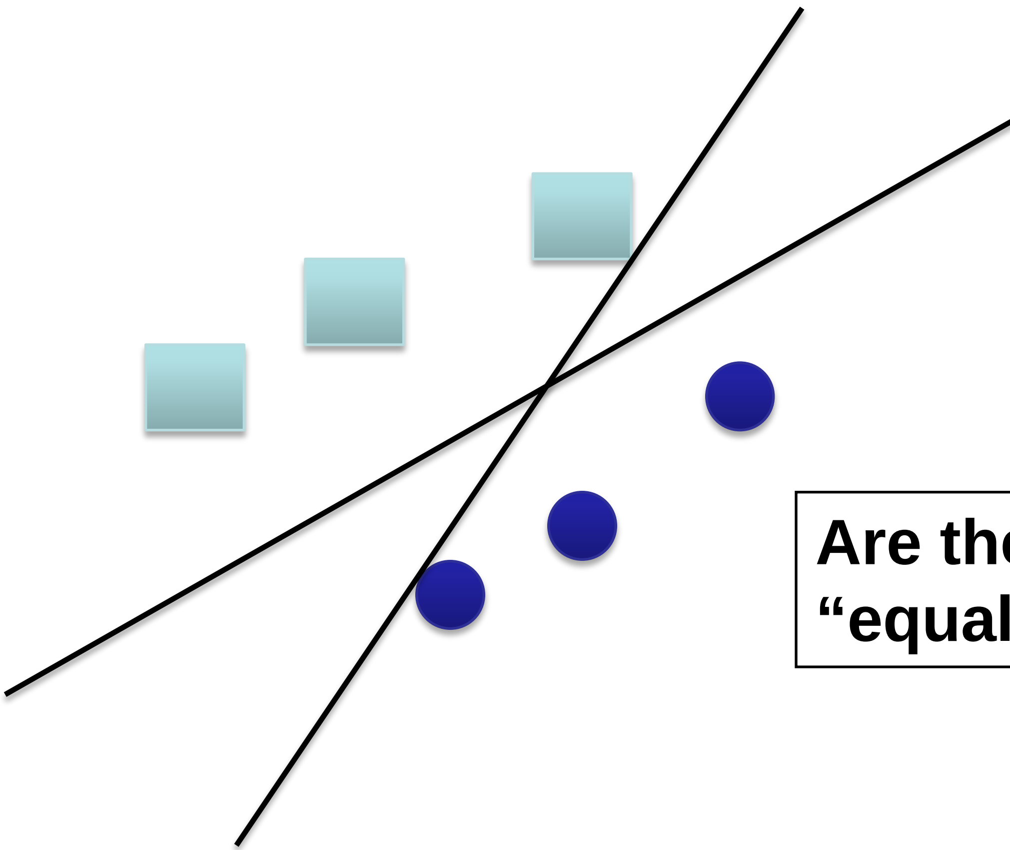
Bernhard E. Boser
University of California, Berkeley

- SVM是由Boser，Guyon，Vapnik等人在COLT-92上首次提出的[1]
- SVM是一种基于统计学习理论的机器学习方法[2]
- SVM目前已经在许多智能信息获取与处理领域都取得了成功的应用．

【1】B.E. Boser, ***et al. A Training Algorithm for Optimal Margin Classifiers. Proceedings of the Fifth Annual Workshop on Computational Learning Theory, vol. 5, pp. 144-152, Pittsburgh, 1992.***
【2】L. Bottou, ***et al. Comparison of classifier methods: a case study in handwritten digit recognition. Proceedings of the 12th IAPR International Conference on Pattern Recognition, vol. 2, pp. 77-82, 1994.***

SVM的思想：最大间隔原则

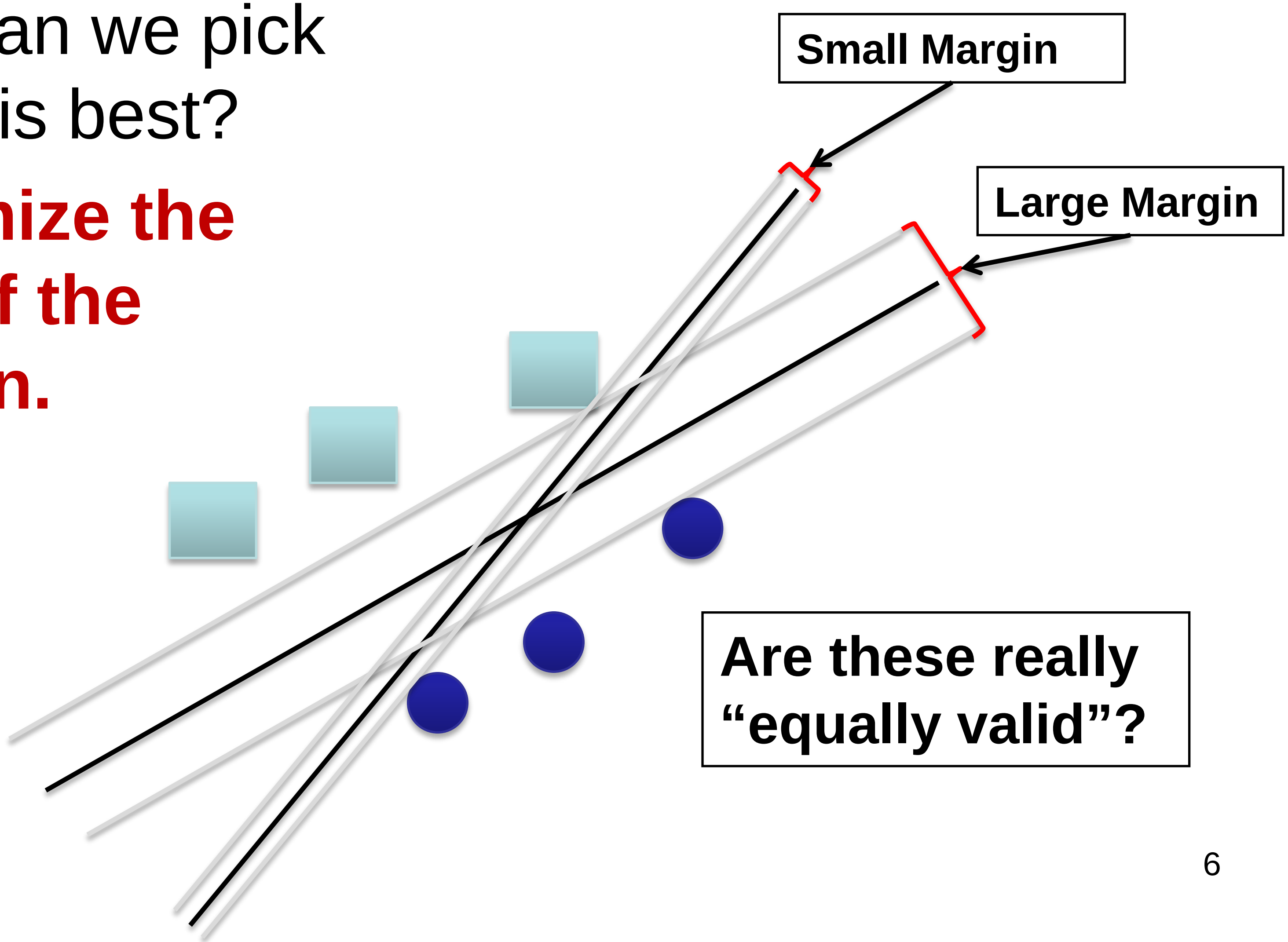
- 感知器或其他线性分类器确定的**决策面可以有多种等价的解！**



Are these really “equally valid”?

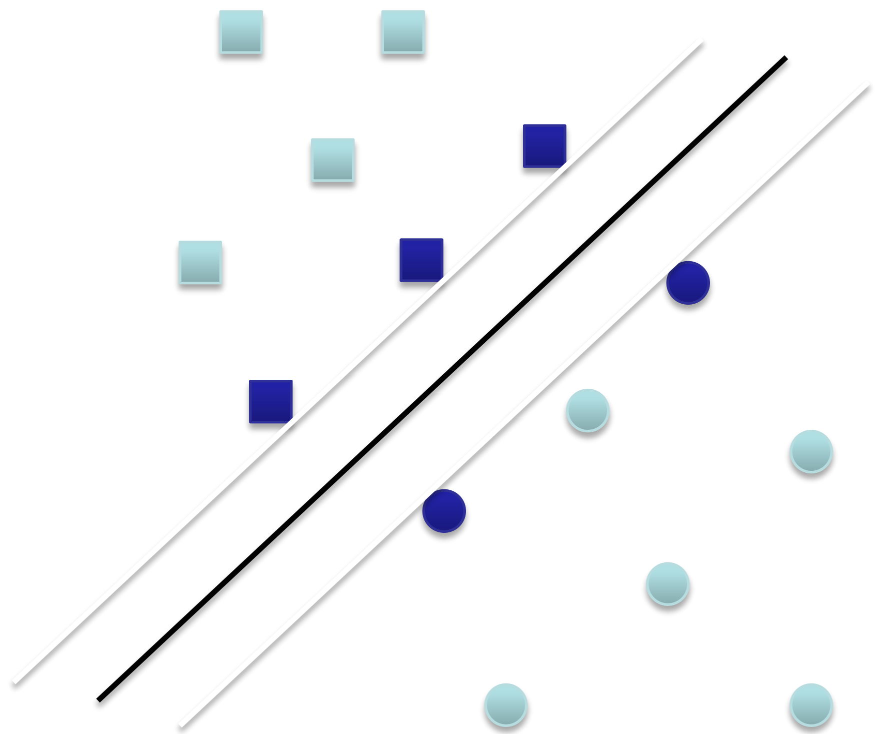
SVM的思想：最大间隔原则

- How can we pick which is best?
- **Maximize the size of the margin.**



支持向量（ Support Vectors ）

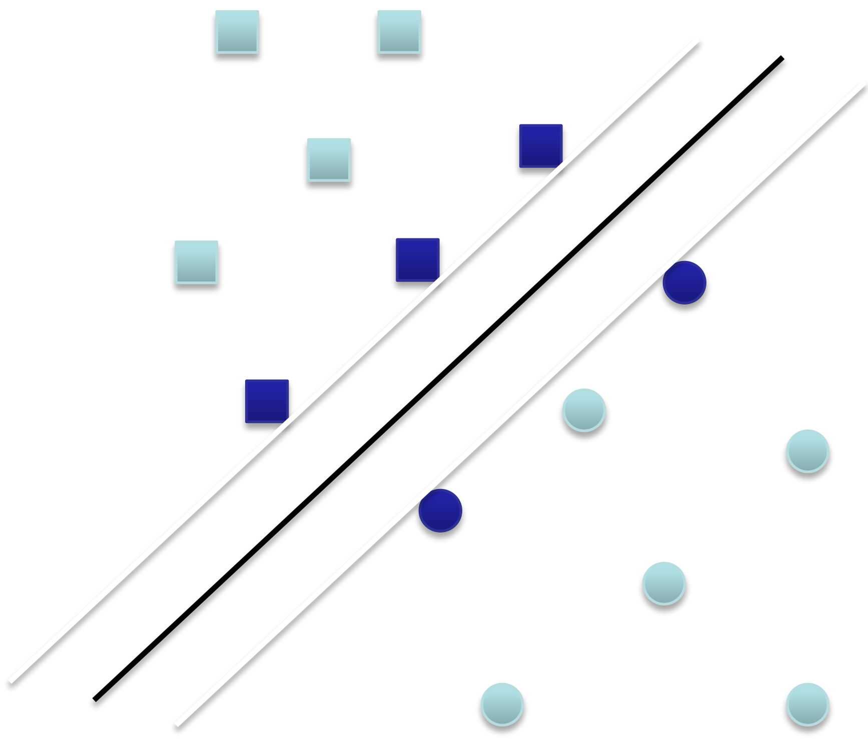
- Support Vectors are those input points (vectors) closest to the decision boundary
- 1. They are vectors
- 2. They “support” the decision hyperplane



支持向量（ Support Vectors ）

- Define this as a decision problem
- The decision hyperplane:

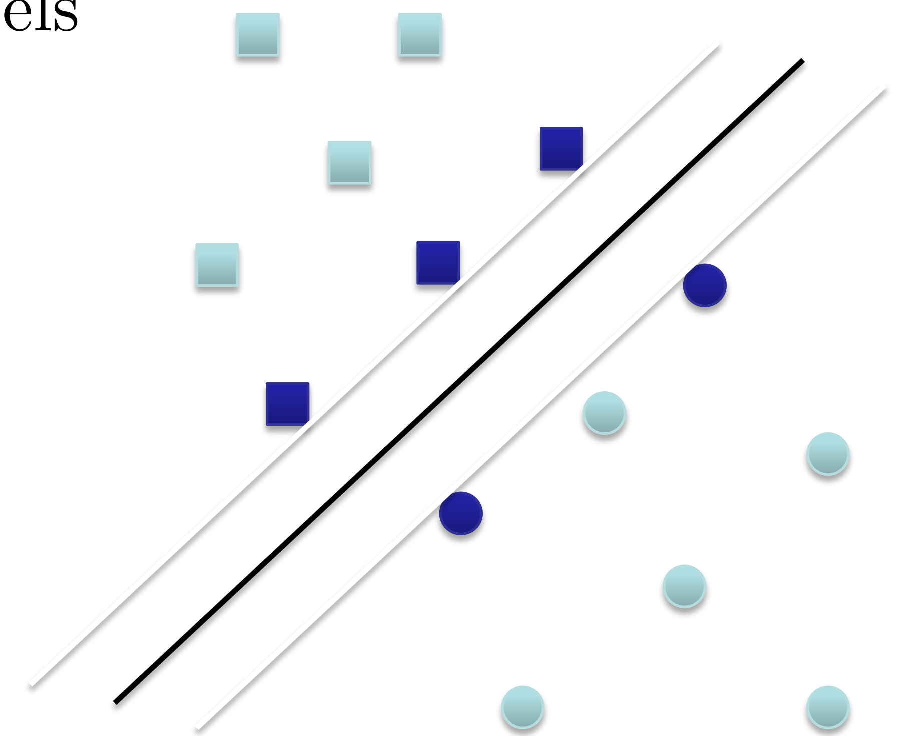
$$\boxed{\vec{w}^T \vec{x} + b = 0}$$



支持向量（ Support Vectors）

$\vec{x}_i$ are the data

$\vec{t}_i \in \{-1, +1\}$ are the labels



- 简单讨论:
  - 为什么使用

    $t_i \in \{-1, +1\}$

  - 可否使用

    $t_i \in \{0, 1\}$

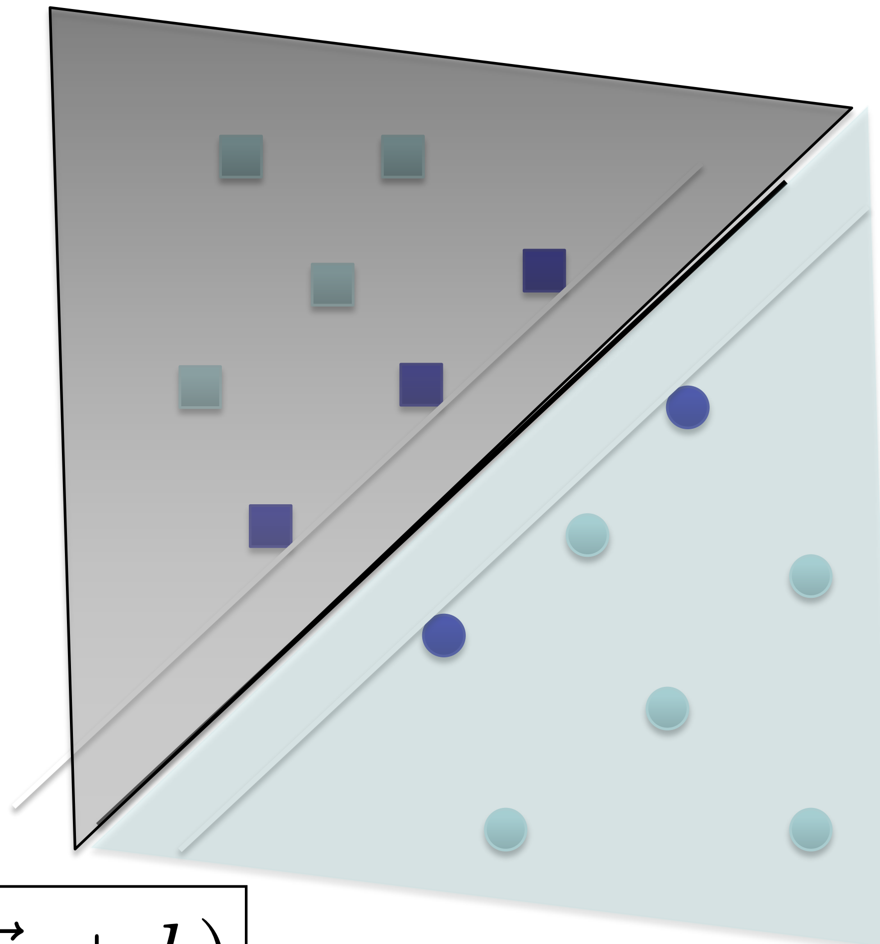
支持向量（ Support Vectors）

- Define this as a decision problem
- The decision hyperplane:

$$\boxed{\vec{w}^T\vec{x} + b = 0}$$

- 决策函数:

$$\boxed{D(\vec{x_i}) = sign(\vec{w}^T\vec{x_i} + b)}$$



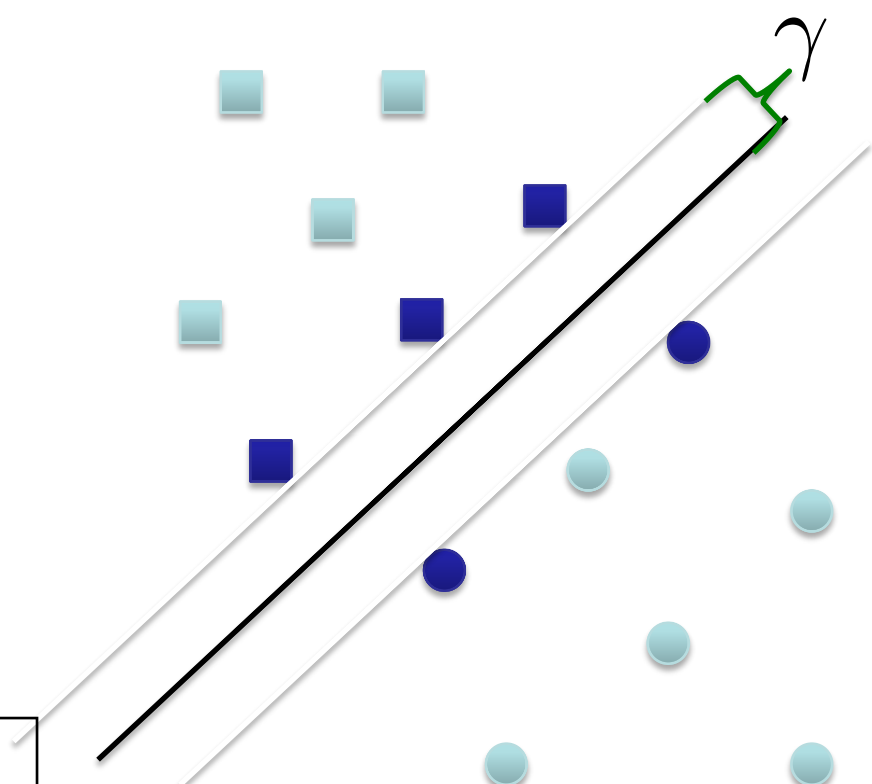
支持向量（ Support Vectors）

- Define this as a decision problem
- The decision hyperplane:

$$\boxed{\vec{w}^T\vec{x} + b = 0}$$

- Margin hyperplanes:

$$\boxed{\begin{aligned} \vec{w}^T\vec{x} + b &= \gamma \\ \vec{w}^T\vec{x} + b &= -\gamma \end{aligned}}$$



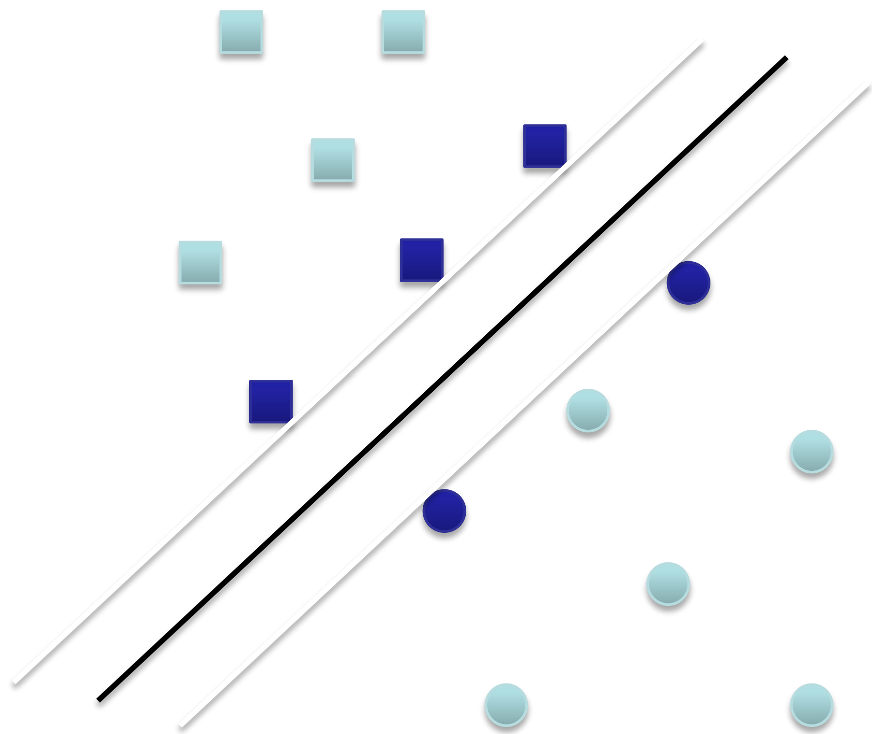
支持向量（Support Vectors）

- The decision hyperplane:

$$\boxed{\vec{w}^T\vec{x} + b = 0}$$

- 缩放无关性：

$$\boxed{c\vec{w}^T\vec{x} + cb = 0}$$



支持向量（ Support Vectors）

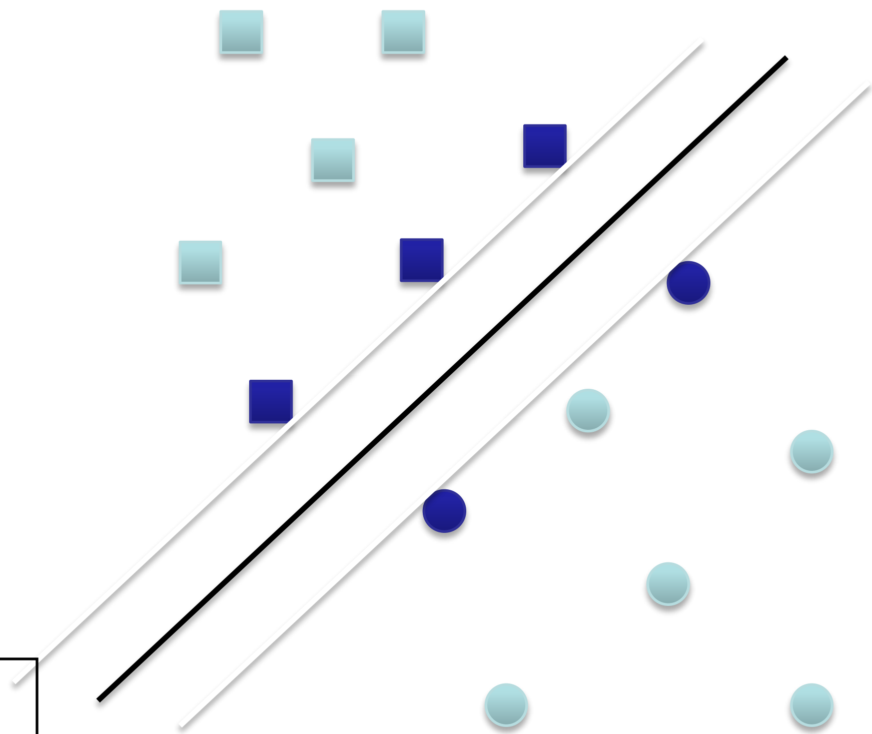
- The decision hyperplane:

$$\boxed{\vec{w}^T\vec{x} + b = 0}$$

- 缩放无关性：

$$\boxed{c\vec{w}^T\vec{x} + cb = 0}$$

$$\boxed{\begin{aligned} \vec{w}^T\vec{x} + b &= \gamma \\ \vec{w}^T\vec{x} + b &= -\gamma \end{aligned}}$$



支持向量（ Support Vectors）

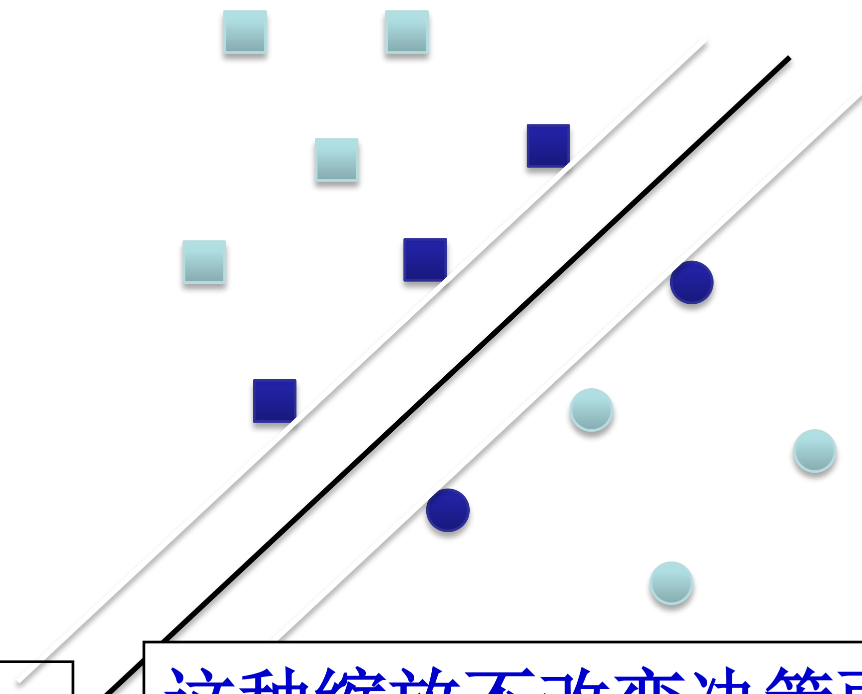
- The decision hyperplane:

$$\boxed{\vec{w}^T\vec{x}+b=0}$$

- 缩放无关性：

$$\boxed{c\vec{w}^T\vec{x}+cb=0}$$

$$\boxed{\begin{aligned}\vec{w^*}^T\vec{x}+b^* &= 1\\ \vec{w^*}^T\vec{x}+b^* &= -1\end{aligned}}$$



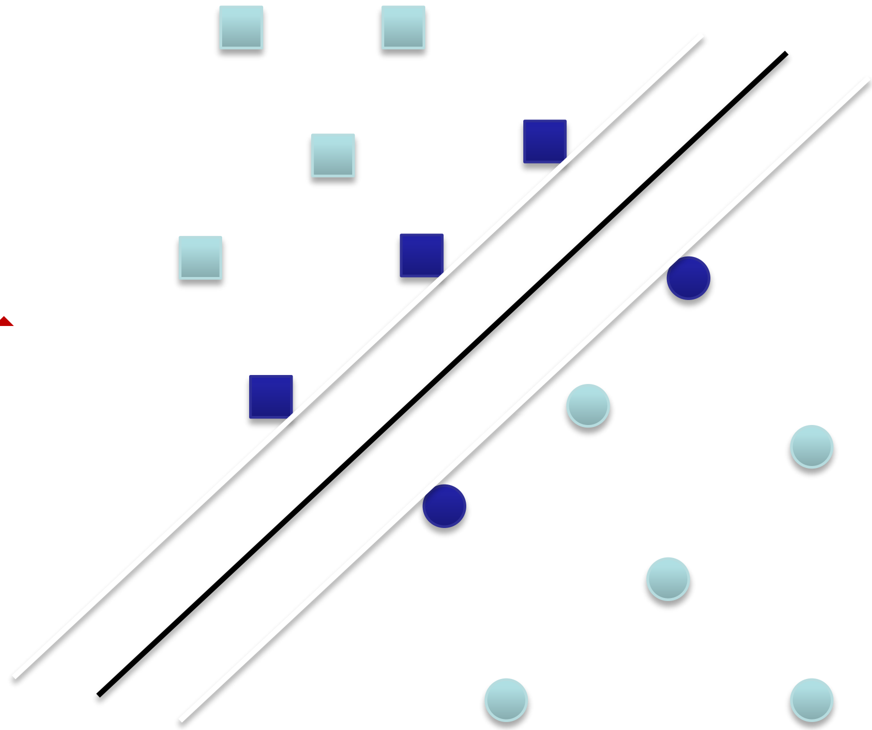
这种缩放不改变决策面和支持向量，却可以减少一个参数！

如何求解?

- 决策面方程为.

$$\boxed{\vec{w}^T\vec{x} + b = 0}$$

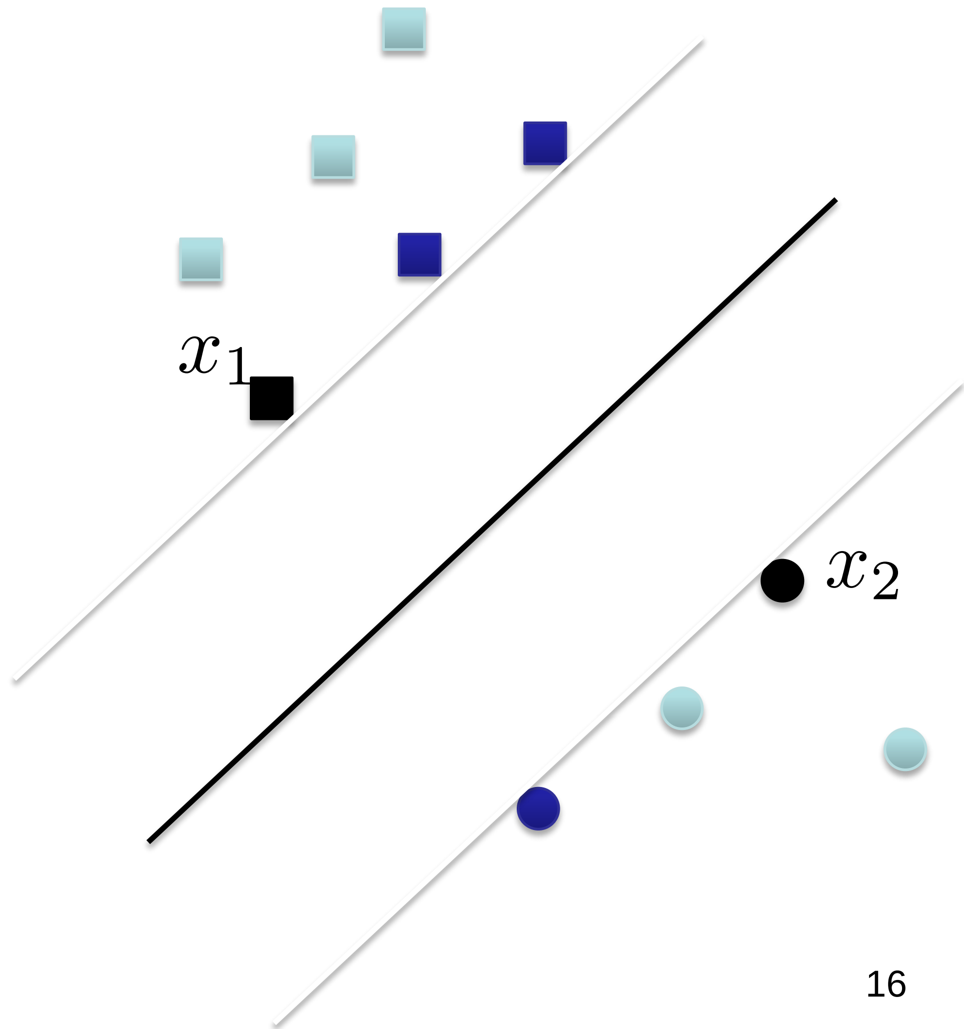
- 如果能将间隔表示为 w 的函数，就可以同时：
  - 识别出决策面
  - 最大化间隔



将间隔表示为 w 的函数

1. There must at least one point that lies on each support hyperplanes

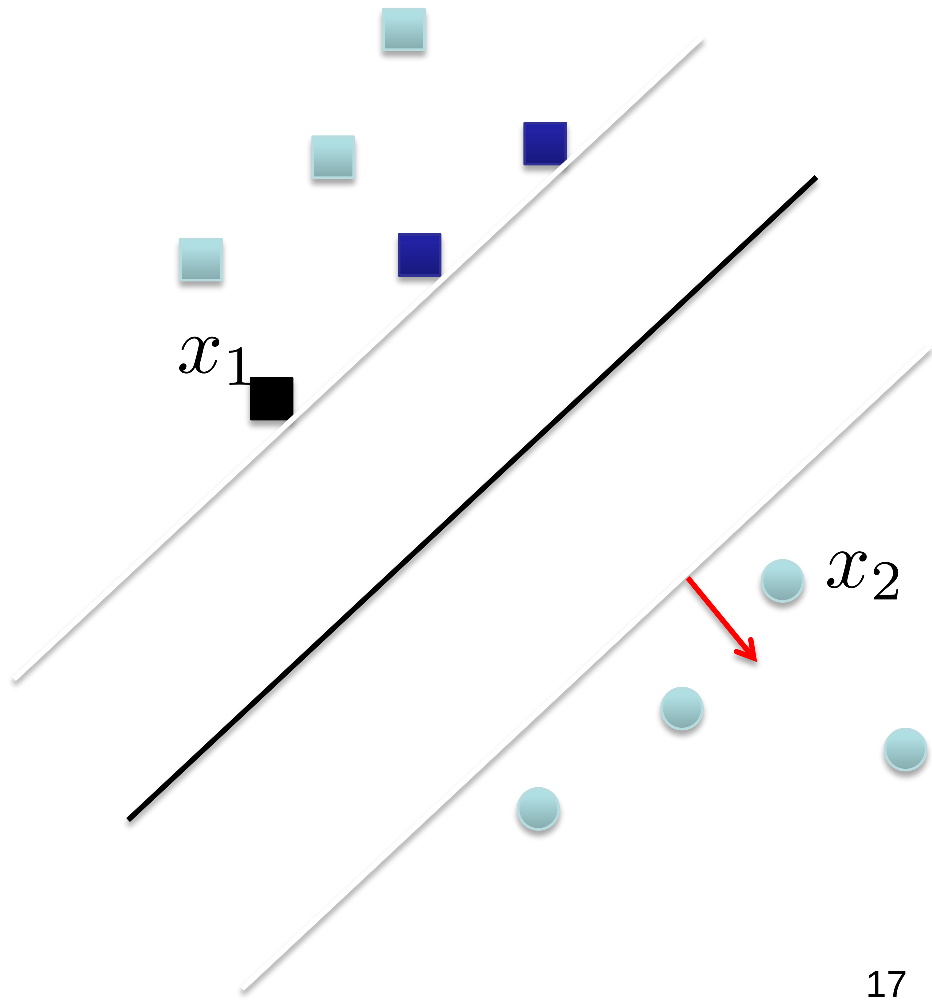
Proof outline: If not, we could define a larger margin support hyperplane that *does* touch the nearest point(s).



将间隔表示为 w 的函数

1. There must at least one point that lies on each support hyperplanes

Proof outline: If not, we could define a larger margin support hyperplane that *does* touch the nearest point(s).



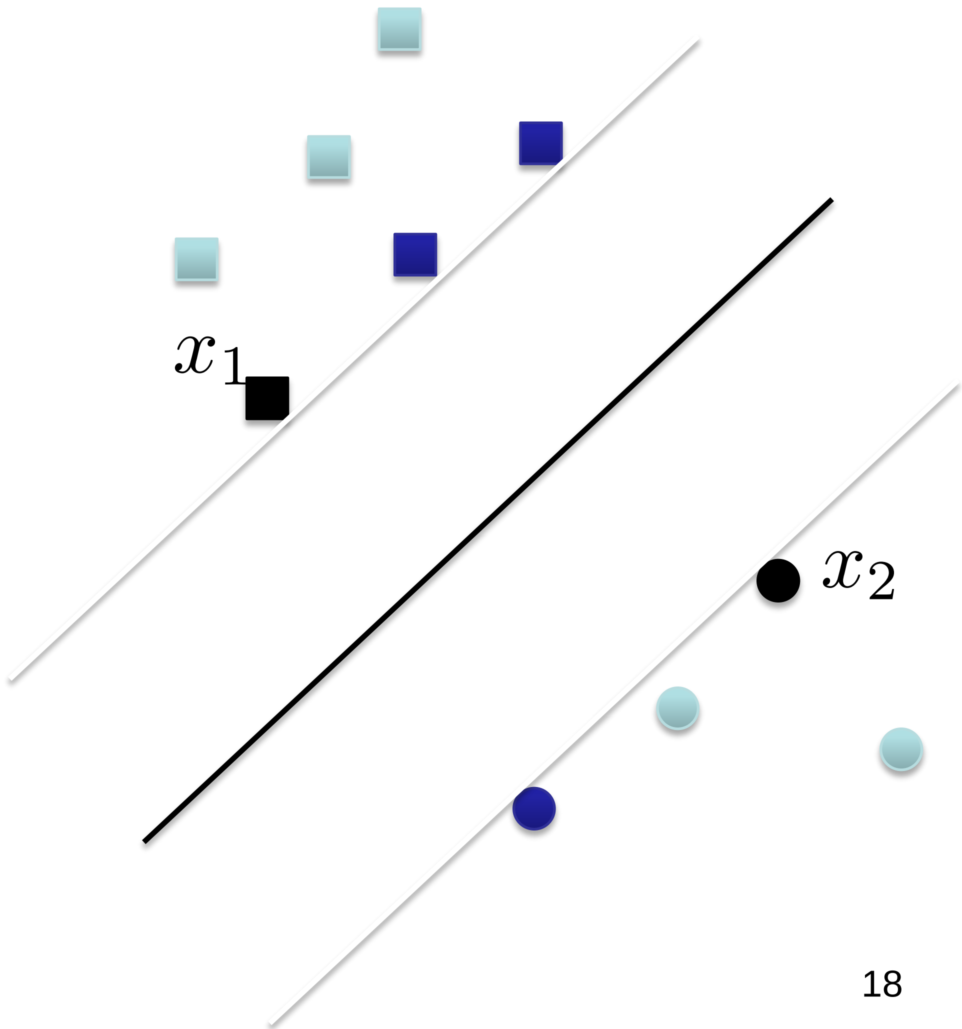
将间隔表示为 w 的函数

1. There must at least one point that lies on each support hyperplanes
2. Thus:

$$\boxed{\begin{aligned} w^T x_1 + b &= 1 \\ w^T x_2 + b &= -1 \end{aligned}}$$

3. **And:**

$$\boxed{w^T(x_1 - x_2) = 2}$$



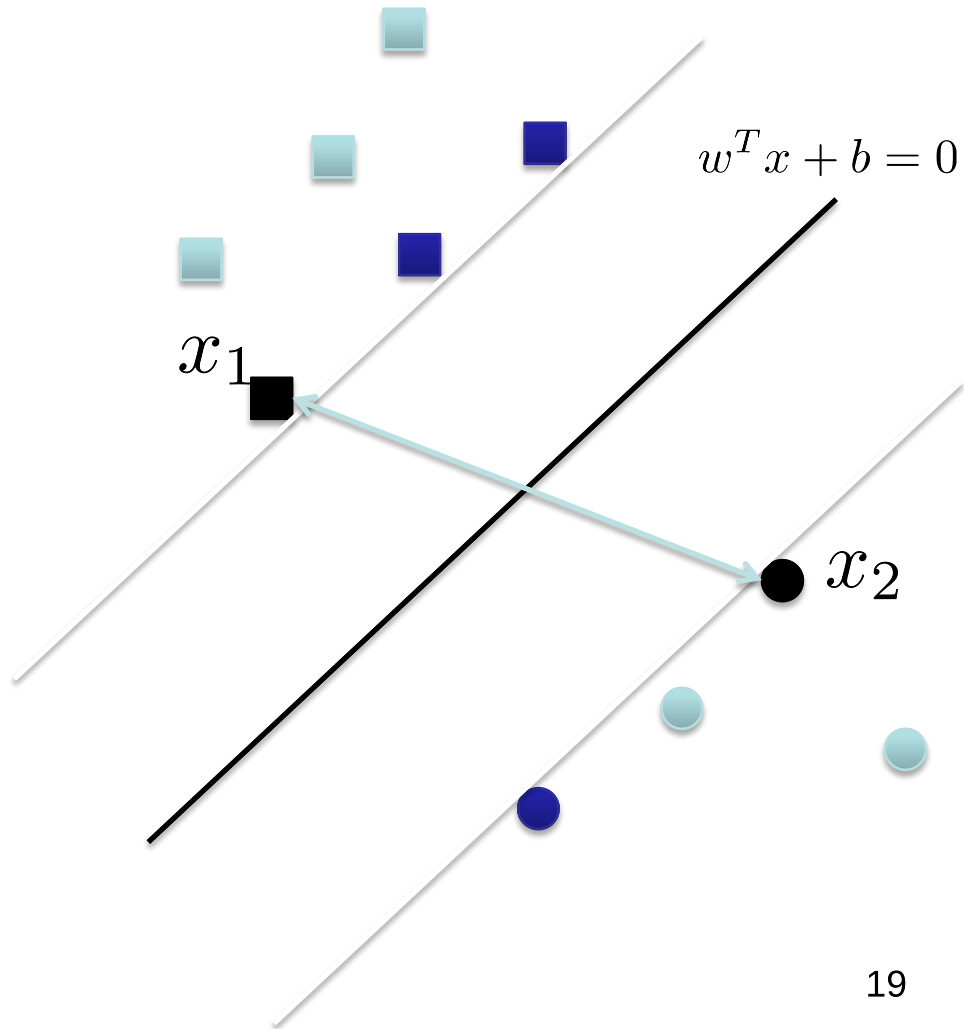
将间隔表示为 w 的函数

1. There must at least one point that lies on each support hyperplanes
2. Thus:

$$\boxed{\begin{aligned} w^T x_1 + b &= 1 \\ w^T x_2 + b &= -1 \end{aligned}}$$

3. **And:**

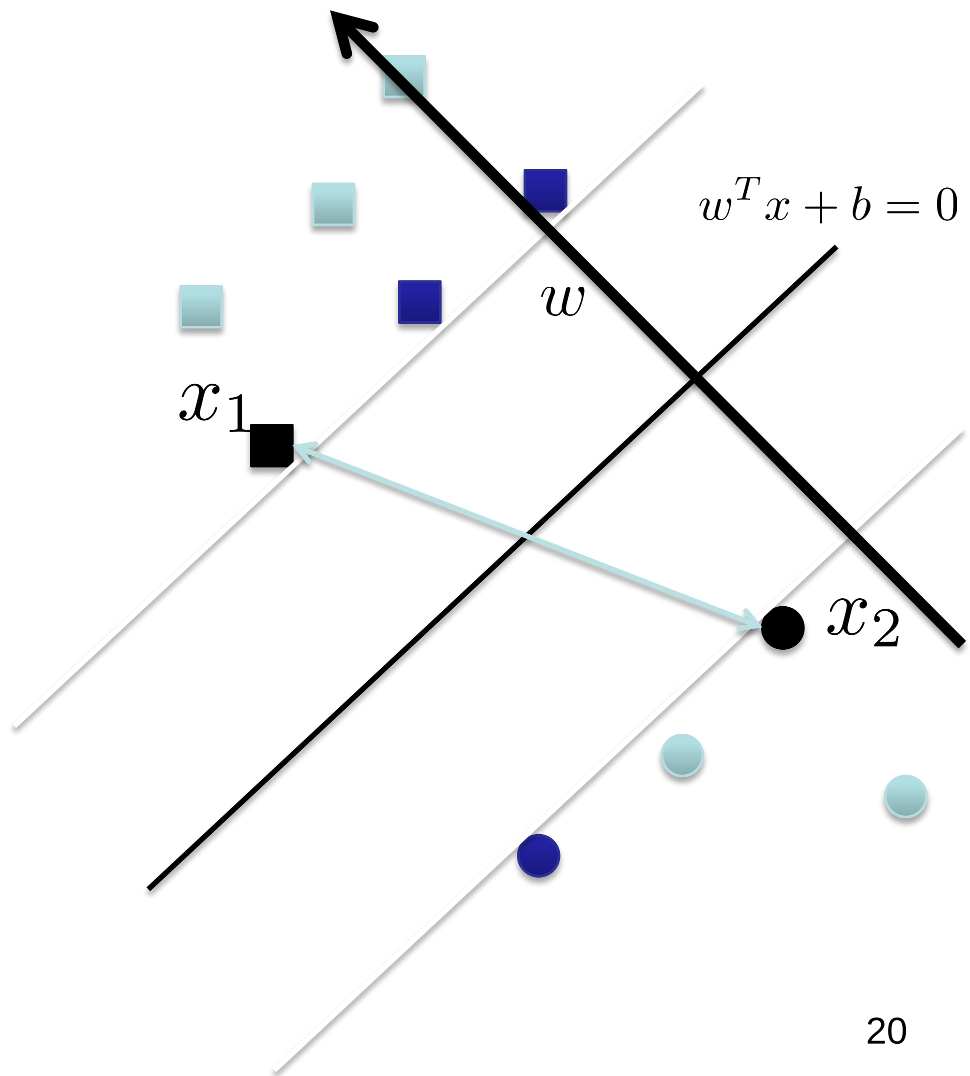
$$\boxed{w^T(x_1 - x_2) = 2}$$



将间隔表示为 w 的函数

- The vector **w** is perpendicular to the decision hyperplane

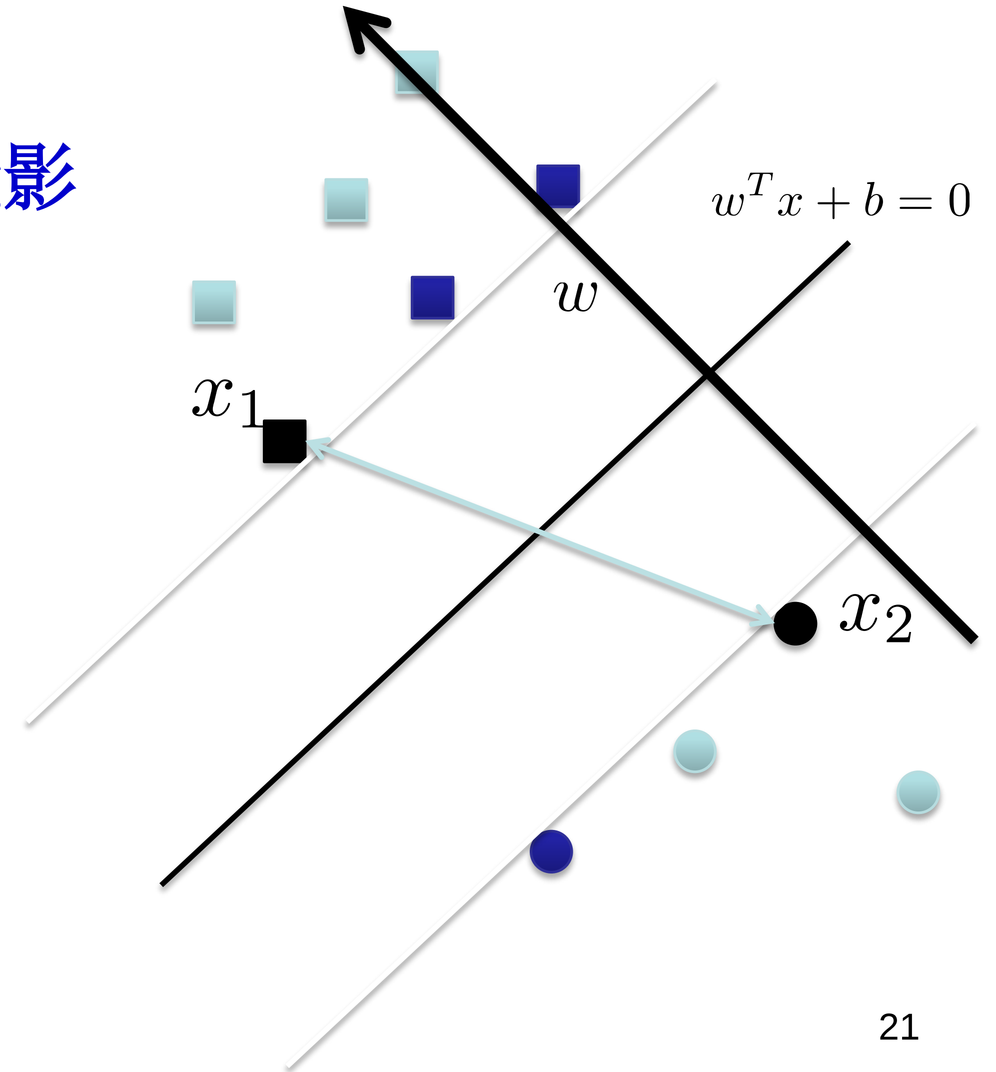
$$\boxed{w^T(x_1 - x_2) = 2}$$



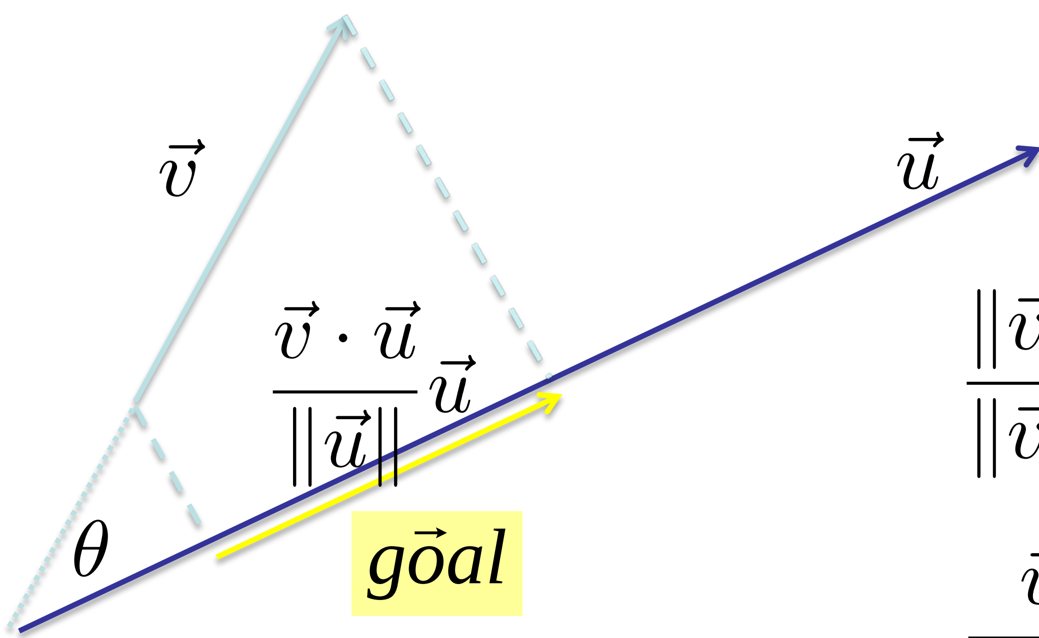
将间隔表示为 w 的函数

- 间隔的大小就是 向量 $x_1 - x_2$ 在 *w* 方向的投影的大小！

$$\boxed{w^T(x_1 - x_2) = 2}$$



向量投影



$$\boxed{\vec{v} \cdot \vec{u} = \|\vec{v}\| \|\vec{u}\| \cos(\theta)}$$

$$cos(\theta) = \frac{\|\vec{goal}\|}{\|\vec{v}\|}$$

$$\frac{\|\vec{v}\| \|\vec{u}\|}{\|\vec{v}\| \|\vec{u}\|} \cos(\theta) = \frac{\|\vec{goal}\|}{\|\vec{v}\|}$$

$$\frac{\vec{v} \cdot \vec{u}}{\|\vec{v}\| \|\vec{u}\|} = \frac{\|\vec{goal}\|}{\|\vec{v}\|}$$

$\vec{v}$ 在 $\vec{u}$ 方向的投影长度为: $\boxed{\frac{\vec{v} \cdot \vec{u}}{\|\vec{u}\|} = \|\vec{goal}\|}$

将间隔表示为 w 的函数

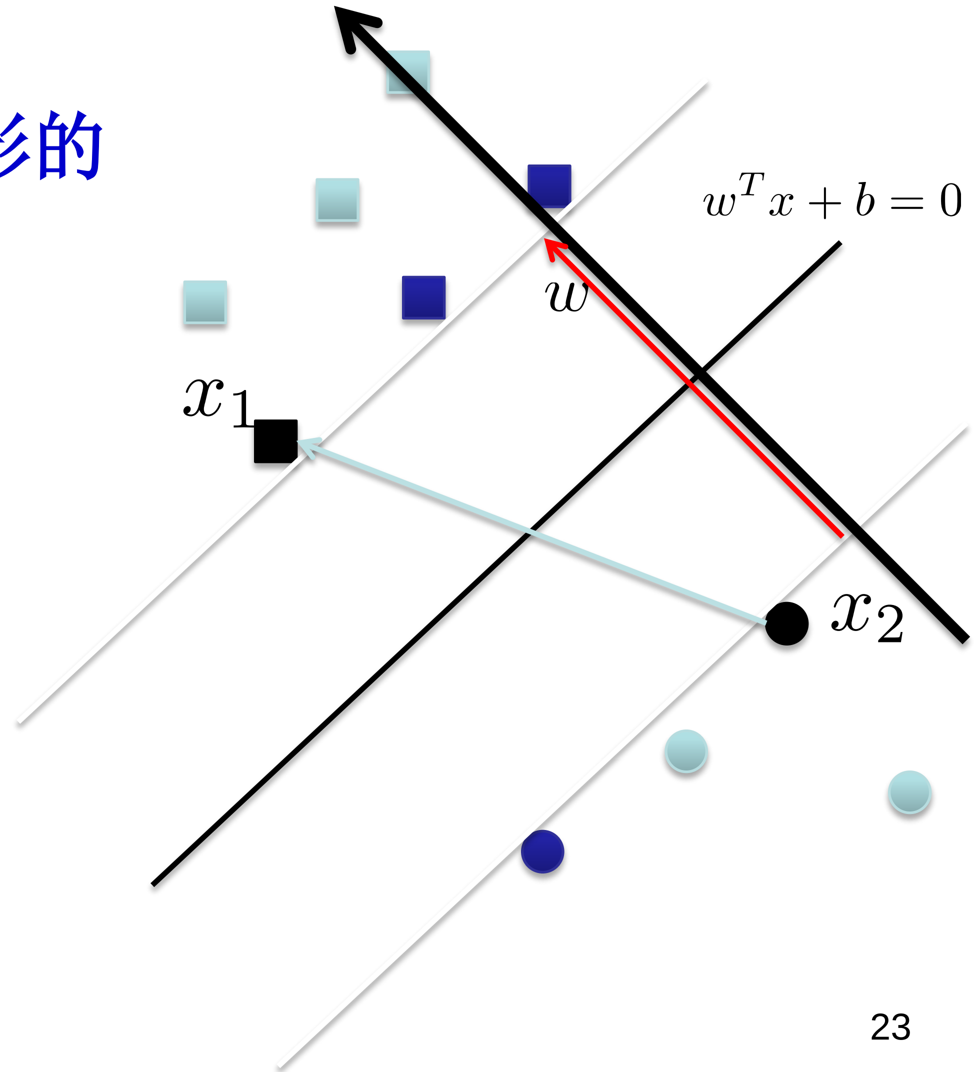
- 间隔的大小就是 向量 $x_1 - x_2$ 在 w 方向的投影的大小！

$$\boxed{w^T(x_1 - x_2) = 2}$$

投影长度: $\dfrac{\vec{v}\cdot\vec{u}}{\|\vec{u}\|}$

$$\frac{w^T(x_1 - x_2)}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|}$$

故间隔大小为: $\dfrac{2}{\|\vec{w}\|}$



最大化间隔

- Goal: maximize the margin

$$\max \frac{2}{\|\vec{w}\|}$$

$$\Leftrightarrow \quad \min \|\vec{w}\|$$

$$\text{where } t_i(\vec{w}^T x_i + b) \geq 1$$

线性可分时：

$$\vec{w}^T x_i + b \geq 1 \quad \text{if} \quad t_i = 1$$
$$\vec{w}^T x_i + b \leq 1 \quad \text{if} \quad t_i = -1$$

拉格朗日乘数法

- If **constraint optimization** then **Lagrange Multipliers**
- Optimize the “Primal”

$$\min \|\vec{w}\|$$

$$\text{where } t_i(\vec{w}^T x_i + b) \geq 1$$

$$L(\vec{w}, b) = \frac{1}{2}\vec{w} \cdot \vec{w} - \sum_{i=0}^{N-1} \alpha_i [t_i((\vec{w} \cdot \vec{x_i}) + b) - 1]$$

拉格朗日乘数法

- Optimize the “Primal”

$$L(\vec{w},b) = \frac{1}{2}\vec{w}\cdot\vec{w} - \sum_{i=0}^{N-1}\alpha_i[t_i((\vec{w}\cdot\vec{x_i})+b)-1]$$

对b求偏导：

$$\begin{aligned}\frac{\partial L(\vec{w},b)}{\partial b} &= 0 \\ \sum_{i=0}^{N-1}\alpha_i t_i &= 0\end{aligned}$$

拉格朗日乘数法

- Optimize the “Primal”

$$L(\vec{w}, b) = \frac{1}{2}\vec{w}\cdot\vec{w} - \sum_{i=0}^{N-1} \alpha_i [t_i((\vec{w}\cdot\vec{x_i}) + b) - 1]$$

对w求偏导：

$$\begin{aligned} \frac{\partial L(\vec{w}, b)}{\partial \vec{w}} &= 0 \\ \vec{w} - \sum_{i=0}^{N-1} \alpha_i t_i \vec{x_i} &= 0 \\ \vec{w} &= \sum_{i=0}^{N-1} \alpha_i t_i \vec{x_i} \end{aligned}$$

拉格朗日乘数法

- Optimize the “Primal”

$$L(\vec{w}, b) = \frac{1}{2}\vec{w}\cdot\vec{w} - \sum_{i=0}^{N-1}\alpha_i[t_i((\vec{w}\cdot\vec{x_i}) + b) - 1]$$

对w求偏导：

$$\begin{aligned} \frac{\partial L(\vec{w}, b)}{\partial \vec{w}} &= 0 \\ \vec{w} - \sum_{i=0}^{N-1}\alpha_i t_i \vec{x_i} &= 0 \\ \vec{w} &= \sum_{i=0}^{N-1}\alpha_i t_i \vec{x_i} \end{aligned}$$

为了求得 α_i,
需将此式代入拉格朗日函数

拉格朗日乘数法

$$L(\vec{w},b)=\frac{1}{2}\vec{w}\cdot\vec{w}-\sum_{i=0}^{N-1}\alpha_i[t_i((\vec{w}\cdot\vec{x_i})+b)-1]$$

$$\vec{w} \quad = \quad \sum_{i=0}^{N-1}\alpha_i t_i \vec{x_i}$$

- 代入后得到**对偶式:**

$$W(\alpha)=\sum_{i=0}^{N-1}\alpha_i-\frac{1}{2}\sum_{i,j=0}^{N-1}\alpha_i\alpha_j t_i t_j(\vec{x_i}\cdot\vec{x_j})$$

$$\text{where } \alpha_i \geq 0 \qquad \sum_{i=0}^{N-1}\alpha_i t_i = 0$$

求解 对偶式

$$W(\alpha)=\sum_{i=0}^{N-1}\alpha_i-\frac{1}{2}\sum_{i,j=0}^{N-1}\alpha_i\alpha_j t_i t_j(\vec{x_i}\cdot\vec{x_j})$$

$$\text{where } \alpha_i \geq 0 \qquad \sum_{i=0}^{N-1}\alpha_i t_i=0$$

- 这是一个带约束条件的**二次规划问题**（ quadratic programming）
- **可以证明该问题是一个凸问题，有唯一解！**
- **求解该问题就可得到 α 的值，进而得到决策面的方向 *w* .**

在 C, C++, Matlab, Python, Java and R等语言中都有这种二次规划问题的标准求解办法！

二次规划问题

$$\text{minimize } f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} + c^T\vec{x}$$

$$\text{subject to (one or more)} \quad A\vec{x} \leq k$$

$$B\vec{x} = l$$

- 如果 Q 是半正定矩阵, **f(x)** 就是一个凸函数.
- 如果 **f(x)** 是凸函数, 就具有唯一的极小值.

$$W(\alpha) = \sum_{i=0}^{N-1} \alpha_i - \frac{1}{2}\sum_{i,j=0}^{N-1} \alpha_i\alpha_j t_i t_j (\vec{x_i} \cdot \vec{x_j})$$

$$\text{where } \alpha_i \geq 0$$

Matlab 求解二次规划问题示例

求如下函数的最小值：

$$f(x)=\frac{1}{2}x_1^2+x_2^2-x_1x_2-2x_1-6x_2$$

$$\text{Constraints}:\begin{cases} x_1+x_2\le 2 \\ -x_1+2x_2\le 2 \\ 2x_1+x_2\le 3 \\ x_1\ge 0,\quad x_2\ge 0 \end{cases}$$

$$f(x)=\frac{1}{2}X^THX+F^TX$$

$$H=\begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix},\quad F=\begin{bmatrix} -2 \\ -6 \end{bmatrix},\quad X=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Matlab 代码：

```
H = [1 -1; -1 2];
F = [-2; -6];
A = [1 1; -1 2; 2 1];
b = [2; 2; 3];
lb = zeros(2,1);

options = optimoptions('quadprog' , …
'Algorithm','interior-point-convex','Display','off');

[x,fval,exitflag] = quadprog(H,F,A,b,[ ],[ ],lb,[ ],[ ],options);
```

运行结果：

x =[0.6667, 1.3333]
fval = -8.2222
exitflag = 1

支持向量的含义

$$\vec{w} = \sum_{i=0}^{N-1} \alpha_i t_i \vec{x_i}$$

新的决策函数：

$$\begin{aligned} D(\vec{x}) &= sign(\vec{w}^T \vec{x} + b) \\ &= sign\left(\left[\sum_{i=0}^{N-1} \alpha_i t_i \vec{x_i}\right]^T \vec{x} + b\right) \\ &= sign\left(\left[\sum_{i=0}^{N-1} \alpha_i t_i (\vec{x_i}^T \vec{x})\right] + b\right) \end{aligned}$$

与x的维数无关!

- 当 α_i 非零是，x_i 就是一个支持向量
- 当 α_i =0，x_i 不是支持向量，与决策无关！

Kuhn-Tucker Conditions

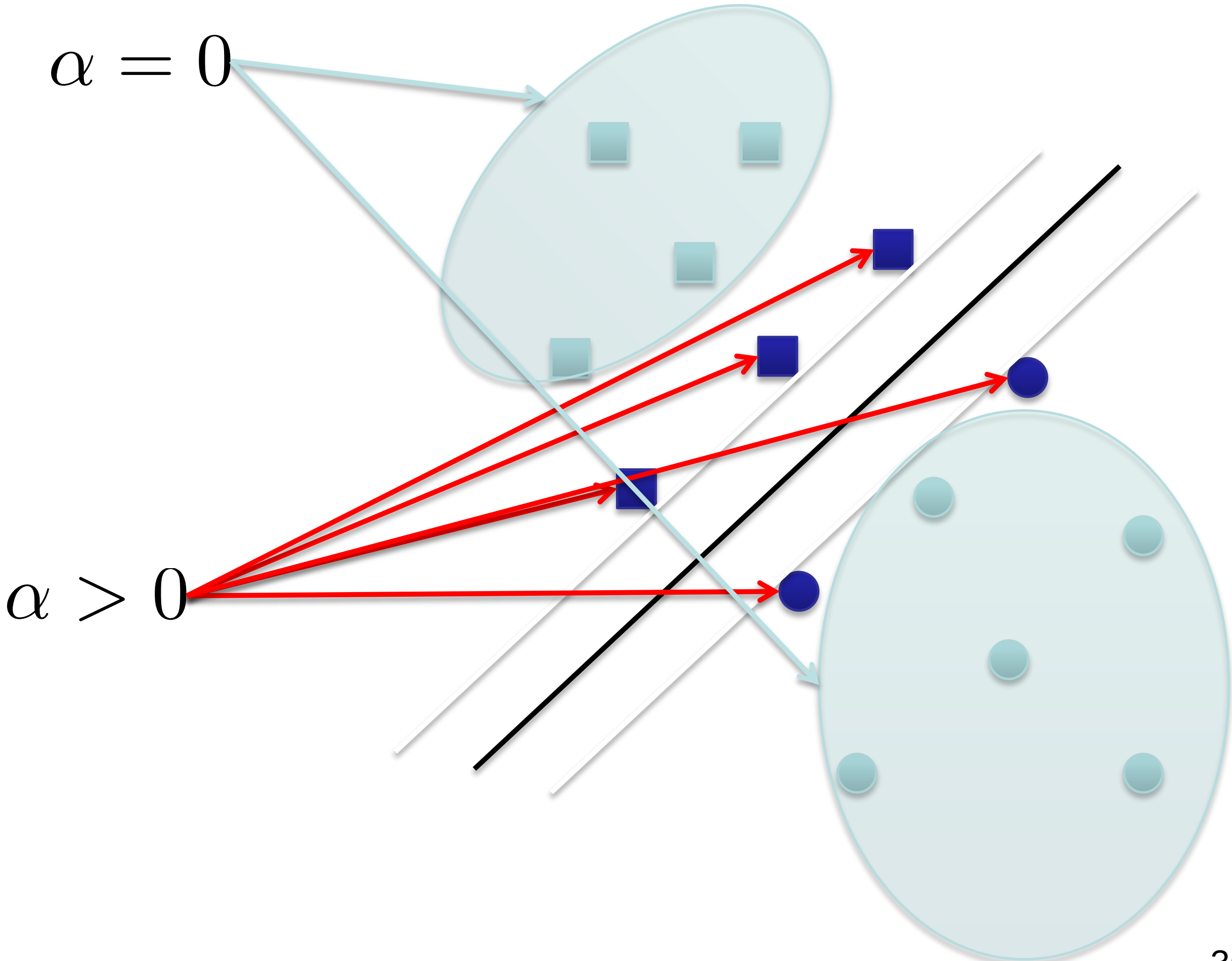
- In constraint optimization: At the optimal solution
  - Constraint * Lagrange Multiplier = 0

$$\alpha_i(1 - t_i(\vec{w}^T \vec{x}_i + b)) = 0$$

$$\text{if } \alpha_i \neq 0 \quad \rightarrow \quad t_i(\vec{w}^T \vec{x}_i + b) = 1$$

只有决策边界上的点对问题的求解有贡献!

支持向量的含义



SVM 参数的进一步理解

- What else can we tell from α’ s?
  - If α is large, then the associated data point is quite important.
  - It’s either an outlier, or incredibly important.

核方法（kernel method）简介

$$W(\alpha)=\sum_{i=0}^{N-1}\alpha_i-\frac{1}{2}\sum_{i,j=0}^{N-1}\alpha_i\alpha_j t_i t_j(\vec{x_i}\cdot\vec{x_j})$$

$$\boxed{\vec{w} \quad = \quad \sum_{i=0}^{N-1}\alpha_i t_i \vec{x_i}}$$

- 由于目标函数只和向量的点积有关，
- 决策过程与数据维数无关！
- 可以将数据映射到线性可分的高维空间！

核方法（kernel method）简介

$$W(\alpha) = \sum_{i=0}^{N-1} \alpha_i - \frac{1}{2} \sum_{i,j=0}^{N-1} \alpha_i \alpha_j t_i t_j (\vec{x_i} \cdot \vec{x_j})$$

- 利用点积的对应关系进行空间映射：

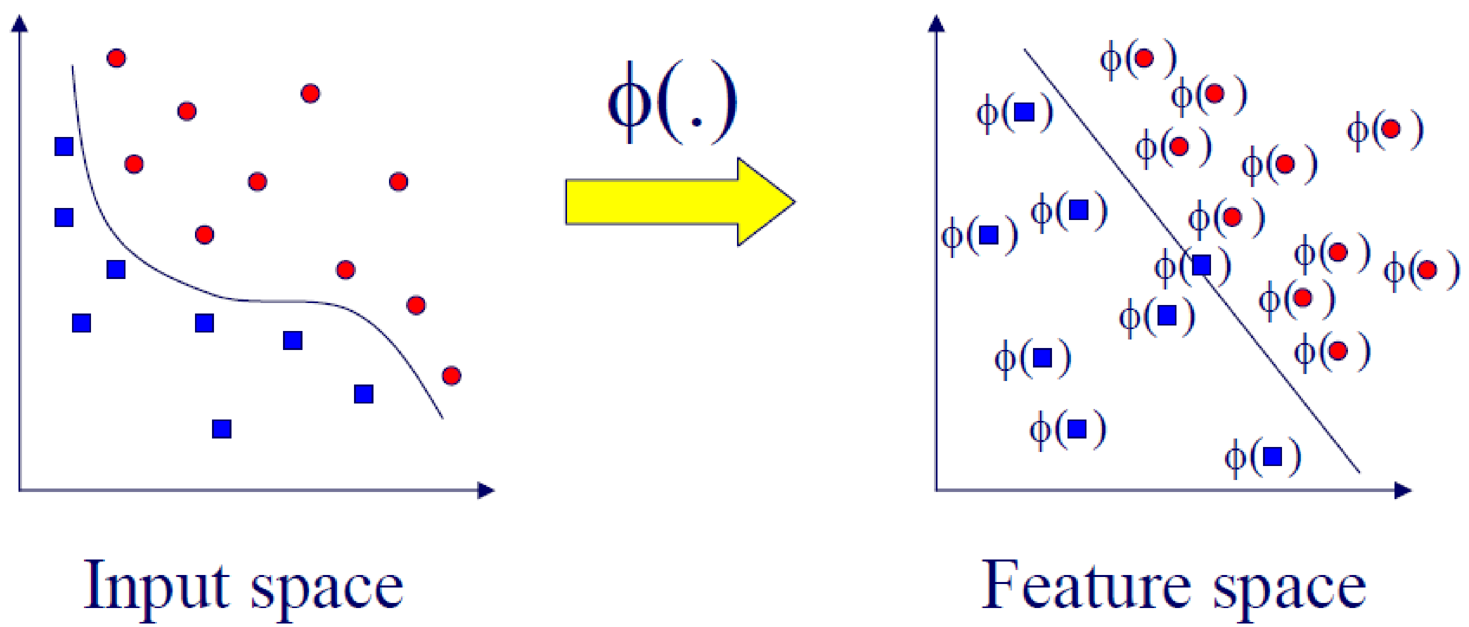
$$\vec{x_i} \cdot \vec{x_j} \longrightarrow \phi(\vec{x_i}) \cdot \phi(\vec{x_j})$$

$$W(\alpha) = \sum_{i=0}^{N-1} \alpha_i - \frac{1}{2} \sum_{i,j=0}^{N-1} \alpha_i \alpha_j t_i t_j (\phi(\vec{x_i}) \cdot \phi(\vec{x_j}))$$

- 核函数：. $K(\vec{x}_i, \vec{x}_j) = \phi(\vec{x}_i) \cdot \phi(\vec{x}_j)$

核方法（kernel method）简介

将数据映射到线性可分的高维空间



软间隔分类
Soft margin classification

- 软间隔：容许数据位于分类间隔内侧
- 优化方案：引入惩罚项 ξ

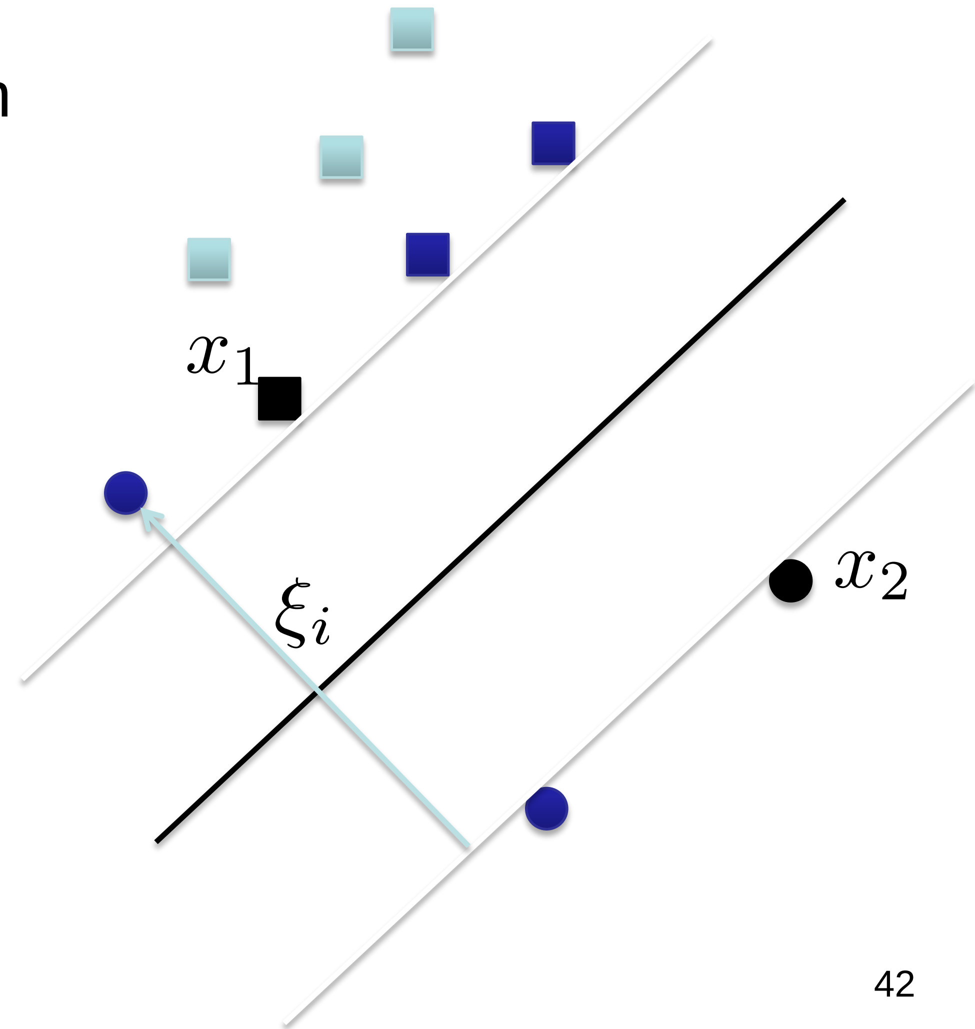
$$\min \|\vec{w}\| + C\sum_{i=0}^{N-1}\xi_i$$

$$\text{where } t_i(\vec{w}^T x_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

$$L(\vec{w}, b) = \frac{1}{2}\vec{w}\cdot\vec{w} + C\sum_{i=0}^{N-1}\xi_i - \sum_{i=0}^{N-1}\alpha_i[t_i((\vec{w}\cdot\vec{x_i}) + b) + \xi_i - 1]$$

软间隔分类（Soft margin）

- Points are allowed within the margin, but cost is introduced.



对偶式（Soft Max Dual）

$$\min \|\vec{w}\| + C\sum_{i=0}^{N-1}\xi_i$$

$$\text{where } t_i(\vec{w}^T x_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

$$\boxed{L(\vec{w},b) = \frac{1}{2}\vec{w}\cdot\vec{w} + C\sum_{i=0}^{N-1}\xi_i - \sum_{i=0}^{N-1}\alpha_i[t_i((\vec{w}\cdot\vec{x_i}) + b) + \xi_i - 1]}$$

仍然是二次规划问题!

$$\boxed{W(\alpha) = \sum_{i=0}^{N-1}\alpha_i - \frac{1}{2}\sum_{i,j=0}^{N-1} t_i t_j \alpha_i \alpha_j (x_i \cdot x_j)}$$

$$\text{where } 0 \leq \alpha_i \leq C \qquad \sum_{i=0}^{N-1}\alpha_i t_i = 0$$

SVM 的效率

- 训练 – O（n^3）
  - – Quadratic Programming efficiency
- 测试 – O（n）
  - – Need to evaluate against each support vector (potentially n)

SVM中的学习理论

- SVM中测试误差的理论界限：
  - The upper bound doesn't depend on the dimensionality of the space
  - The lower bound is maximized by maximizing the margin, γ, associated with the decision boundary.

为何人们喜欢SVM？

- They work
  - Good generalization
- Easily interpreted.
  - Decision boundary is based on the data in the form of the **support vectors**.
    - Not so in multilayer perceptron networks
- Principled bounds on testing error from Learning Theory

SVM 与 概率模型

- SVM 得到的是决策函数
  - 决策模型: $f(x) = \mathrm{argmax}_c\ p(c|x)$
- SVM 不是基于数据的概率密度函数的！
- **SVM 没有使用概率模型。**

SVM 和 多层感知机（MLP）的比较

- SVMs have many fewer parameters
  - SVM: Maybe just a kernel parameter
  - MLP: Number and arrangement of nodes and eta learning rate

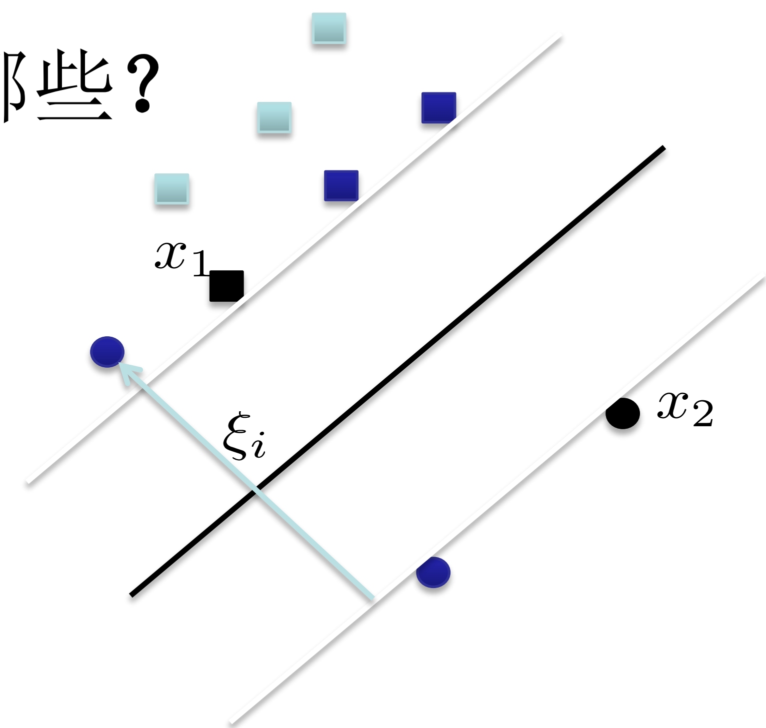
- SVM: Convex optimization task
  - MLP: likelihood is non-convex -- local minima

$$R(\theta) = \frac{1}{N}\sum_{n=0}^{N}\frac{1}{2}\left(y_n - g\left(\sum_k w_{kl}\, g\left(\sum_j w_{jk}\, g\left(\sum_i w_{ij} x_{n,i}\right)\right)\right)\right)^2$$

讨论

- SVM 的**软间隔分类法**是否可以解决**线性不可分问题**？

- **软间隔分类法**的**好处**有哪些？

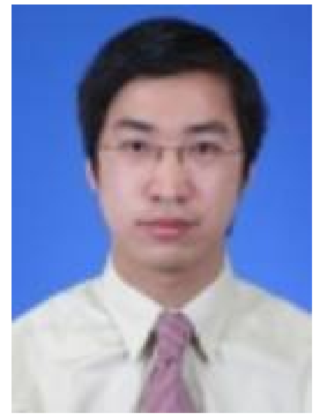


参考资料

Andrew Rosenberg

Assistant Professor
Computer Science
Queens College (CUNY)

Machine Learning PPT



黄开竹
kzhuang@nlpr.ia.ac.cn

http://liama.ia.ac.cn/wiki/projects:pal:course:pr

中科院自动化所博士生模式识别课程讲义