



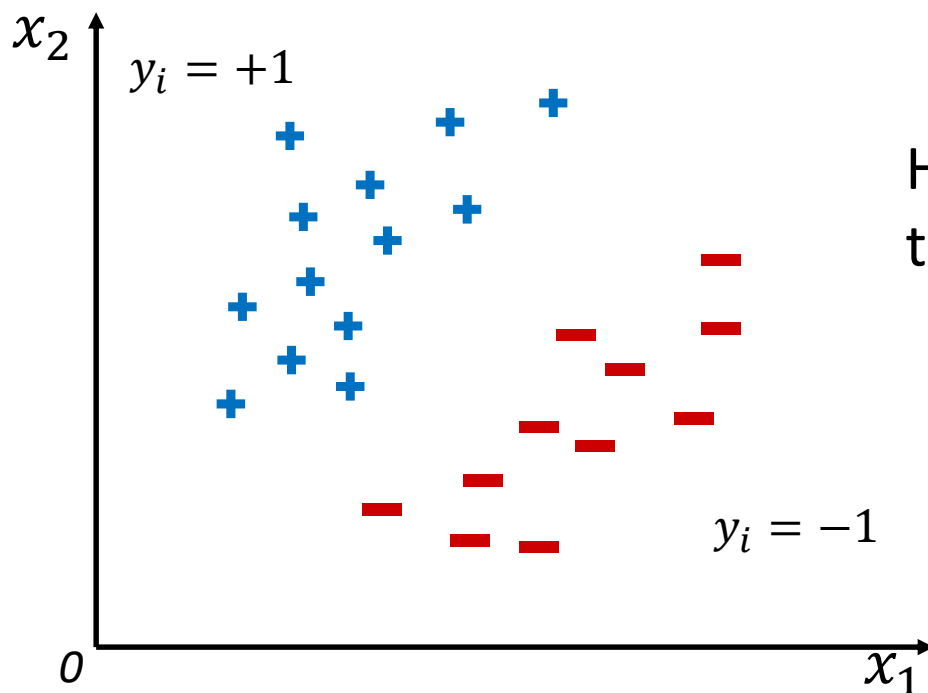
# Support vector machine

# Outline

- Margin and support vector
- Dual problem
- Kernel function
- Soft margin
- Support Vector Regression
- Usage of LIBSVM with matlab

# Linear Classifier

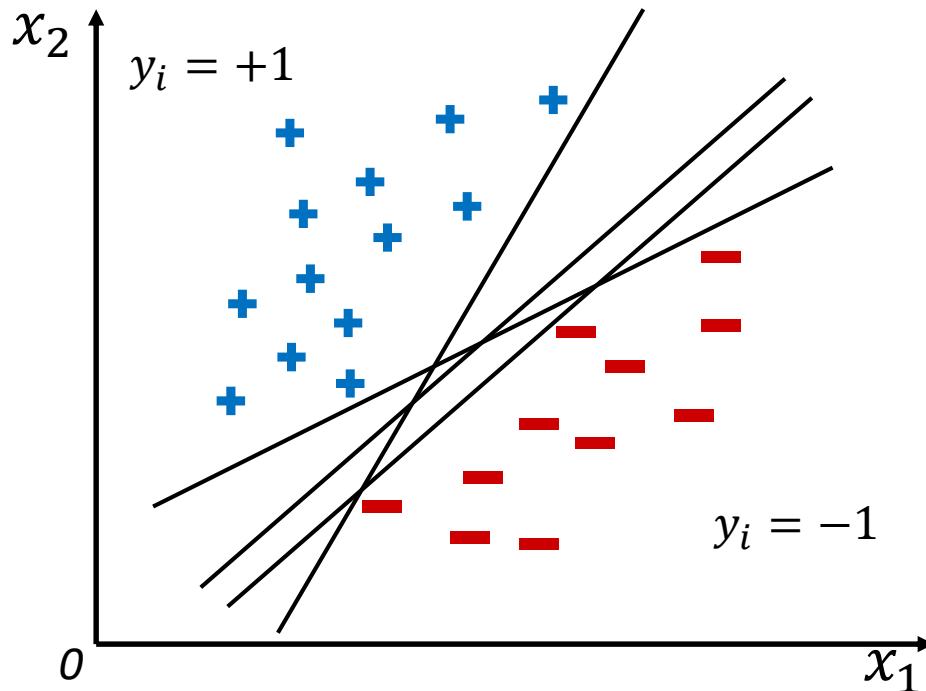
Train data set  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ,  $y_i \in \{-1, +1\}$



How would you classify this data **correctly** ?

# Linear Classifier

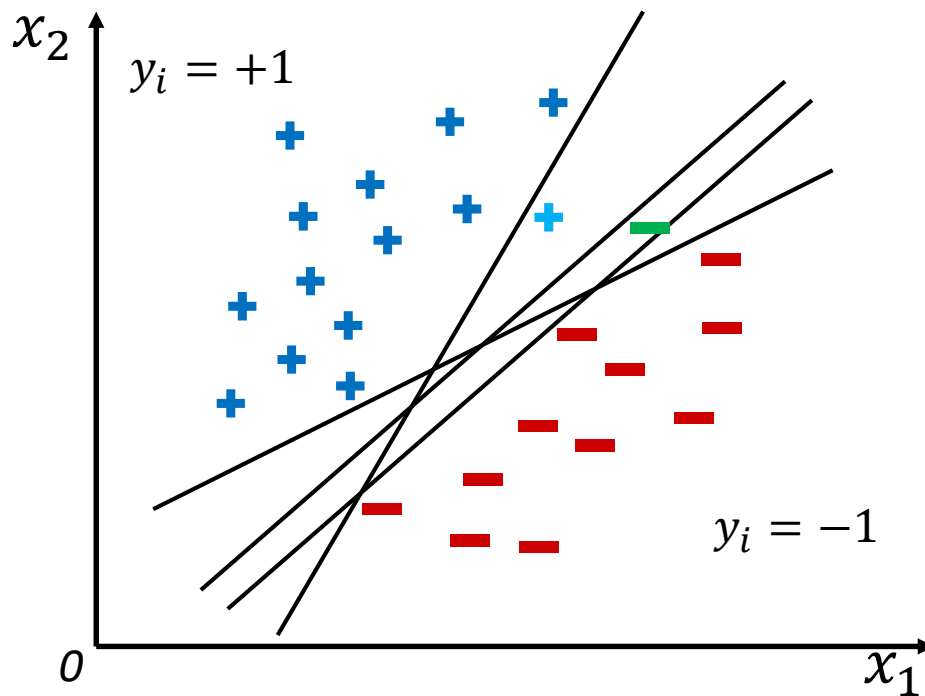
Train data set  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ,  $y_i \in \{-1, +1\}$



Any of these  
would be fine...

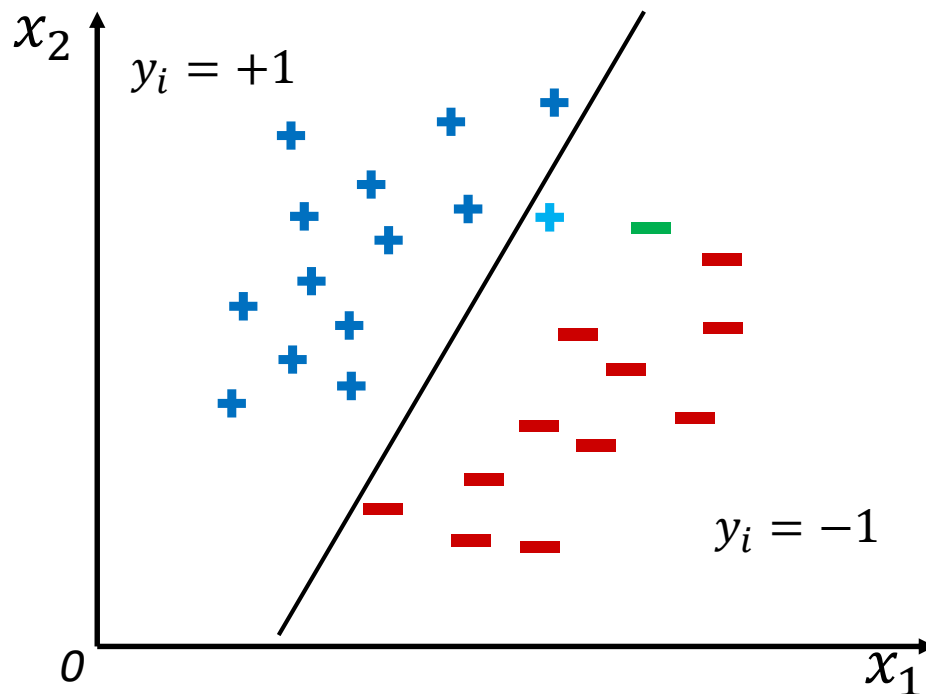
...but which one is  
the best?

# Linear Classifier



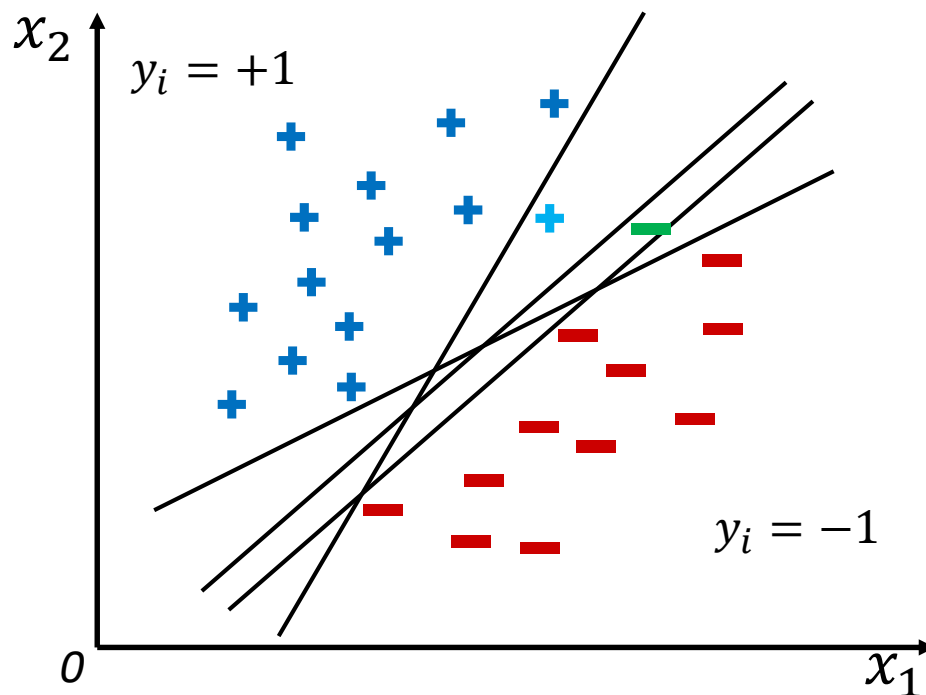
- In addition to correction, we manage to find a **robust** classifier which has good **generalization ability**(泛化能力) to new data.

# Linear Classifier



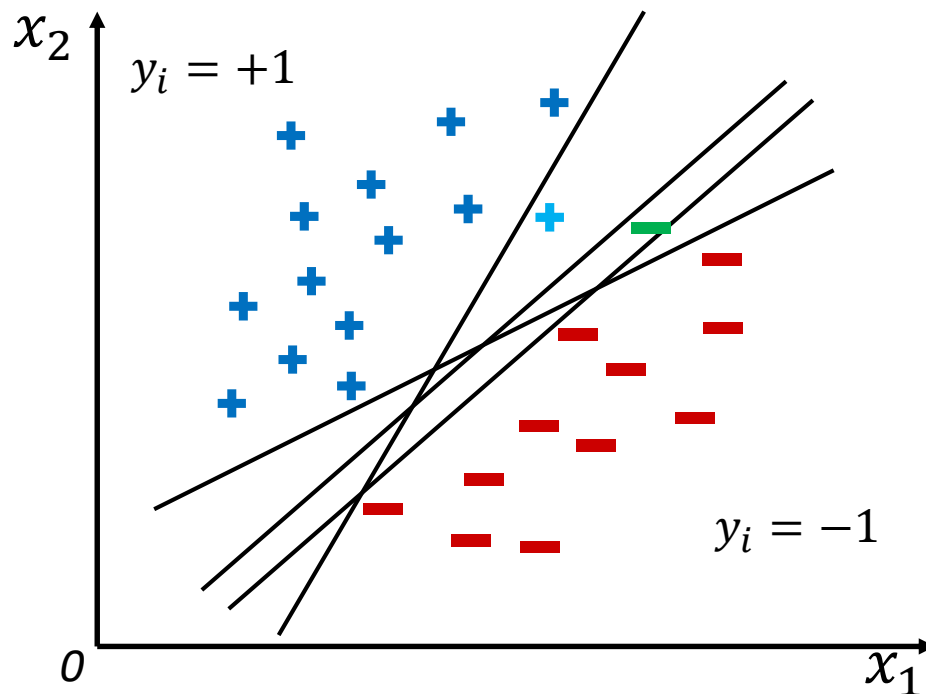
- In addition to correction, we manage to find a **robust** classifier which has good **generalization ability**(泛化能力) to new data.

# Linear Classifier



- In addition to correction, we manage to find a **robust** classifier which has good **generalization ability**(泛化能力) to new data.

# Linear Classifier

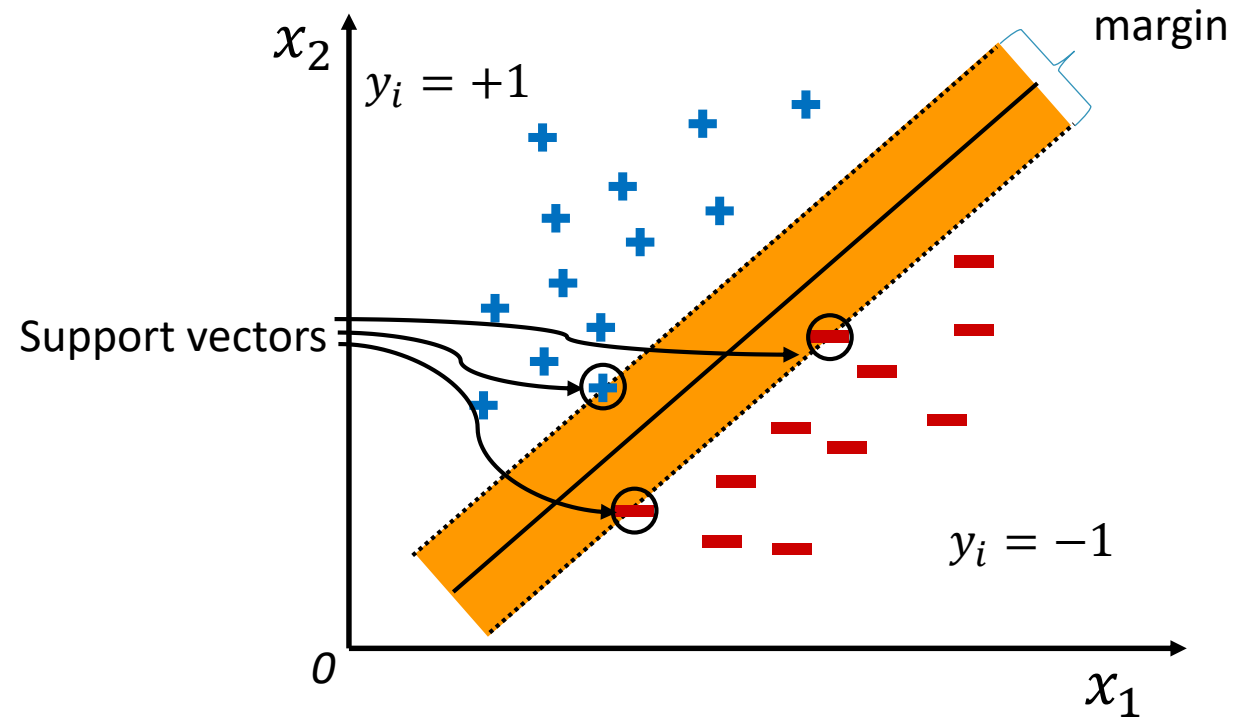


In order to classify new data closer to the line correctly, the minimum distance between the line and training samples should be as far as possible.

- In addition to correction, we manage to find a **robust** classifier which has good **generalization ability**(泛化能力) to new data.

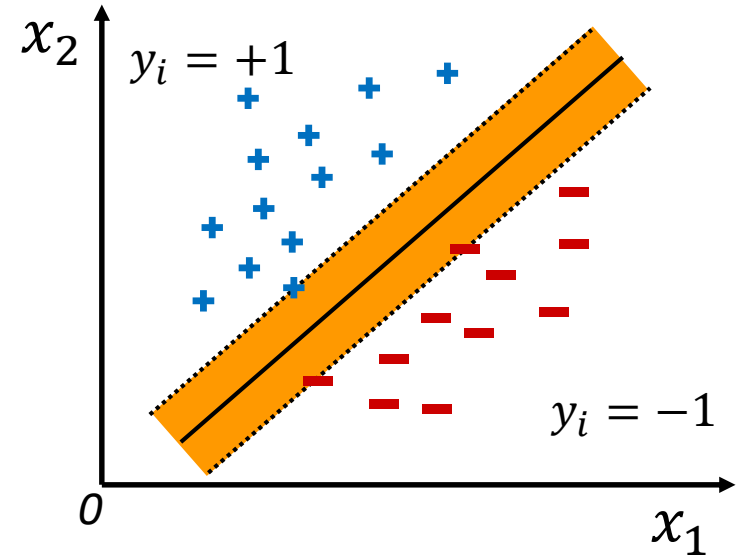
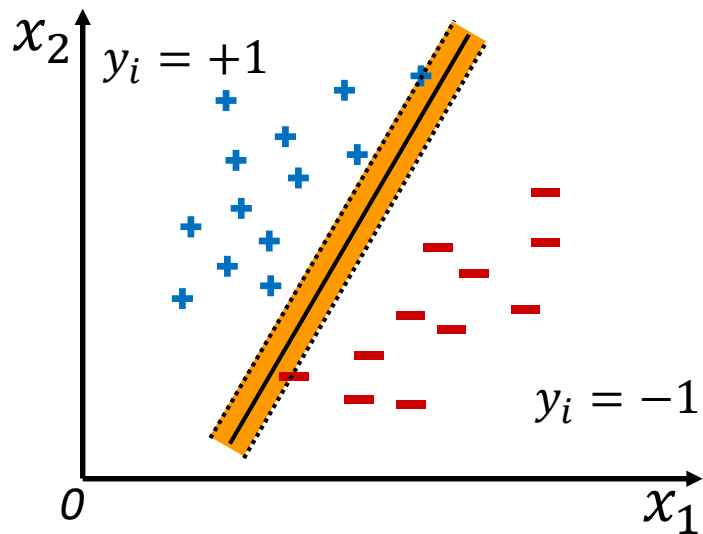


# Margin(间隔)



Define the **margin** of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

# Maximum margin classifier

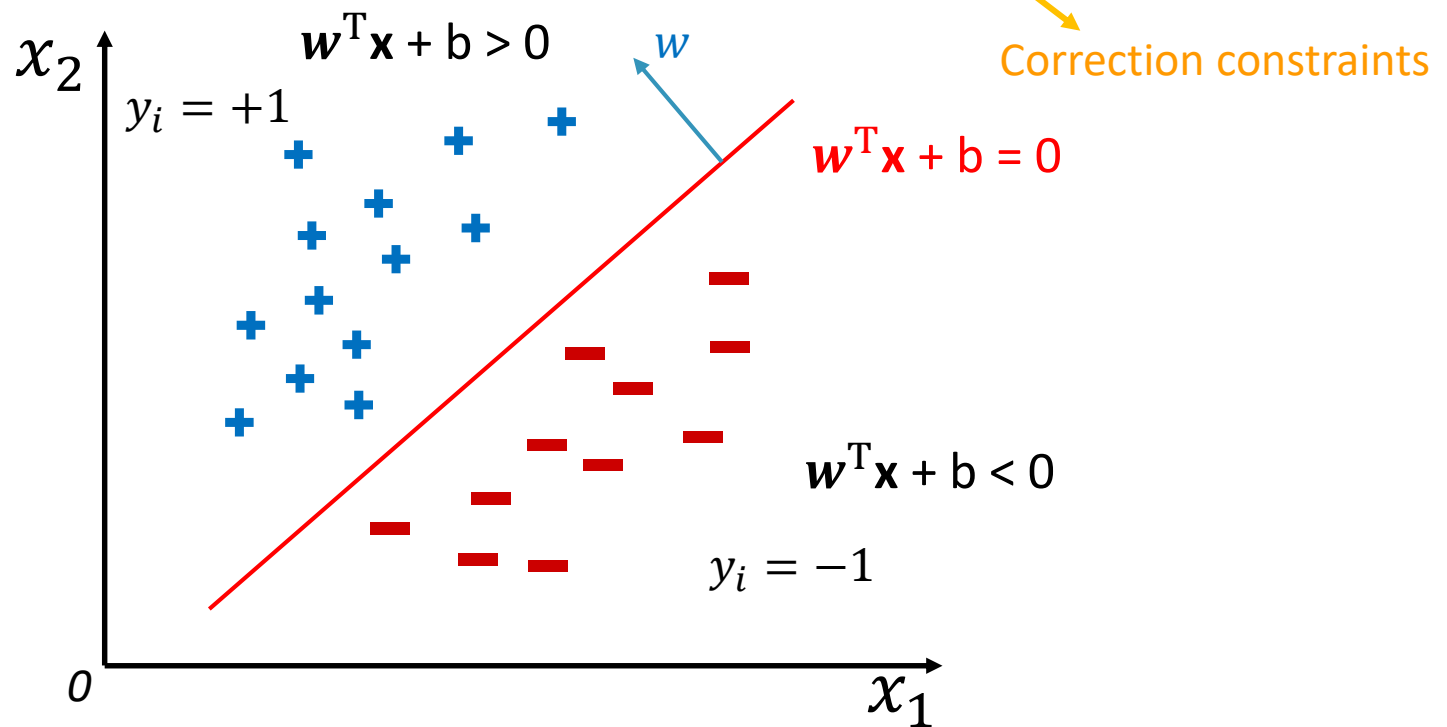


Each classifier corresponds to a margin.

Pick the one with the **largest** margin !

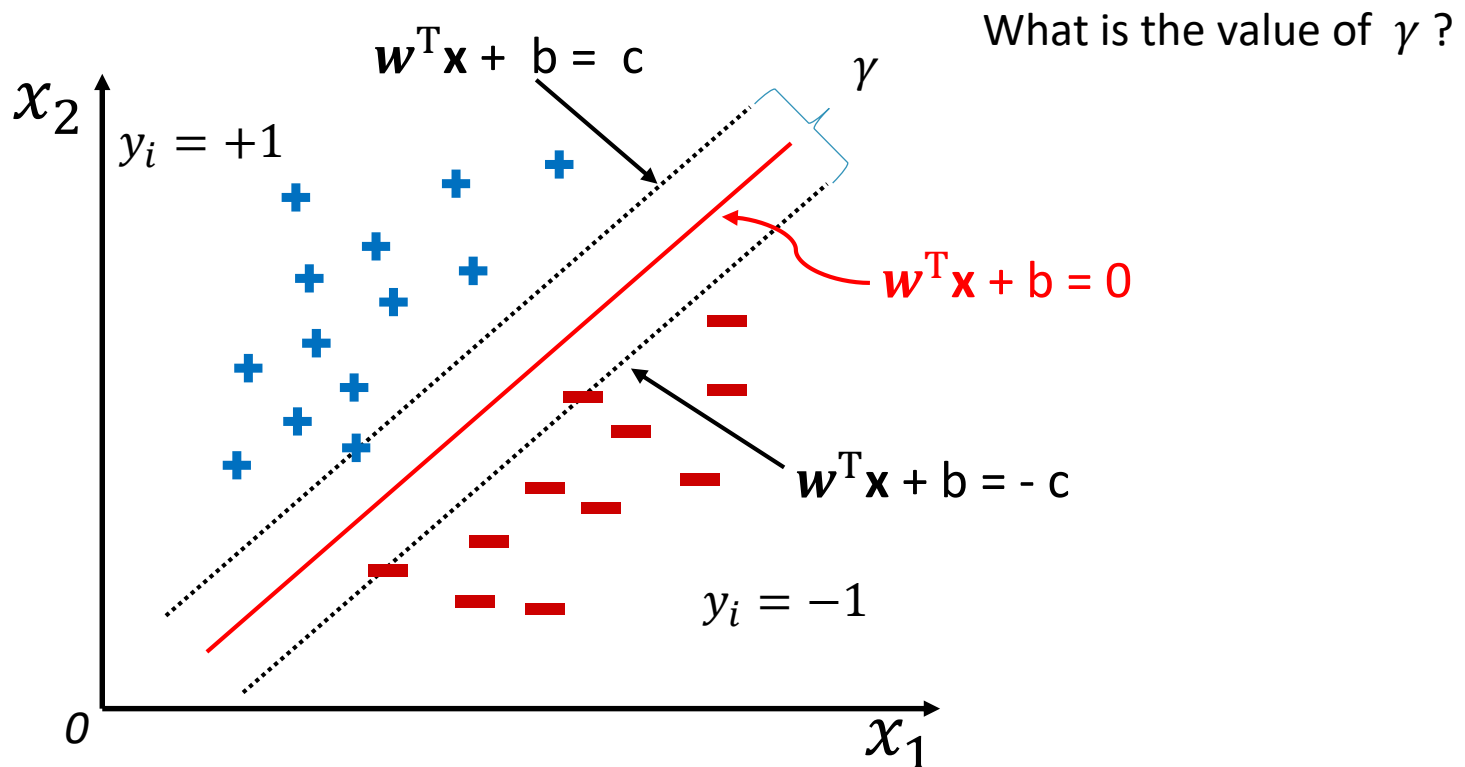
# Specifying a hyper-plane

- A hyper-plane:  $\mathbf{w}^T \mathbf{x} + b = 0$
- For  $(x_i, y_i) \in D$ , if  $y_i = +1$ ,  $\mathbf{w}^T \mathbf{x}_i + b > 0$ ; if  $y_i = -1$ ,  $\mathbf{w}^T \mathbf{x}_i + b < 0$



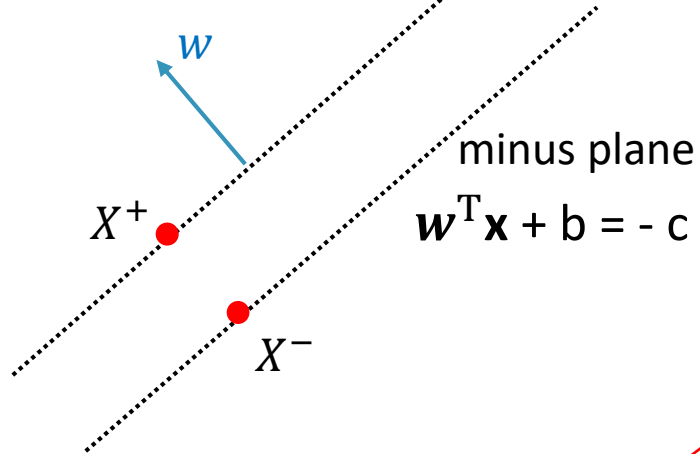
# Maximum margin

- Margin:  $\gamma$
- Goal :  $\max\{\text{margin}\}$



# Maximum margin

plus plane:  $\mathbf{w}^T \mathbf{x} + b = c$



- Let  $X^-$  be any point on the minus plane
- Let  $X^+$  be the closest plus-plane-point to  $X^-$ .

Claim:  $X^+ = X^- + \alpha \mathbf{w}$  for some value of  $\alpha$ .

Why?

What we know:

- $\mathbf{w}^T X^+ + b = c$
- $\mathbf{w}^T X^- + b = -c$
- $X^+ = X^- + \alpha \mathbf{w}$
- $|X^+ - X^-| = \gamma$

$$\mathbf{w}^T (X^- + \alpha \mathbf{w}) + b = c$$

$\Rightarrow$

$$\mathbf{w}^T X^- + b + \alpha \mathbf{w}^T \mathbf{w} = c$$

$\Rightarrow$

$$-c + \alpha \mathbf{w}^T \mathbf{w} = c$$

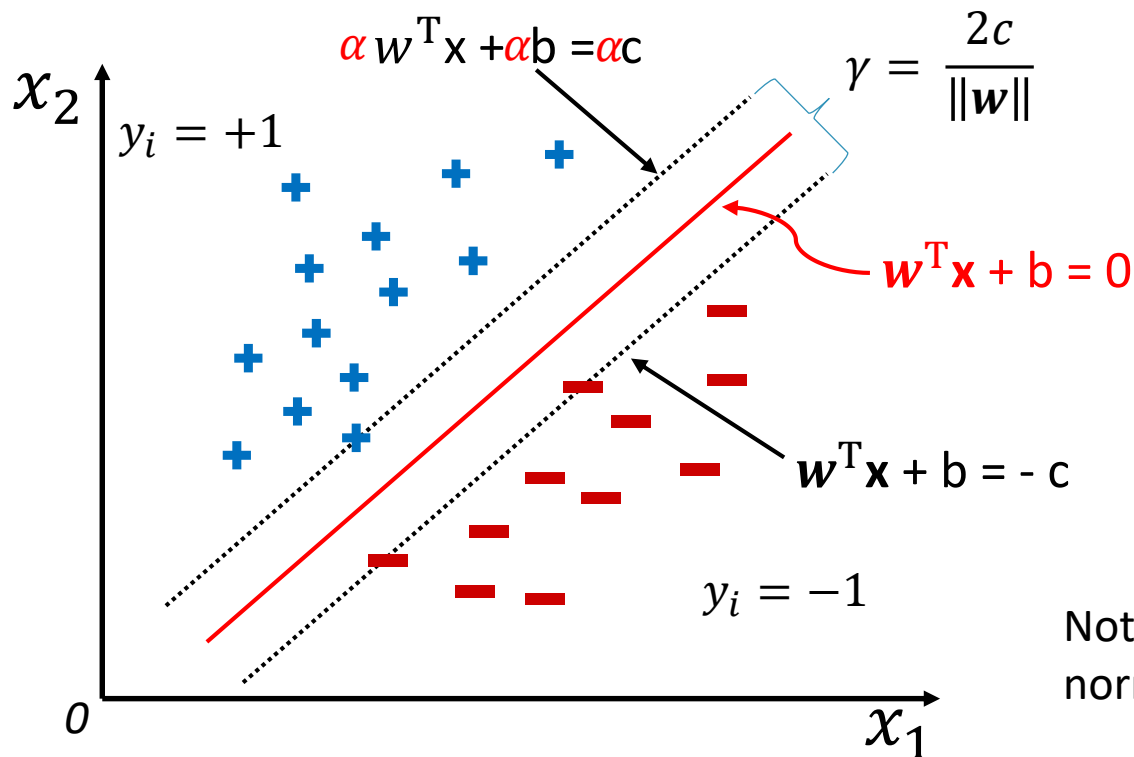
$\Rightarrow$

$$\alpha = \frac{2c}{\mathbf{w}^T \mathbf{w}} = \frac{2c}{\|\mathbf{w}\|^2}$$

$$\gamma = |X^+ - X^-| = |\alpha \mathbf{w}| = \alpha \|\mathbf{w}\| = \frac{2c}{\|\mathbf{w}\|^2} \|\mathbf{w}\| = \frac{2c}{\|\mathbf{w}\|}$$

# Maximum margin

- Margin  $\gamma = \frac{2c}{\|w\|}$
- Goal : max{margin}

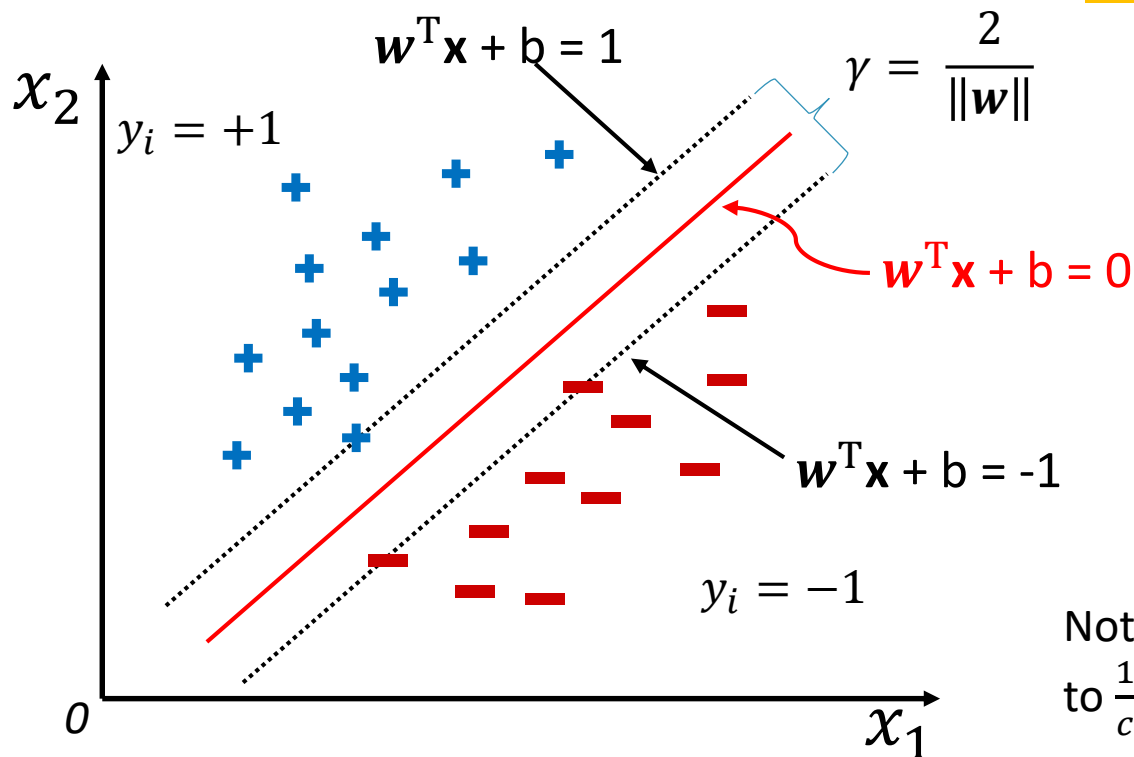


Note: 'c' is arbitrary (can normalize equations by c)

# Maximum margin

- Margin  $\gamma = \frac{2}{\|w\|}$
- Goal : max{margin}

$$\alpha = \frac{1}{c}$$



Note: the ' $w$ ' here is equal to  $\frac{1}{c}$   $w$  of last slide, ' $b$ ' too

# Primal optimization problem

- $\mathbf{w}^T \mathbf{x}_i + b \geq 1$ , if  $y_i = +1$
  - $\mathbf{w}^T \mathbf{x}_i + b \leq -1$ , if  $y_i = -1$
- $\longrightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, i = 1, 2, \dots, m$

$$\max_{\mathbf{w}, b} \frac{2}{\|\mathbf{w}\|}$$

$$\text{s.t. } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m$$



$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2$$

$$\text{s.t. } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m$$



# Outline

- Margin and support vector
- **Dual problem**
- Kernel function
- Soft margin and regularization
- Support Vector Regression
- Usage of LIBSVM with matlab

# Lagrangian duality

- Primal optimization problem:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m \end{aligned} \quad (*)$$

It is a convex quadratic programming problem, can be resolved by existed tool.  
But we have more efficient method:

- Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$$

$$\alpha = (\alpha_1; \alpha_2; \dots; \alpha_m), \alpha_i \geq 0$$

$$\max_{\alpha: \alpha_i \geq 0} L(\mathbf{w}, b, \alpha) = \begin{cases} \frac{1}{2} \|\mathbf{w}\|^2, & \text{if } w, b \text{ satisfies primal constraints} \\ \infty, & \text{otherwise} \end{cases}$$

- re-written (\*) :  $\min_{w,b} \max_{\alpha: \alpha_i \geq 0} L(\mathbf{w}, b, \alpha)$

# Lagrangian duality

- Recall the primal problem:

$$\min_{\mathbf{w}, b} \max_{\alpha: \alpha_i \geq 0} L(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$$

- The dual problem:

$$\max_{\alpha: \alpha_i \geq 0} \min_{\mathbf{w}, b} L(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$$

- We change to resolve the dual problem, why?
  - Easier to resolve (can use an efficient algorithm better than generic QP software)
  - Allow us to use kernels used in linearly inseparable problem See later...
- Can we do that ?

$$d^* = \max_{\alpha: \alpha_i \geq 0} \min_{\mathbf{w}, b} L(\mathbf{w}, b, \alpha) \leq \min_{\mathbf{w}, b} \max_{\alpha: \alpha_i \geq 0} L(\mathbf{w}, b, \alpha) = p^*$$

but, under certain conditions:  $d^* = p^*$

Fortunately, our problem satisfies the “certain conditions”.

# Lagrangian duality

- The dual problem:

$$\max_{\alpha: \alpha_i \geq 0} \min_{w, b} L(w, b, \alpha) = \frac{1}{2} \|w\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i (w^T x_i + b))$$

- We minimize  $L$  with respect to  $w$  and  $b$  first:

$$\frac{\partial L}{\partial w} = 0 \longrightarrow w = \sum_{i=1}^m \alpha_i y_i x_i \quad (*)$$

$$\frac{\partial L}{\partial b} = 0 \longrightarrow \sum_{i=1}^m \alpha_i y_i = 0 \quad (**)$$

- Plug  $(*)$  back to  $L(w, b, \alpha)$ , and using  $(**)$ , we have:

$$L(w, b, \alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i^T x_j$$

# Dual problem

- Now we have the dual problem:

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0, \\ \alpha_i \geq 0, \quad i = 1, 2, \dots, m$$

- Resolve the above problem, we will get  $\alpha$ , then  $\mathbf{w}$ ,  $b$ , and then the final model:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

how to resolve the dual problem will be told later...

# KKT conditions

- Under “certain conditions” , there must exist  $\mathbf{w}, b, \alpha$  so that  $(\mathbf{w}, b)$  is the solution to the primal problem,  $\alpha$  is the solution to the dual problem,  $(\mathbf{w}, b, \alpha)$  satisfy the *Karush-Kuhn-Tucker (KKT)* conditions, which are as follows:

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

$$0 = \sum_{i=1}^m \alpha_i y_i$$

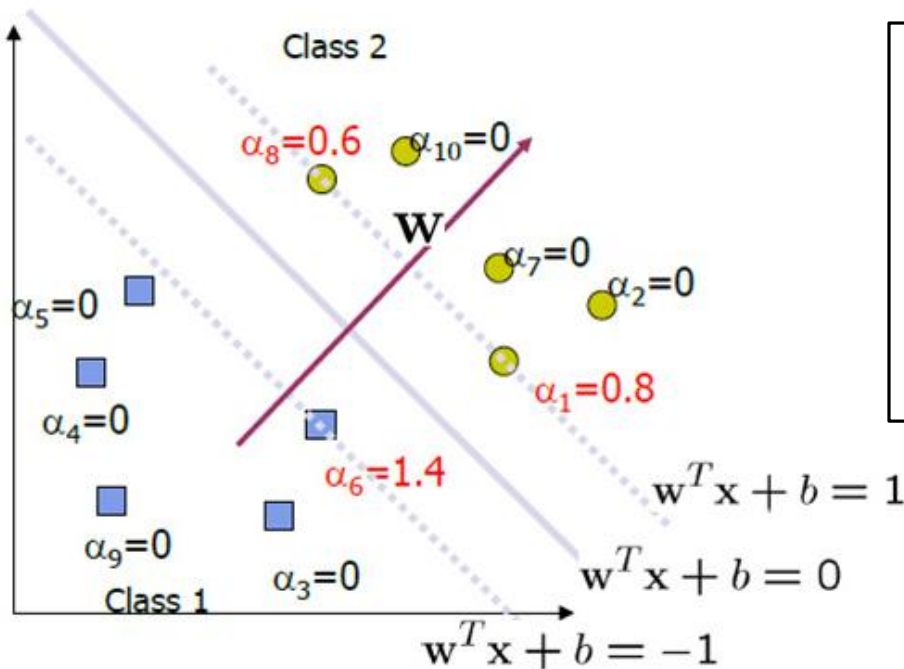
$$\alpha_i \geq 0$$

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \geq 0$$

$$\alpha_i(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1) = 0.$$

# Support vector

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



- $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 > 0 \Rightarrow \alpha_i = 0$
- $\alpha_i > 0 \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$   
 $\mathbf{x}_i$  is on the margin, is a **support vector**

- Call the training data points whose  $\alpha$  are nonzero the **support vectors** (SV)

# Support vector

- The model:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

- Many  $\alpha_i$  are zero.
- The final model  $f(\mathbf{x})$  is determined only by the support vectors , since:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{\alpha_i \neq 0} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

- For a test data  $\mathbf{z}$  , compute:

$$f(\mathbf{z}) = \mathbf{w}^T \mathbf{z} + b = \sum_{\alpha_i \neq 0} \alpha_i y_i \mathbf{x}_i^T \mathbf{z} + b$$

classify  $\mathbf{z}$  as class 1 if the sum is positive, and class -1 otherwise



# SMO algorithm

- Now we back to resolve the dual problem:

$$\max_{\alpha} L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0, \\ \alpha_i \geq 0, \quad i = 1, 2, \dots, m$$

- SMO(sequential minimal optimization) algorithm:

Repeat till convergence:

1. Select some pair  $\alpha_i$  and  $\alpha_j$  to update next
2. Re-optimize  $L(\alpha)$  with respect to  $\alpha_i$  and  $\alpha_j$ , while holding all the other  $\alpha_k$  ( $k \neq i; j$ ) fixed.

# Determine 'w' & 'b'

- $w$ : 
$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

- $b$ : For any support vector  $(\mathbf{x}_s, y_s)$ ,

$$y_s \left( \sum_{i \in t} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_s + b \right) = 1$$

$t = \{i | \alpha_i > 0, i = 1, 2, \dots, m\}$ , the subscript set of support vectors

$$\sum_{i \in t} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_s + b = \frac{1}{y_s} = y_s$$

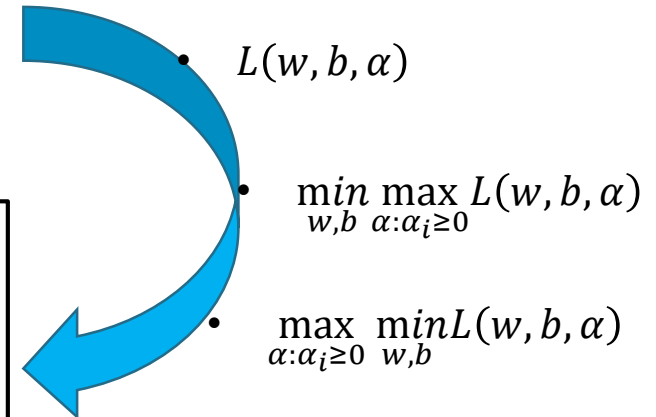
$$b = y_s - \sum_{i \in t} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_s$$

# summary

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|w\|^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \quad i = 1, 2, \dots, m \end{aligned}$$

$$\max_{\alpha} L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\ & \alpha_i \geq 0, \quad i = 1, 2, \dots, m \end{aligned}$$



- Solve the dual problem , get the value of  $\alpha$

- Calculate 'w' and 'b' :  $w = \sum_{i=1}^m \alpha_i y_i x_i$   $b = y_s - \sum_{i \in t} \alpha_i y_i x_i^T x_s$

- The final model:

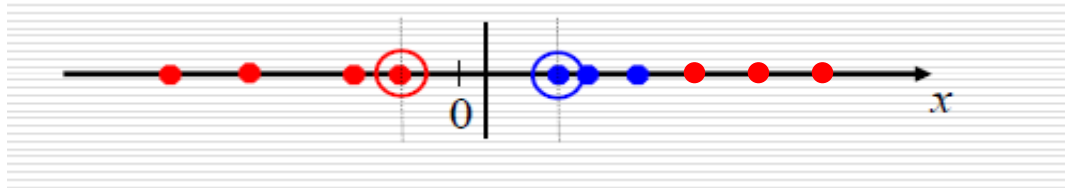
$$f(x) = w^T x + b = \sum_{i=1}^m \alpha_i y_i x_i^T x + b = \sum_{\alpha_i \neq 0} \alpha_i y_i x_i^T x + b$$

# Outline

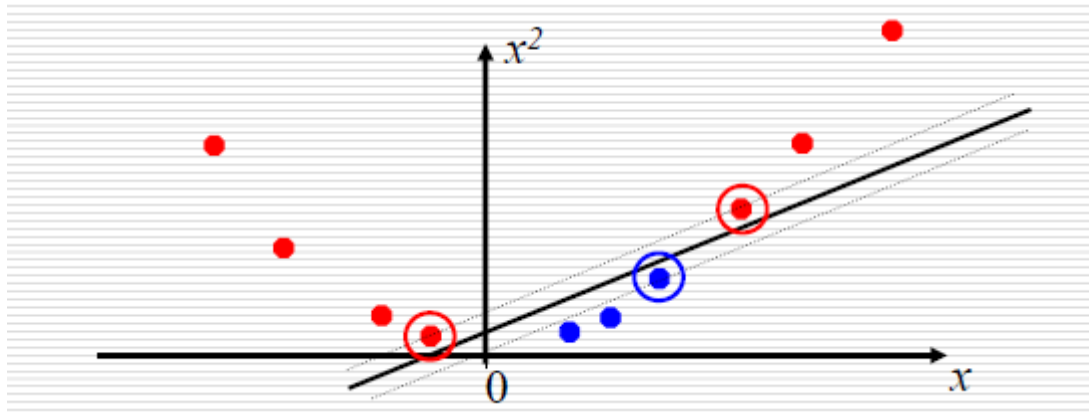
- Margin and support vector
- Dual problem
- **Kernel function**
- Soft margin
- Support Vector Regression
- Usage of LIBSVM with matlab

# Kernel function

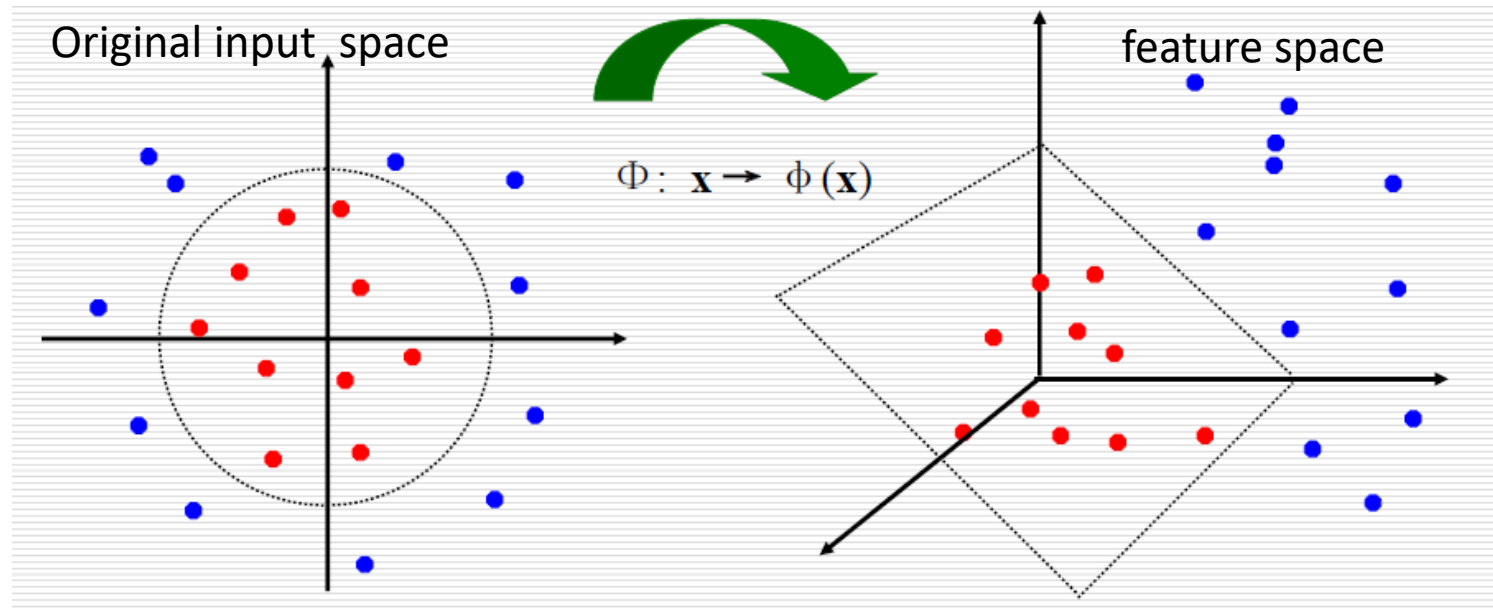
- Datasets are linearly inseparable.



- How about mapping data to a higher dimensional space :



# Kernel function



- General idea : the original input space can be mapped into some higher dimensional feature space where the training set can be separable.

# Kernel function

With this mapping ,our dual optimization problem becomes :

$$\max_{\alpha} L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \boxed{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)} \leftarrow \boxed{\mathbf{x}_i^T \mathbf{x}_j}$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \quad i = 1, 2, \dots, m$$

- Calculating the **inner product** of feature vectors in the feature space can be costly because it is high dimensional.
- The kernel trick comes to rescue:

$$\text{Kernel function} \leftarrow \boxed{k(\mathbf{x}_i, \mathbf{x}_j)} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

the inner product of feature vectors in the feature space  $\rightarrow$  calculation in the original input space by function  $k(\cdot, \cdot)$ .

# Kernel function

- An example:

3-dimension to 9-dimension

Suppose  $x, z \in R^3$ , and

$$\phi(x) = [x_1x_1, x_1x_2, x_1x_3, x_2x_1, x_2x_2, x_2x_3, x_3x_1, x_3x_2, x_3x_3]$$

Then,

$$\phi(x)^T \phi(z) = \sum_{i=1}^3 \sum_{j=1}^3 x_i x_j z_i z_j$$

We can use the kernel

$$K(x, z) = \sum_{i=1}^3 \sum_{j=1}^3 x_i x_j z_i z_j = (x_1 z_1 + x_2 z_2 + x_3 z_3)^2 = (x^T z)^2$$

$$K(x, z) = (x^T z)^2 = \phi(x)^T \phi(z)$$



# Kernel function

- Re-written dual optimization problem:

$$\max_{\alpha} L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \boxed{k(\mathbf{x}_i, \mathbf{x}_j)} \leftarrow \boxed{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \quad i = 1, 2, \dots, m$$

- Resolve it and we can get:

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i=1}^m \alpha_i y_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + b = \sum_{i=1}^m \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$$

# Kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly.

What kind of function  $k$  can be used as a kernel function?

- **Theorem (Mercer).** Let  $k : R^n \times R^n \rightarrow R$  be given. Then for  $k$  to be a valid (Mercer) kernel, it is necessary and sufficient that for any data  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ , ( $m < \infty$ ), the corresponding **kernel matrix** is **symmetric positive semi-definite**(对称 正半定).

**kernel matrix:**

$$\begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & \cdots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}$$

# Kernel function

- Examples of commonly-used kernel functions:

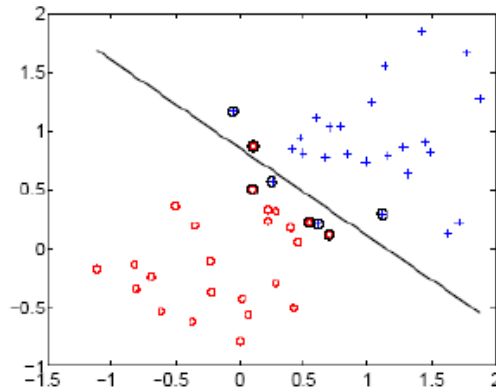
Linear kernel:  $K(x_i, x_j) = x_i^T x_j$

Polynomial kernel:  $K(x_i, x_j) = (1 + x_i^T x_j)^p$

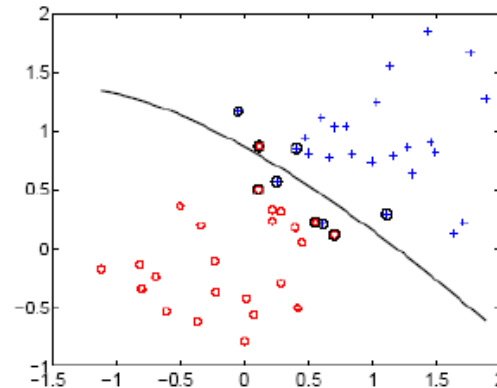
Gaussian (Radial-Basis Function(RBF)) kernel:  $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$

Sigmoid kernel:  $K(x_i, x_j) = \tanh(\beta_0 x_i^T x_j + \beta_1)$

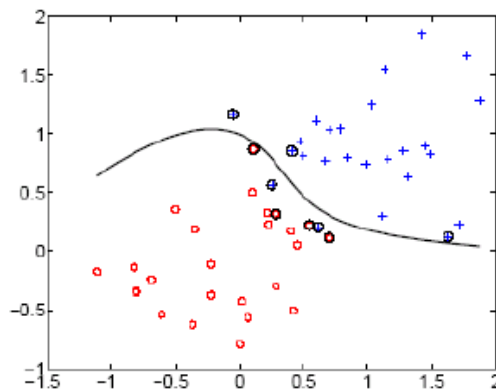
# SVM examples



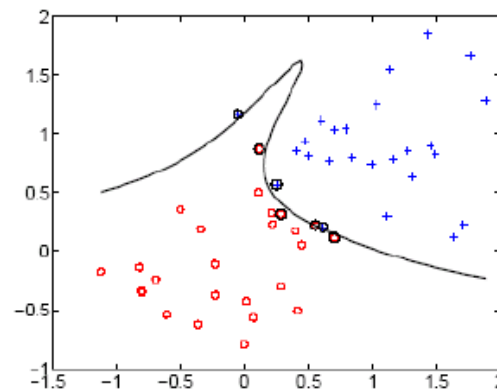
linear



2<sup>nd</sup> order polynomial



4<sup>th</sup> order polynomial

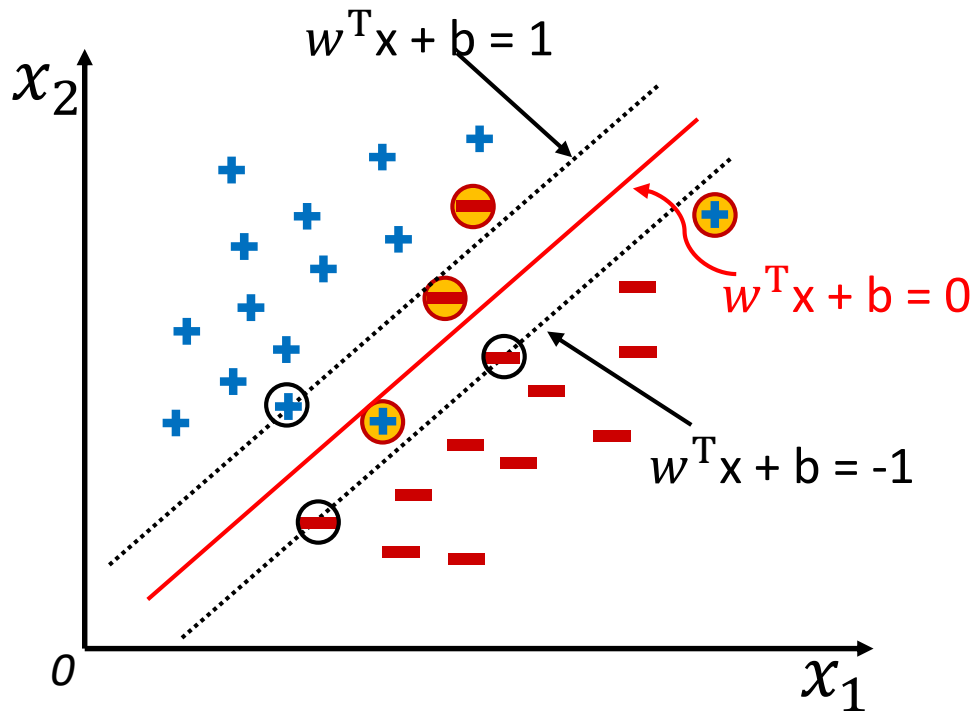


8<sup>th</sup> order polynomial

# outline

- Margin and support vector
- Dual problem
- Kernel function
- **Soft margin**
- Support Vector Regression
- Usage of LIBSVM with matlab

# Data with noise



- Allow some training samples don't satisfy the constraint:  
$$y_i(w^T x_i + b) \geq 1$$
- Maximize margin and minimize # training samples that don't satisfy the constraint

# Optimization object

Maximize margin

Minimize # training samples that don't satisfy the constraint

$$\min_{w,b} \left( \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m l_{0/1}(y_i(w^T x_i + b) - 1) \right)$$

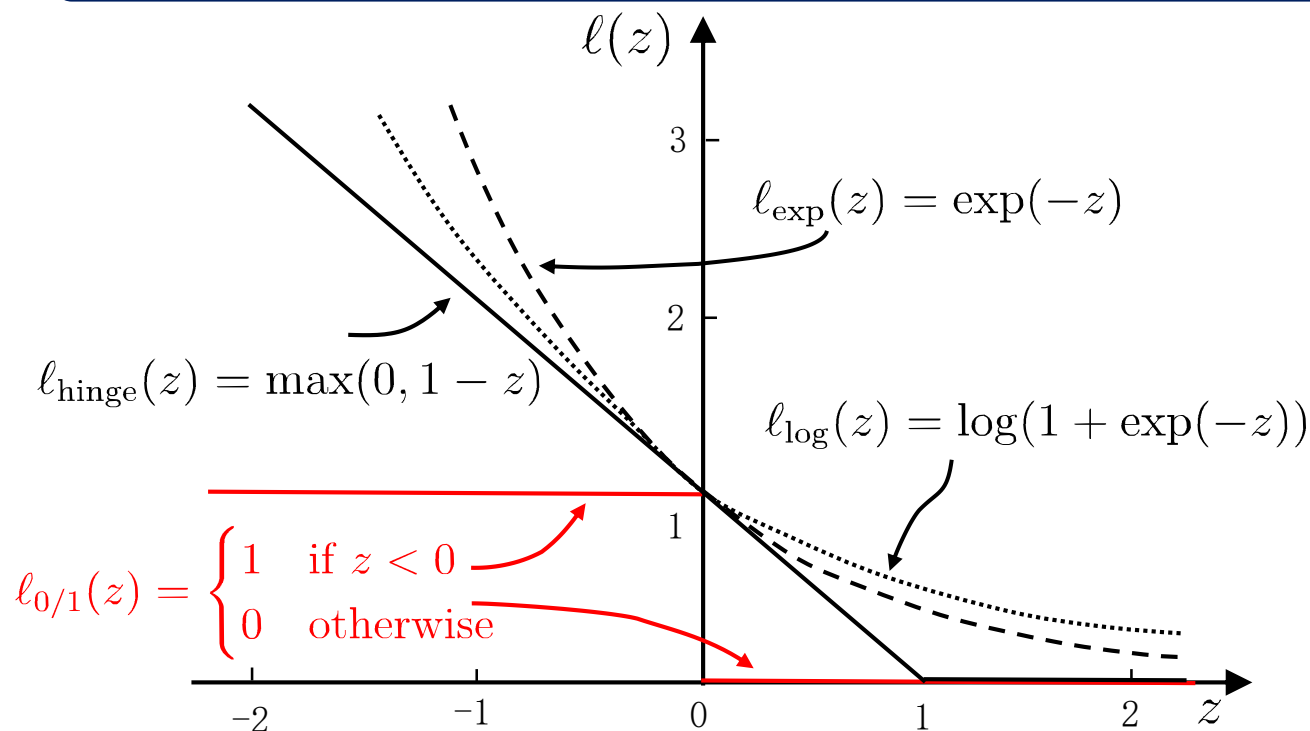
- $l_{0/1}$ ---loss function

$$l_{0/1}(z) = \begin{cases} 1, & \text{if } z < 0; \\ 0, & \text{otherwise.} \end{cases}$$

$C > 0$ , tradeoff parameter;  $C = \infty \longrightarrow$  hard margin SVM

But  $l_{0/1}$  is not convex, discontinuous (非凸、非连续)

# Surrogate loss function(替代损失)



hinge loss:  $\ell_{\text{hinge}}(z) = \max(0, 1 - z)$

exponential loss(指数损失):  $\ell_{\text{exp}}(z) = \exp(-z)$

logistic loss(对率损失):  $\ell_{\text{log}}(z) = \log(1 + \exp(-z))$



# Soft margin SVM

If we apply hinge loss , the optimization object becomes:

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \max(0, 1 - y_i(w^T x_i + b)) \quad (*)$$

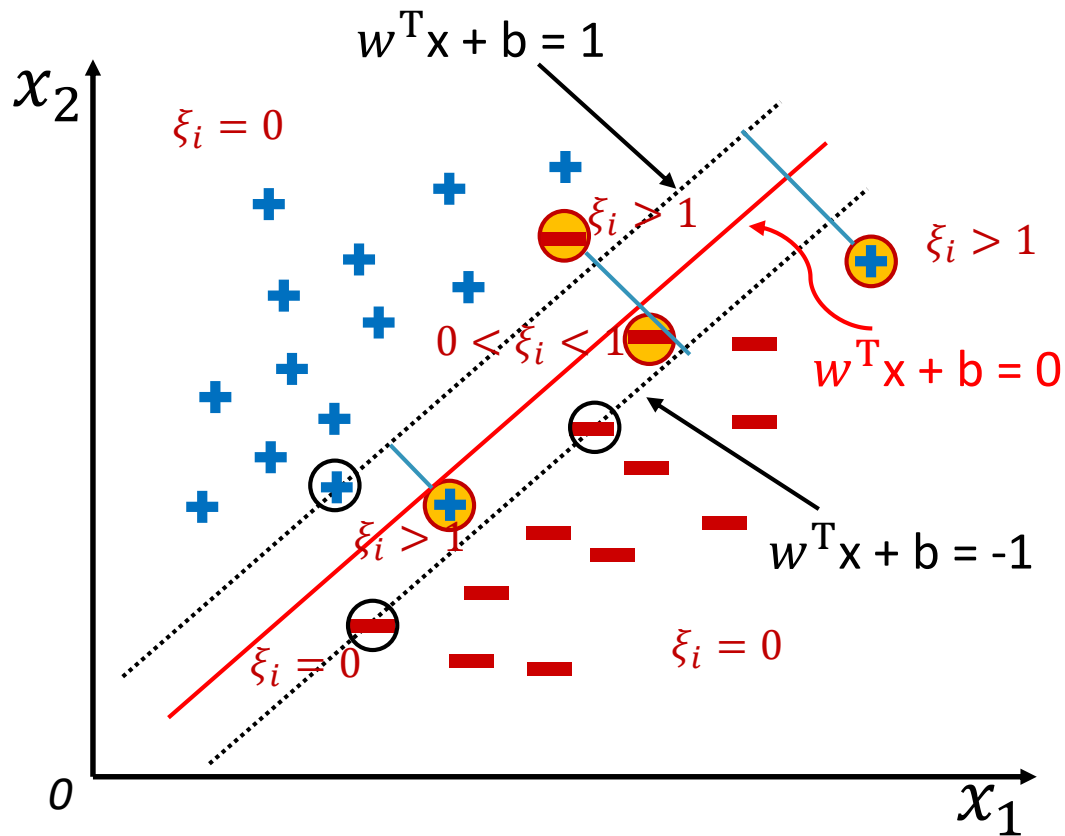
Introduce 'slack variables'  $\xi_i$ , rewrite (\*):

Soft margin SVM

$$\begin{aligned} \min_{w,b,\xi_i} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, 2, \dots, m \end{aligned}$$

- every sample have a  $\xi_i$  ,  $\xi_i$  denotes the degree of disobeying constraints.

# Soft margin SVM



Soften the constraints:

$$y_i(w^T x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, i = 1, 2, \dots, m$$

Penalty for misclassifying:

$$C\xi_i$$

# Lagrangian multipliers method

- Lagrangian:

$$L(w, b, \alpha, \xi, \mu) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i + \sum_{i=1}^m \alpha_i (1 - \xi_i - y_i(w^T x_i + b)) - \sum_{i=1}^m \mu_i \xi_i$$

$\alpha_i \geq 0, \mu_i \geq 0$  are Lagrangian multipliers.

let  $L$ 's partial derivatives with respect to  $w, b, \xi_i$  to zero, we can get :

$$\begin{aligned} w &= \sum_{i=1}^m \alpha_i y_i x_i \\ 0 &= \sum_{i=1}^m \alpha_i y_i \\ c &= \alpha_i + \mu_i \end{aligned}$$

# The Optimization Problem

- The dual of this new constrained optimization problem is:

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0,$$

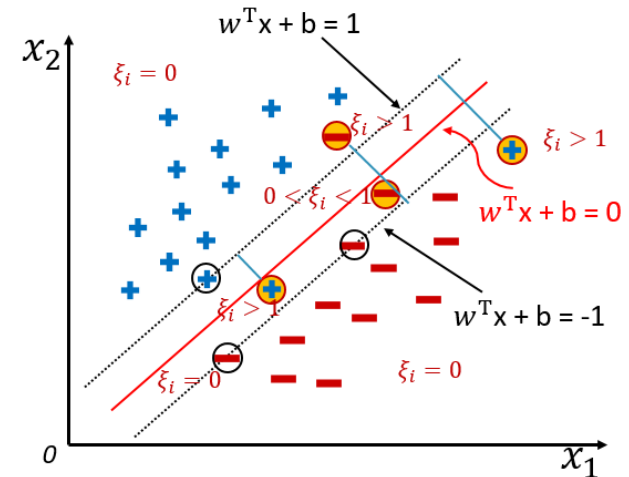
$$0 \leq \alpha_i \leq c, \quad i = 1, 2, \dots, m$$

- This is very similar to the optimization problem in the linear separable case, except that there is an upper bound  $c$  on  $\alpha_i$  now
- Again , SMO algorithm can be used to find  $\alpha_i$

# KKT conditions

- KKT conditions:

$$\left\{ \begin{array}{l} \alpha_i \geq 0, \mu_i \geq 0 \\ y_i(w^T x_i + b) - 1 + \xi_i \geq 0 \\ \alpha_i(y_i f(x_i) - 1 + \xi_i) = 0 \\ \xi_i \geq 0, \mu_i \xi_i = 0 \end{array} \right.$$



For every sample  $(x_i, y_i)$ , always have:  $\alpha_i = 0$  or  $y_i f(x_i) = 1 - \xi_i$

If  $\alpha_i = 0$ , the sample has no effect on  $f(x)$

If  $\alpha_i > 0$ , then  $y_i f(x_i) = 1 - \xi_i$ , the sample is a support vector

If  $\alpha_i < C$ , then  $\mu_i > 0$ , and then  $\xi_i = 0$ , the sample is on the margin

If  $\alpha_i = C$ , then  $\mu_i = 0$

If  $\xi_i \leq 1$ , the sample is in the margin

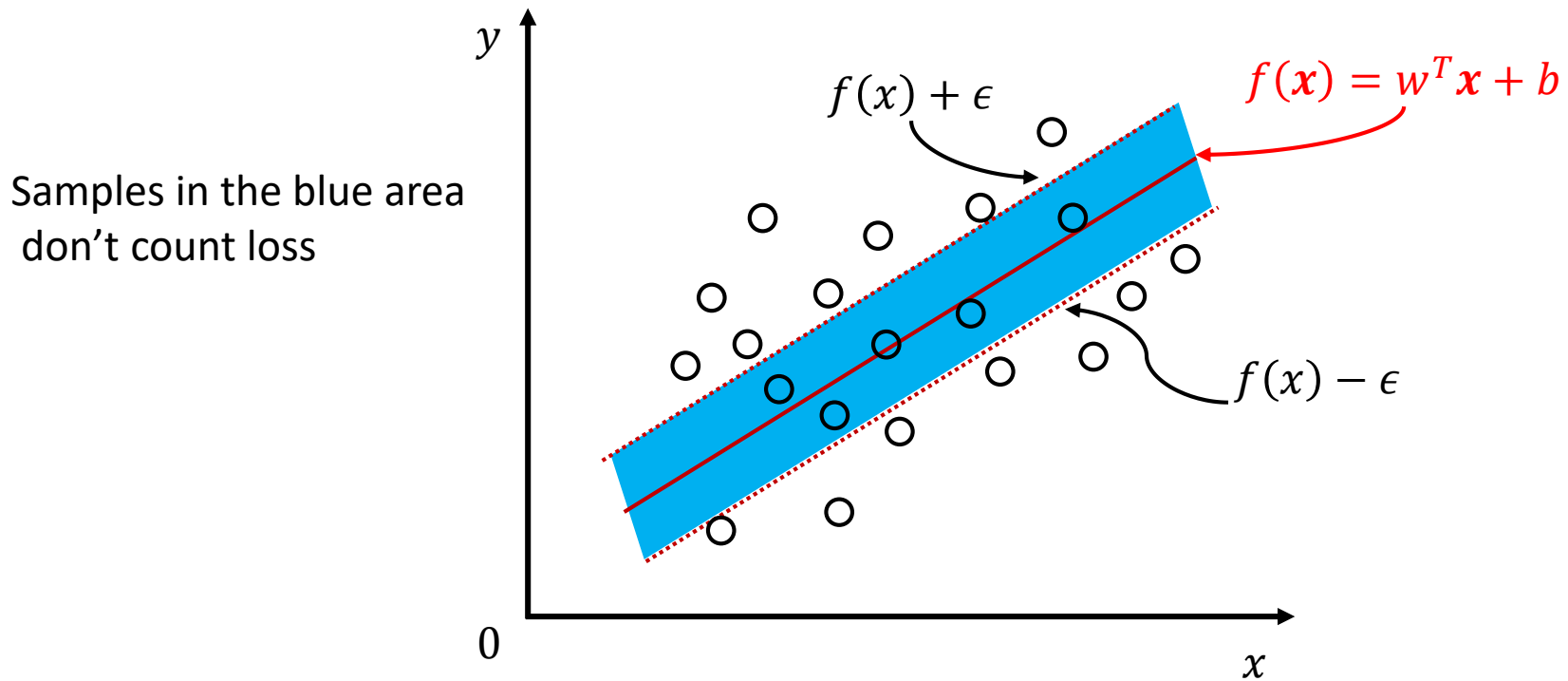
If  $\xi_i > 1$ , the sample is misclassified

# Outline

- Margin and support vector
- Dual problem
- Kernel function
- Soft margin
- **Support Vector Regression**
- Usage of LIBSVM with matlab

# Support Vector Regression

- Data set  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ,  $y_i \in \mathbb{R}$
- We want to find a model  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$  that  $f(\mathbf{x}) \sim y$



# Formulation

$$\min_{w,b} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m l_{\epsilon}(f(x_i) - y_i) \quad (*)$$

$$l_{\epsilon}(z) = \begin{cases} 0, & \text{if } |z| \leq \epsilon \\ |z| - \epsilon, & \text{otherwise} \end{cases}$$

Introduce 'slack variables'  $-\xi_i$  and  $\widehat{\xi}_i$ , rewrite (\*):

The slack degree of two sides can be different

$$\begin{aligned} \min_{w,b,\xi_i,\widehat{\xi}_i} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \widehat{\xi}_i) \\ \text{s.t.} \quad & f(x_i) - y_i \leq \epsilon + \xi_i \\ & y_i - f(x_i) \leq \epsilon + \widehat{\xi}_i \\ & \xi_i \geq 0, \widehat{\xi}_i \geq 0, i = 1, 2, \dots, m \end{aligned}$$



# Resolve the problem

- Again

lagrangian multiplier method

dual problem

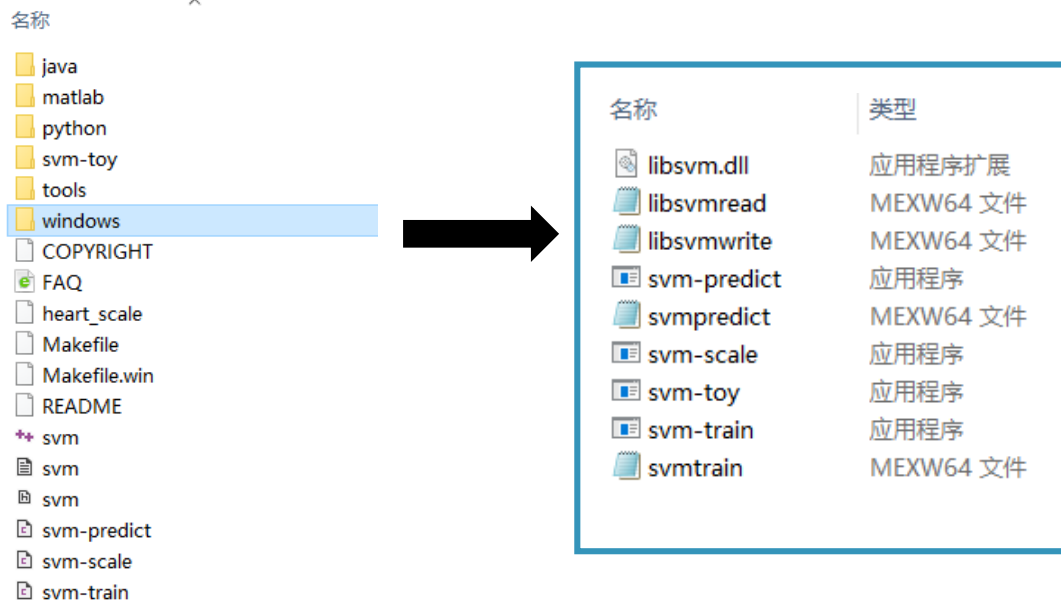
KKT conditions

# Outline

- Margin and support vector
- Dual problem
- Kernel function
- Soft margin
- Usage of LIBSVM with matlab

# LIBSVM

- Download LIBSVM : <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>  
(Generally, we need “windows” file folder only.)



- Add path in matlab (设置路径—添加并包含子文件夹—libsvm-3.22 )
- Now, we can use LIBSVM to train our model.

# A toy example



100 images: 60 for training, 40 for testing

120 images: 70 for training, 50 for testing

- Data preparation

data\_train.mat:  $130 \times n$ , the first 60 rows are cat samples;  $n$  is feature dimension

data\_test.mat:  $90 \times n$ , the first 40 rows are cat samples, the rest are dog samples

label\_train.mat:  $130 \times 1$ , first 60 are 1 which indicate cat category, the rest are -1

label\_test.mat:  $90 \times 1$ , first 40 are 1 which indicate cat category, the rest are -1

- Model training

model = **svmtrain**(label\_train, data\_train, '-s 0 -t 1 -c 10 -d 3')

参数具体含义见libsvm官网

- Model prediction/testing

[predict\_label, accuracy, dec\_values] = **svmpredict**(label\_test, data\_test)

# Take Home Message

- 'Maximum margin' idea
- Dual problem
- The KKT condition and support vectors
- Feature mapping—deal with linearly inseparable problem ; kernel function
- Soft margin—deal with data with noise

# Resources

- <http://www.csie.ntu.edu.tw/~htlin/mooc> 台湾大学林轩田
- <http://www.powercam.cc/home.php?user=chli&f=slide&v=list&fid=4097>(李政轩)——kernel method & SVM
- <http://cs229.stanford.edu/materials.html> ——Andrew Ng's Lecture Notes(斯坦福机器学习课程笔记).
- 一些博客:
  - [http://blog.pluskid.org/?page\\_id=683](http://blog.pluskid.org/?page_id=683)
  - [http://blog.csdn.net/v\\_july\\_v/article/details/7624837](http://blog.csdn.net/v_july_v/article/details/7624837)
- <https://www.csie.ntu.edu.tw/~cjlin/libsvm/> ——LIBSVM
- <https://www.csie.ntu.edu.tw/~cjlin/liblinear/> ——LIBLINEAR -- A Library for Large Linear Classification

# Homework

- 自己试着推导, 从  $\min_{w,b} \frac{1}{2} \|\mathbf{w}\|^2$   
s.t.  $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m$

到

$$\max_{\alpha} L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0, \\ \alpha_i \geq 0, \quad i = 1, 2, \dots, m$$



*Thanks !*