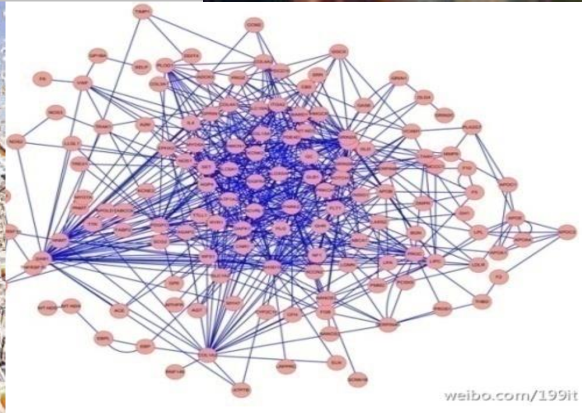




Chapter 8 Support Vector Machine

2025 Autumn

Lei Sun

01 Hard Margin

02 Soft Margin

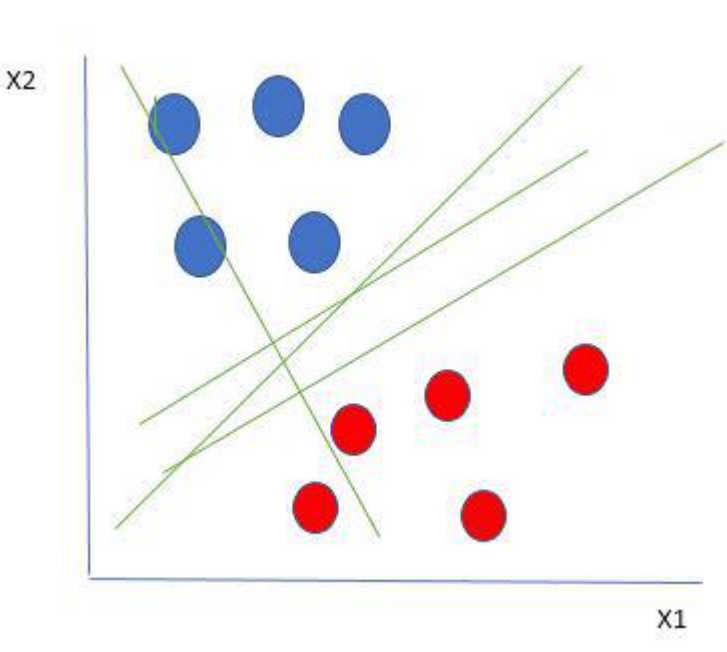
03 Kernal Approach

04 Multi-class SVM

05 Summary

Basic Idea

SVM algorithms are very effective as we try to find the maximum separating hyperplane(超平面) between the different classes .

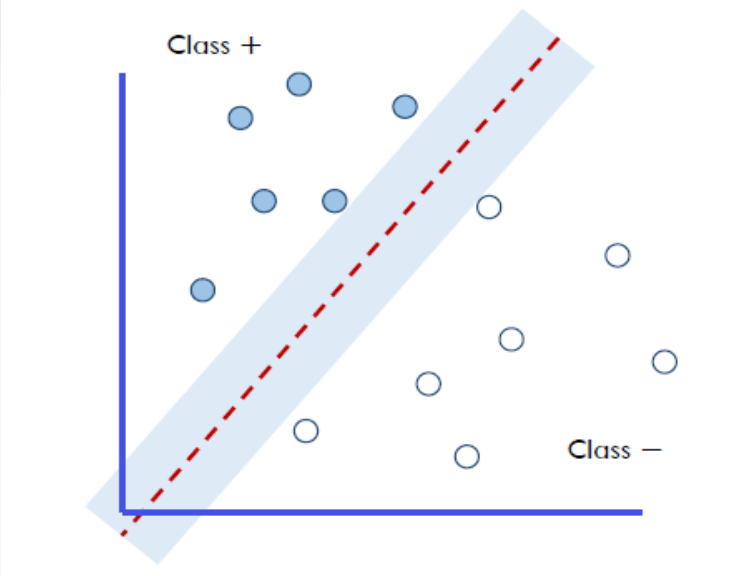


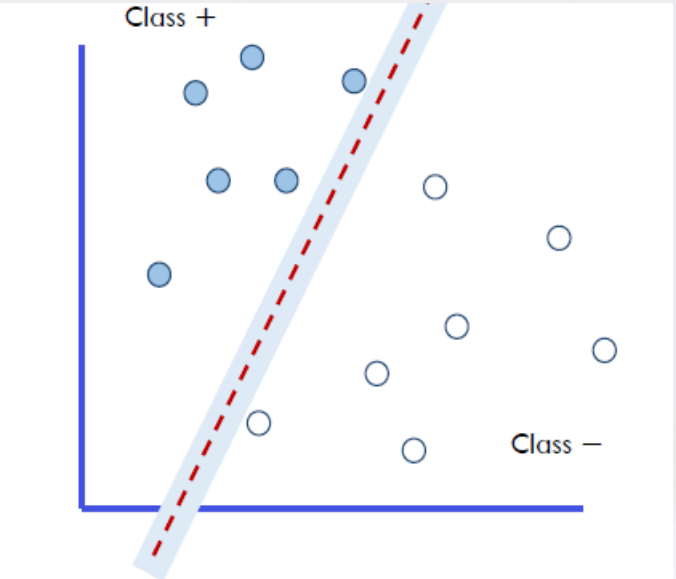
There are multiple lines (our hyperplane here is a line because of two input features x1, x2) that segregate our data points or do a classification between red and blue circles.

So how do we choose the best line or in general the best hyperplane that segregates our data points?

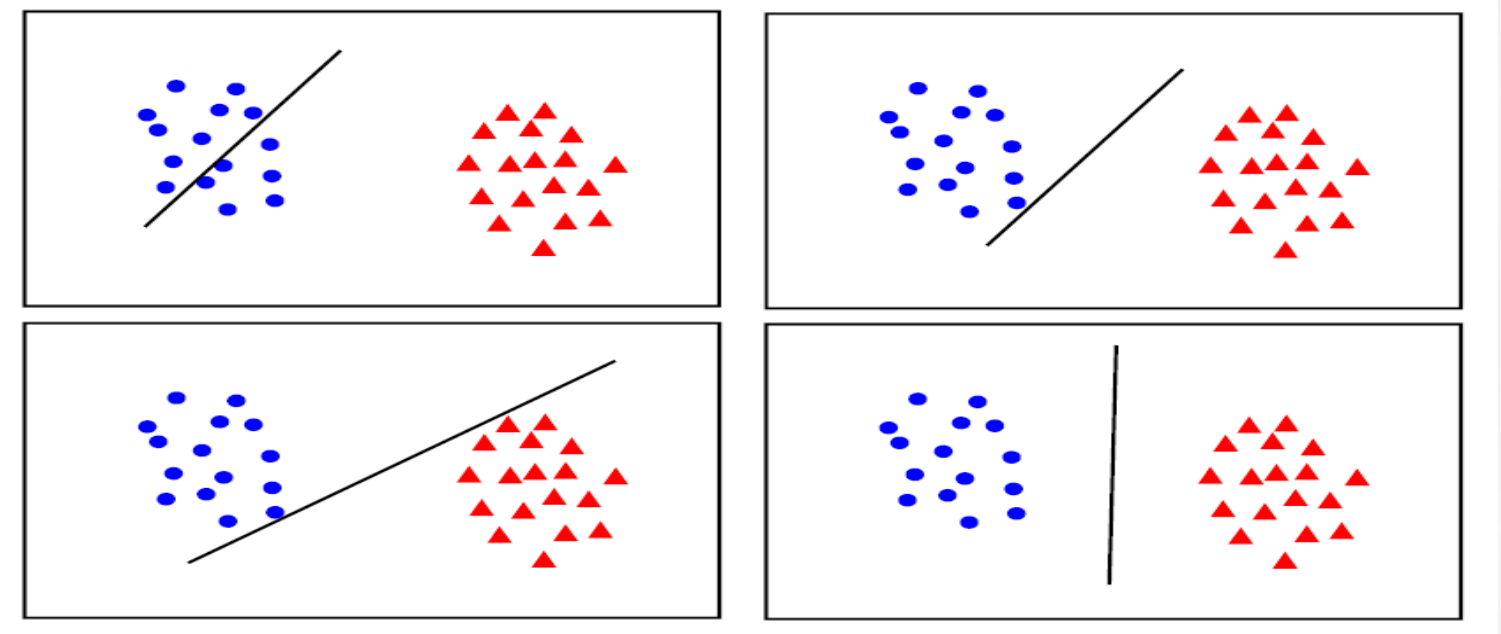
Basic Idea

- ✓ **Margin(间隔)** of a linear classifier is the width of a band around the classifier without any training examples
- ✓ Intuition: If we select a classifier with low margin then there is high chance of misclassification for a new data (Classifier is not robust)





Basic Idea



Both **direction** and **location** of hyperplane affects the margin

- ✓ error tolerance of these three lines is different
- ✓ the margin is as large as possible to overcome error and eliminate some overfitting

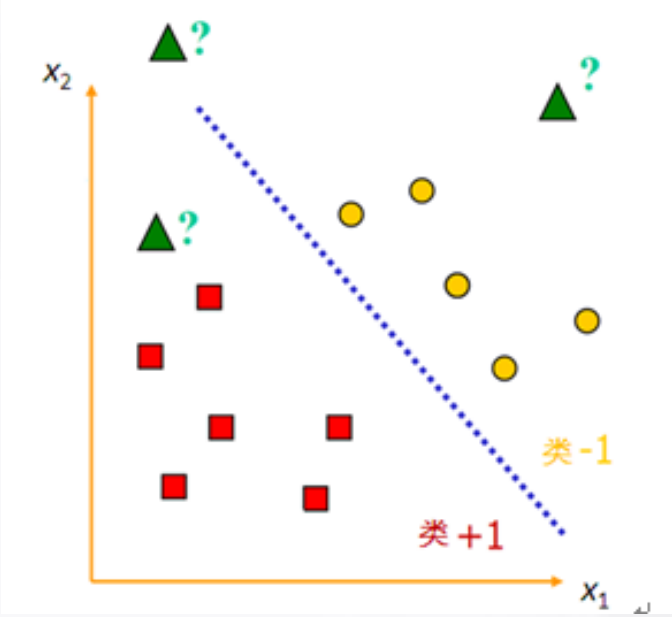
we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the maximum-margin hyperplane/hard margin.

Basic Idea

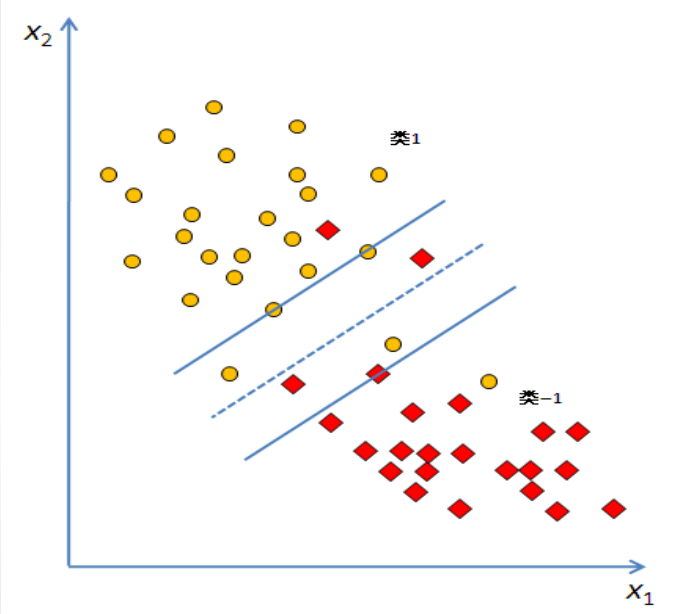
- ✓ **Hyperplane:** decision boundary that is used to separate the data points of different classes in a feature space. In the case of linear classifications, it will be a linear equation i.e. $wx+b = 0$.
- ✓ **Support Vectors:** the closest data points to the hyperplane, which makes a critical role in deciding the hyperplane and margin.
- ✓ **Margin:** the distance between the support vector and hyperplane. The main objective of the support vector machine algorithm is to maximize the margin. The wider margin indicates better classification performance.

Basic Idea

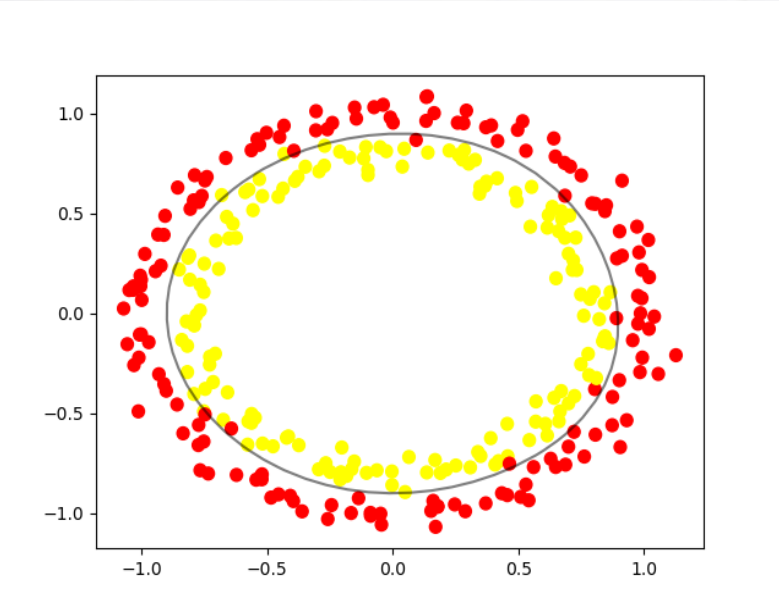
Types of SVM



Hard margin



Soft margin



Non-linear SVM using RBF kernel

01

Basic Idea

Hard Margin:

The maximum-margin hyperplane or the hard margin hyperplane is a hyperplane that properly separates the data points of different categories without any misclassifications.

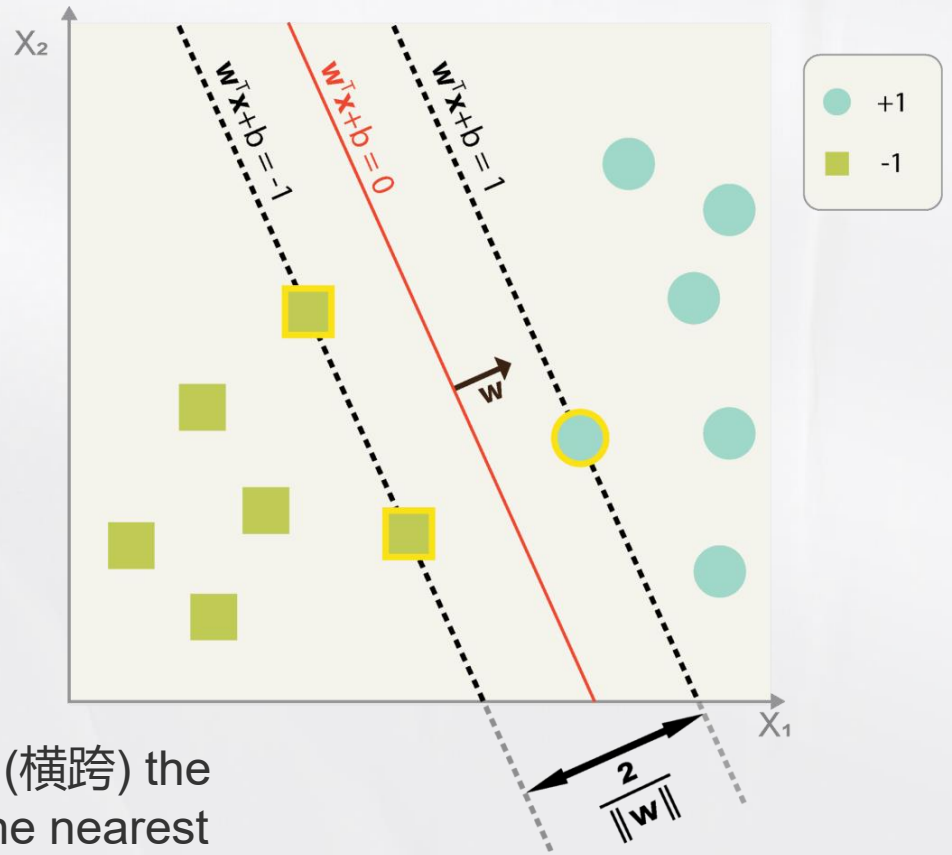
Soft Margin:

When the data is not perfectly separable or contains outliers, SVM permits a soft margin technique and certain misclassifications or violations

Kernel trick:

a method used in SVMs to enable them to classify non-linear data using a linear classifier.

Hard Margin--LSVM



✓ How to compute hyperplane

Any hyperplane can be written as a set of all points x that satisfy

$w^T x+b=0$

linearly separable: define two parallel hyperplanes that straddle (横跨) the nearest point of either class. Two parallel hyperplanes run through the nearest point of either class, such that for some w and b:

wTx+b=1 and wTx+b=−1

Goal: define and maximize the distance

$$\frac{b+1}{\|\omega\|}-\frac{b-1}{\|\omega\|}=\frac{2}{\|\omega\|}$$

$$\lambda=\frac{2}{\|W\|}=\text{margin}$$

$$\begin{cases} Ax+By+C1=0 \\ Ax+By+C2=0 \end{cases}$$

$$d=\frac{|C1-C2|}{\sqrt{A^2+B^2}}$$

Hard Margin--LSVM

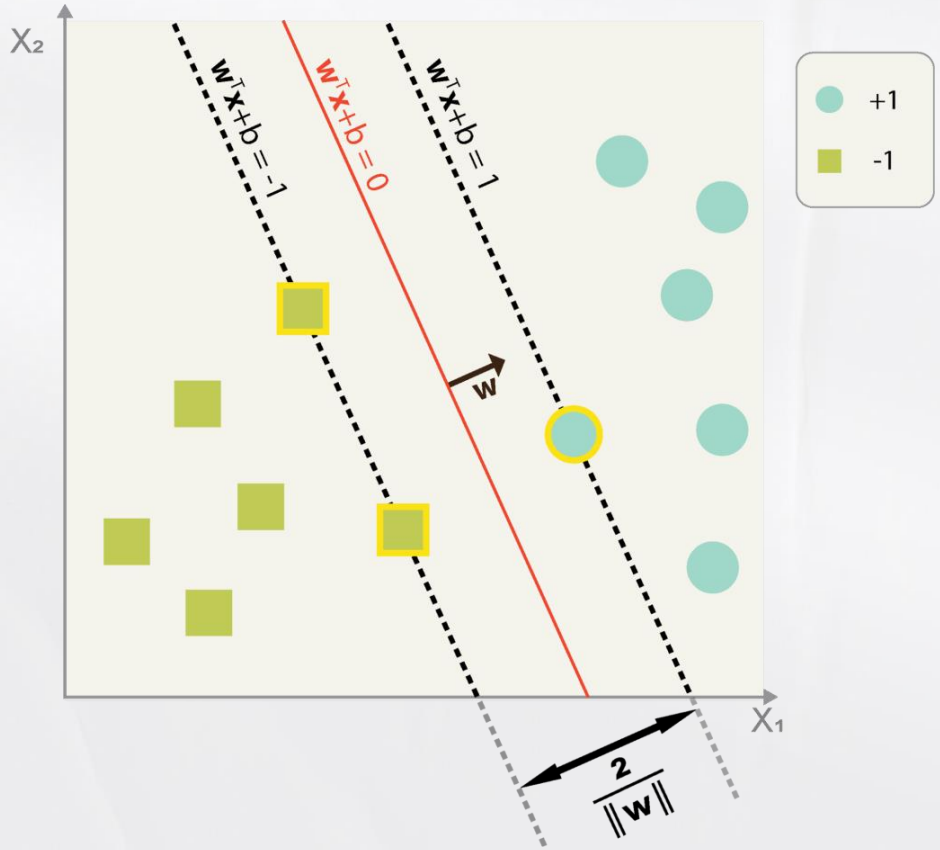
$$\lambda = \frac{2}{\|W\|}$$ → Geometric margin

maximize $\frac{2}{\|\omega\|}$, which is equivalent to minimizing ‖w‖.

For every data point, enforce the constraint:

$$W^T X + b \geq 1 \quad if \;\; y_i = 1$$
$$W^T X + b \leq -1 \quad if \;\; y_i = -1$$

→ $\forall i, \quad y_i(W^T X + b) \geq 1$



✓ Find w and b such that

$$\min \quad \frac{1}{2}\|w\|^2$$
$$\text{s.t.} \quad y_i(w.x_i + b) \geq 1 \quad \text{for all } i$$

Functional margin

Need to optimize a quadratic function（二次函数） subject to linear constraints.

Hard Margin--LSVM

$$\min(x_1^2 + x_2^2)$$
$$s.t. \quad x_1 + x_2 = 1$$

Duality(对偶)--Example

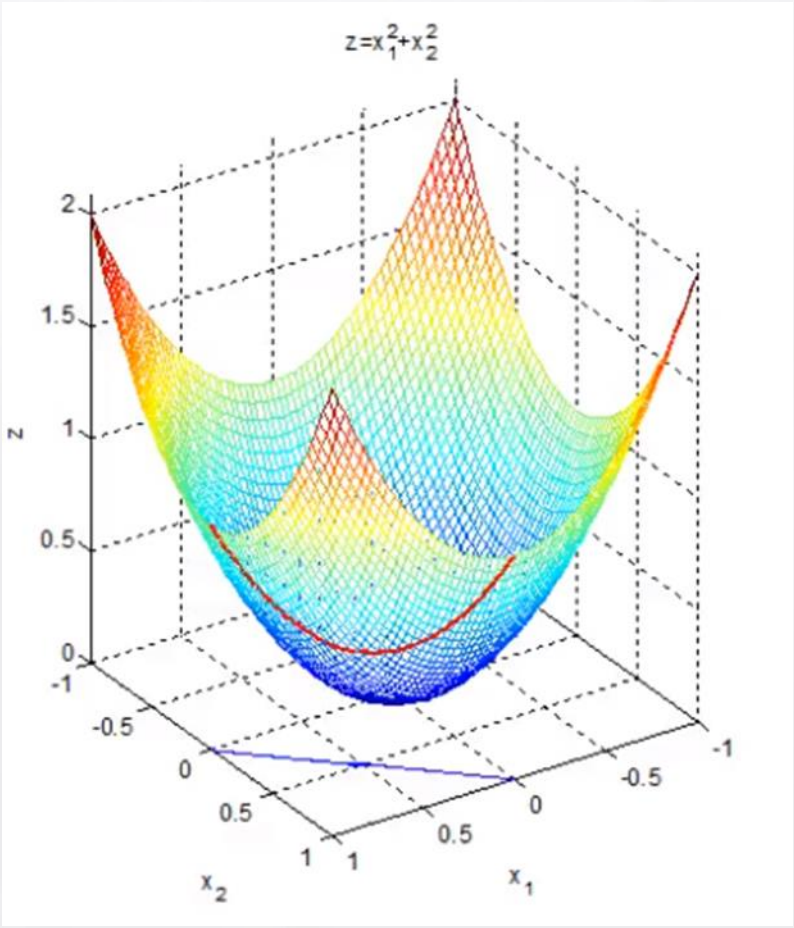
$$x_1 = t \qquad x_2 = 1 - t$$

$$\min\ [t^2 + (1 - t^2)]$$

$$\frac{d[t^2 + (1 - t^2)]}{dt} = 2t - 2(1 - t)$$

$$2t - 2 + 2t = 0 \qquad t = \frac{1}{2}$$

$$x_1 = x_2 = \frac{1}{2}$$



Hard Margin--LSVM

The lagrangian is

$$L(x_1, x_2, \alpha) = (x_1^2 + x_2^2) + \alpha(x_1 + x_2 - 1)$$

So

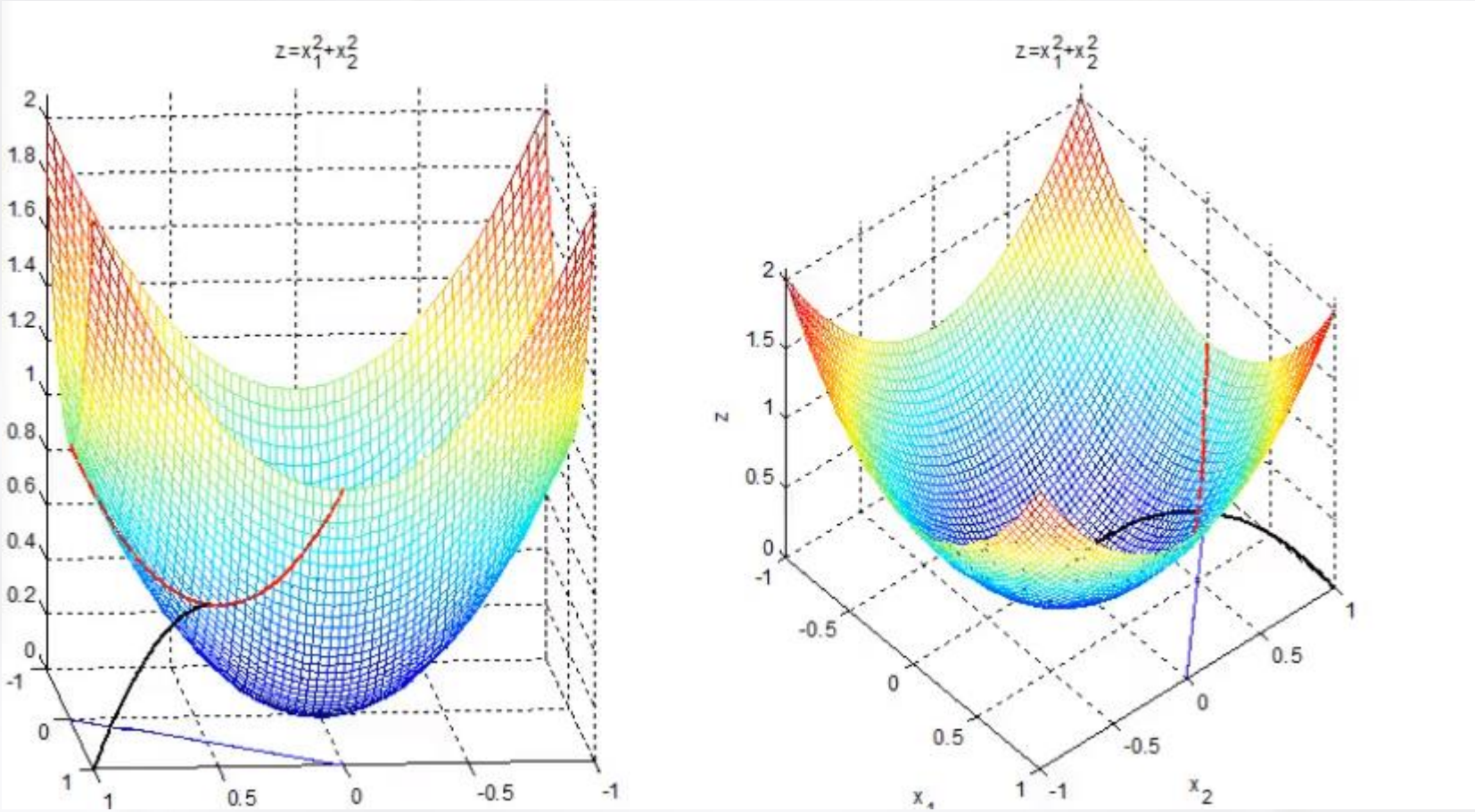
$$W(\alpha) = \min_{x_1, x_2} L(x_1, x_2, \alpha)$$

$$\frac{\partial L}{\partial x_1} = 2x_1 + \alpha = 0 \Rightarrow x_1 = -\frac{\alpha}{2}$$
$$\frac{\partial L}{\partial x_2} = 2x_2 + \alpha = 0 \Rightarrow x_2 = -\frac{\alpha}{2}$$

Hence

$$W(\alpha) = (-\frac{\alpha}{2})^2 + (-\frac{\alpha}{2})^2 + \alpha\left(-\frac{\alpha}{2} - \frac{\alpha}{2} - 1\right) = -\frac{\alpha^2}{2} - \alpha$$

Hard Margin--LSVM



The maximum value of the black curve W (α) is the minimum value of the red curve

Hard Margin--LSVM

To obtain optimal α^*

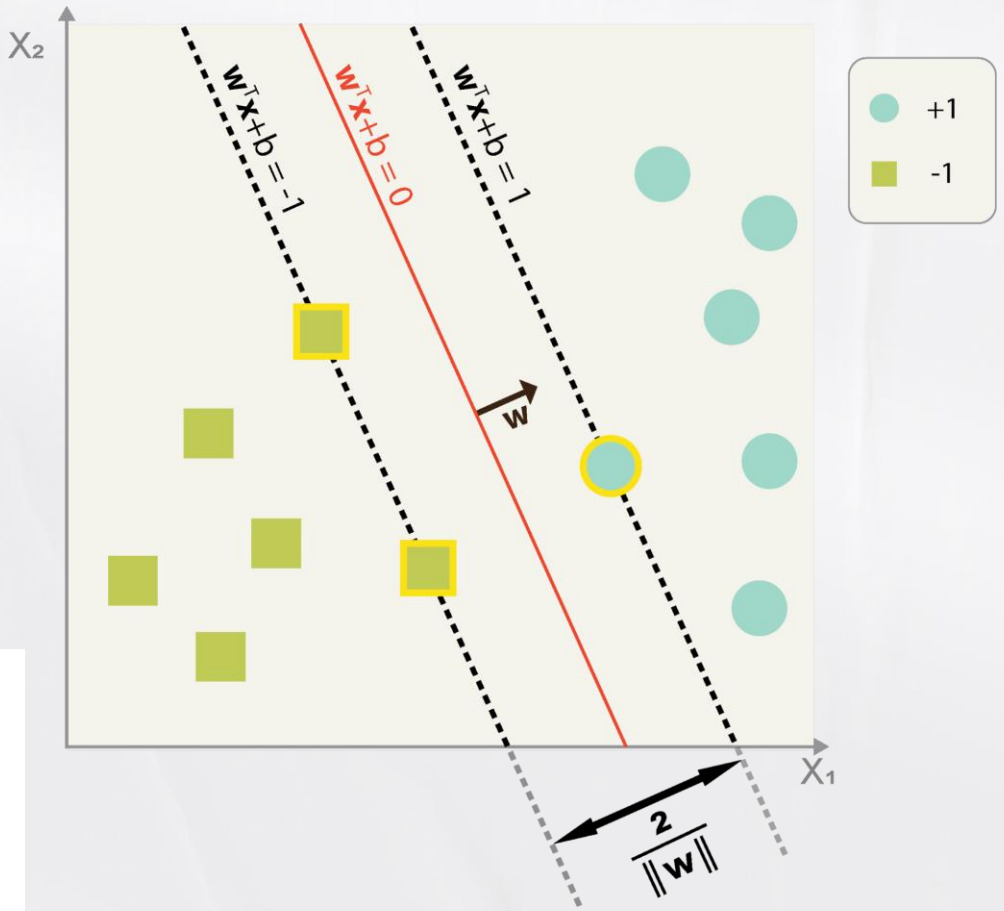
$$\max_{\alpha} W(\alpha) = -\frac{1}{2}\alpha^2 - \alpha$$

That is

$$W'(\alpha) = -\alpha - 1 = 0 \Rightarrow \alpha^* = -1$$

Therefore,

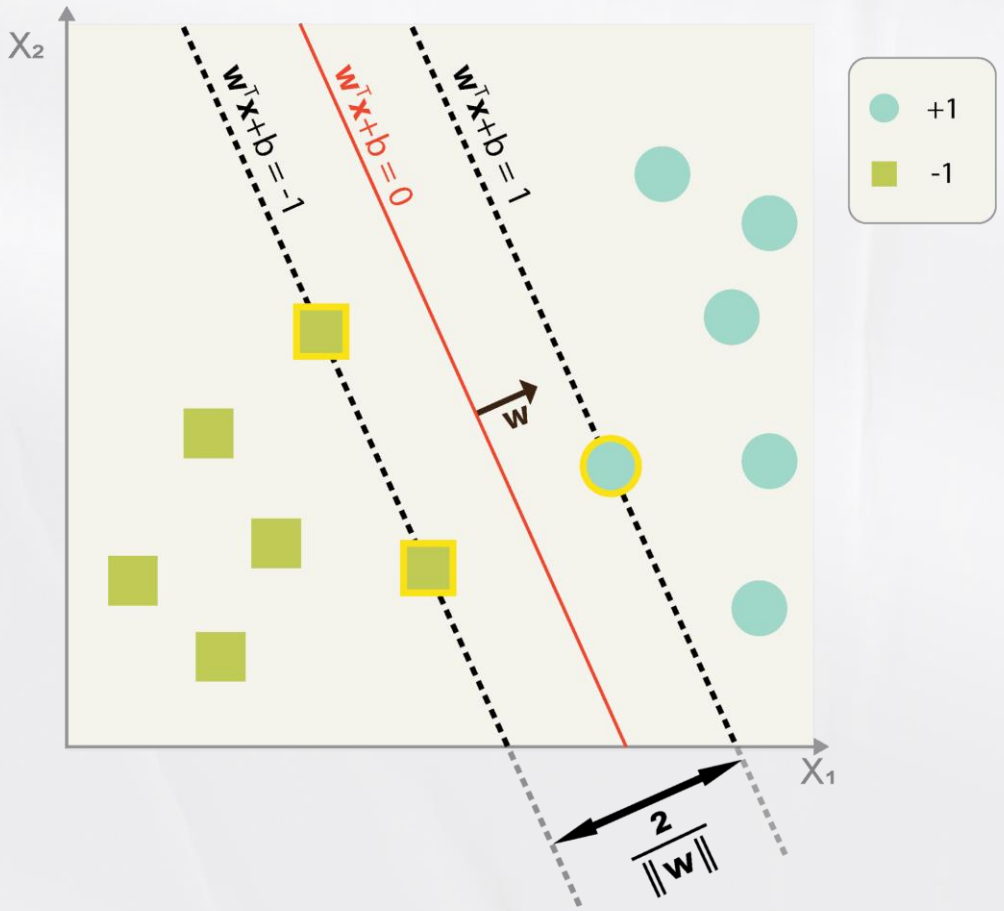
$$x_1^* = x_2^* = -\frac{-1}{2} = \frac{1}{2}$$



Hard Margin--LSVM

Primal problem:

$$\min x_1^2 + x_2^2$$
$$\text{s.t. } x_1 + x_2 = 1$$



Dual problem:

$$\max_{\alpha} W(\alpha)$$

Where:

$$W(\alpha) = \min_{x_1, x_2} L(x_1, x_2, \alpha) = \min_{x_1, x_2} (x_1^2 + x_2^2) + \alpha(x_1 + x_2 - 1)$$

Hard Margin--LSVM

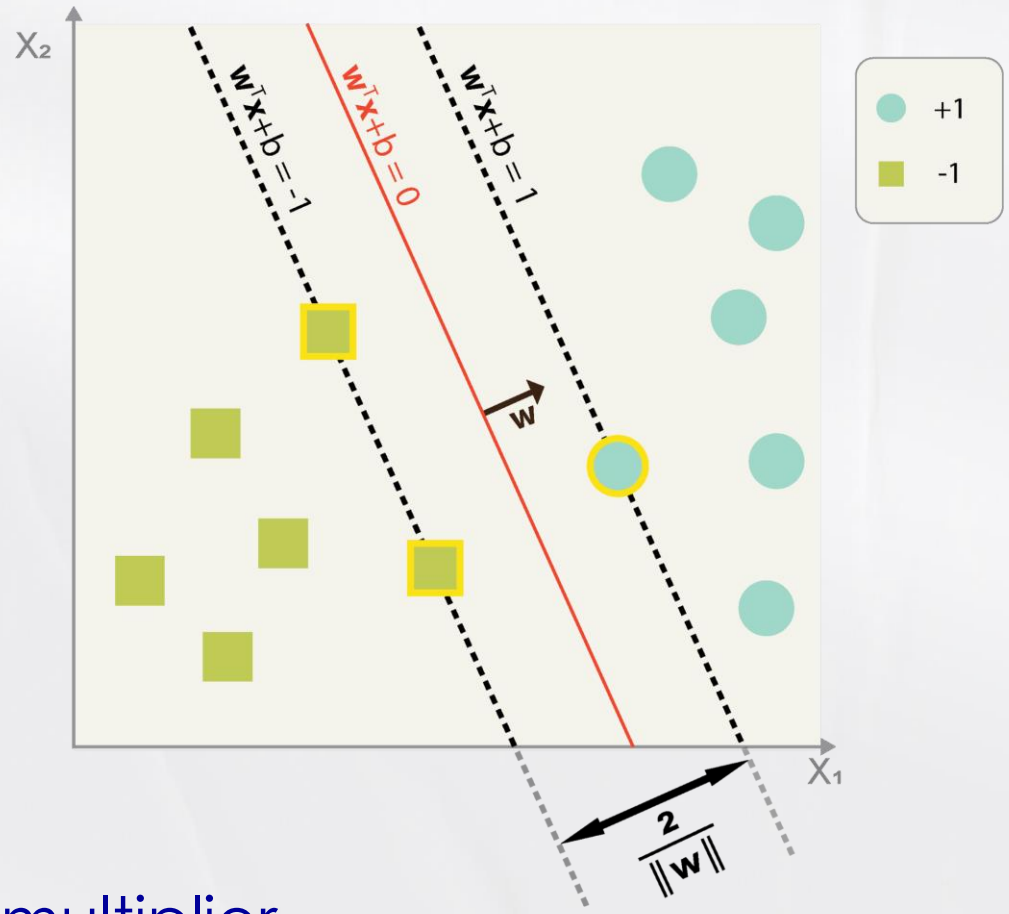
Converting constrained problem to an unconstrained problem with help of certain unspecified parameters known as lagrangian multipliers

$$\min \mathcal{F}(x)$$

$$\text{s.t. } h(x) \leq 0$$

Lagrangian multiplier

$$\min \mathcal{L}(x, \alpha) = \mathcal{F}(x) + \alpha h(x)$$

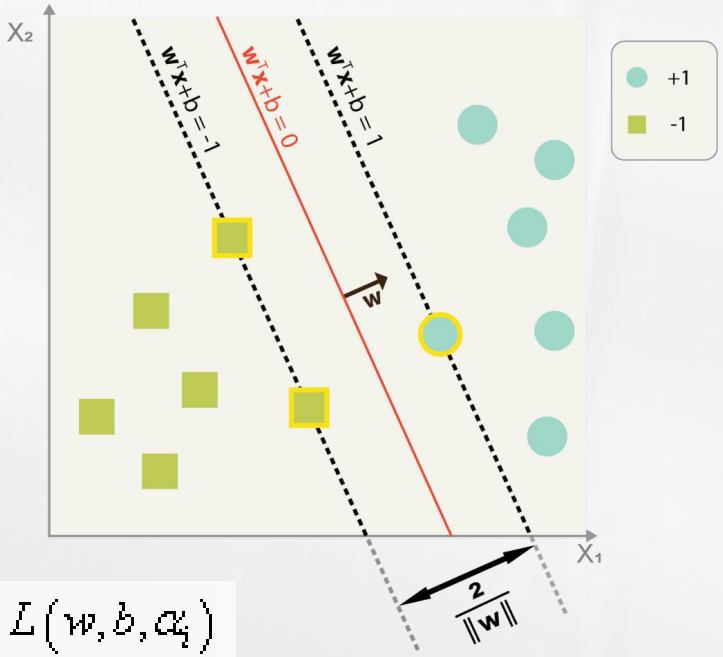


We use lagrangian method to solve the SVM optimization problem

Hard Margin--LSVM

$$\min \quad \frac{1}{2}\|w\|^2$$
$$\text{s.t.} \quad y_i(w.x_i + b) \geq 1 \quad \text{for all } i$$

$$\min_{w,b} \max_{\alpha_i \geq 0} L(w, b, \alpha_i)$$



$$\min_{w,b} \max_{\alpha_i \geq 0} L(w, b, \alpha_i) = \max_{\alpha_i \geq 0} \min_{w,b} L(w, b, \alpha_i)$$

Integrating the constraints to the Lagrange form we get

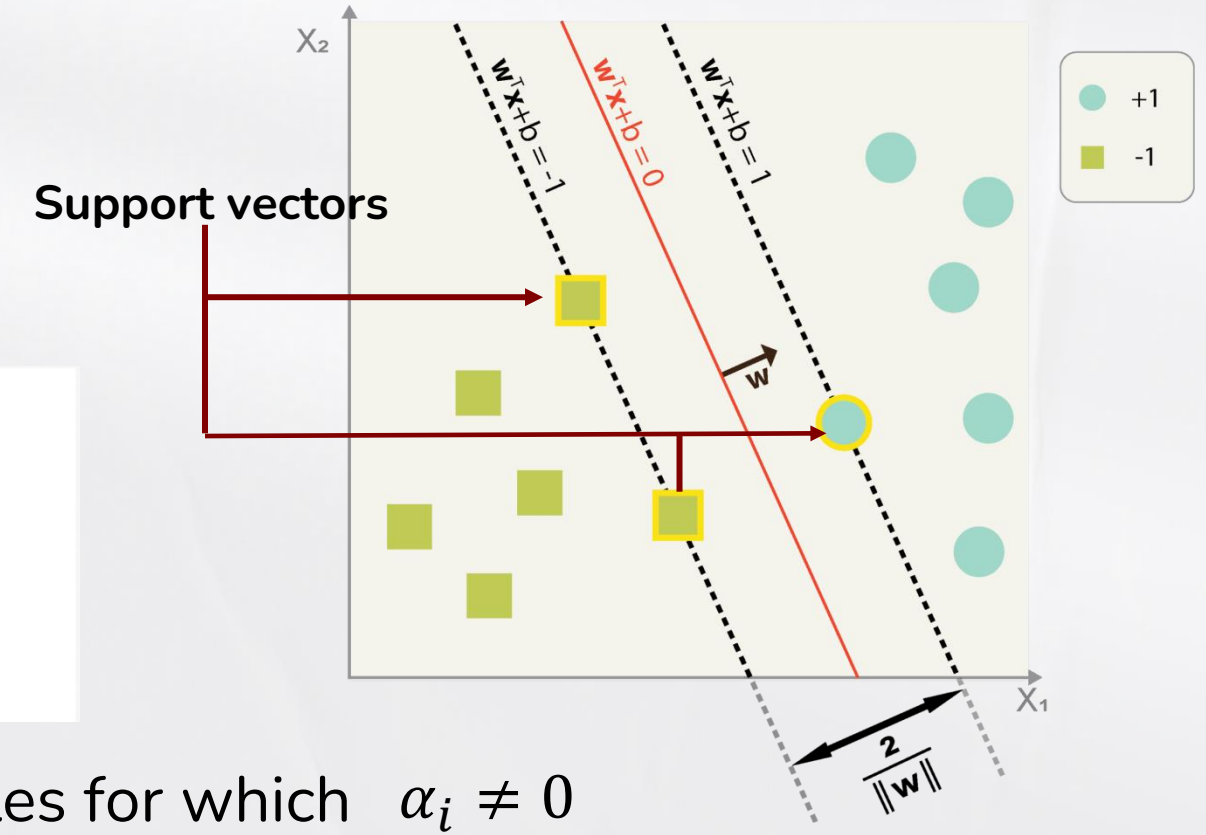
$$\max_{\alpha \geq 0} \min_{w,b} L(w, b, \alpha) = \frac{1}{2} w.w - \sum_i \alpha_i y_i (w.x_i + b) + \sum_i \alpha_i$$

Lagrangian multiplier

Hard Margin--LSVM

$$\frac{\partial L(w,b,\alpha)}{\partial w} = 0, \qquad \frac{\partial L(w,b,\alpha)}{\partial b} = 0$$

$$(1)\ w = \sum_i \alpha_i y_i x_i \qquad (2)\ \sum_i \alpha_i y_i = 0$$



Therefore, *w* only depends on those samples for which $\alpha_i \neq 0$

Additionally, The KKT conditions require that the solution satisfies:

$$\alpha_i(y_i(W \cdot x_i + b) - 1) = 0$$

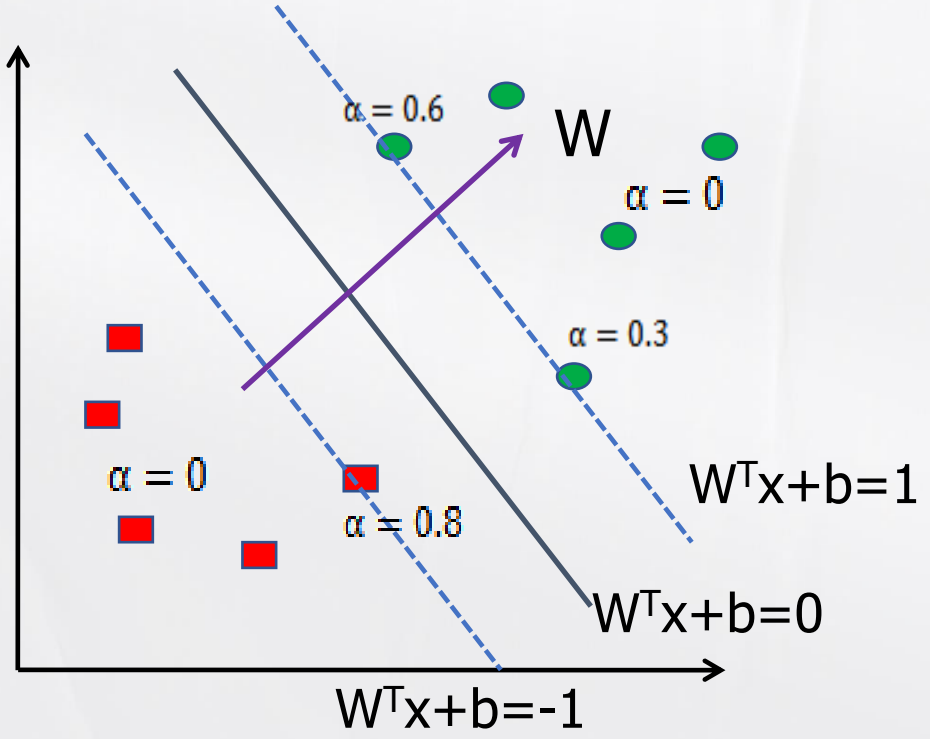
In order to compute b, the constraint for sample I must be tight, i.e. $\alpha_i>0$, so that

$$y_i(W \cdot x_i + b) - 1 = 0$$ support vectors

Hence, b depends only on those samples for which $\alpha_i>0$.

Therefore, we can conclude that the solution depends on all samples for which $\alpha i>0$.

Hard Margin--LSVM



- ✓ Closest points x_i to the plane that achieve the margin are called support vectors

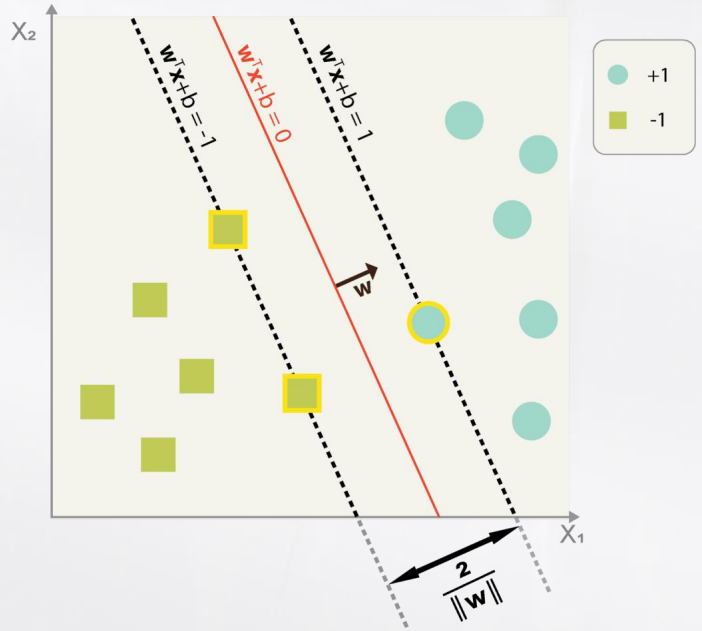
$$y_i(W \cdot x_i + b) - 1 = 0$$

- ✓ The lagranian multiplier corresponding to the support vectors is nonzero: α_i>0

- ✓ only the SVs change, the hyperplane correspondingly move.

  Solution does not change if we delete other instances

Hard Margin--LSVM

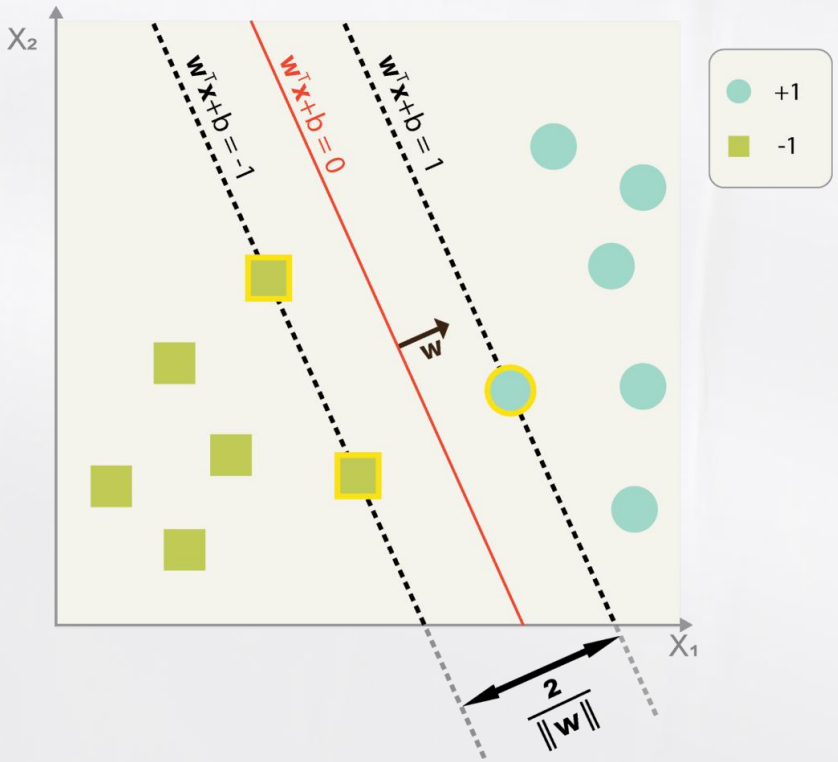


What are support vectors?

Support vectors:
special because they are the training points that define the maximum margin of the hyperplane to the data set and they therefore determine the shape of the hyperplane.

If you were to move one of them and retrain the SVM, the resulting hyperplane would change. The opposite is the case for non-support vectors (provided you don't move them too much, or they would turn into support vectors themselves).

Hard Margin--LSVM



Using conditions (1), (2) we get

$$L(w,b,\alpha) = \frac{1}{2} w.w - \sum_i \alpha_i y_i (w.x_i + b) + \sum_i \alpha_i =$$

$$\sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (x_i.x_j)$$

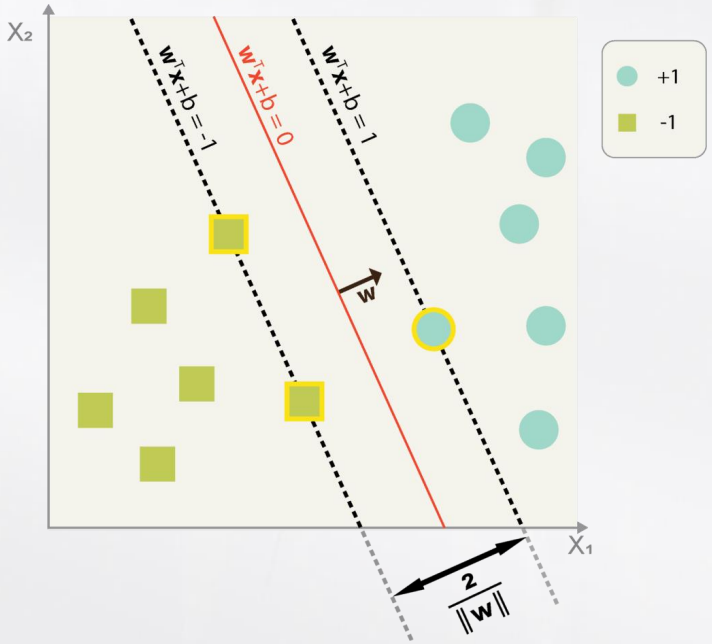
Therefore α can be found by solving

Known as dual problem ←

$$\max \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (x_i.x_j)$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0 \quad \text{for all } i$$

Hard Margin--LSVM



Input: $T=\{(x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N)\}$, $y=\{-1,1\}$

(1) Construct optimized problem

$$\min_{\alpha}\varphi(x)=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)-\sum_{i}^{N}\alpha_i$$

$$\sum_{i}^{N}\alpha_i y_i=0,\quad \alpha_i\geq 0,\quad i=1,2,\ldots,N$$

(2) compute all αi

(3) Compute w and b:

$$\begin{cases} W=\sum_{i=1}^{l}a_i y_i x_i \\ b=y_i-W^T x_i \end{cases}$$

(4) Output hyperlane:

$$w\cdot x+b=0$$

(5) Classification function:

$$f(x)=\mathrm{sign}(w\cdot x+b)$$

Hard Margin--LSVM

Example

	Y label	X1	X2
x1	圆形 1	0	0
x2	圆形 1	1	0
x3	三角形 -1	0	1
x4	三角形 -1	1	1

Hard Margin--LSVM

$$\min_{\alpha} \varphi(x) = Min(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i}^{N}\alpha_i) \qquad \sum_{i}^{N}\alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1,2,\dots,N$$

- Feed four instance points into equation(将四个实例点带入：)

$$\begin{aligned}\mathcal{L}(w,b,\alpha) = \frac{1}{2}(&\alpha_1^2 \times 1 \times 0 + \alpha_1\alpha_2 \times 1 \times 0 + \alpha_1\alpha_3(-1) \times 0 + \alpha_1\alpha_4(-1) \times 0 + \alpha_2^2 \times 1 \times 1 \\ &+ \alpha_2\alpha_1 \times 1 \times 0 + \alpha_2\alpha_3(-1) \times 0 + \alpha_2\alpha_4(-1) \times 1 + \alpha_3^2 \times 1 \times 1 + \alpha_3\alpha_1 \times (-1) \\ &\times 0 + \alpha_3\alpha_2 \times (-1) \times 0 + \alpha_3\alpha_4 \times 1 \times 1 + \alpha_4^2 \times 1 \times 2 + \alpha_4\alpha_1(-1) \times 0 \\ &+ \alpha_4\alpha_2(-1) \times 1 + \alpha_4\alpha_2(-1) \times 1 + \alpha_4\alpha_3 1 \times 1) - \alpha_1 - \alpha_2 - \alpha_3 - \alpha_4 \\ &= \frac{1}{2}(\alpha_2^2 + \alpha_3^2 + 2\alpha_4^2 - 2\alpha_2\alpha_4 + 2\alpha_3\alpha_4) - \alpha_1 - \alpha_2 - \alpha_3 - \alpha_4\end{aligned}$$

$$\begin{cases}\alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0 \\ \alpha_i \geq 0\end{cases} \Rightarrow \begin{cases}\alpha_1 = \alpha_3 + \alpha_4 - \alpha_2 \\ \alpha_i \geq 0\end{cases}$$

Hard Margin--LSVM

- 将上式带入 $\min_{\alpha} \varphi(x) = Min(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i}^{N}\alpha_i)$

$$\min_{\alpha} \mathcal{L}(w,b,\alpha) = \mathrm{Min}(\frac{1}{2}(\alpha_2^2 + \alpha_3^2 + 2\alpha_4^2 - 2\alpha_2\alpha_4 + 2\alpha_3\alpha_4) - 2(\alpha_3 - \alpha_4))$$

- Find partial derivative: (求偏导数)

$$\begin{cases} \frac{\partial \mathcal{L}}{\alpha_2} = 0 \\ \frac{\partial \mathcal{L}}{\alpha_3} = 0 \\ \frac{\partial \mathcal{L}}{\alpha_4} = 0 \end{cases} \Rightarrow \begin{cases} \alpha_2 - \alpha_4 = 0 \\ \alpha_3 + \alpha_4 = 2 \\ 2\alpha_4 - \alpha_2 + \alpha_3 = 2 \end{cases} \qquad \begin{cases} \alpha_1 = 2 - \alpha_4 \\ \alpha_2 = \alpha_4 \\ \alpha_3 = 2 - \alpha_4 \\ \alpha_4 = \alpha_4 \end{cases}$$

$$w = \sum_{i=1}^{N}\alpha_i y_i x_i = (2-\alpha_4)(0,0) + \alpha_4(1,0) - (2-\alpha_4)(0,1) - \alpha_4(1,1)$$

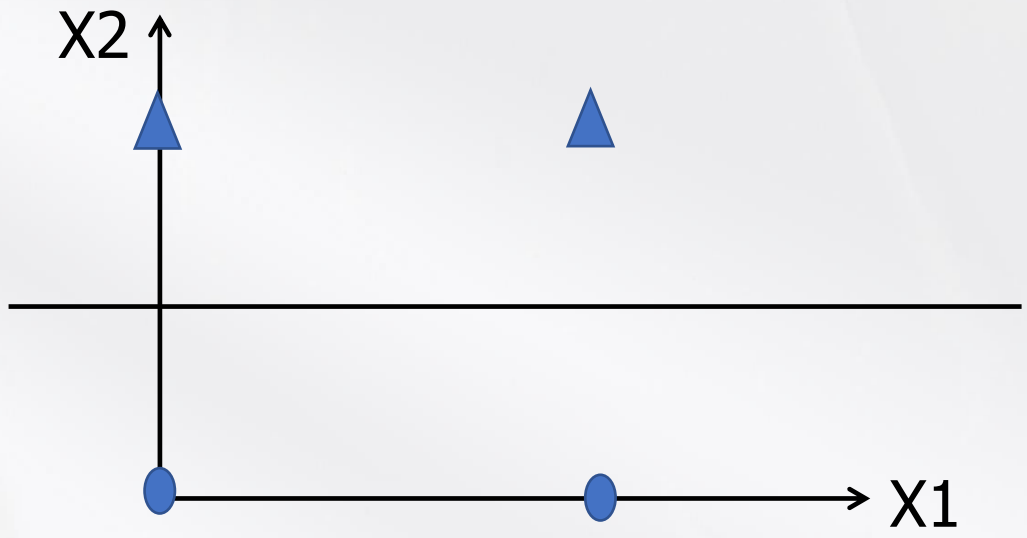
$$= (0 + \alpha_4 - 0 - \alpha_4,\ 0 + 0 - 2 + \alpha_4 - \alpha_4) = (0,\ -2)$$

$$w \cdot x = (0,-2)(x1,x2)^T = -2x2$$

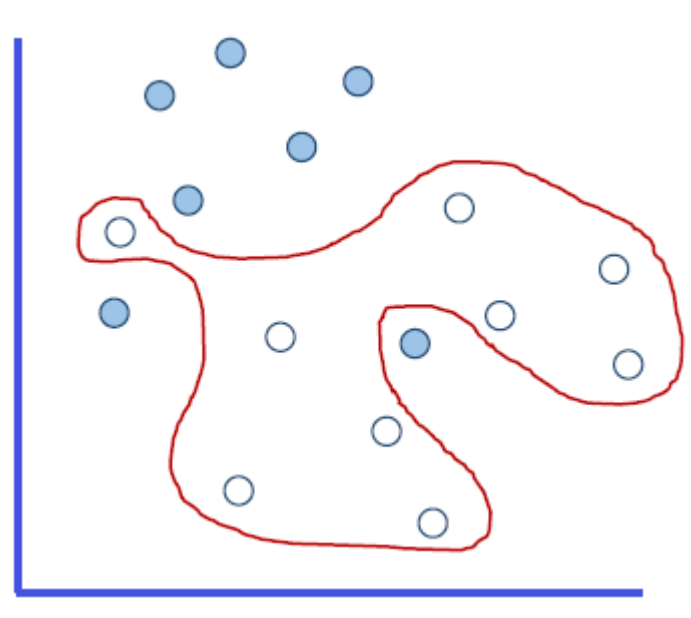
Hard Margin--LSVM

$$b = y_j - w \cdot x_j = y_j - x_j \cdot \sum_{i=1}^{N} \alpha_i y_i x_i$$

- 选取x1=(0,0),y=1带入：$b = 1 - (0,0)(0,-2)^T = 1 \quad \Rightarrow \quad -2x2 + 1 = 0$



Soft Margin



What if our data isn't linearly separable?

Data are unlikely to be linearly separable due to:

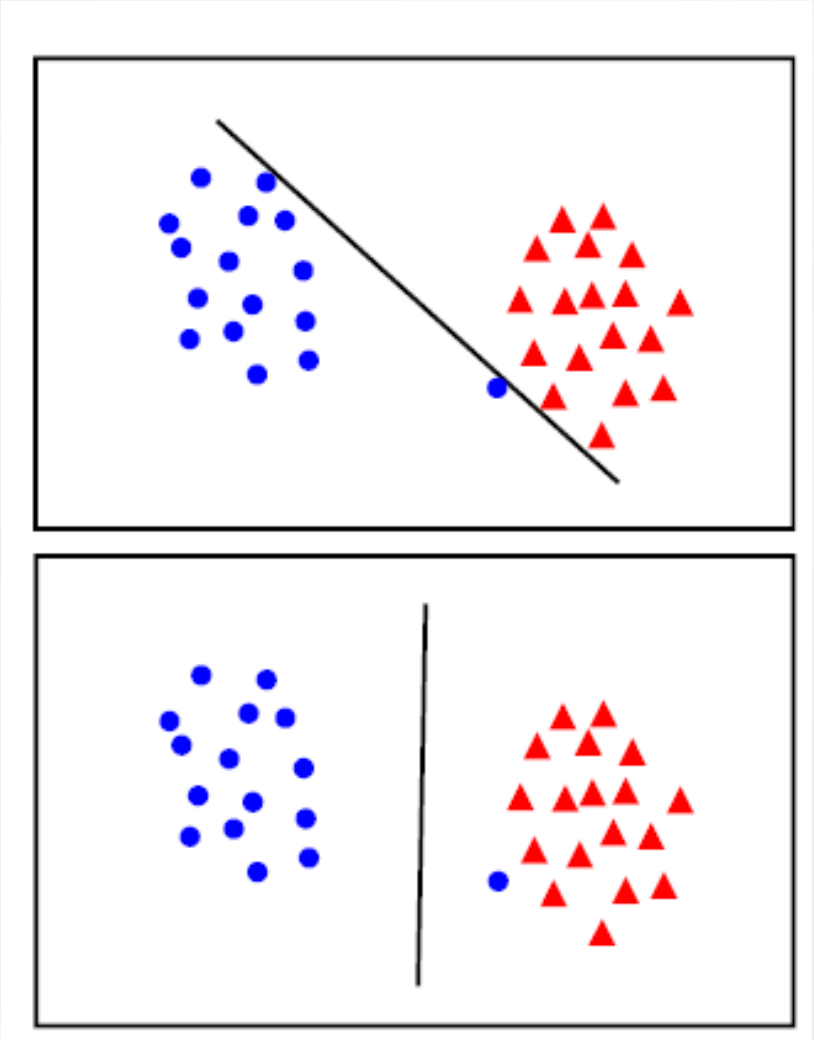
- underlying patterns(潜在模式) themselves are not linearly separable.
- noise has polluted the data, creating a non-linearly separable permutation(排列) of the underlying pattern.

Classic linear SVM cannot adequately deal with the former problem, but for the latter, we can use the soft margin extension to improve the classic SVM method.

Soft Margin—what is the best?

The points can be linearly separated but there is a very narrow margin.

But possibly the large margin solution is better, even though one constraint is violated.



In general there is a trade off between the margin and the number of mistakes on the training data

Soft Margin

Solution： to loosen some of the constraints by introducing slack variables

- Slack variables are introduced to allow certain constraints to be violated.
- The number of points within the margin is to be as small as possible.

Introduce a slack variable ξi , one for each datapoint i.

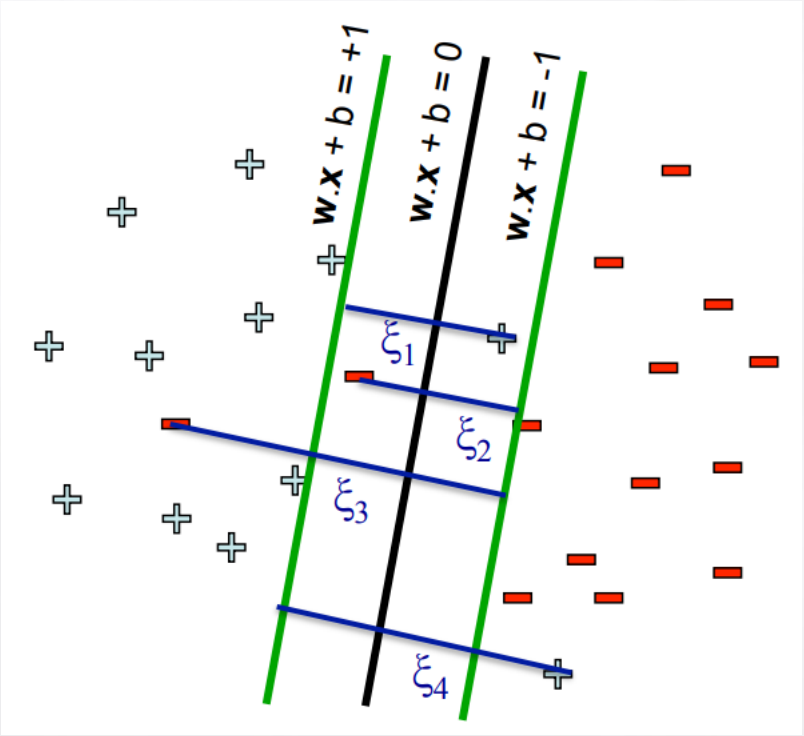
Soft Margin--LSVM

Allow for slack(松弛)

The slack variable ξi allows the input x_i to be closer to the hyperplane (or even be on the wrong side), but there is a penalty in the objective function for such "slack".

$$\min\left(\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i\right), \qquad C > 0$$

$$s.t. \quad \xi_i \geq 0, \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n$$



The minimum functional distance between the samples on one side of the hyperplane and the hyperplane is 1. When subtracting ξ, it indicates that some samples are allowed to cross the support vector. Error is allowed

Soft Margin

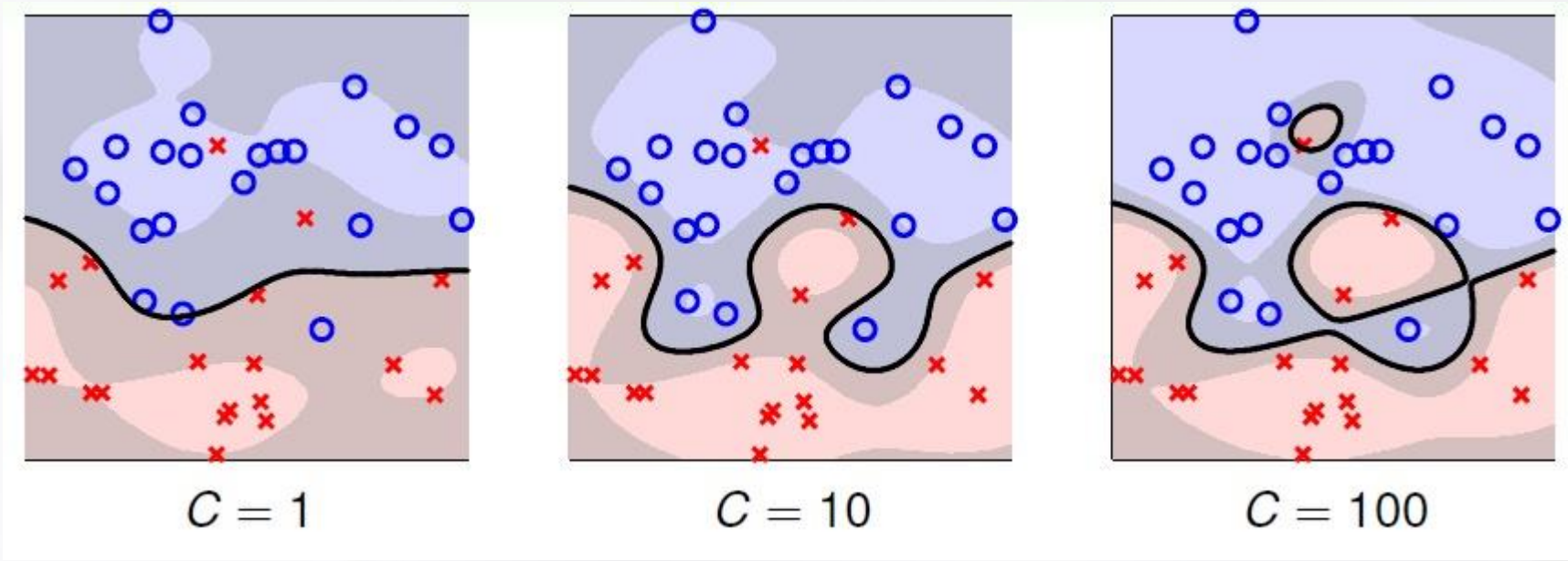
$$\min\left(\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i\right), \qquad C > 0$$

$$subject\ to \qquad \xi_i \geq 0, \qquad y_i(w^T x_i + b) \geq 1 - \xi_i, \qquad i = 1,2,\dots,n$$

	Small C	Big C	Infinity
desire	becomes very loose and may "sacrifice" some points to obtain a simpler solution	the SVM becomes very strict, keeps most slack variables small, tries to get all points to be on the right side of the hyperplane.	Hard margin
margin	large	small	
model	simple	complex	
danger	underfitting	overfitting	
boundary	flat	Sinuous(蜿蜒曲折)	

Soft Margin

Guassian kernel/soft-margin with different value of C：



如果模型在训练数据上表现良好，但在测试数据上表现不佳，可能是过拟合现象。此时，可以尝试_____C值来降低模型复杂度。

A.减小　　B. 增加

03

Soft Margin

$$\min\left(\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i\right), \qquad C > 0$$

$$subject\ to \qquad \xi_i \geq 0, \qquad y_i(w^T x_i + b) \geq 1 - \xi_i, \qquad i = 1,2,\dots,n$$

$$L(b, w, \alpha, \beta, \xi) = \frac{1}{2} W^T W + C \cdot \sum_{i=1}^{N} \xi_i + \sum_{i=1}^{N} \alpha_i \left(1 - \xi_i - y_i(W^T x_i + b)\right) - \sum_{i=1}^{N} \beta_i \xi_i$$

$$\mathcal{L}(w, b, \xi, \alpha, \beta) = \frac{1}{2}w^T w + \sum_{i=1}^{n} \alpha_i\left(1 - y_i(w^T x_i + b)\right) + \sum_{i=1}^{n} (C - \alpha_i - \beta_i)\xi_i$$

$$\frac{\partial L(w, b, \xi, \alpha, \beta)}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \qquad \frac{\partial L(w, b, \xi, \alpha, \beta)}{\partial b} = 0 \rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\frac{\partial \mathcal{L}(w, b, \xi, \alpha, \beta)}{\partial \xi} = 0 \Longrightarrow C - \alpha_i - \beta_i = 0 \Longrightarrow \begin{cases} \alpha_i = C - \beta_i \\ \beta_i = C - \alpha_i \end{cases} \qquad 0 \leq \alpha_i, \beta_i \leq C$$

Soft Margin

$$\text{Max}\ \left(\sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\, x_i^T x_j\right)$$

$$\text{or}\ \ \text{Min}\ \left(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\, x_i^T x_j - \sum_{i=1}^{N}\alpha_i\right)$$

$$\text{s.t.}\sum_{i=1}^{N} y_i\,\alpha_i = 0 \qquad 0 \le \alpha_i \le C \qquad W = \sum_{i=1}^{N}\alpha_i y_i x_i \qquad \beta_i = C - \alpha_i$$

- ✓ prediction

$$h(x) = \text{sign}(b + W^T x) = \text{sign}\left(b + \sum_{i=1}^{N}\alpha_i y_i\,(x_i^T x_j)\right)$$

- ✓ Neither the slack variables, nor their lagrange multipliers appear in the dual.
- ✓ The only change is the additional constraint on αi

$$0 \le \alpha_i \le C$$

Soft Margin

example

	Y	X1	X2
x1	圆形 1	0	0
x2	圆形 1	1	0
x3	三角形 -1	0	1
x4	三角形 -1	1	1
x5	三角形 -1	0.5	0

Soft Margin

$$\min_{\alpha} \varphi(x) = Min(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i}^{N}\alpha_i) \qquad \sum_{i}^{N}\alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1,2,\dots,N$$

- 将5个实例点带入

$$\mathcal{L}(w,b,\alpha) = \alpha_2^2 + \alpha_3^2 + 2\alpha_4^2 + \frac{1}{4}\alpha_5^2 - 2\alpha_2\alpha_4 - \alpha_2\alpha_5 + 2\alpha_3\alpha_4 + \alpha_4\alpha_5 - \alpha_1 - \alpha_2 - \alpha_3 - \alpha_4 - \alpha_5$$

$$\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4 + \alpha_5$$

$$\mathcal{L}(w,b,\alpha) = \alpha_2^2 + \alpha_3^2 + 2\alpha_4^2 + \frac{1}{4}\alpha_5^2 - 2\alpha_2\alpha_4 - \alpha_2\alpha_5 + 2\alpha_3\alpha_4 + \alpha_4\alpha_5 - 2\alpha_3 - 2\alpha_4 - 2\alpha_5$$

$$\begin{cases} \frac{\partial \mathcal{L}}{\alpha_2} = 0 \\ \frac{\partial \mathcal{L}}{\alpha_3} = 0 \\ \frac{\partial \mathcal{L}}{\alpha_4} = 0 \\ \frac{\partial \mathcal{L}}{\partial \alpha_5} = 0 \end{cases} \Rightarrow \begin{cases} \alpha_2 = \alpha_4 + \alpha_5 \\ \alpha_3 + \alpha_4 = 1 \\ 4\alpha_4 - 2\alpha_2 + 2\alpha_3 + \alpha_5 = 2 \\ \alpha_5 - 4\alpha_2 + 2\alpha_4 = 4 \end{cases} \qquad \begin{cases} \alpha_1 = 3 \\ \alpha_2 = -2 \\ \alpha_3 = 3 \\ \alpha_4 = -2 \\ \alpha_5 = 0 \end{cases}$$

Soft Margin

$$0 \leq \alpha_i \leq C$$

令C=4

$$\begin{cases} \alpha_1 = 3 \\ \alpha_2 = 0 \\ \alpha_3 = 3 \\ \alpha_4 = 0 \\ \alpha_5 = 0 \end{cases}$$

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i = \alpha_1 y_1 x_1 + \alpha_2 y_2 x_2 + \alpha_3 y_3 x_3 + \alpha_4 y_4 x_4 + \alpha_5 y_5 x_5$$

$$= 3(0,0) + 0(1,0) - 3(0,1) - 0(1,1) - 0(0.5,0) - 0(0.5,0) = (0,-3) \Rightarrow \quad w \cdot x = (0,-3)(x1,x2)$$

$$= -3x2$$

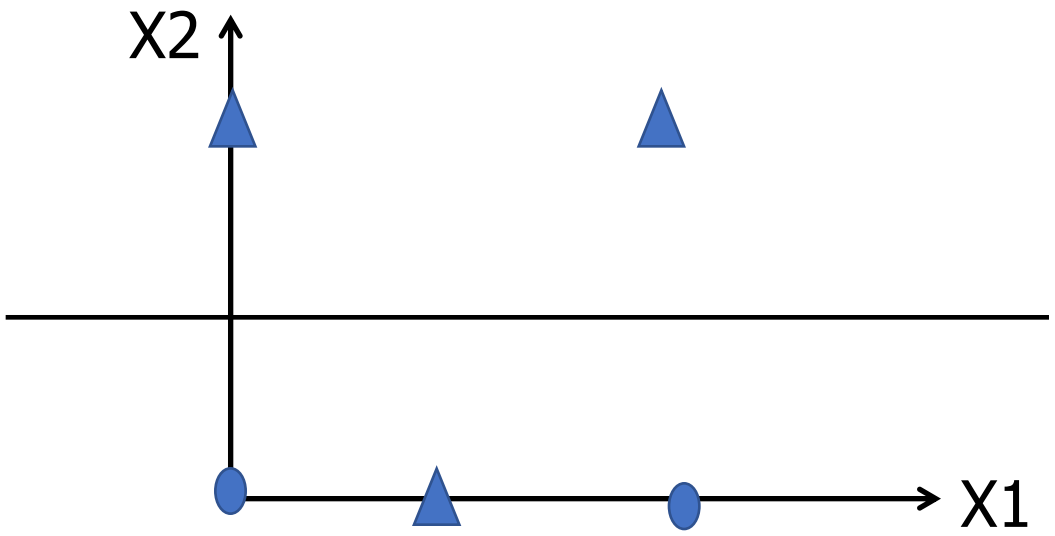
$$b_2 = y_3 - x_3 \sum_{i=1}^{N} \alpha_i y_i x_i \qquad b_1 = y_1 - x_1 \sum_{i=1}^{N} \alpha_i y_i x_i = 1 - 0 = 1$$

$$= -1 - [3(0,0) \cdot (0,1)^T + 0(1,0) \cdot (0,1)^T - 3(0,1) \cdot (0,1)^T - 0(1,1) \cdot (0,1)^T - 0(0.5,0) \cdot (0,1)^T] = -1 + 3 = 2$$

Example

$$\bar{b} = \frac{b_1 + b_2}{2} = \frac{3}{2} \qquad -3x2 + \frac{3}{2} = 0 \quad \Rightarrow \quad x2 = \frac{1}{2}$$

$$f(x) = \text{sign}(x2 - \frac{1}{2})$$



Soft Margin

Which of the following statements are true?

A. The solution of an SVM will always change if we remove some instances from the training set.

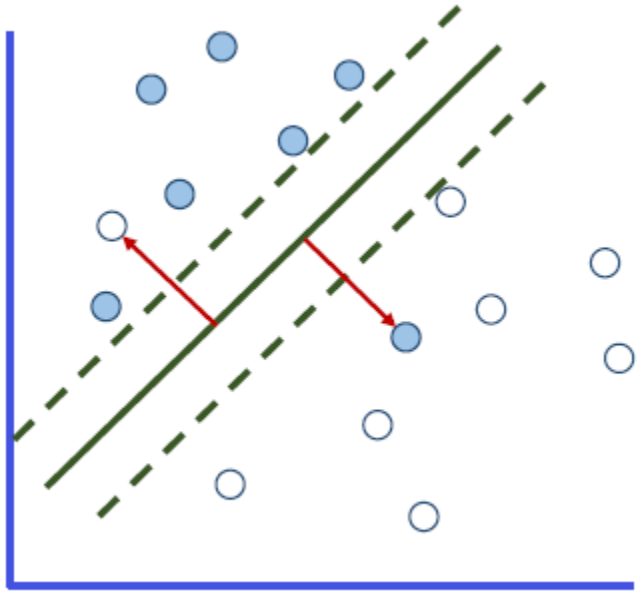
B. If we know that our data is linearly separable, then it does not make sense to use slack variables.

C. If you only had access to the labels $\{y_i\}$ and the inner products $\{x_i^T x_j\}$, we can still find the solution to the SVM

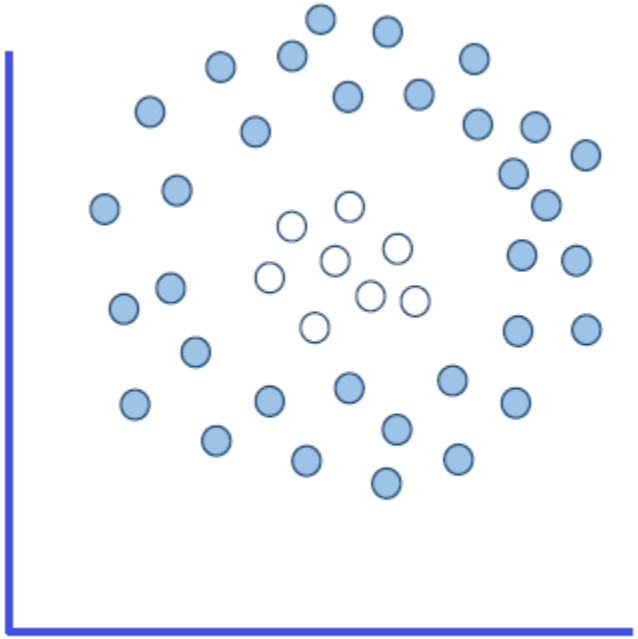
A: False, B: False, C: True

Kernel Trick

Noise in Data

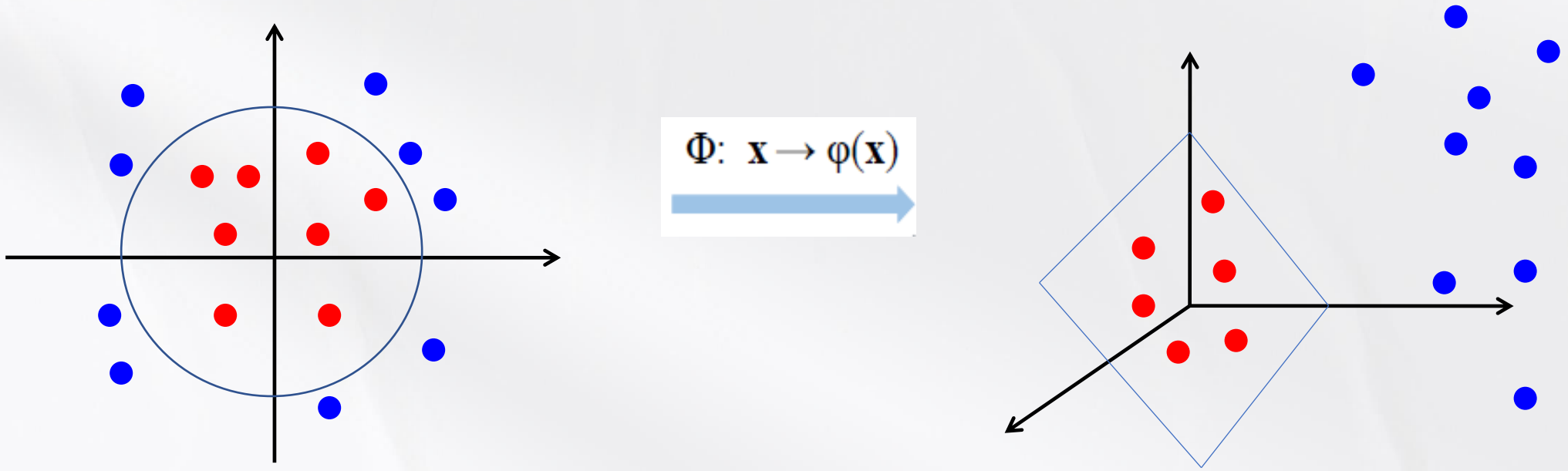


Non linear boundary



Kernel Trick

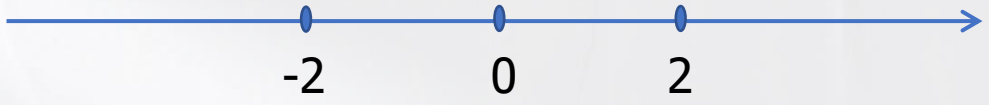
- ✓ **General idea:** The original input space can be mapped to some higher-dimensional feature space where the training set is separable.



φ is a nonlinear mapping to possibly high dimensional space

Kernel Trick

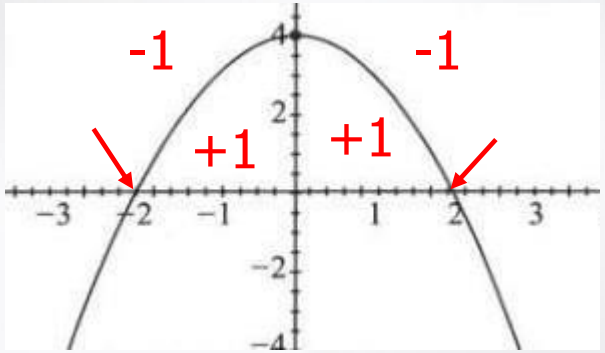
transformation



$\Phi: R^1 \rightarrow R^2$

$x \in class + 1, \quad if -2 \leq x \leq 2$

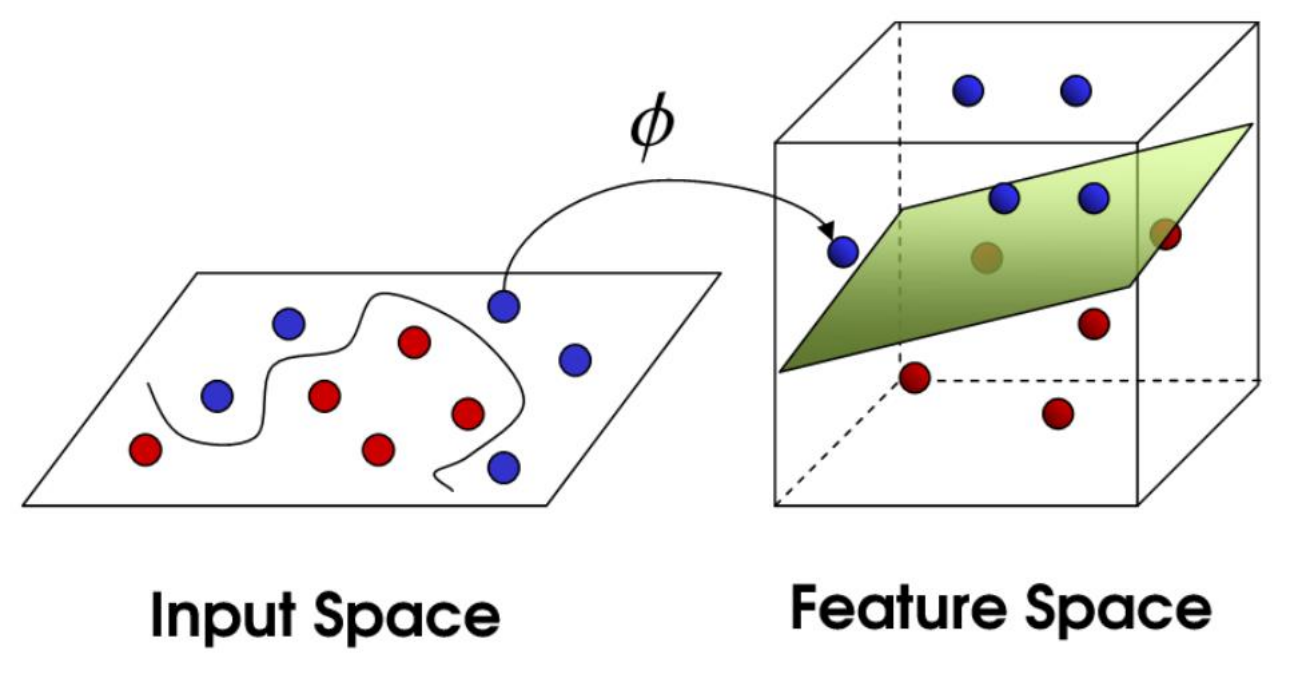
$x \in class - 1, \quad otherwise$



$y = -x^2+4$

$$f(x) = \begin{cases} 1 & -x^2 + 4 > 0 \\ -1 & -x^2 + 4 \leq 0 \end{cases}$$

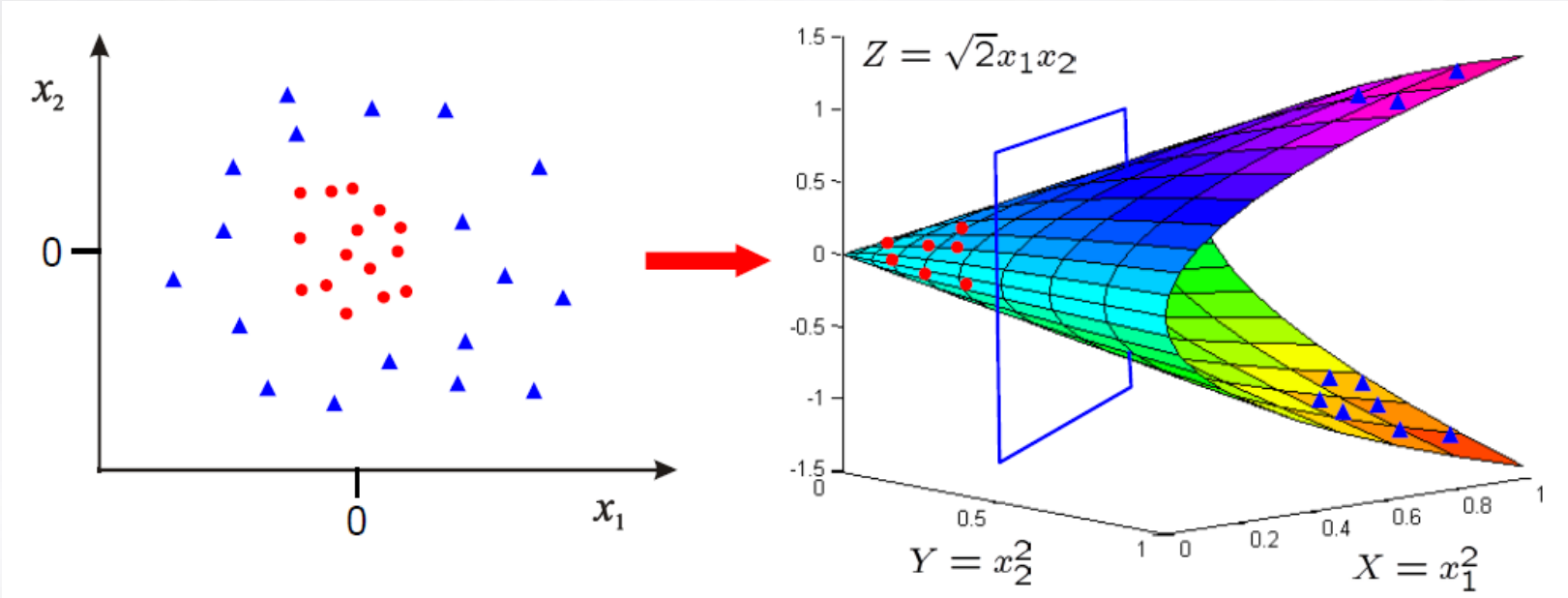
Kernel Trick



Kernel Trick

$$\phi: \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix}$$

$$R^2 \rightarrow R^3$$



- Data is linearly separable in 3D
- This means that the problem can still be solved by a linear classifier

Kernel Trick

Kernel function

A function that takes as its inputs vectors in the original space and returns the dot product(点积) of the vectors in the feature space is call a kernel function.

More formally, if we have data $x, z \in X$ and a map, $\Phi: X \rightarrow R^N$ then

$$k(x, z) = \langle \phi(x), \phi(z) \rangle$$

is a kernel function

Kernel Trick

inner products

$$\min_{\alpha} \varphi(x) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i}^{N}\alpha_i$$

Recall our SVM form:

$$s.t. \sum_{i=1}^{N} y_i \alpha_i = 0 \quad 0 \le \alpha_i \le C$$

- Training only relies on inner products: $x_i^T x_j$
  - Using SVM on the feature space : $\phi(x_i)$ only need: $\phi(x_i)^T \phi(x_j)$
  - Therefore, no need to design $\phi(\cdot)$ only need to design

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Kernel

Feature Maps

Kernel Trick

Data vectors occur only as dot product in SVM learning

Training phase

$$\max \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \left(\varphi(x_i) \cdot \varphi(x_j) \right)$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0 \quad \text{for all } i$$

Testing phase

$$\text{Label} = \text{sign}(w \cdot \varphi(x_{test}) + b) =$$

$$\text{sign}\left(\sum_i \alpha_i y_i \varphi(x_i) \cdot \varphi(x_{test}) + b \right)$$

Kernel Trick

If we can find a function $k(x,y)$, which is equivalent to $\varphi(x) \cdot \varphi(y)$, we can avoid explicit mapping to high dimensions.

$$\max \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0 \quad \text{for all } i$$

$$\text{Label} = \text{sign}\left(\sum_i \alpha_i y_i k(x_i, x_{test}) + b\right)$$

Kernel Trick

- ✓ Is ϕ necessary? Not necessary!
- ✓ Can we only apply K? YES!
- ✓ Given a feature mapping ϕ, can we find a corresponding kernel function K calculating the inner product in feature space? YES!
- ➢ Given a kernel function K, can we construct a feature space H (i.e., a feature mapping ϕ)such that we use K to compute the inner product in H? YES!

Kernel Trick

Kernel Types: Polynomial

- Fix degree *d* and constant c: $k(x, x') = (x^T x' + c)^d$
- What are $\phi(x)$?
- Expand the expression to get $\phi(x)$

$$\forall x, x' \in R^2 \qquad K(x, x') = (x_1 x_1' + x_2 x_2' + c)^2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1 x_2 \\ \sqrt{2c}x_1 \\ \sqrt{2c}x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} x_1'^2 \\ x_2'^2 \\ \sqrt{2}x_1' x_2' \\ \sqrt{2c}x_1' \\ \sqrt{2c}x_2' \\ c \end{bmatrix}$$

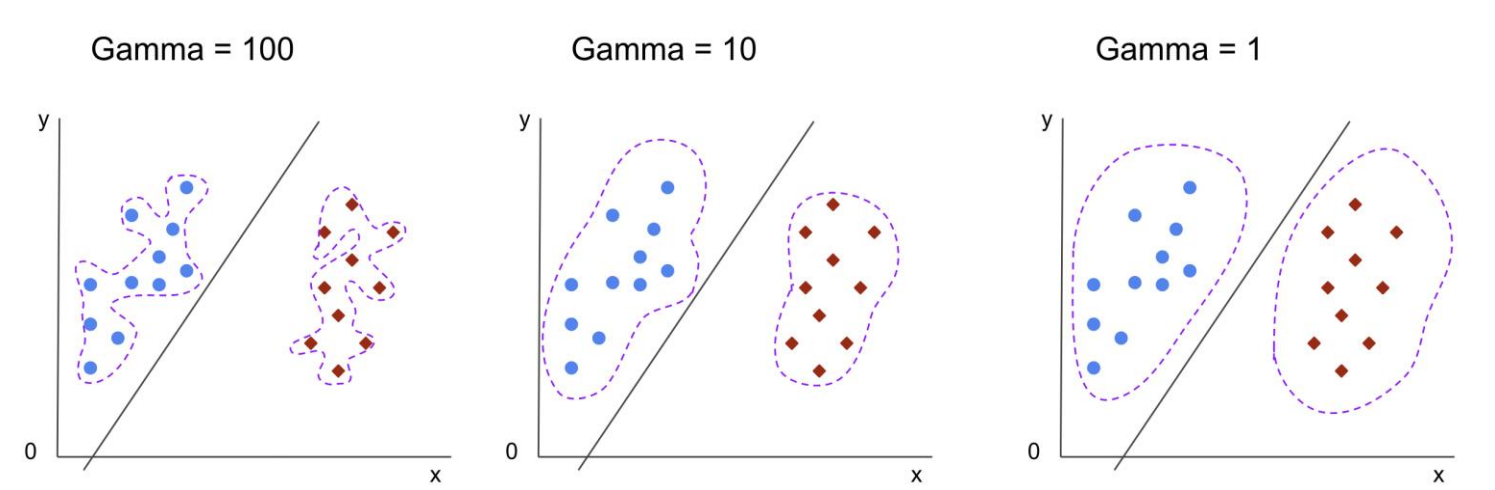
Kernel Trick

Kernel Types: Gaussian

With RBF kernels, you are projecting to an infinite dimensional space

$$k(x, x') = \exp(-\gamma \|x - x'\|^2) \qquad \gamma = \frac{1}{\sigma^2}$$

With RBF kernels, you are projecting to an infinite dimensional space



- **High γ :** complex boundary, overfitting.
- **Low γ :** smooth boundary, underfitting.

$$\text{when } x = (x), K(x, x') = \exp\left(-(x - x')^2\right)$$

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

Taylor

$$= \exp\left(-(x)^2\right)\exp\left(-(x')^2\right)\exp\left(2xx'\right)$$

$$= \exp\left(-(x)^2\right)\exp\left(-(x')^2\right)\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!}$$

$$= \sum_{i=0}^{\infty} \exp\left(-(x)^2\right)\exp\left(-(x')^2\right)\sqrt{\frac{2^i}{i!}}\sqrt{\frac{2^i}{i!}}(x)^i(x')^i = \phi(x)^T\phi(x')$$

$$\phi(x) = \exp\left(-x^2\right)\left(1, \sqrt{\frac{2}{1!}}x, \sqrt{\frac{2^2}{2!}}x^2, \ldots\right)$$

More generally, Gaussian kernel

$$K(x, x') = \exp\left(-\gamma \left\| x - x' \right\|^2\right) \text{ with } \gamma > 0$$

Kernel Trick

$$\sigma \to 0 \Rightarrow \gamma = \frac{1}{2\sigma^2} \to \infty \Rightarrow \gamma \left\| x - x' \right\|^2 \to \infty \Rightarrow \mathcal{K}(x, x') \to 0$$

$$|\phi(x) - \phi(x')|^2 = \mathcal{K}(x,x) - 2\mathcal{K}(x,x') + \mathcal{K}(x',x')$$

$$= 2 - 2\mathcal{K}(x,x') = 2$$

The distance between every two mapped points is equal ($\sqrt{2}$). That is, there is no clustering phenomenon (intuitively, each point itself is a group, the distance between them is small), so that each sample point will be separately clustered.

所有映射后的点彼此之间的距离均相等（为 $\sqrt{2}$ ），即不存在聚类现象（直观上理解就是各个点自己聚在一起，之间的距离很小），这样每个样本点将被单独形成一个类别。

增加γ可提高预测精度，但可能导致过拟合。

Kernel Trick

$$\sigma \to \infty \Rightarrow \gamma = \frac{1}{2\sigma^2} \to 0 \Rightarrow \gamma \left\| x - x' \right\|^2 \to 0 \Rightarrow \mathcal{K}(x, x') \to 1$$

$$|\phi(x) - \phi(x')|^2 = \mathcal{K}(x,x) - 2\mathcal{K}(x,x') + \mathcal{K}(x',x')$$
$$= 2 - 2\mathcal{K}(x,x') = 0$$

If σ is too large （gamma is too small），two different points are mapped to become the same point in the high-dimensional space (the distance between them is 0). So all the samples are in the same category.

σ很大，两个不同的点经过映射后，成为高维空间上的同一个点（相互之间距离为0）。那么，所有的样本点将被划分成同一个类，无法区分开。 总结：参数σ越小，分的类别会越细，容易导致过拟合；参数σ越大，分的类别会越粗，导致无法将数据区分开来。

Increasing σ , decreasing accuracy of prediction, but be careful of underfitting

Kernel Trick

Overfitting with High Gamma Values

One of the most common pitfalls is setting the gamma value too high. A high gamma value results in a complex and wiggly decision boundary, leading to overfitting, where the model performs poorly on unseen data.

Underfitting with Low Gamma Values

A low gamma value results in a smoother decision boundary that may be too simplistic to capture the underlying patterns in the data. This can lead to poor model performance.

04

Kernel Trick

Ignoring the Interaction with the C Parameter

→

The gamma parameter should not be tuned in isolation. It is crucial to consider the interaction between gamma and the gamma.

C parameter controls the trade-off between achieving a low training error and a low testing error. Ignoring this interaction can lead to suboptimal tuning and poor model performance.

Kernel Trick

Which of the following statements are true?

A. SVMs with nonlinear kernels transform the low dimensional features to a high dimensional space and then performing linear classification in that space.

B. The “Kernel trick” refers to computing this transformation and then applying the dot product between the transformed points

A: True, B: False

Kernel Trick

Consider the kernel for $k(x, x') = (xx' + 1)^3$. Give an explicit expression for a feature map ϕ such that

$$\phi(x)^\top \phi(x') = k(x, x').$$

1. $\phi(x)^\top = [x^3, x^2, x, 1]$
2. $\phi(x)^\top = [x^3, \sqrt{3}x^2, \sqrt{3}x, 1]$
3. $\phi(x)^\top = [x^3, \sqrt{3}x^2, x, \sqrt{3}]$
4. $\phi(x)^\top = [x^3, \sqrt{3}x^2, \sqrt{3}x]$

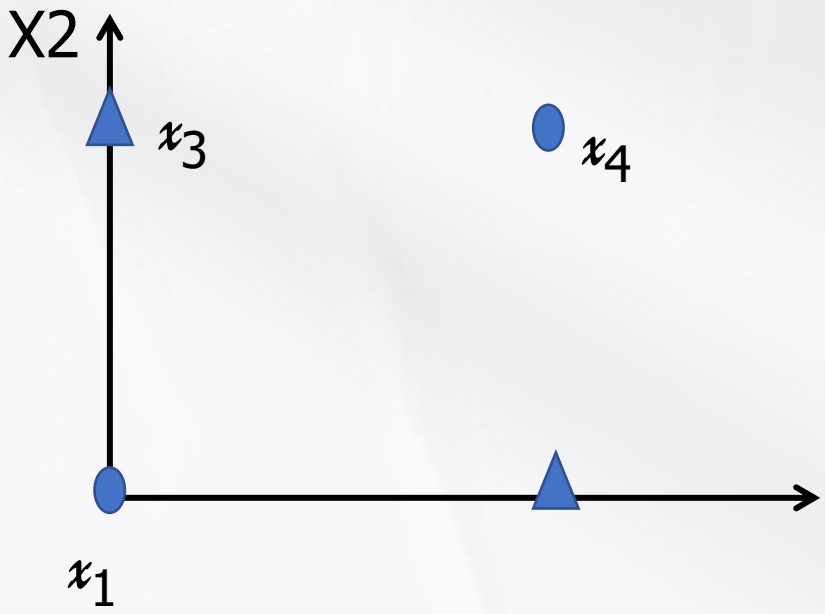
Answer: 2

$$k(x, x') = (xx' + 1)^3$$
$$= (xx')^3 + 3(xx')^2 + 3xx' + 1$$
$$= \begin{bmatrix} x^3 & \sqrt[3]{3}x^2 & \sqrt[3]{3}x & 1 \end{bmatrix} \begin{bmatrix} (x')^3 \\ \sqrt[3]{3}(x')^2 \\ \sqrt[3]{3}x' \\ 1 \end{bmatrix}$$

Kernel Trick

example

	Y	X1	X2
x_1	圆形 1	0	0
x_2	三角形 -1	1	0
x_3	三角形 -1	0	1
x_4	圆形 1	1	1



$$\kappa(x_i \cdot x_j) = (x_i \cdot x_j + 1)^2$$

Polynomial Kernel

$$\min_{\alpha}\varphi(x)=Min(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \varkappa(x_i\cdot x_j)-\sum_{i}^{N}\alpha_i) \qquad \sum_{i}^{N}\alpha_i y_i=0,\quad \alpha_i\geq 0,\quad i=1,2,\dots,N$$

$$0\leq\alpha_i\leq C$$

$$\mathcal{L}(w,b,\alpha)=\frac{1}{2}(\alpha_1^2+4\alpha_2^2+4\alpha_3^2+9\alpha_4^2-2\alpha_1\alpha_2-2\alpha_1\alpha_3+2\alpha_1\alpha_4+2\alpha_2\alpha_3-8\alpha_2\alpha_4-8\alpha_3\alpha_4)-\alpha_1$$
$$-\alpha_2-\alpha_3-\alpha_4$$

由于： $\alpha_1+\alpha_2=\alpha_3+\alpha_4$ $\mathcal{L}(w,b,\alpha)=\frac{3}{2}\alpha_2^2+\frac{3}{2}\alpha_3^2+4\alpha_4^2-3\alpha_2\alpha_4-3\alpha_3\alpha_4-2\alpha_2-2\alpha_3$

$$\begin{cases}\frac{\partial\mathcal{L}}{\alpha_2}=0\\ \frac{\partial\mathcal{L}}{\alpha_3}=0\\ \frac{\partial\mathcal{L}}{\alpha_4}=0\end{cases}\Rightarrow\begin{cases}3\alpha_2=3\alpha_4+2\\ 3\alpha_3=3\alpha_4+2\\ 8\alpha_4-3\alpha_2-3\alpha_3=0\end{cases}\qquad\begin{cases}\alpha_1=\frac{4}{3}\\ \alpha_2=\frac{8}{3}\\ \alpha_3=\frac{8}{3}\\ \alpha_4=2\end{cases}$$

令：C=3　$0 \leq \alpha_i \leq C$　i=1,2,3,4 都满足条件，对偶函数可以取得最小值

$$b_1 = y_1 - \sum_{i=1}^{N} \alpha_i y_i \kappa(x_1 \cdot x_i)$$

$$= 1 - [\alpha_1 y_1 (x_1 \cdot x_1 + 1)^2 + \alpha_2 y_2 (x_2 \cdot x_1 + 1)^2 + \alpha_3 y_3 (x_3 \cdot x_1 + 1)^2 + \alpha_4 y_4 (x_4 \cdot x_1 + 1)^2]$$

$$= 1 - \left(\frac{4}{3} - \frac{8}{3} - \frac{8}{3} + 2\right) = 3$$

此时超平面的w和x是隐式的，但可以通过核函数求出w和x的内积从而求出决策函数：

$$f(x) = \text{sign}\left[\left(\sum_{i=1}^{N} \alpha_i y_i \kappa(x_i \cdot x_j)\right) + b\right]$$

$$f(x) = \text{sign}\{\alpha_1 y_1[(X1,X2)\cdot(0,0)^T+1]^2+\alpha_2 y_2[(X1,X2)\cdot(1,0)^T+1]^2$$
$$+\alpha_3 y_3[(X1,X2)\cdot(0,1)^T+1]^2+\alpha_4 y_4[(X1,X2)\cdot(1,1)^T+1]^2+b$$

$$=\frac{4}{3}-\frac{8}{3}(X1^2+1)-\frac{8}{3}(X2^2+1)+2(X1+X2+1)^2+3$$

$$\longrightarrow \quad f(x)=\text{sign}(1-\frac{2}{3}X1^2-\frac{2}{3}X2^2-\frac{2}{3}X1-\frac{2}{3}X2+4X1X2)$$

$$\begin{cases} f(x_1)=1 \\ f(x_2)=-1 \\ f(x_3)=-1 \\ f(x_4)=1 \end{cases}$$

	Y	X1	X2
x_1	圆形 1	0	0
x_2	三角形 -1	1	0
x_3	三角形 -1	0	1
x_4	圆形 1	1	1

Multi- Class SVM

one-versus-one (1-1)

breaks down the multiclass problem into **multiple binary classification** problems. A binary classifier per each pair of classes.

we need a hyperplane to separate between every two classes, neglecting the points of the third class.

This means the separation takes into account only the points of the two classes in the current split.

Multi- Class SVM

```
> head(SvmFit$decision.values)
        1/2       1/0         2/0
1 1.033036 1.2345269 -0.61225558
2 1.600637 1.2219439  0.76098974
3 1.068253 1.0112116  0.59276079
4 1.047869 0.9999145  0.05666298
5 2.146043 1.4892178  1.23321397
6 1.031256 1.2279855 -1.10302134
```

Let the K-th class be+1 class, and the rest be -1 class in sequence, and construct binary classification separately. K: number of classes

令第K类是+1类，其余依次为-1类，分别构建二值分类。

In the One-to-One approach, the classifier can use K(K-1)/2 SVMs.

1 VS 1	+: y1 － : y2	+: y1 － : y3	+: y1 － : y4	+: y2 －: y3	+: y2 － : y4	+: y3 － : y4
classify	+	+	－	+	－	+

For new observations, category prediction should be based on the mode category provided by K(K-1)/2 support vector classifications.

对于新观测，类别预测应该是 K(K-1)/2 个支持向量分类给出的众数类别

Multi- Class SVM

one-versus-one (1-1)

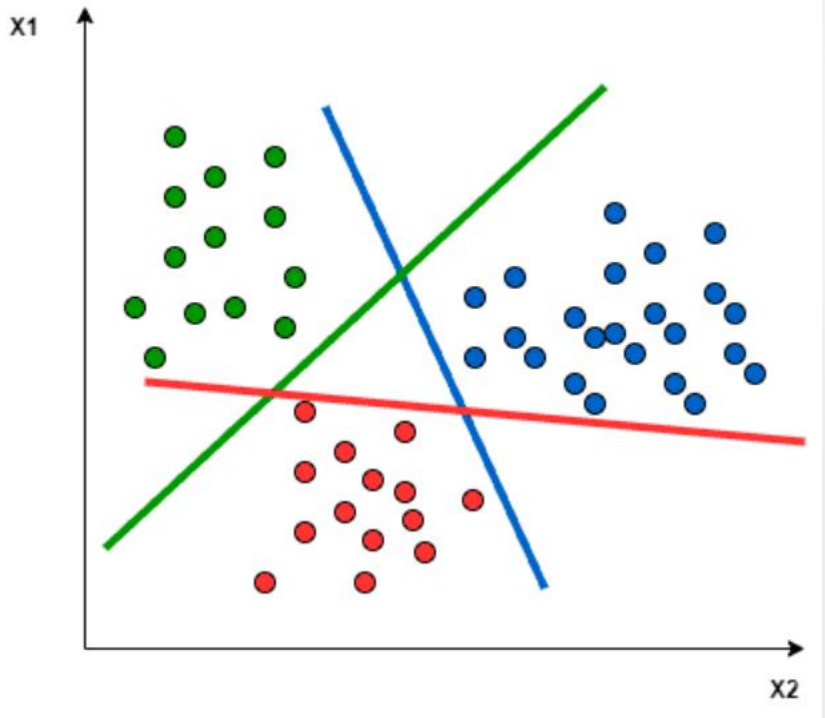
This is much less sensitive to the problems of imbalanced datasets but is much more computationally expensive.

Multi- Class SVM

One-to-Rest approach

we need a hyperplane to separate between **a class and all others at once.** This means the separation takes all points into account, dividing them into two groups;

a group for the class points and a group for all other points.



For example, the green line tries to maximize the separation between green points and all other points at once.

Multi- Class SVM

新观测带入到K个超平面方程，其类别的预测值为 $\max(W_K^T X + b_k)$ 对应的类别K.

One-to-Rest approach

1 VS r	+: (y1 y2) y3	+: (y1 y3) y2	- : (y2 y3) y1
classify	$\max(W_K^T X + b_k)$		

K -classifiers

The new observation is introduced into K hyperplane equations, and the predicted class label of it is the maximum value of the hyperplane equation corresponding class K.

Samples unbalance: Because the data of a certain type of sample is much smaller than the sum of the data of other categories of samples.

Summary

Pros of Kernelized SVM	Cons of Kernelized SVM
They perform very well on a range of datasets.	Efficiency (running time and memory usage) decreases as the size of the training set increases. O(n^3) 序列最小优化（SMO）算法
They are versatile: different kernel functions can be specified, or custom kernels can also be defined for specific datatypes.	Needs careful normalization of input data and parameter tuning.
They work for both high and low dimensional data. Effective in cases of limited data.	**Computationally intensive.** SVMs can be computationally expensive, especially when dealing with large data sets. The training time and memory requirements increase significantly with the number of training samples.

06

Summary

支持向量中的核函数

1.线性不可分的数据：在现实世界中，很多数据集并不是线性可分的，直接应用线性SVM无法找到有效的分类边界。

2.映射到高维空间：将原始数据通过某种非线性映射（函数）转换到一个更高维的空间处理线性不可分的数据，但计算上会有维度灾难问题。这种映射可以是显式的，也可以是隐式的。

3.计算效率（核技巧）：核函数能够隐式地计算两个样本之间的内积，而不需要实际执行映射。核函数在原空间的值等于在高维空间中样本的内积，从而大大简化了计算过程。同时也解决了维度灾难问题。