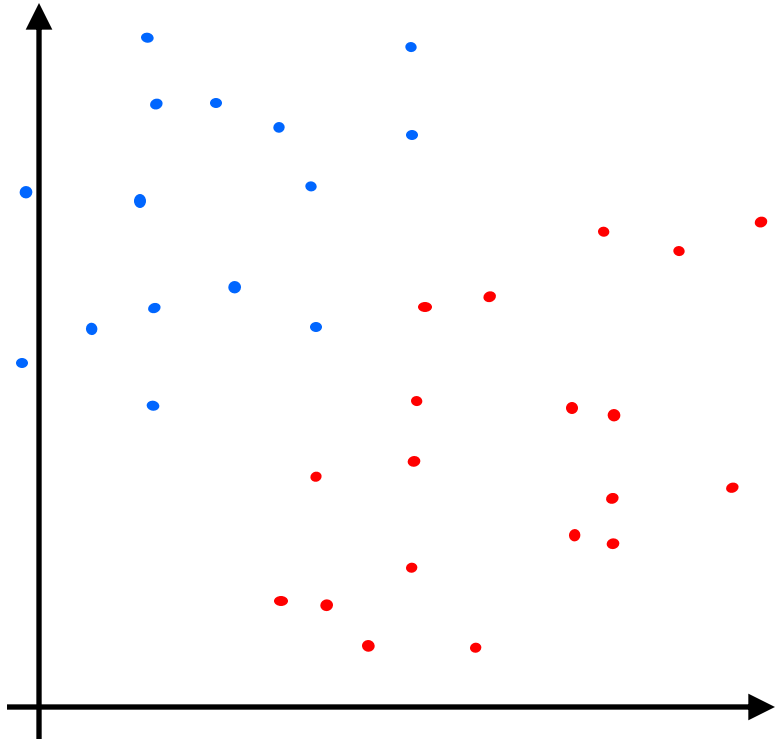
Support Vector Machine

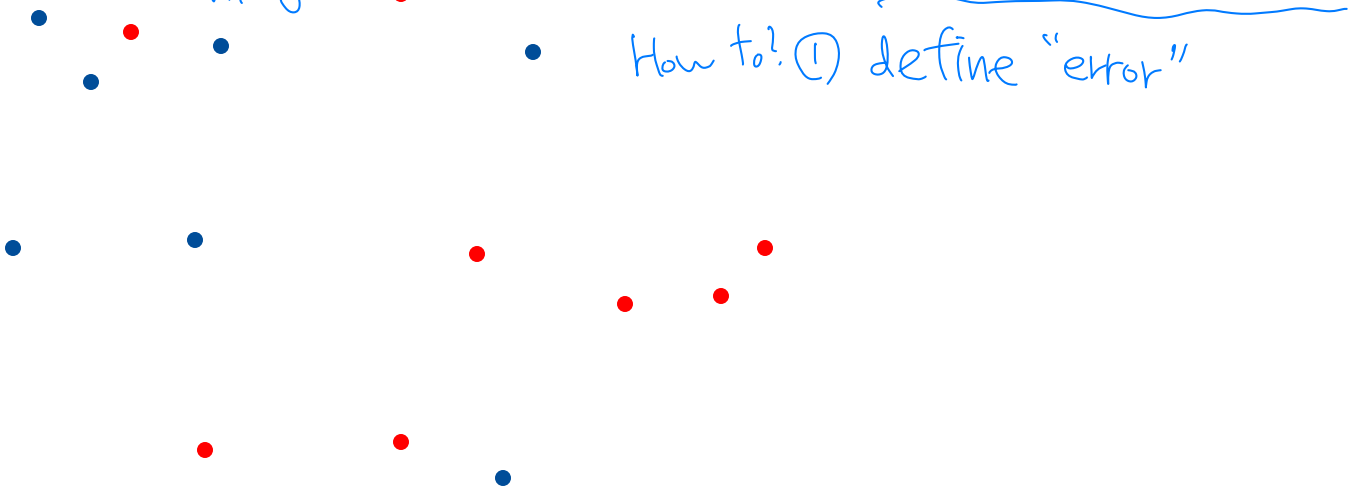
Linear SVM with Soft Margin

- **Find a linear boundary**



Linear SVM with Soft Margin

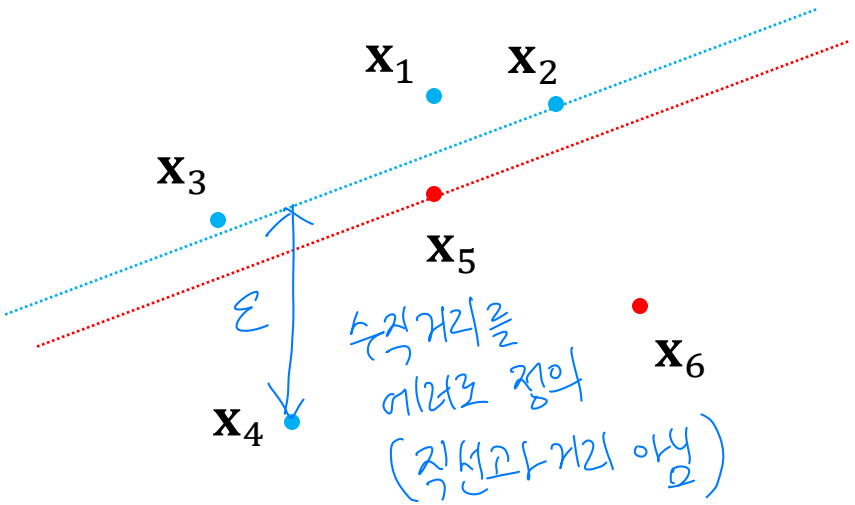
- **Non-Linearly Separable Problem**
  - Impossible to find a linear boundary without error



Let's allow ERROR

Linear SVM with Soft Margin

- **Non-Linearly Separable Problem**
  - No linear boundaries without errors
  - We cannot build the SVM formula



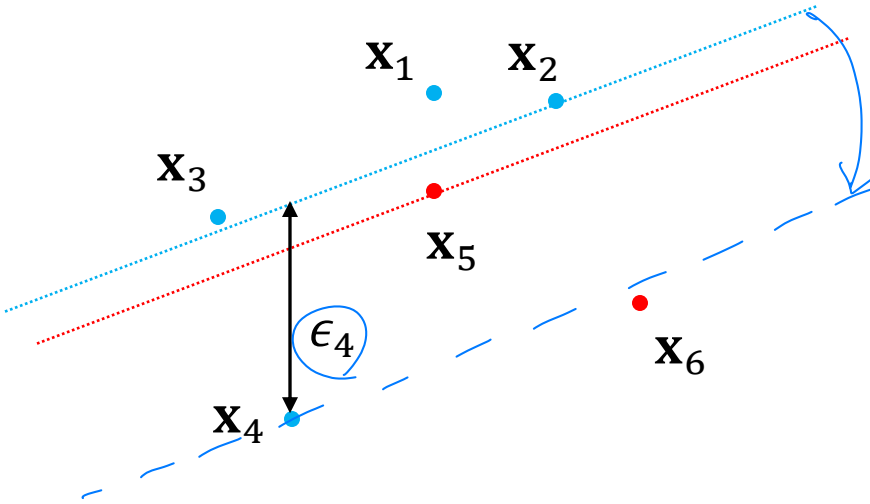
$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to
$$y_1(\mathbf{w}\cdot\mathbf{x}_1 + b) \geq 1$$
$$y_2(\mathbf{w}\cdot\mathbf{x}_2 + b) \geq 1$$
$$y_3(\mathbf{w}\cdot\mathbf{x}_3 + b) \geq 1$$
violated $$y_4(\mathbf{w}\cdot\mathbf{x}_4 + b) \geq 1$$
$$y_5(\mathbf{w}\cdot\mathbf{x}_5 + b) \geq 1$$
$$y_6(\mathbf{w}\cdot\mathbf{x}_6 + b) \geq 1$$

We cannot solve this!!

Linear SVM with Soft Margin

- **Non-Linearly Separable Problem**
  - Let's modify the conditions, and build the formular for SVM



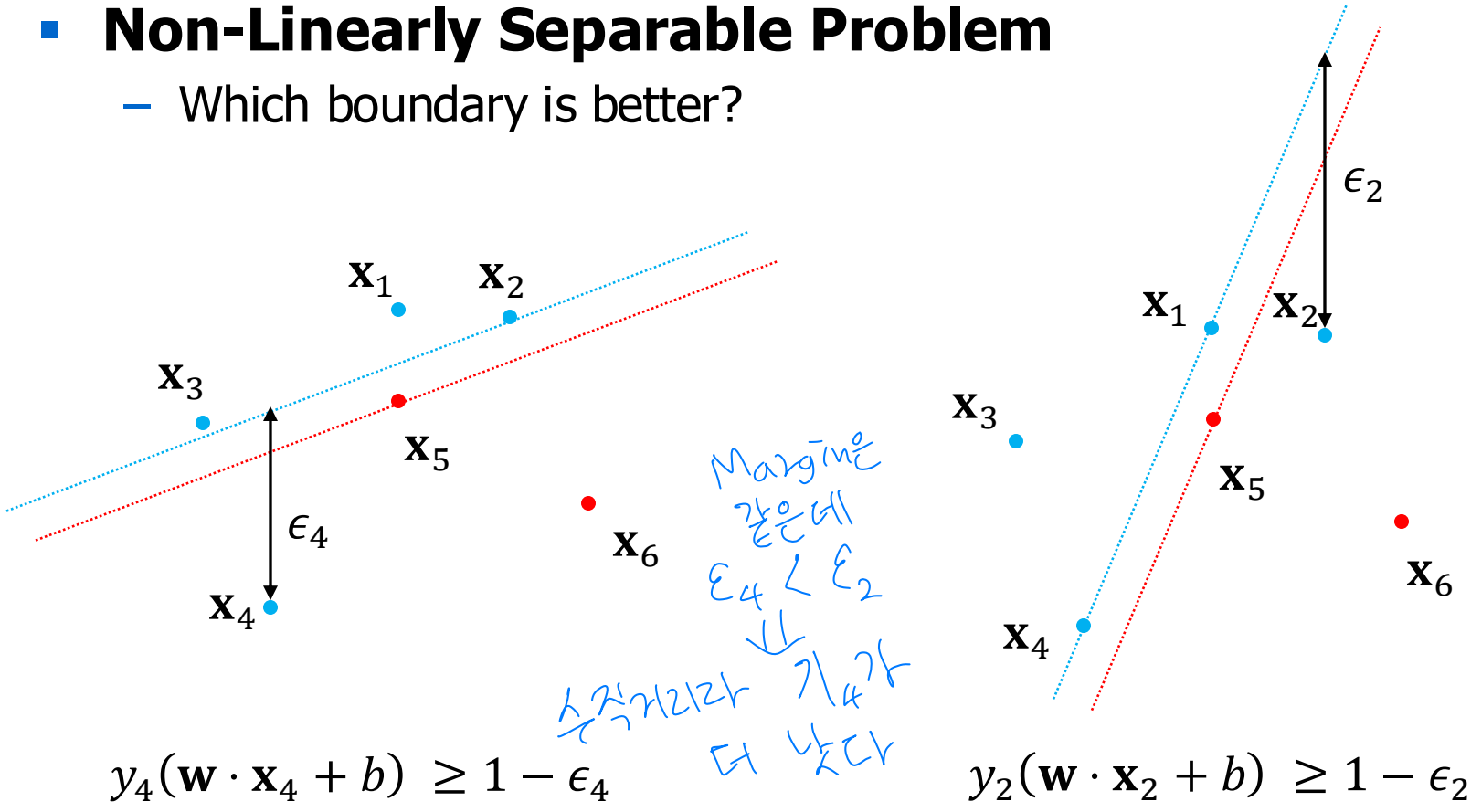
$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \epsilon_4$$

subject to

$$y_1(\mathbf{w}\cdot\mathbf{x}_1 + b) \geq 1$$
$$y_2(\mathbf{w}\cdot\mathbf{x}_2 + b) \geq 1$$
$$y_3(\mathbf{w}\cdot\mathbf{x}_3 + b) \geq 1$$
$$y_4(\mathbf{w}\cdot\mathbf{x}_4 + b) \geq 1 - \epsilon_4$$
$$y_5(\mathbf{w}\cdot\mathbf{x}_5 + b) \geq 1$$
$$y_6(\mathbf{w}\cdot\mathbf{x}_6 + b) \geq 1$$
$$\epsilon_4 > 0$$

Linear SVM with Soft Margin

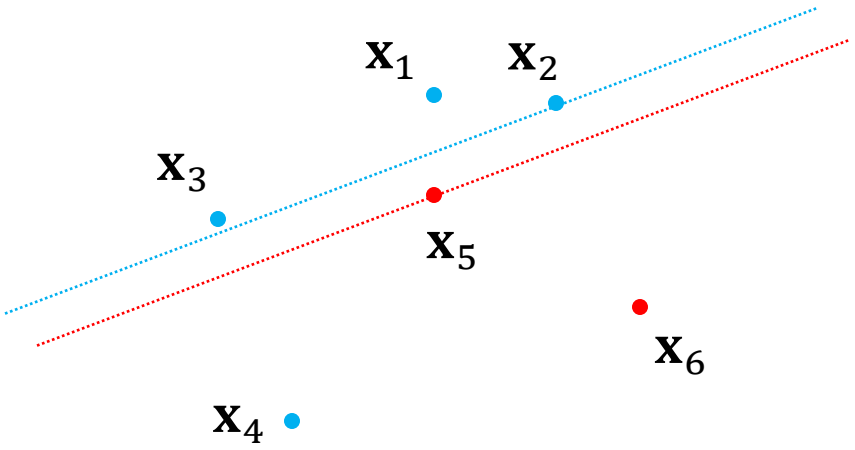
- **Non-Linearly Separable Problem**
  - Which boundary is better?



We want to modify the conditions as little as possible

Linear SVM with Soft Margin

- **How to formulate Non-Linearly Separable Problem**
  - Additional objective



Conditions in Formula

$$y_1(\mathbf{w}\cdot\mathbf{x}_1 + b) \geq 1 - \epsilon_1$$
$$y_2(\mathbf{w}\cdot\mathbf{x}_2 + b) \geq 1 - \epsilon_2$$
$$y_3(\mathbf{w}\cdot\mathbf{x}_3 + b) \geq 1 - \epsilon_3$$
$$y_4(\mathbf{w}\cdot\mathbf{x}_4 + b) \geq 1 - \epsilon_4$$
$$y_5(\mathbf{w}\cdot\mathbf{x}_5 + b) \geq 1 - \epsilon_5$$
$$y_6(\mathbf{w}\cdot\mathbf{x}_6 + b) \geq 1 - \epsilon_6$$

몇개가 violate해 있는지 알 수 없게

Permit error for all, and

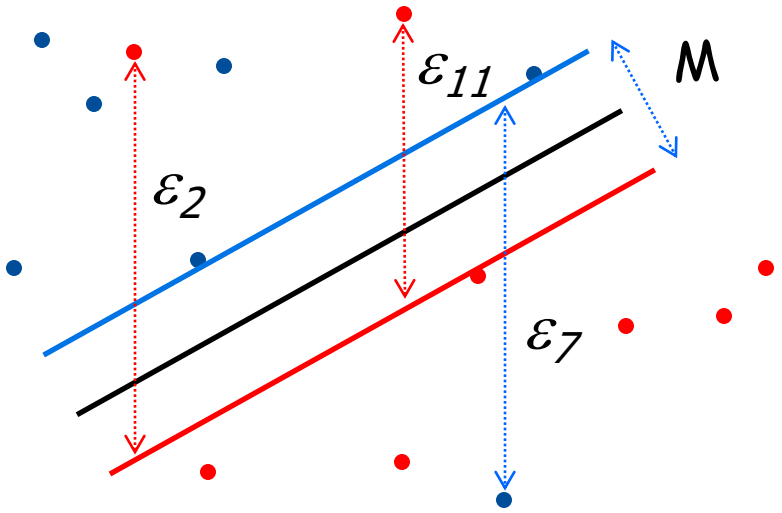
We want to modify the conditions as little as possible

$\Leftrightarrow$

$$\text{minimize} \sum_{i=1}^{n} \epsilon_i$$

Linear SVM with Soft Margin

- **Two Objectives**
  - Maximize the margin of the boundary with the minimum modification on the conditions



$$\underset{\mathbf{w},\boldsymbol{\epsilon},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \epsilon_i$

$$\underset{\mathbf{w},\boldsymbol{\epsilon},b}{\operatorname{argmin}} \sum_{i=1}^{n} \varepsilon_i$$

subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \epsilon_i$

Linear SVM with "Soft Margin"

- **Formulation**
  - We want to find a maximum margin boundary with minimum modification on conditions

Margin

Error

$$\underset{\mathbf{w},\epsilon,b}{\operatorname{argmin}} \left(\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\varepsilon_i \right)$$

$$\text{subject to } \begin{cases} y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1-\varepsilon_i, \ i = 1,\ldots,n \\ \varepsilon_i \geq 0, i = 1,\ldots,n \end{cases}$$

- C: given by experts, controlling overfitting

Linear SVM with Soft Margin

- **Dual form of the modified formula**
  - For given $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ *where* $y_i \in \{-1, +1\}$

$$\underset{\alpha_1,\ldots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, (\mathbf{x}_i \cdot \mathbf{x}_j) \right)$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n \end{cases}$$

이부분만 H·M과 다르다

  - Solution

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

$$b = y_k - \mathbf{w} \cdot \mathbf{x}_k \text{ for any } x_k \text{ such that } 0 < \alpha_k < C$$

Linear SVM with Soft Margin

- **Effect of C**

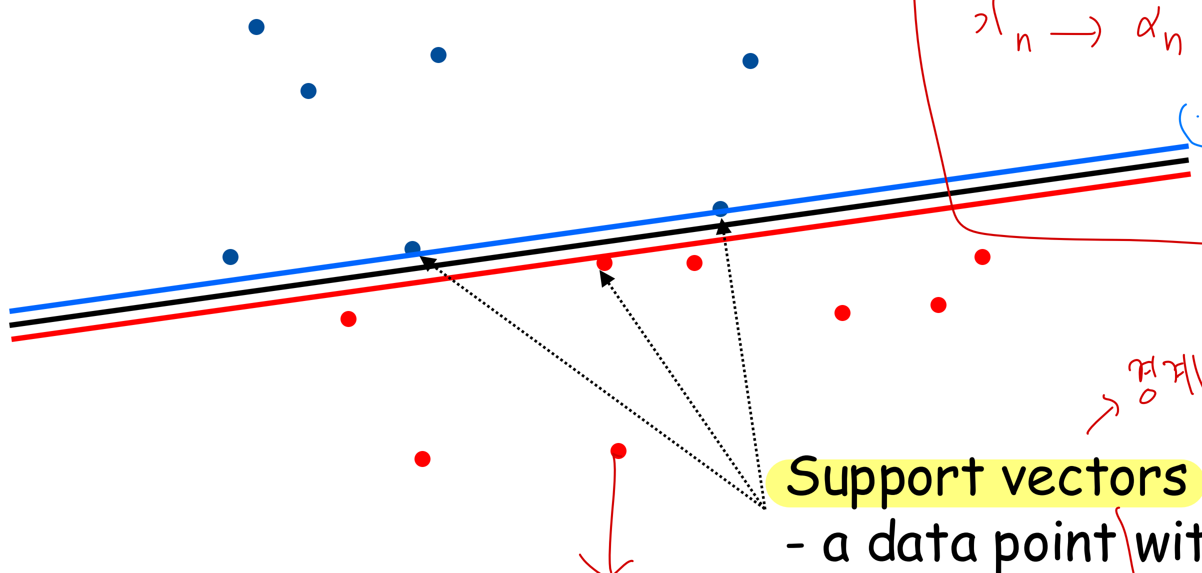
$$\underset{\mathbf{w}}{\operatorname{argmin}} \left(\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{k=1}^{R}\varepsilon_k \right)$$

$$subject\ to \begin{cases} y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \varepsilon_i \quad i = 1, \ldots, n \\ \varepsilon_i \geq 0 \quad i = 1, \ldots, n \end{cases}$$

- C = infinite
  - No allowance for error -> Narrow margin
  - Equivalent to hard margin SVM
  - Over-fitting
- C = 0
  - Maximum allowance for error -> Maximum margin
  - Over-generalization

Linear SVM with Soft Margin

- **Effect of C**
  - C = infinite : No allowance for error



data sample 변수
$x_1 \rightarrow \alpha_1$
$x_2 \rightarrow \alpha_2$
⋮
$x_n \rightarrow \alpha_n$

$\alpha_n \geq 0$ 이므로 $\alpha_n = 0$ / $\alpha_n > 0$으로 나뉨
$\alpha > 0$ 에 해당하는 x들은 Support vector가 된다
(∵ $\alpha = 0$에 해당하는 애들이 S.V가 아니기 때문)

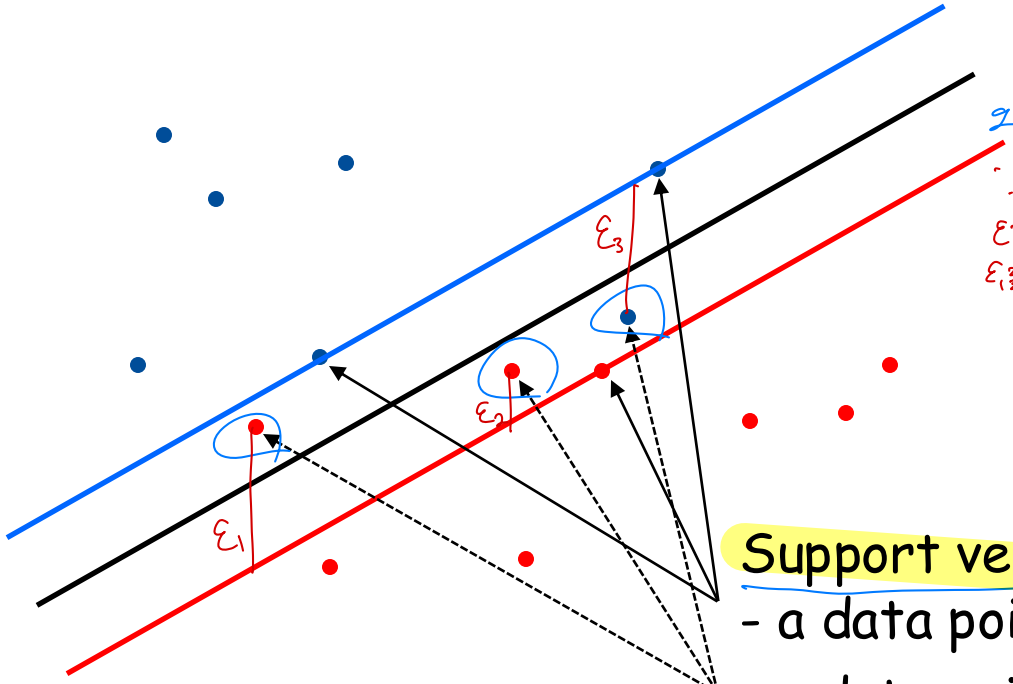
→ 경계선을 결정한다 (경계선 위 존재)

직선을 위아래로 움직이면서 처음 닿는 데이터

얘를 삭제해도 경계선 변하지 않지만
S.V를 삭제하면 경계선 변함

Linear SVM with Soft Margin

- **Effect of C**
  - C = Some Value: Some allowance for error



이표시한 애들도 S.V이다
왜? 애네들은 삭제하면 경계선이 변화함
∴ 잘못된 값이라 ξ가 존재해서 삭제되면
ξ가 사라지므로 E 함수가 달라지기 때문에
ξ_1를 지운다하면 $E=\xi_1+\xi_2+\xi_3 \Rightarrow E=\xi_2+\xi_3$

둘 이 같은 솔루션을 찾는다는 보장이없음

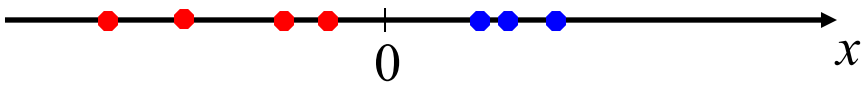
Support vectors
- a data point with $0 < \alpha < C$ → 경계선 위쪽
- a data point with $\alpha = C$ → 잘못된 값

all the other point : $\alpha = 0$

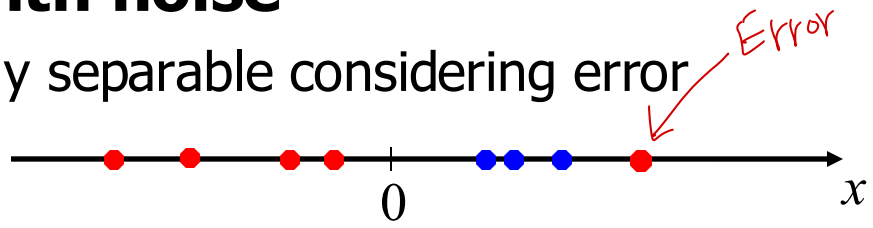
  - # of support vectors increases as *C* becomes small

Non-Linear SVM

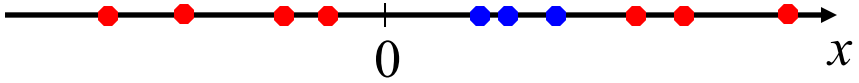
- **Data that are linearly separable**



- **Data with noise**
  - linearly separable considering error



- **What about this?** ⇒ Non-linear boundary



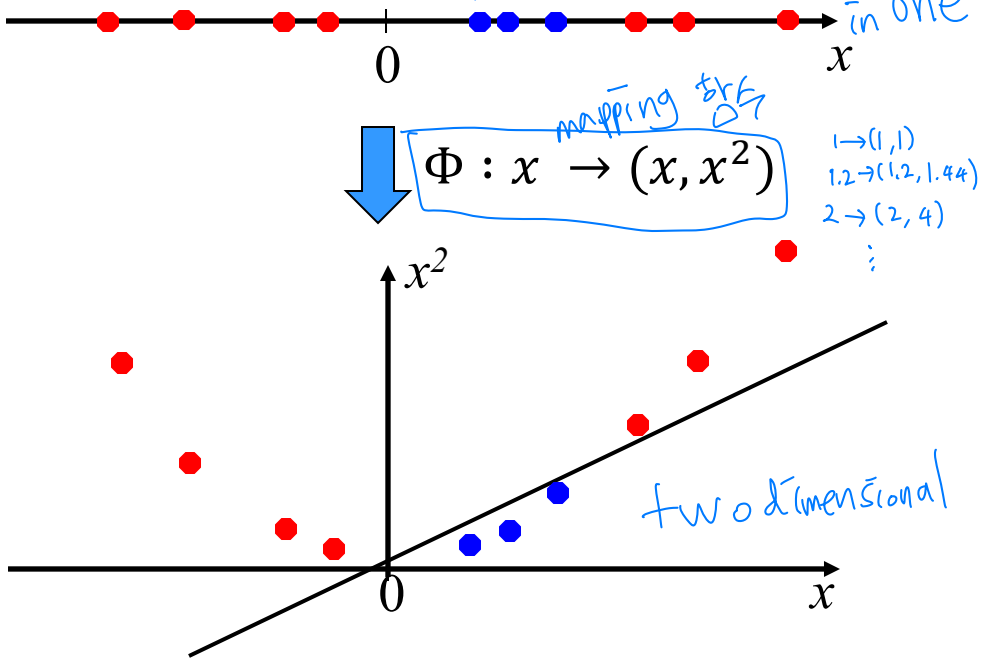
Trick!

It can be linearly separable in a HIGH dimensional space!

Uhh?

Non-Linear SVM

- **Trick**
  1. Map data to a higher-dimensional space

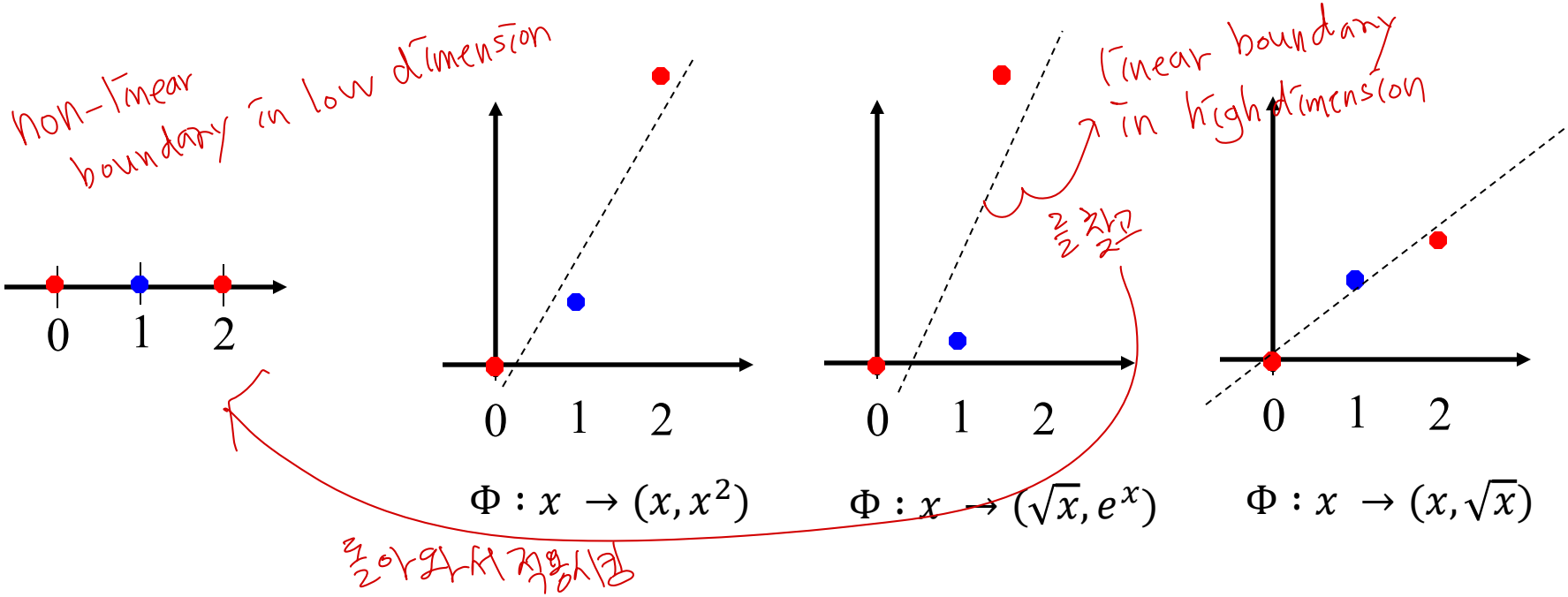


  2. Find a linear boundary in the higher-dimensional space

Can we find such mapping easily?

Non-Linear SVM

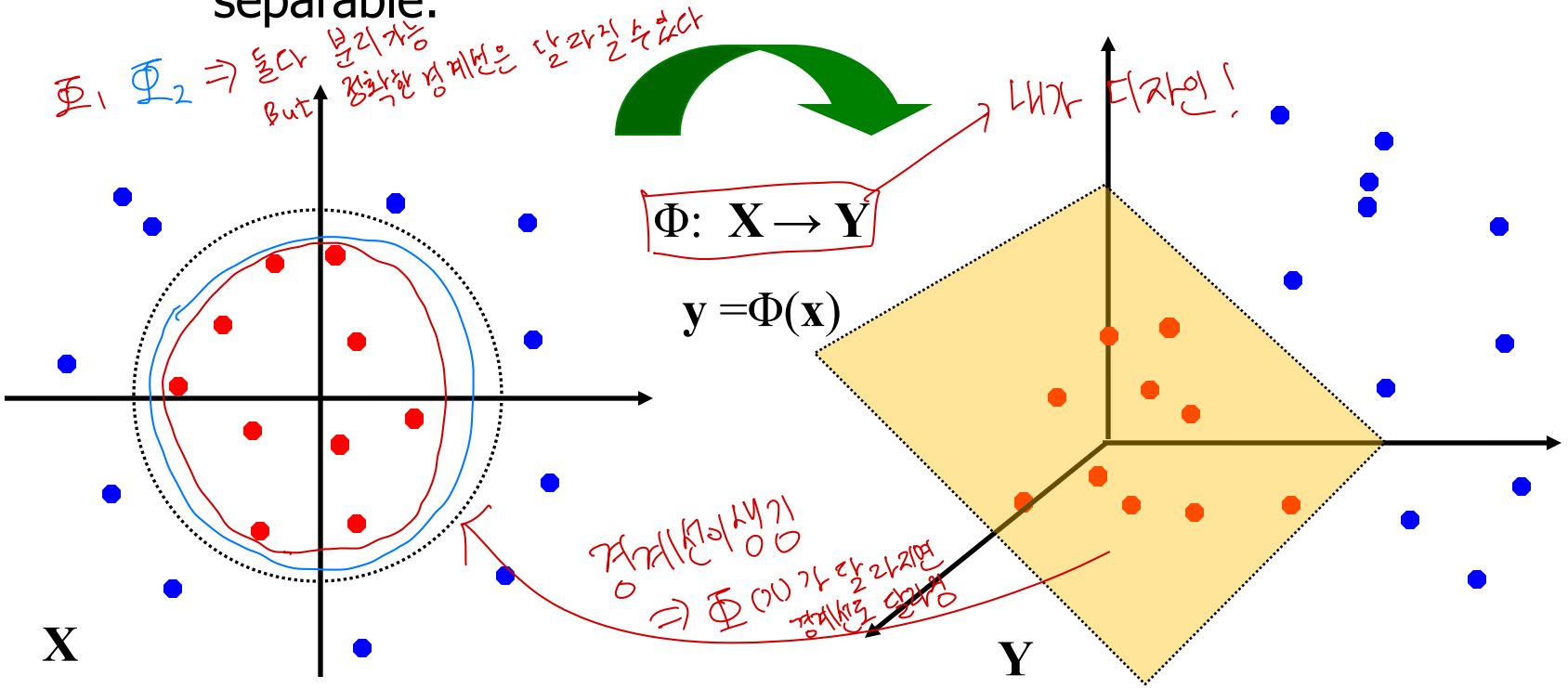
- **Most of non-linear mapping do this!!**



Then, How high dimension?
- The higher dimension, the better

Non-Linear SVM

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



Non-Linear SVM

- **Remind: Formulation for linear boundary**
  - For given $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $y_i \in \{-1, +1\}$

$$\underset{\alpha_1,\ldots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, (\mathbf{x}_i \cdot \mathbf{x}_j) \right)$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \ldots, n \end{cases}$$

with error

($C \sum_{i=1}^{n} \xi_i$ 에서 (값이 큰))

  - Solution

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

$$b = y_k - \mathbf{w} \cdot \mathbf{x}_k \text{ for any } x_k \text{ such that } 0 < \alpha_k < C$$

Non-Linear SVM

- **Formulation for non-linear boundary**

samples in low dim

1. For given $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ where $y_i \in \{-1, +1\}$
2. Define Φ for $\Phi : \mathbf{x} \rightarrow \Phi(\mathbf{x})$
3. Convert data using Φ

samples in high dim

$$D = \{(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_n), y_n)\} \; where \; y_i \in \{-1, +1\}$$

4. Solve (Find out linear boundary in high dim)

linear로 SVM 적용가능

$$\underset{\alpha_1,\ldots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)\right) \right)$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n \end{cases}$$

Non-Linear SVM

- **Formulation for non-linear boundary**

  5. Solution for the boundary

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w} \cdot \Phi(\mathbf{x}_k)$$

$$= y_k - \sum_{i=1}^{n} \alpha_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_k) \text{for any } \Phi(\mathbf{x}_k) \text{ such that } 0 < \alpha_k < C$$

Non-Linear SVM

- **Example**

$$D = \{(1,1,-1),(2,2,-1),(3,1,-1),(1,2,1),(3,2,1)\}$$

$$\Phi: (x_1, x_2) \rightarrow (x_1, x_2, x_1^2, x_2^2, x_1^3, x_2^3, x_1x_2, x_1x_2^2, x_1^2x_2)$$

$$\Phi(D) = \left\{ \begin{array}{l} (1,1,1,1,1,1,1,1,1,-1),(2,2,4,4,8,8,4,8,8,-1), \\ (3,1,9,1,27,1,3,2,9,-1),(1,2,1,4,1,8,2,4,2,\ 1), \\ (3,2,9,4,27,8,6,12,18,1) \end{array} \right\}$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \right) \right)$$

$$\text{subject to} \left\{ \begin{array}{l} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{array} \right.$$

Non-Linear SVM

- **Observation: Any severe difference??**
  - Original Formulation

$$D = \{(1,\ 1,\ -1), (2,\ 2,\ -1), (3,\ 1,\ -1), (1,\ 2,\ 1), (3,\ 2,\ 1)\}$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \right), \text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

  - After mapping into a higher space

$$\Phi(D) = \{(1,1,1,1,1,1,1,1,1,-1), (2,2,4,4,8,8,4,8,8,-1), (3,1,9,1,27,1,3,2,9,-1), (1,2,1,4,1,8,2,4,2,1), (3,2,9,4,27,8,6,12,18,1)\}$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) \right), \text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

Non-Linear SVM

- **Observation: Which one do we need?**
  - Mapping data into a higher space

$$\Phi: \mathbf{x} \rightarrow \Phi(\mathbf{x})$$

  - Inner products of mapped data

$$\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

$$\Phi(D) = \begin{Bmatrix} (1,1,1,1,1,1,1,1,1,-1), (2,2,4,4,8,8,4,8,8,-1), (3,1,9,1,27,1,3,2,9,-1), \\ (1,2,1,4,1,8,2,4,2,1), (3,2,9,4,27,8,6,12,18,1) \end{Bmatrix}$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\text{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \right) \right), \text{subject to } \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

Kernel Trick

2차원과 Trick

- **Another Transform**
  - Instead of

$$\Phi: (x_1, x_2) \rightarrow (x_1, x_2, x_1^2, x_2^2, x_1^3, x_2^3, x_1x_2, x_1x_2^2, x_1^2x_2)$$

  - What about this?

$$\Phi: (x_1, x_2) \rightarrow \left(1, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{3}x_1^2, \sqrt{3}x_2^2, x_1^3, x_2^3, \sqrt{6}x_1x_2, \sqrt{3}x_1x_2^2, \sqrt{3}x_1^2x_2\right)$$

    - Why??
    - Evaluate $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$

Kernel Trick

- **Another Transform**

$$\mathbf{x}_1 = (x_{11}, x_{12}),\ \mathbf{x}_2 = (x_{21}, x_{22})$$

Inner product of samples in high dimension space

$$\Phi(\mathbf{x}_1) = \left(1, \sqrt{3}x_{11}, \sqrt{3}x_{12}, \sqrt{3}x_{11}^2, \sqrt{3}x_{12}^2, x_{11}^3, x_{12}^3, \sqrt{6}x_{11}x_{12}, \sqrt{3}x_{11}x_{12}^2, \sqrt{3}x_{11}^2x_{12}\right)$$

$$\Phi(\mathbf{x}_2) = \left(1, \sqrt{3}x_{21}, \sqrt{3}x_{22}, \sqrt{3}x_{21}^2, \sqrt{3}x_{22}^2, x_{21}^3, x_{22}^3, \sqrt{6}x_{21}x_{22}, \sqrt{3}x_{21}x_{22}^2, \sqrt{3}x_{21}^2x_{22}\right)$$

$$\begin{aligned}\Phi(\mathbf{x}_1)\cdot\Phi(\mathbf{x}_2) &= 1 + 3x_{11}x_{21} + 3x_{12}x_{22} + 3x_{11}^2x_{21}^2 + 3x_{12}^2x_{22}^2 + x_{11}^3x_{21}^3 + x_{12}^3x_{22}^3 \\ &\quad + 6x_{11}x_{12}x_{21}x_{22} + 3x_{11}x_{12}^2x_{21}x_{22}^2 + 3x_{11}^2x_{12}x_{21}^2x_{22} \\ &= (x_{11}x_{21} + x_{12}x_{22})^3 + 3(x_{11}x_{21} + x_{12}x_{22})^2 \\ &\quad + 3(x_{11}x_{21} + x_{12}x_{22})^2 + 1 \\ &= ((x_{11}x_{21} + x_{12}x_{22}) + 1)^3 \\ &= (\mathbf{x}_1\cdot\mathbf{x}_2 + 1)^3\end{aligned}$$

Inner product of samples in low dimension space

What does it mean??

If a transform function is well designed, we can easily evaluate the inner products!!

Kernel Trick

- **Example**
  - Compare the computations!

$$\Phi(\mathbf{x}_1) = (1,\ \sqrt{3},\ \sqrt{3},\ \sqrt{3},\ \sqrt{3},\ 1,\ 1,\ \sqrt{6},\ \sqrt{3},\ \sqrt{3})$$
$$\Phi(\mathbf{x}_2) = (1,\ 2\sqrt{3},\ 2\sqrt{3},\ 4\sqrt{3},\ 4\sqrt{3},\ 8,\ 8,\ 4\sqrt{6},\ 8\sqrt{3},\ 8\sqrt{3})$$

1. mapping

$$\mathbf{x}_1 = (1,\ 1)$$
$$\mathbf{x}_2 = (2,\ 2)$$

2. Inner product

$$\Phi(\mathbf{x}_2) \cdot \Phi(\mathbf{x}_2) = 1 + 6 + 6 + 12 + 12 + 8 + 8 + 24 + 24 + 24 = 125$$

위의 2가지 단계 = (using K) 1단계와 같다

$$K(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1 \cdot \mathbf{x}_2 + 1)^3 = (4 + 1)^3 = 125$$

Kernel Trick

- **Another Transform**
  - Instead of solving this

$$D = \{(x_{11}, x_{12}, y_1), \cdots, (x_{n1}, x_{n2}, , y_n)\} \quad \text{where } y_i \in \{-1, +1\}$$

$$\Phi: (x_1, x_2) \rightarrow \left(1, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{3}x_1^2, \sqrt{3}x_2^2, x_1^3, x_2^3, \sqrt{6}x_1x_2, \sqrt{3}x_1x_2^2, \sqrt{3}x_1^2x_2\right)$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\text{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)\right) \right) \text{,subject to } \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

2단계로

  - Solve this. It saves your times!!

$$D = \{(x_{11}, x_{12}, y_1), \cdots, (x_{n1}, x_{n2}, y_n)\} \quad \text{where } y_i \in \{-1, +1\}$$

$$\underset{\alpha_1,\cdots,\alpha_n}{\text{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i \cdot \mathbf{x}_j + 1\right)^3 \right) \text{,subject to } \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

1단계로

Kernel Trick

- **Kernel**
  - A function that corresponds to a dot product of two feature vectors in some expanded feature space
  - You have two functions $\Phi(\mathbf{x})$ and $K(\mathbf{x}, \mathbf{y})$ and it happens that $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y}) = K(\mathbf{x}, \mathbf{y})$ Then, K is called a kernel function

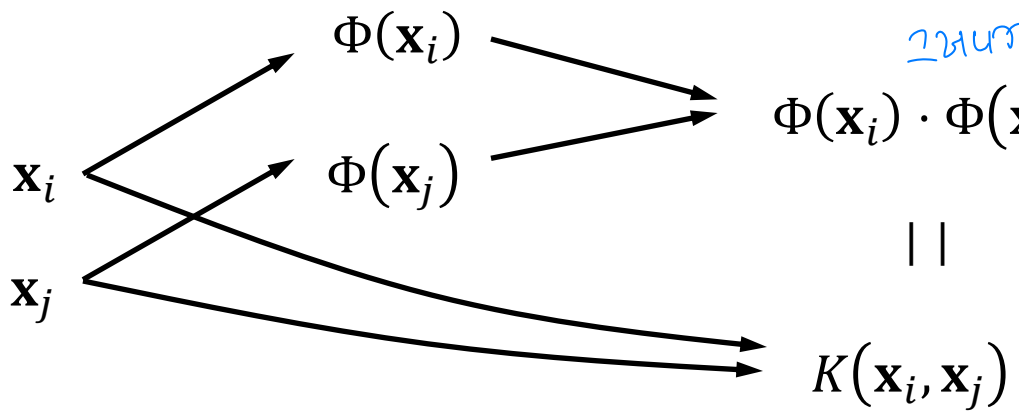
SVM에서 매핑함수가 필요한게 아니라 들어내적함이 필요함

모든함수가 되는것은 아님

기존 매핑함수로 매핑할수있는것만

(there exist a function 또)

그러니까 매핑함수값 = 커널함수값인 매핑함수가 존재해야 커널함수라고 할 수있다.



Kernel Trick

- **Final Formulation of SVM**
  - For given $D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_1, y_n)\}$ where $y_i \in \{-1, +1\}$
    Choose C and K, and Solve

$$\underset{\alpha_1,\cdots,\alpha_n}{\text{argmax}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \right)$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad i = 1, \cdots, n \end{cases}$$

  - Solution for the boundary

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w} \cdot \Phi(\mathbf{x}_k)$$

$$= y_k - \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_k) \text{ for any } k \text{ such that } 0 < \alpha_k < C$$

For unknown x

$$\text{Class of } \mathbf{x} = \begin{cases} +1 & \text{if } y(\mathbf{x}) \ge 0 \\ -1 & \text{if } y(\mathbf{x}) < 0 \end{cases}$$

$$\text{where } y(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Some More on Kernels

- **Examples of commonly-used kernel functions:**
  - Linear Kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$$

  - Homogeneous Polynomial Kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$$

  - Polynomial Kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^d$$

  - Gaussian Kernel (RBF Kernel)

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

  - Sigmoid Kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha \mathbf{x}_i \cdot \mathbf{x}_j + b)$$

Some More on Kernels

- **Comments on Some Kernels**
  - Homogeneous Polynomial Kernel

$$K(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d \qquad \Phi(\mathbf{x}) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right) \qquad \Phi(\mathbf{y}) = \left(y_1^2, \sqrt{2}y_1y_2, y_2^2\right)$$

  - RBF Kernel

$$\begin{aligned} K(\mathbf{x},\mathbf{y}) &= \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2}\right) \\ &= \exp\left(-\frac{(x_1-y_1)^2+(x_2-y_2)^2}{2}\right) \\ &= \exp\left(-\frac{x_1^2-2x_1y_1+y_1^2+x_2^2-2x_2y_2+y_2^2}{2}\right) \\ &= \exp\left(-\frac{\|x\|^2}{2}\right)\exp\left(-\frac{\|y\|^2}{2}\right)\exp(\mathbf{x}\cdot\mathbf{y}) \\ &= \exp\left(-\frac{\|x\|^2}{2}\right)\exp\left(-\frac{\|y\|^2}{2}\right)\sum_{n=0}^{\infty}\frac{(\mathbf{x}\cdot\mathbf{y})^n}{n!} \end{aligned}$$

Discussion

- **Pros:**
  - Find the optimal separation hyperplane.
  - Can deal with very high dimensional data.
  - Usually work very well.
- **Cons:**
  - Require both positive and negative examples.
  - Need to select a good kernel function.
  - Require lots of memory and CPU time.
  - There are some numerical stability problems in solving the constrained QP