Pattern Recognition

SVM 개념 잡기

Yukyung Choi

yk.choi@rcv.sejong.ac.kr

What is SVM?

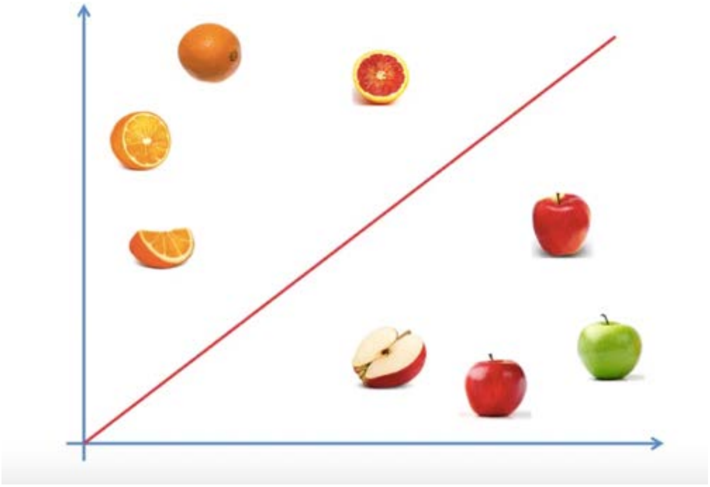
- **S**upport **V**ector **M**achine ➔ SVM
- Traditional Classifier
- Until now, favorite classifier to everyone
    - Wondering why? **Kernel Trick!!!**

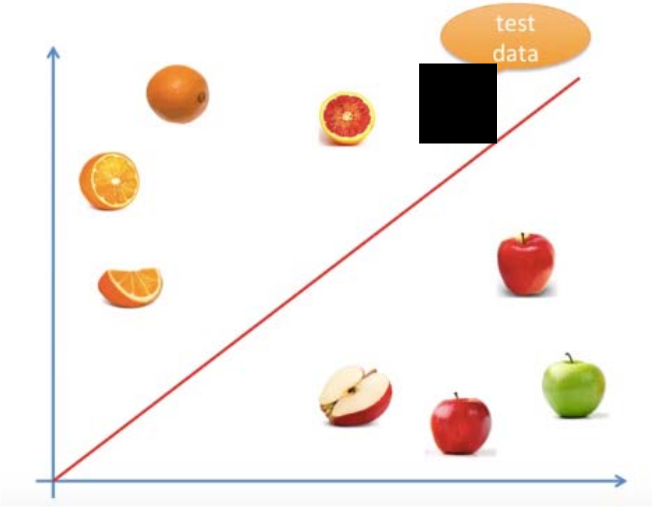
**"만약, 문제에 어떠한 알고리즘을 사용할지 모르겠다면,
SVM은 좋은 출발선이 될 수 있음"**

Classifier

- **Classifier** is a hypothesis or discrete-valued function that is used to **assign (categorical) class labels to particular data points**.

- In the email classification example, this classifier could be a hypothesis for labeling emails as **spam** or **non-spam**.

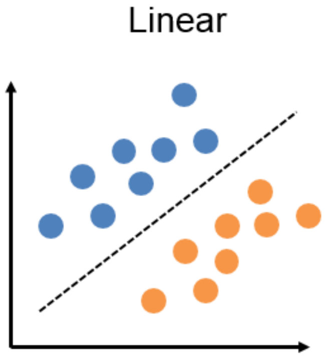


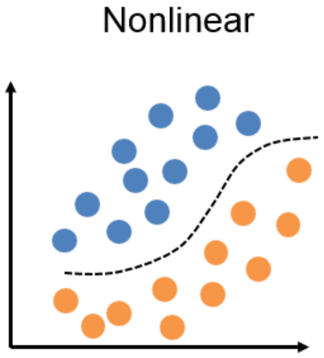


Classifier

- y = label, x = data, y = f(x), f: classifier

- If **decision function** is linear, this classifier (f) is linear classifier
- If not, this classifier (f) is non-linear classifier


Linear


Nonlinear

y = f(x)

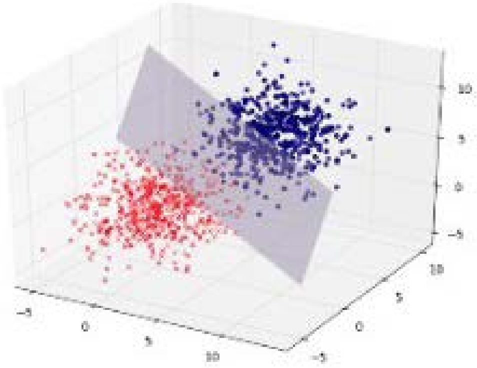
데이터를 구획해주는 이 **점선의 함수 (decision boundary)**를 우리는 **판별 함수 (decision function)**라 부른다.

Classifier

- **Hyperplane**
  - In geometry, a hyperplane is a subspace whose dimension is one less than that of its ambient space. If a space is **3-dimensional** then its hyperplanes are the **2-dimensional planes**, while if the space is **2-dimensional**, its hyperplanes are the **1-dimensional lines**.

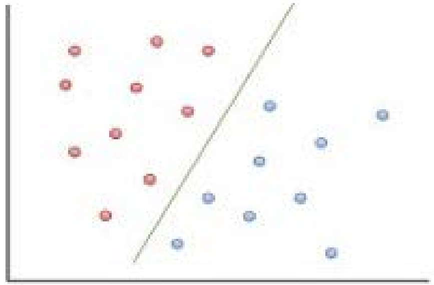
$$\mathbf{w}^T\mathbf{x} = 0$$

Hyperplane



$$y = ax + b$$

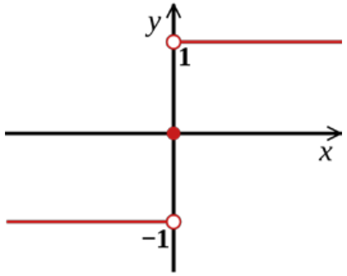
Line



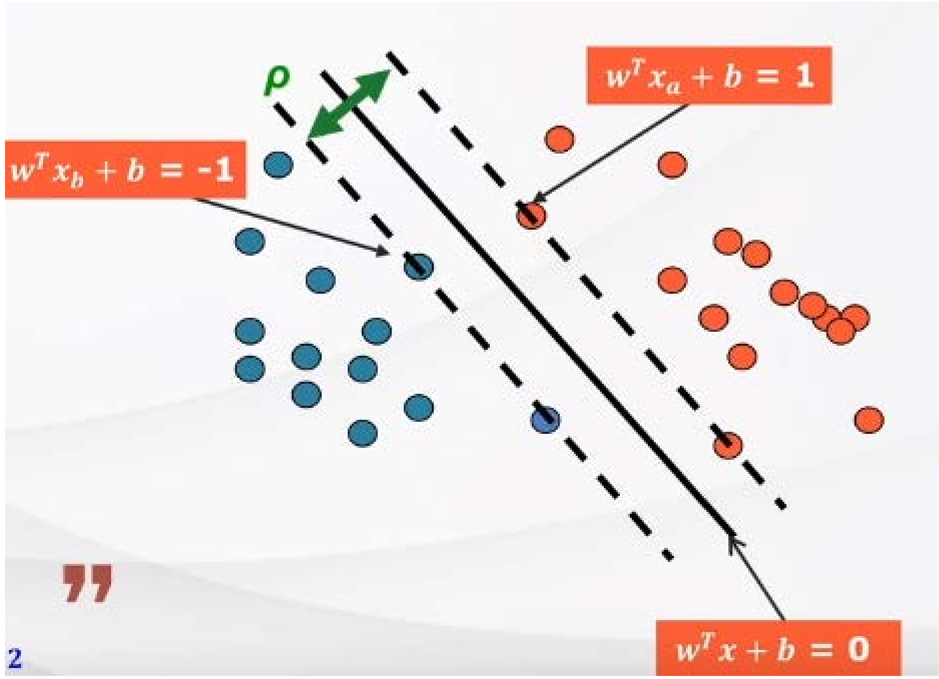
SVM Classifier

- W : vector for hyperplane
- x_i : i_{th} data, y_i : label (class) of i_{th} data

- $\mathbf{Y} = \mathbf{sign}(W^T\mathbf{X}+b) = f(\mathbf{X})$
  - $Y_i = +1$ when $W^T\mathbf{X}_i+b > 1$
  - $Y_i = -1$ when $W^T\mathbf{X}_i+b < -1$

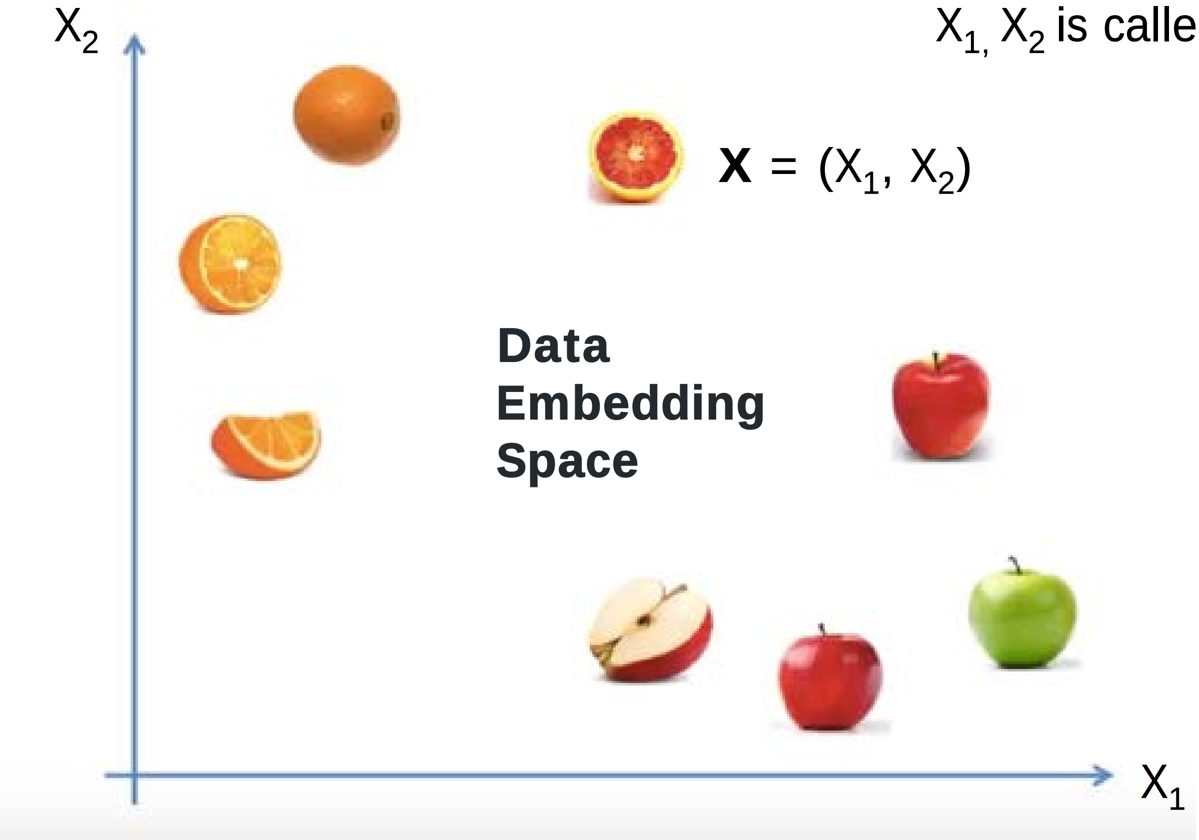


Sign function



Apple, orange classifier

- **Data Embedding Space**



X_2

$\mathbf{X} = (X_1, X_2)$

Data Embedding Space

X_1

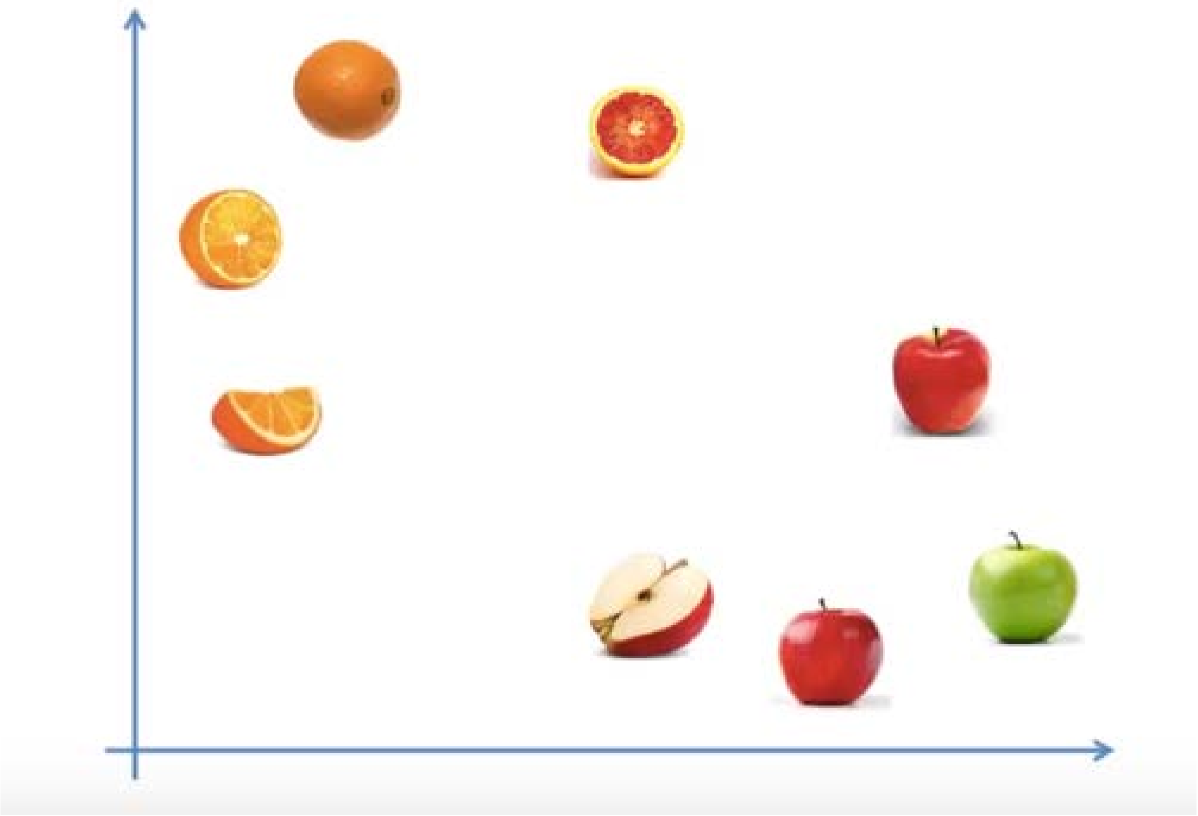
$X_{1,}$ X_2 is called feature or attribute.

Data Embedding
범주형 자료를 **벡터 형태**로 바꾸는 것

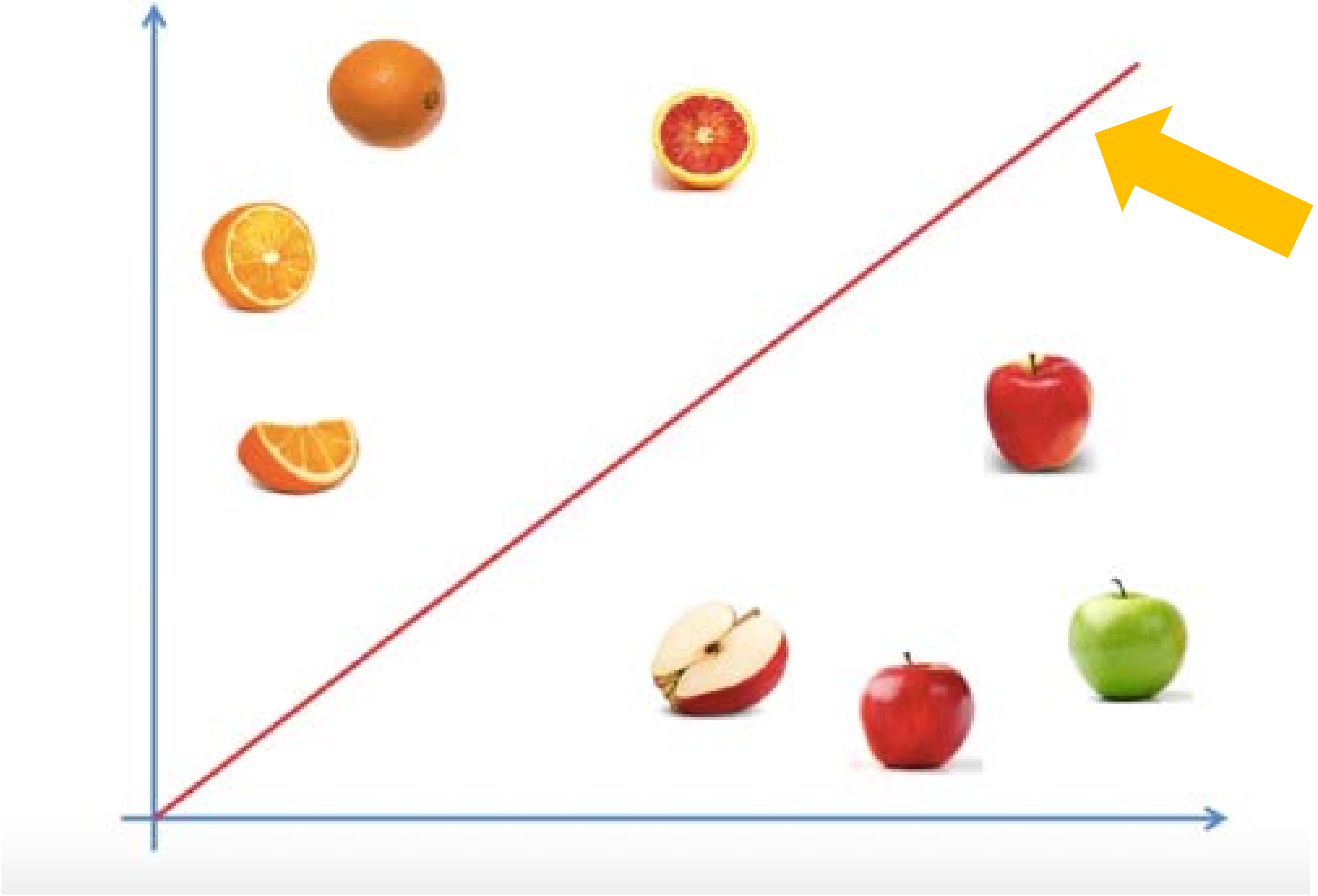
Categorical Data
범주형 데이터란 몇 개의 범주로 나누어진 데이터 예) 남/여, A/B/O/AB

Apple, orange classifier

- Which hyperplane can we choose?

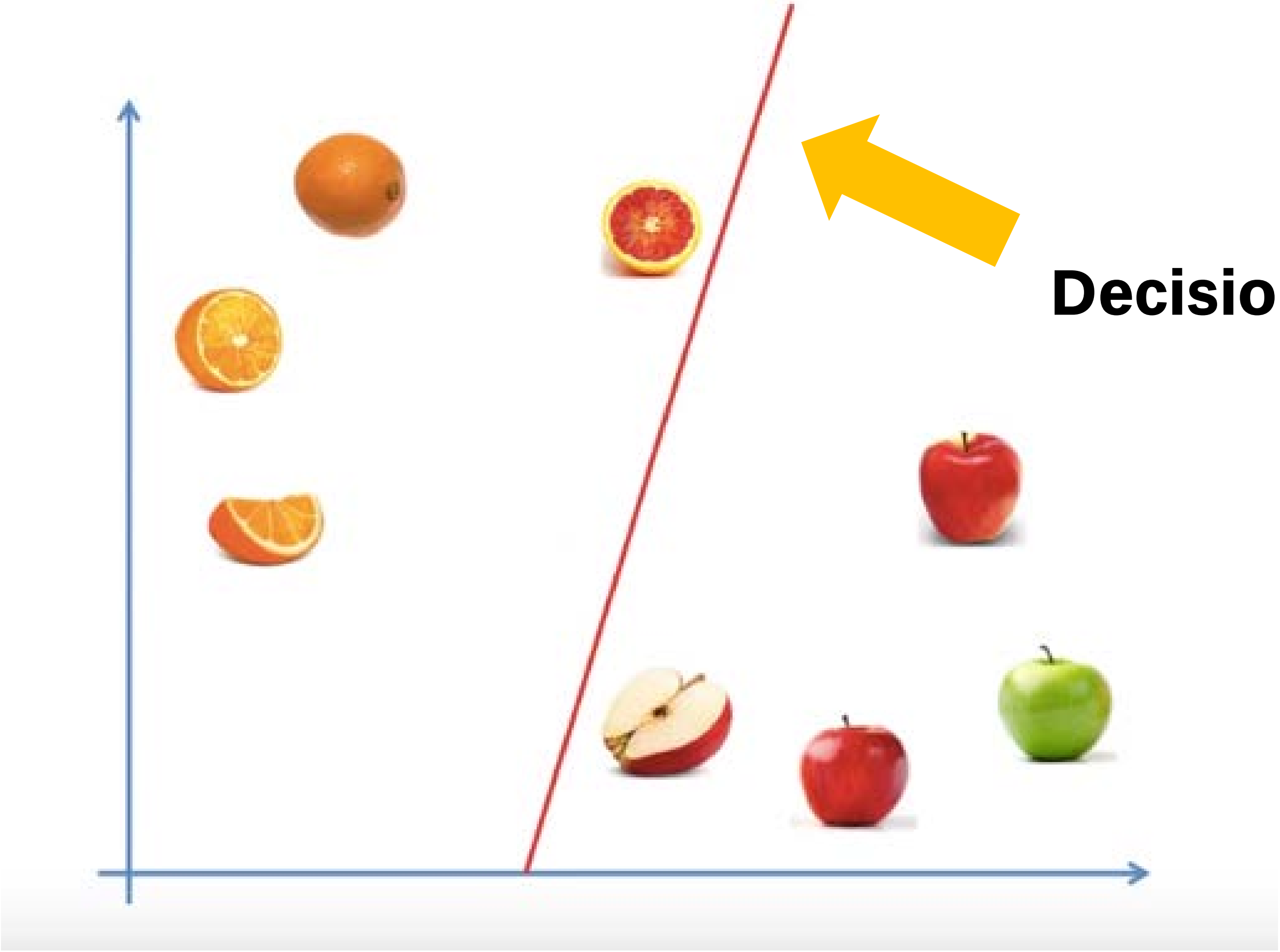


Apple, orange classifier



Decision Boundary

Apple, orange classifier

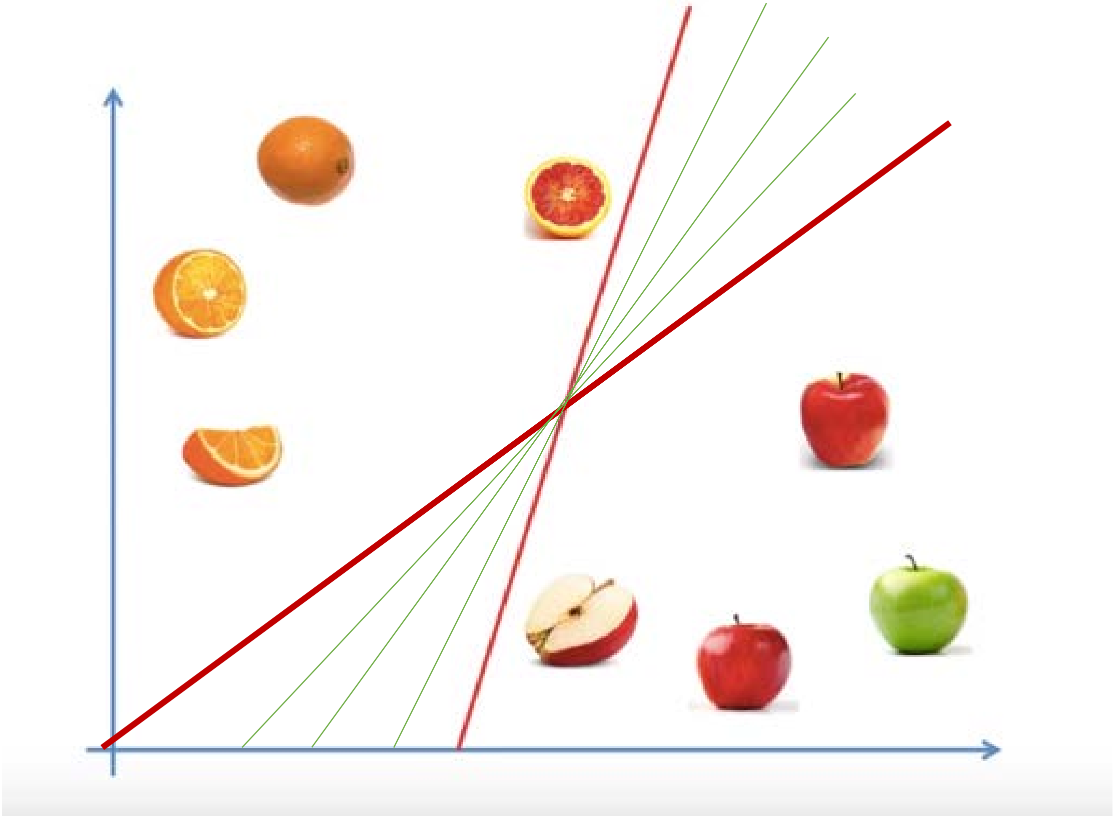


Decision Boundary

Apple, orange classifier

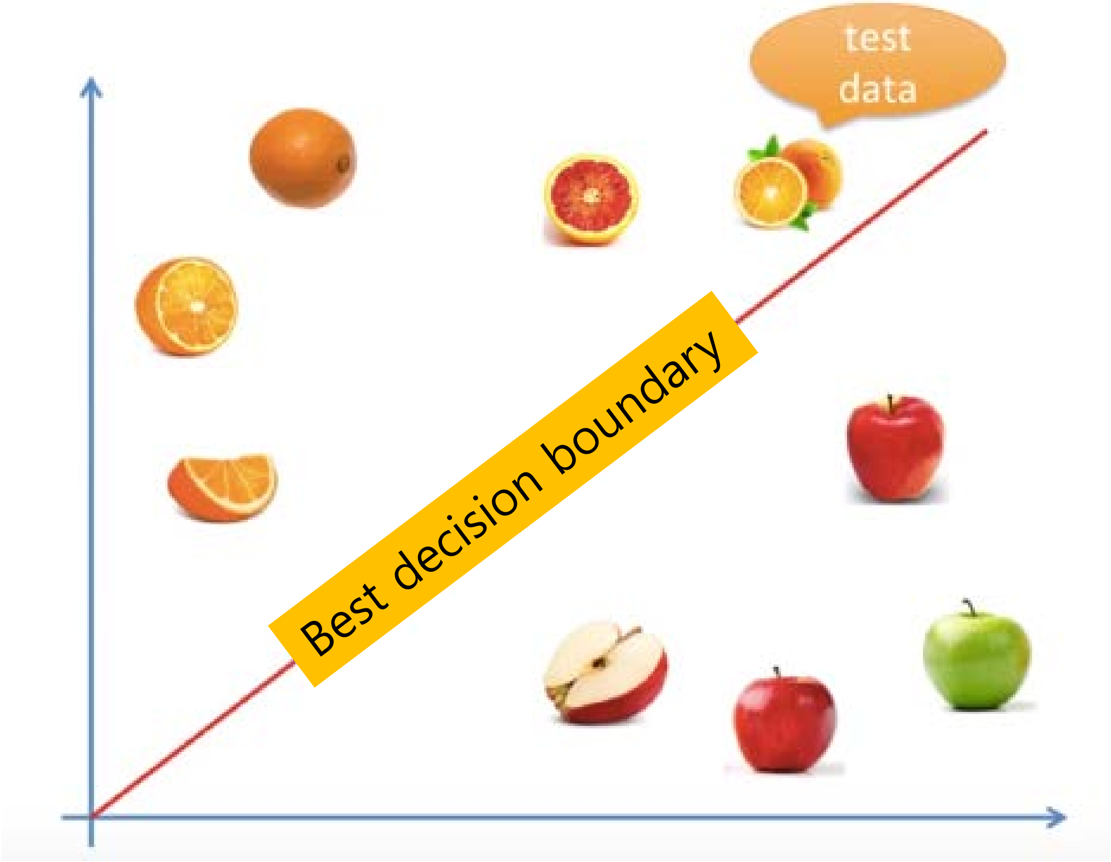
- Which one is better?
  - Classifier should have dealt with **unseen data**

Train sample data ➔ seen data
Test sample data ➔ unseen data



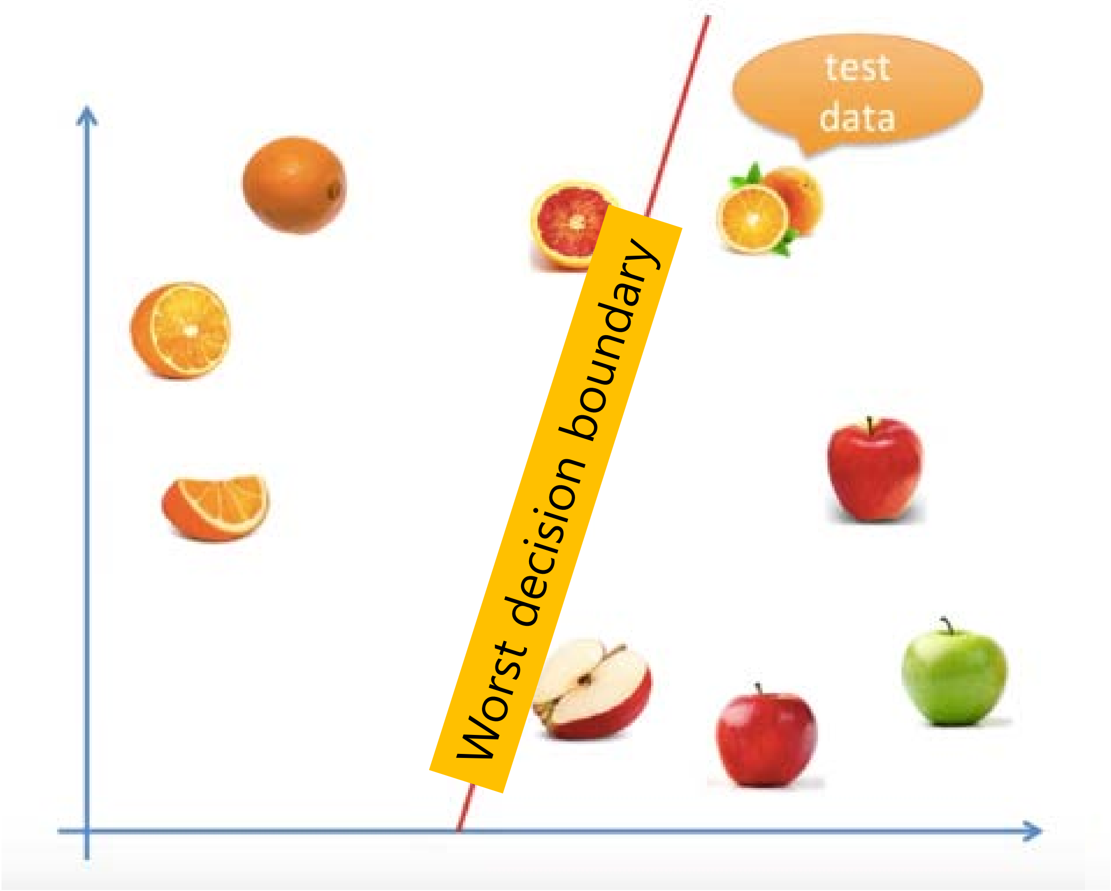
How can we decide decision boundary?

- Test data predicted well (O)



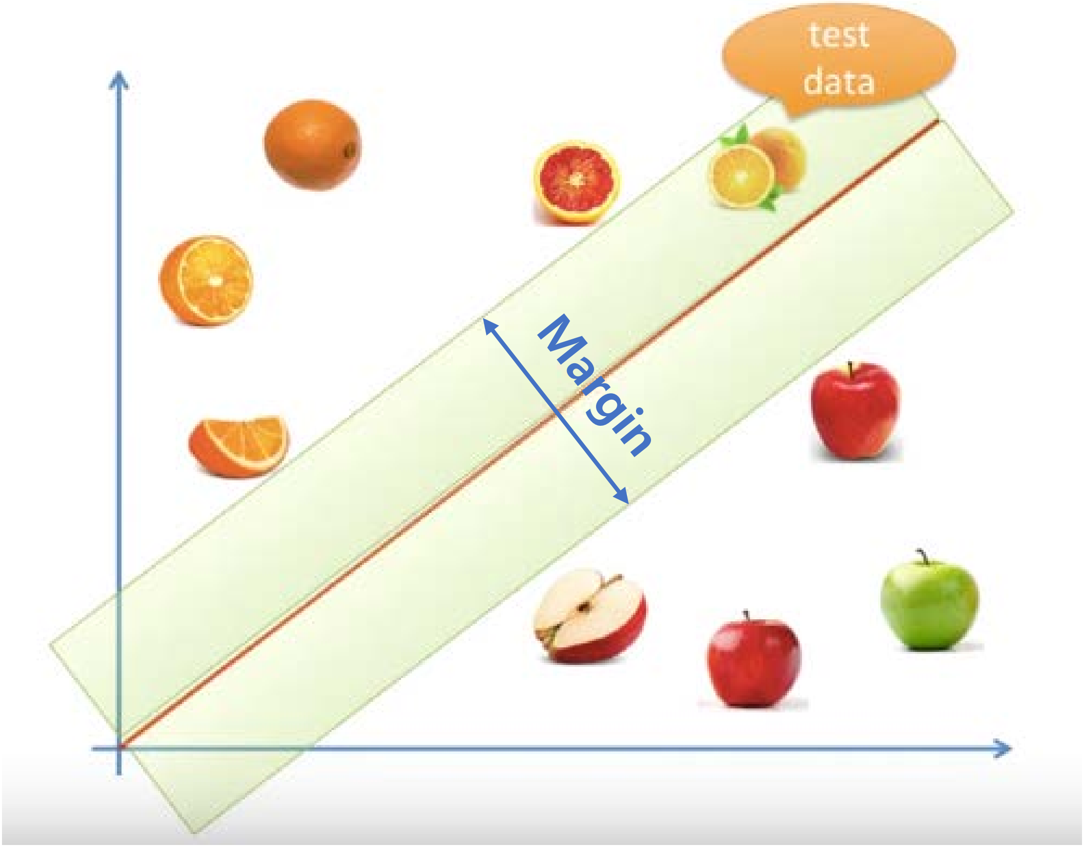
How can we decide decision boundary?

- Test data predicted well (X)



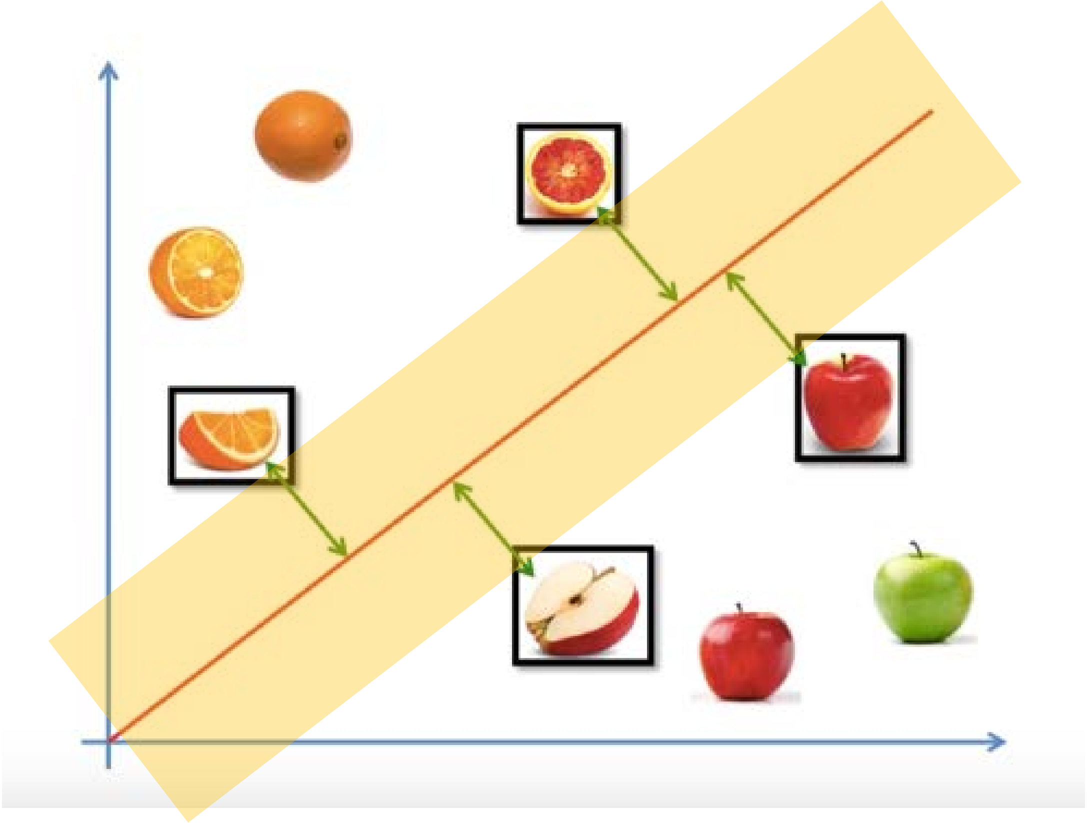
How can we decide decision boundary?

- The answer is “Large Margin”!!



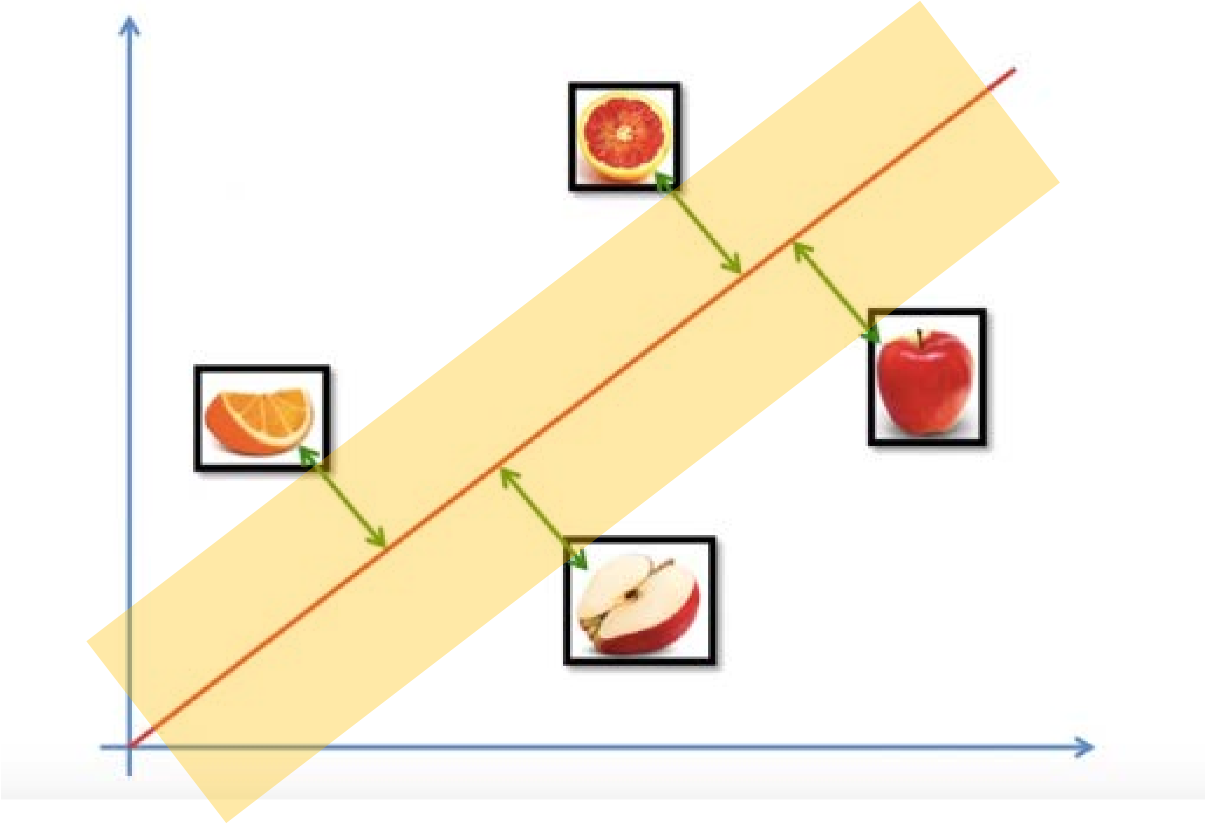
Support Vector

- **Support Vector**
  - Samples on the margin are called the support vectors.

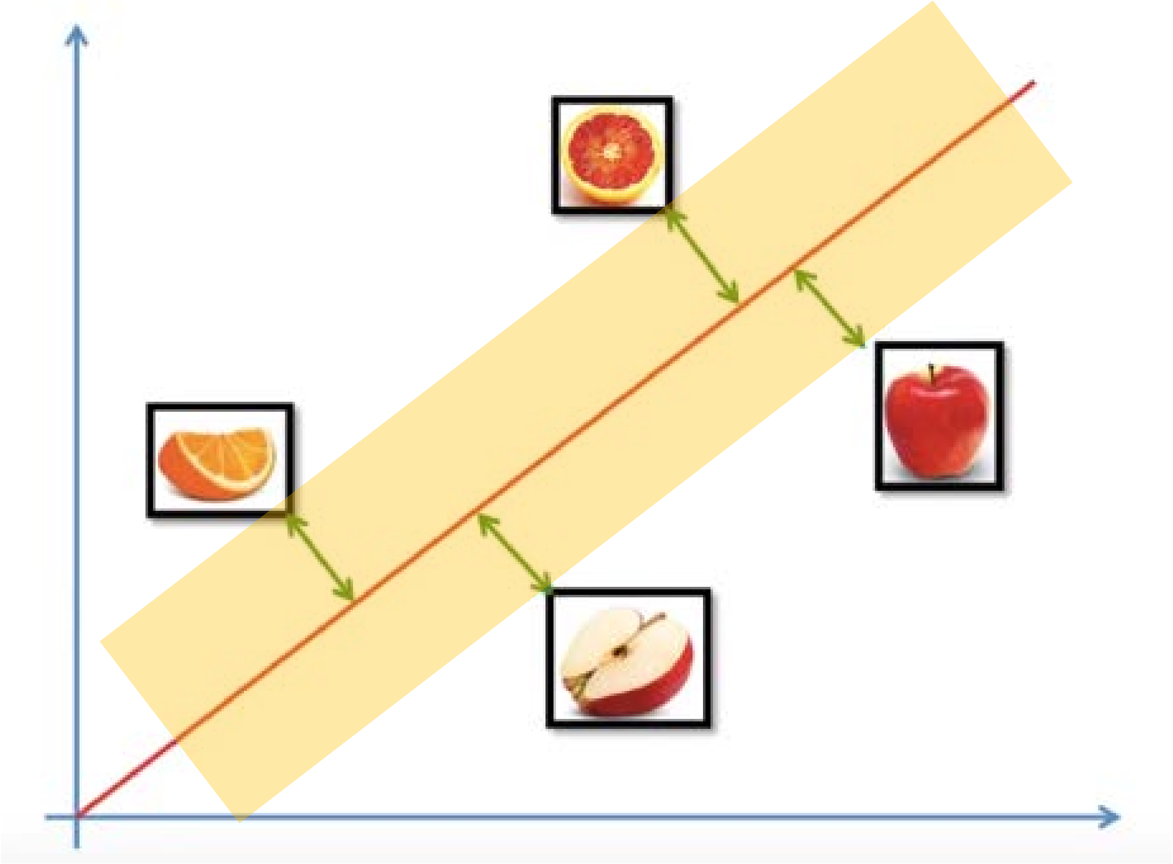


Support Vector

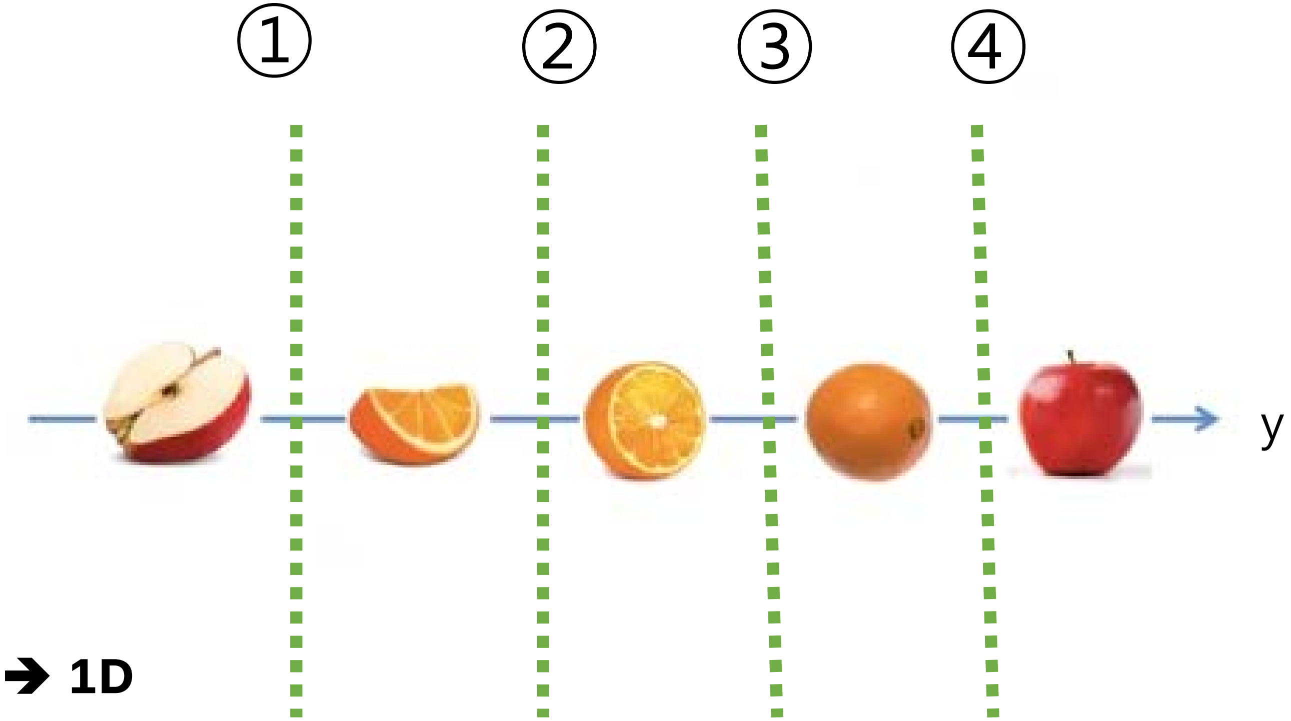
- SVM only uses support vector for prediction
  - Less computation!!!



Linearly Separable or not



What if data is not linearly separable?

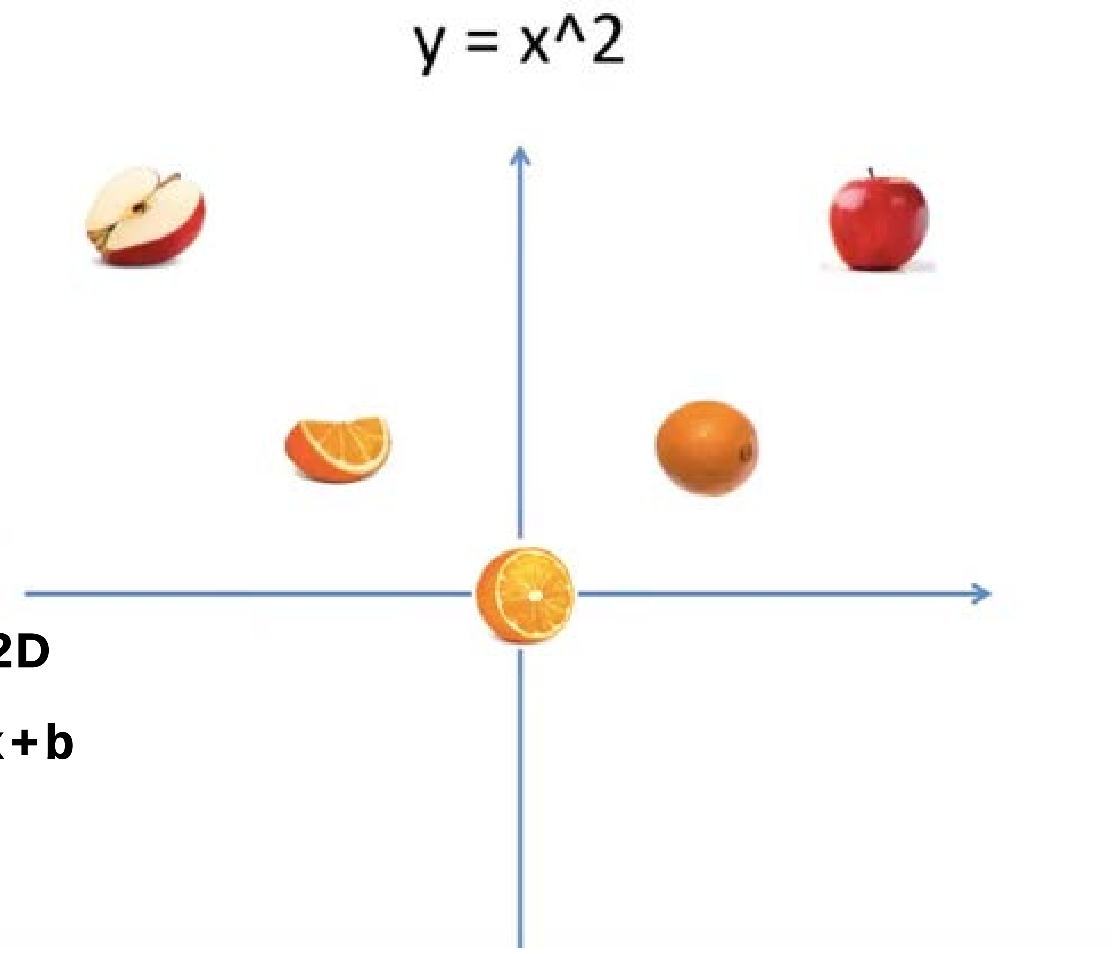


Data Embedding Space ➔ 1D

Hyperplane ➔ 0D ➔ y = b

What if data is not linearly separable?

- Mapping lower dimension to high dimension

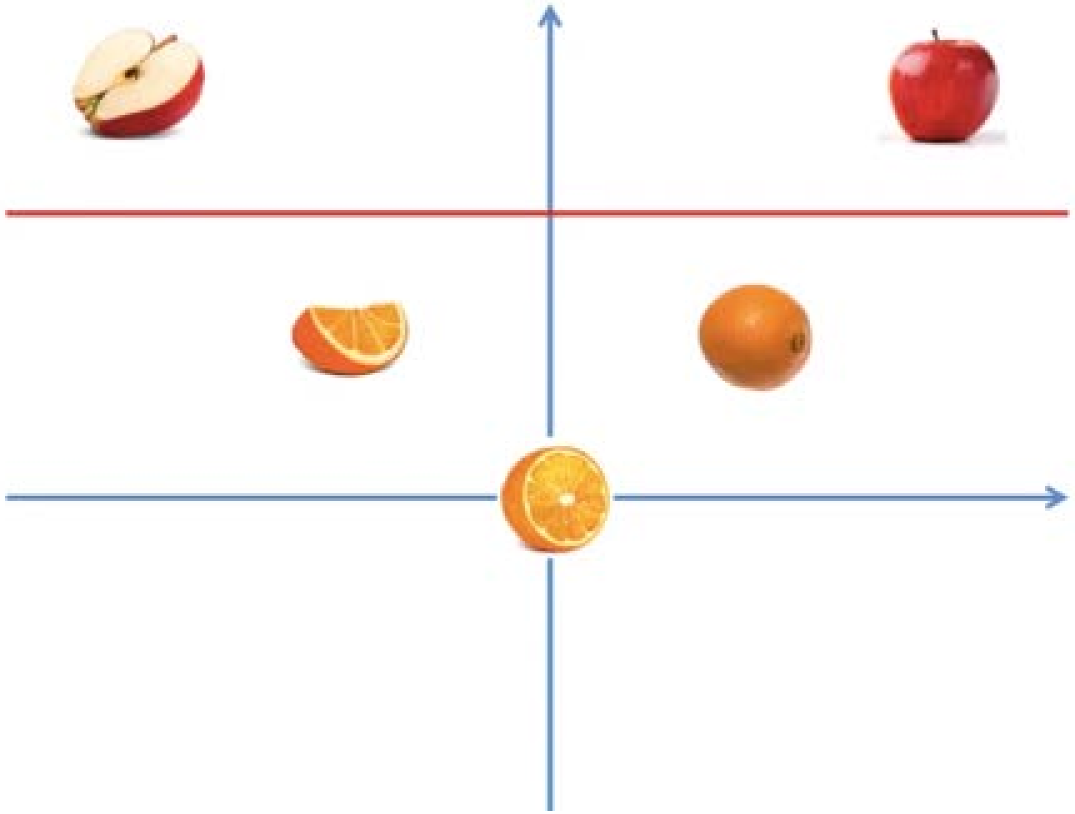


Data Embedding Space ➔ 2D

Hyperplane ➔ 1D ➔ y = ax+b

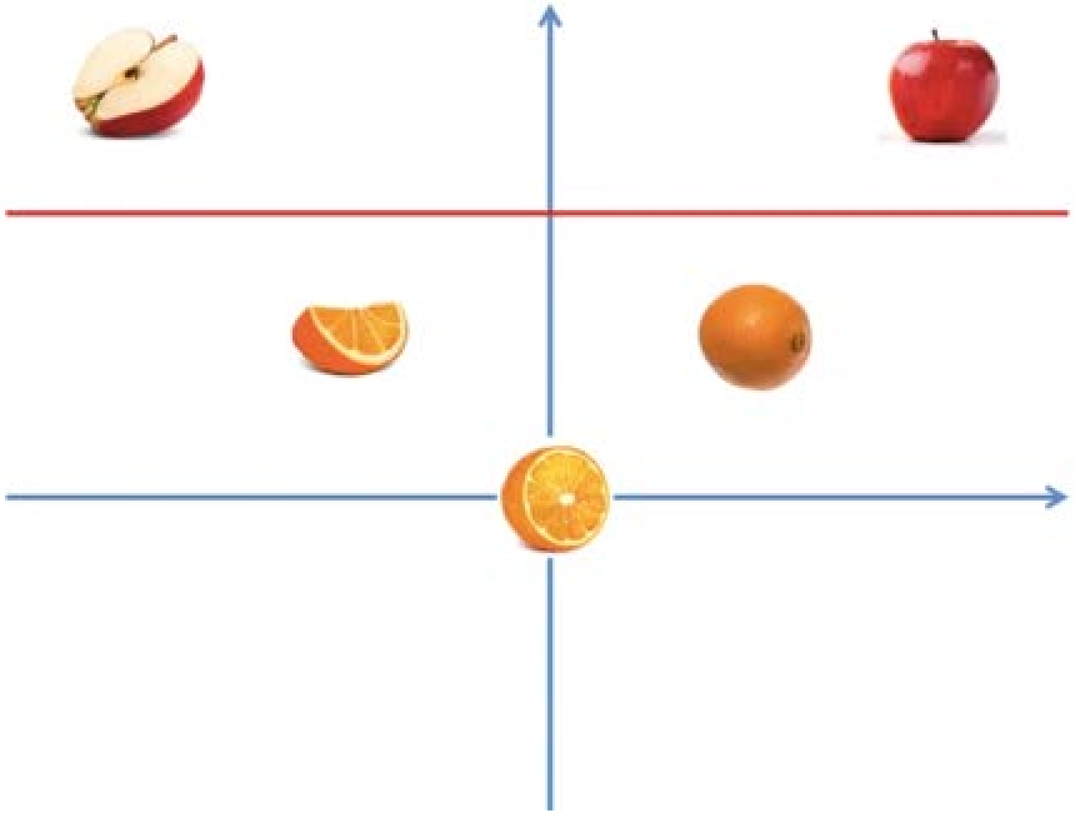
What if data is not linearly separable?

- Now it is linearly separable in higher dimension
  - Mapping to high dimension requires **much computation!**



What if data is not linearly separable?

- **Kernel trick** in SVM do this without explicitly
  - Move data point to higher dimension with **low computation!**



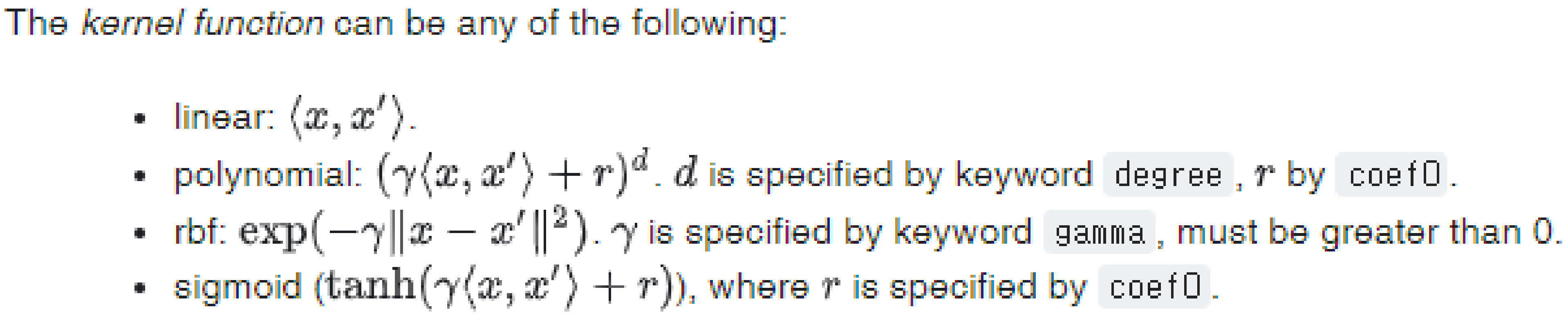
Kernel Trick

- The **kernel trick** avoids the explicit mapping that is needed to get linear learning algorithms.

- **Kernel methods** owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by **simply computing the inner products** between the images of all pairs of data in the feature space

Kernel Trick

- Kernel Function ➔ **simply computing the inner products**



The *kernel function* can be any of the following:

- linear: $\langle x, x'\rangle$.
- polynomial: $(\gamma\langle x, x'\rangle + r)^d$. d is specified by keyword `degree`, r by `coef0`.
- rbf: $\exp(-\gamma\|x - x'\|^2)$. γ is specified by keyword `gamma`, must be greater than 0.
- sigmoid $(\tanh(\gamma\langle x, x'\rangle + r))$, where r is specified by `coef0`.

Mapping 함수의 inner-product.. Mapping (m➔n)

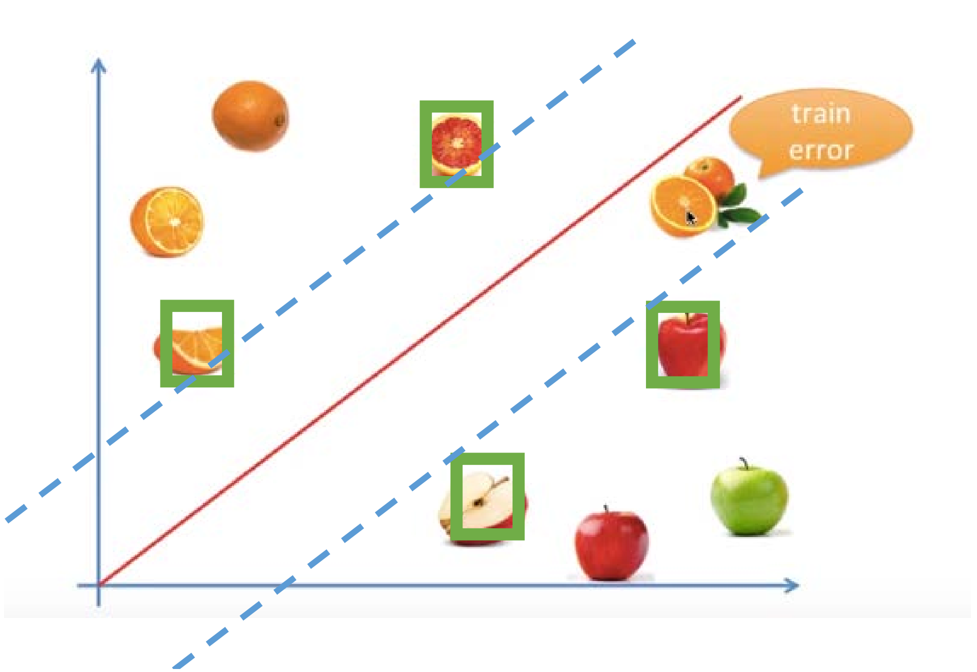
$$K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j) = x_i^T A^T A x_j$$

https://scikit-learn.org/stable/modules/svm.html

SVM Parameter - Cost

Manually hand-tuned

- Cost is small == Margin is large



$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \quad \text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, i = 1, \ldots, n$$

Margin width misclassification

C is small

Training error is allowed

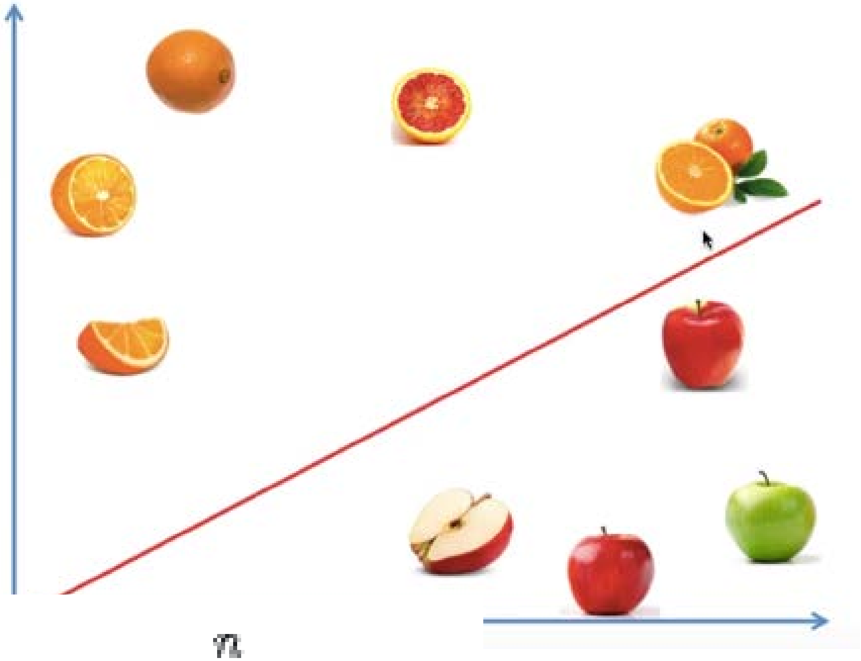
Overfitting is not allowed

Margin is large

Testing error is small

SVM Parameter - Cost

- Cost is large == Margin is small

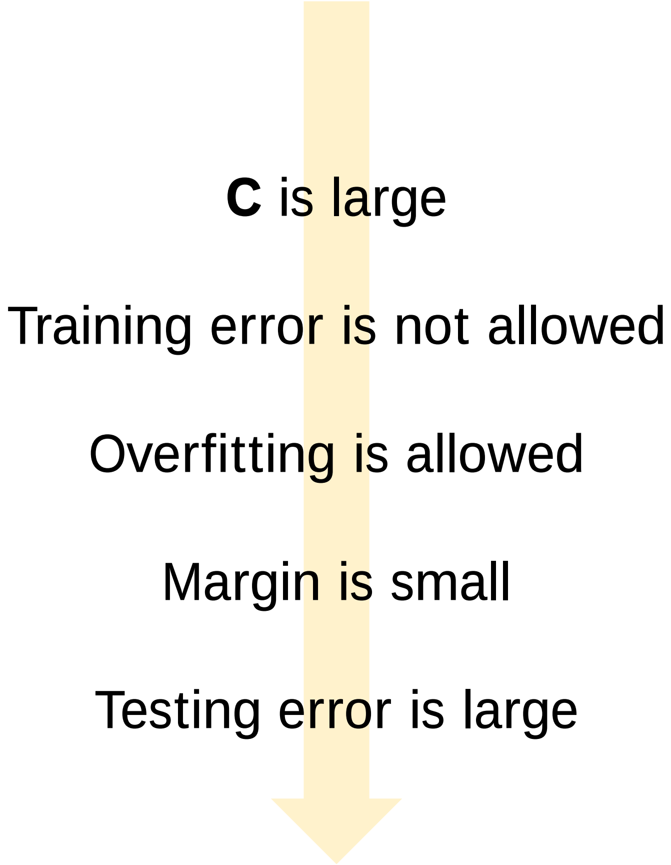


$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$$

$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i,$$
$$\zeta_i \geq 0, i = 1, \ldots, n$$

Margin width

misclassification



C is large

Training error is not allowed

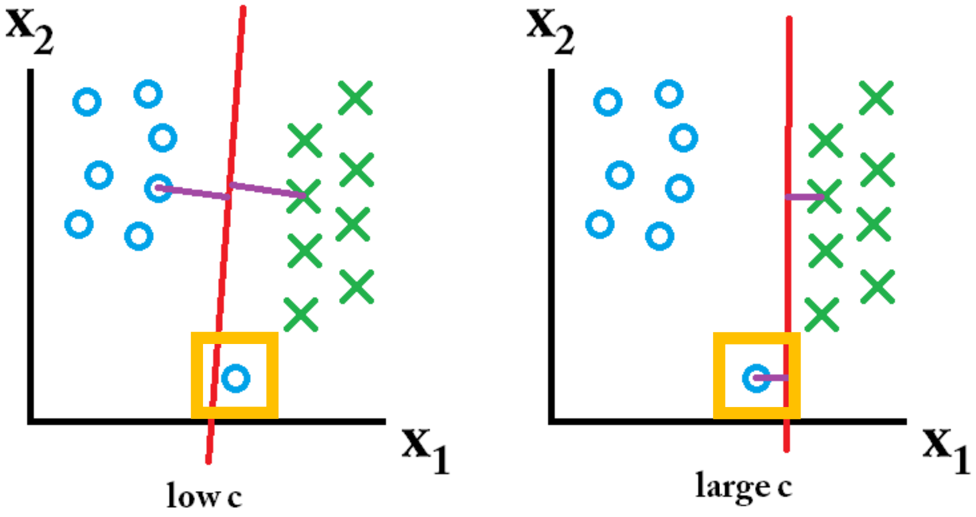
Overfitting is allowed

Margin is small

Testing error is large

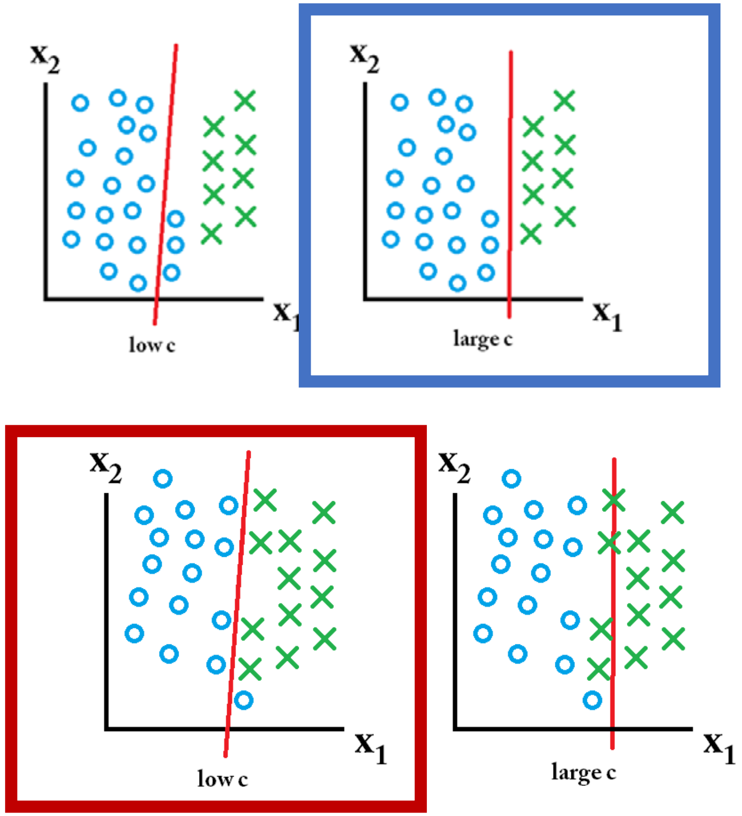
SVM Parameter - Cost

- We **assume** that some samples caused by train error are the **outlier**.
- Therefore, we generally select a **large margin** for decision boundary.
- But, if not?



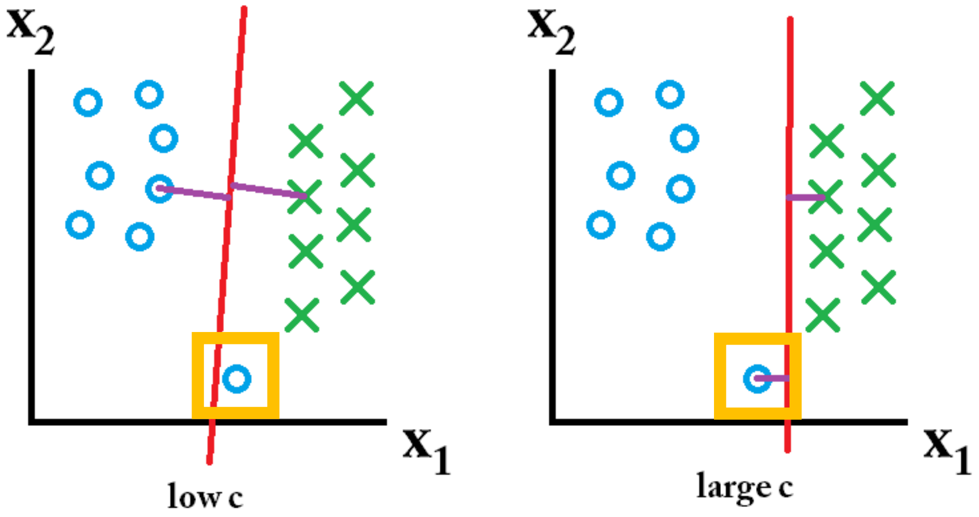
Inlier or outlier

?

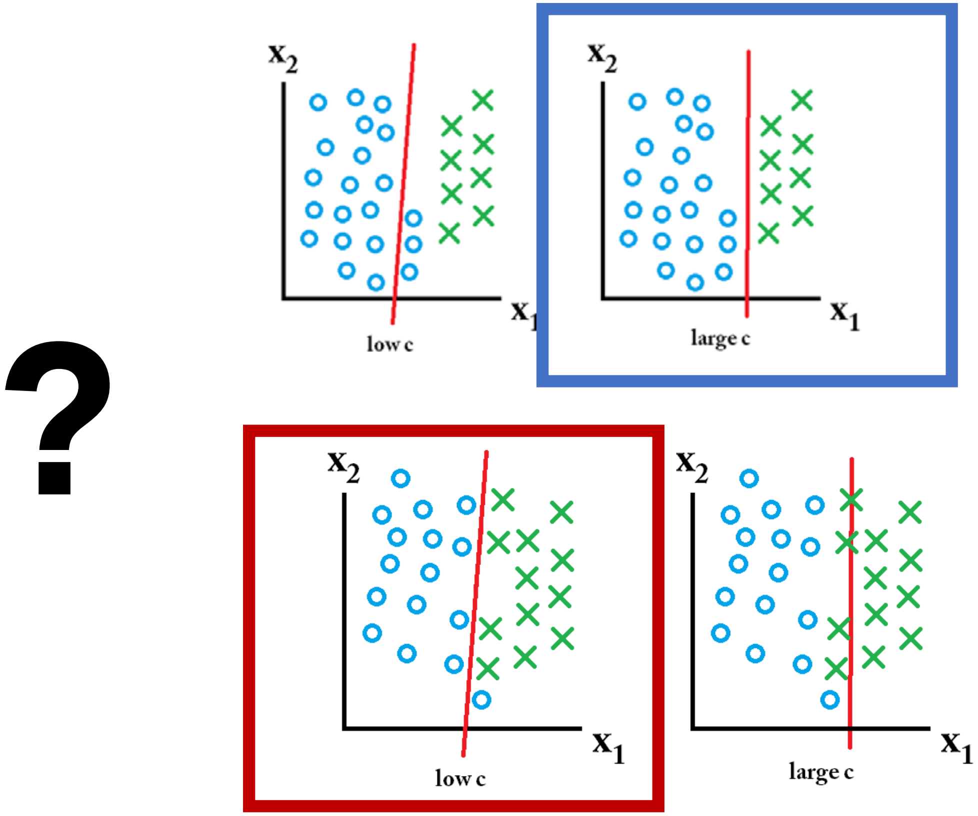


SVM Parameter - Cost

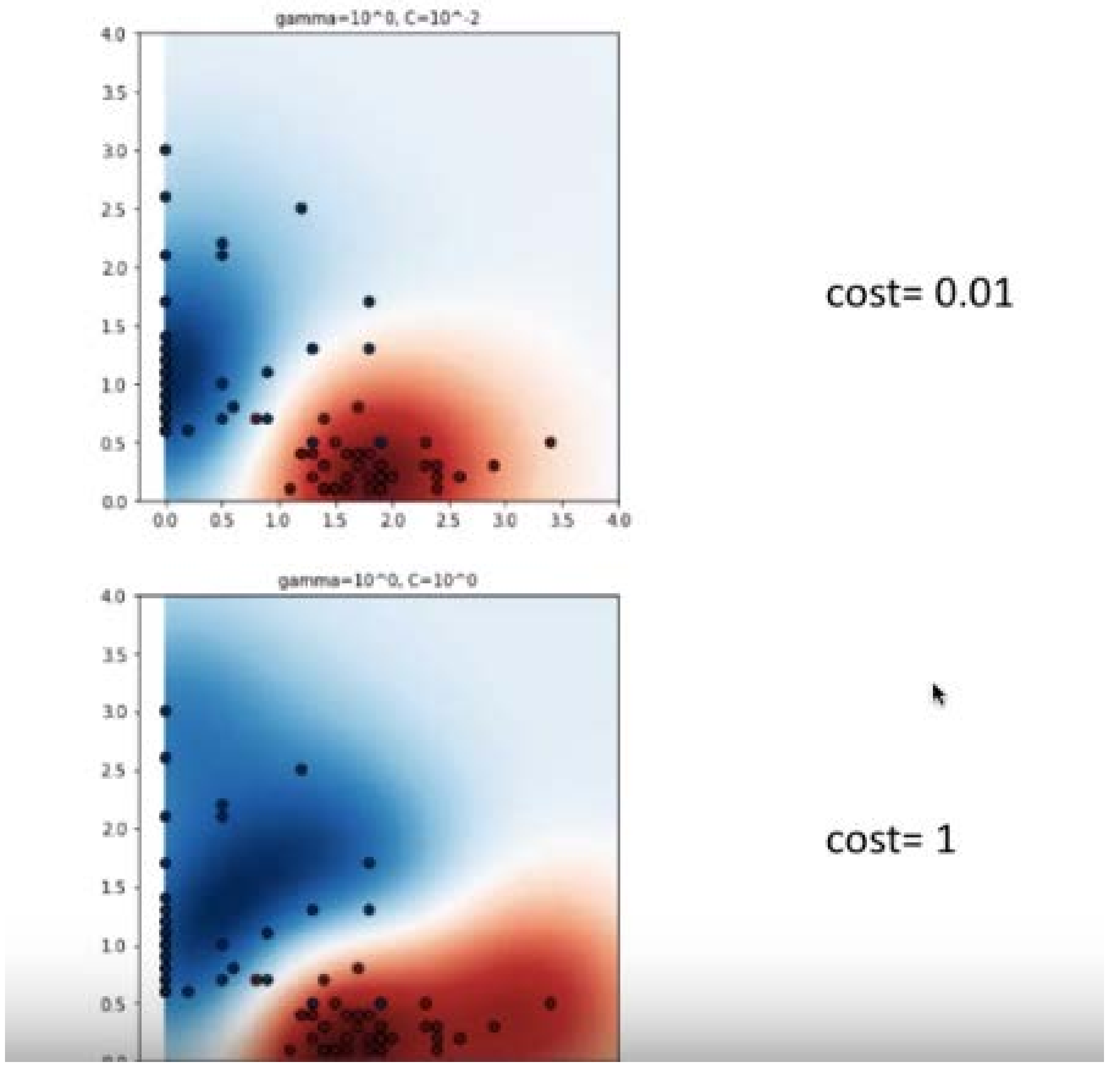
- Therefore, we cannot argue that we should choose large C, but we must make a decision through **data analysis**.



Inlier or outlier



SVM Parameter - Cost



Cost is small
Training error is allowed
Overfitting is not allowed
Decision boundary is simple

Cost is large
Training error is allowed
Overfitting is allowed
Decision boundary is complex

SVM parameter – Gamma in RBF kernel

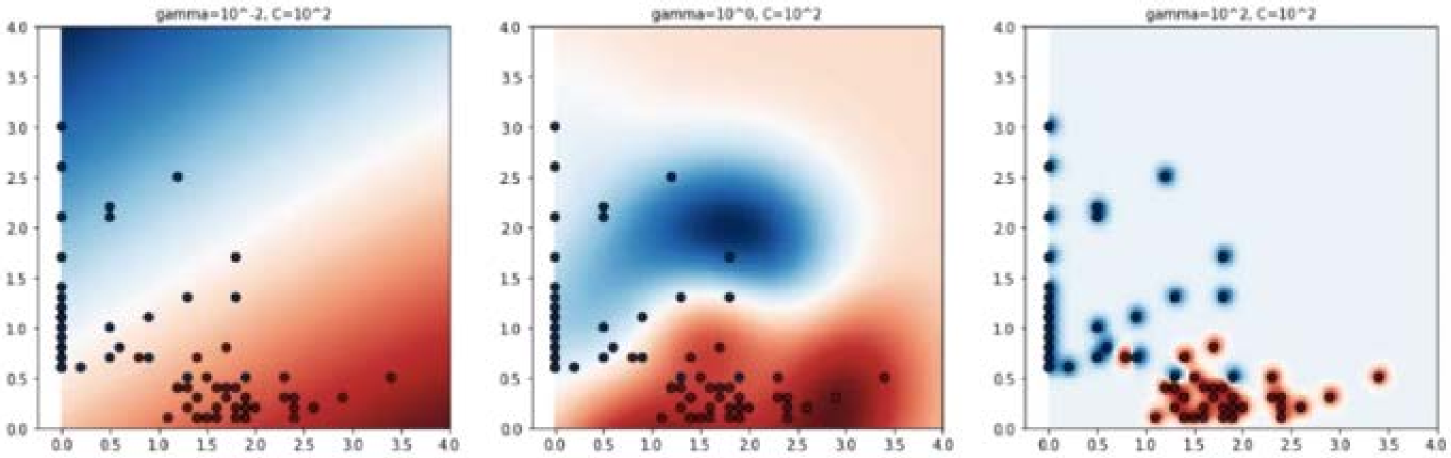
- Intuitively, the **gamma parameter** defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'.

Radial Base Function (also called Gaussian Kernel)

```
K(x,x') = exp(-gamma * ||x-x'||^2)
```

Far

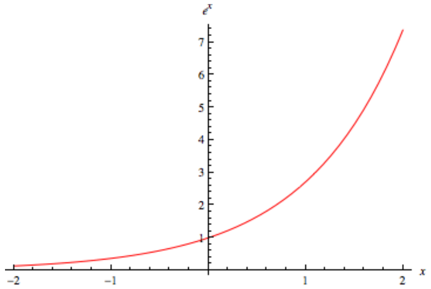
Close



gamma = 0.01

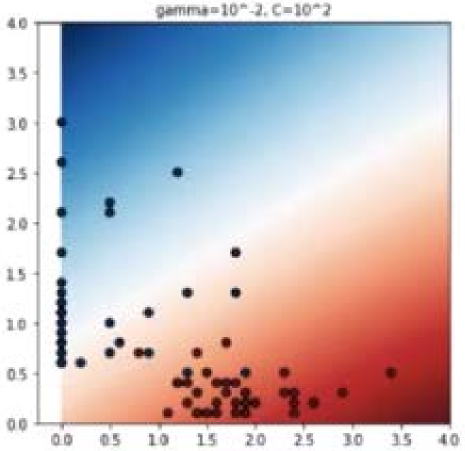
gamma = 1

gamma = 100

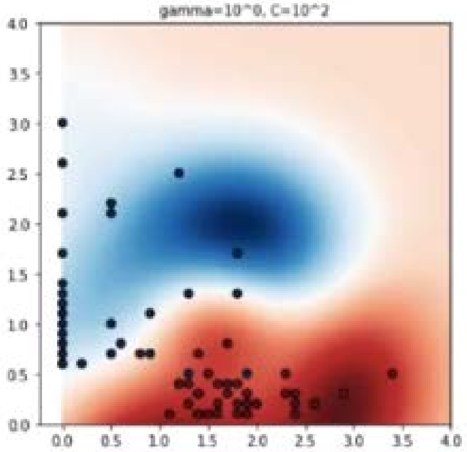


SVM parameter – Gamma in RBF kernel

Far

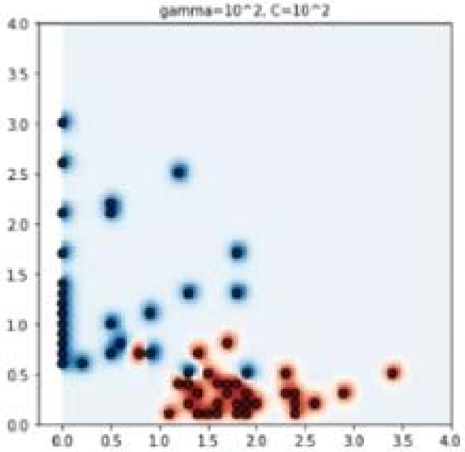


gamma = 0.01



gamma = 1

Close



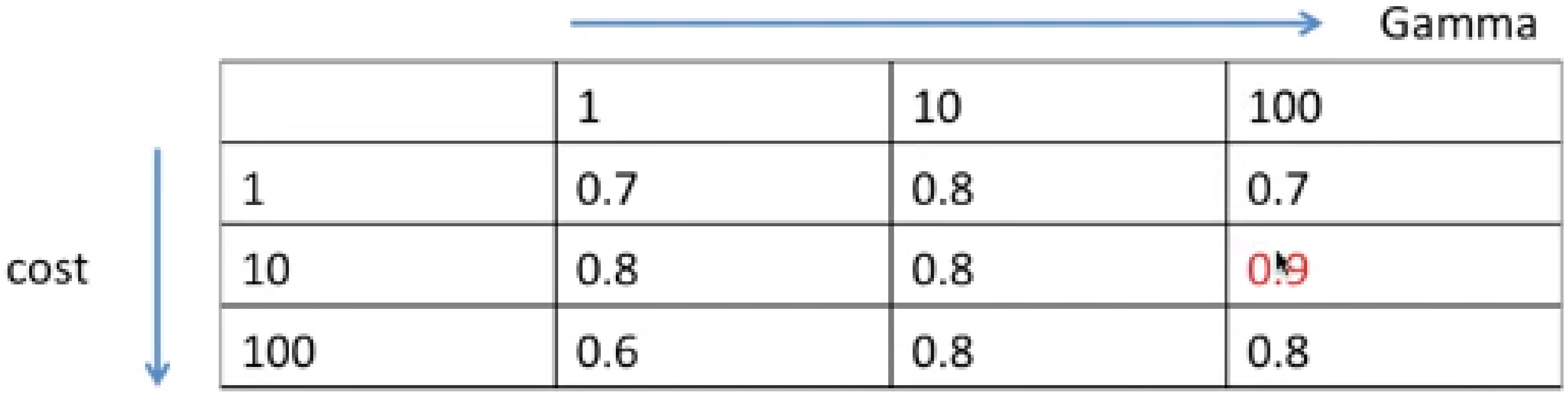
gamma = 100

Gamma is small
Influence is large
Margin is large
Similarly to a linear model

Gamma is large
Influence is small
Margin is small
Overfitting is allowed

Find optical parameter – **data analysis**

- **Grid Search**
  - **Grid search** builds a model for **every combination** of hyper-parameters specified and evaluates each model.



	1	10	100
1	0.7	0.8	0.7
10	0.8	0.8	0.9
100	0.6	0.8	0.8

Binary Classification

- One vs One

- One vs Rest

Multiple Classification

- How to extend binary to multiple classifier

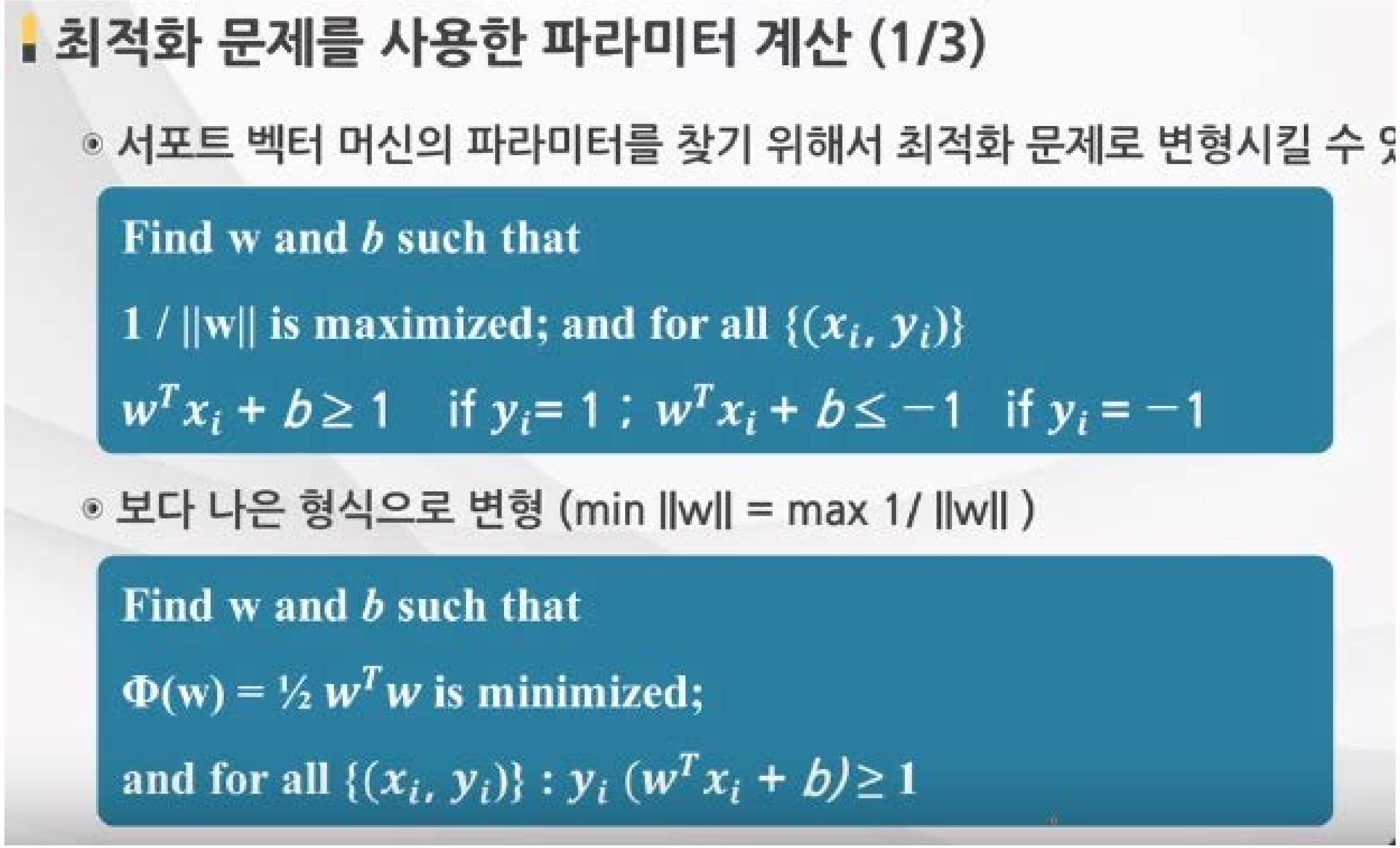
Unbalanced problems

- Sklearn: class_weight

How to design custom kernel

- https://scikit-learn.org/stable/auto_examples/svm/plot_custom_kernel.html#sphx-glr-auto-examples-svm-plot-custom-kernel-py

SVM Optimization



최적화 문제를 사용한 파라미터 계산 (1/3)

◉ 서포트 벡터 머신의 파라미터를 찾기 위해서 최적화 문제로 변형시킬 수 있

Find w and *b* such that

1 / ||w|| is maximized; and for all $\{(x_i, y_i)\}$

$w^T x_i + b \geq 1$ if $y_i = 1$; $w^T x_i + b \leq -1$ if $y_i = -1$

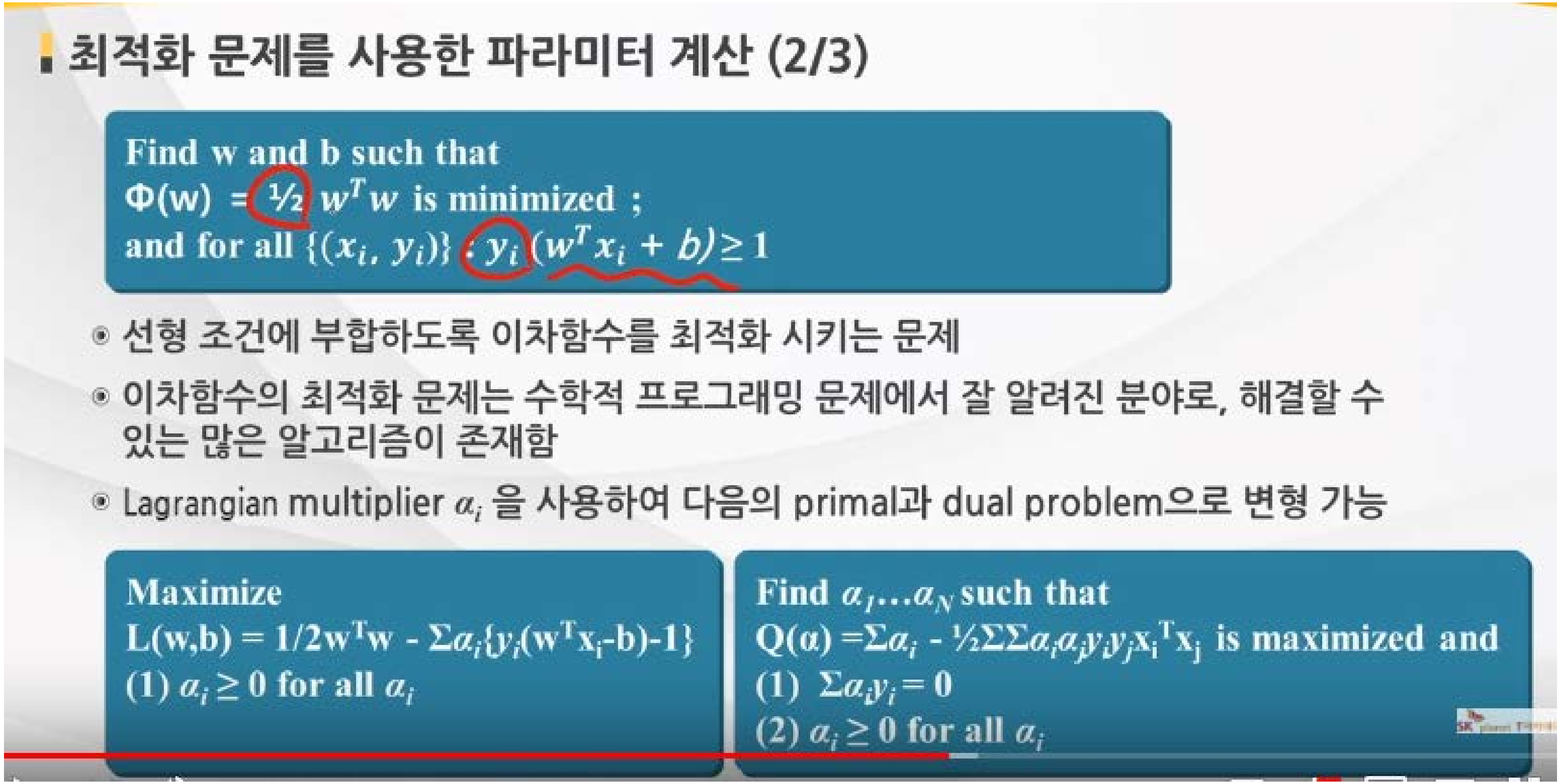
◉ 보다 나은 형식으로 변형 (min ||w|| = max 1/ ||w||)

Find w and *b* such that

$\Phi(\mathrm{w}) = ½\, w^T w$ is minimized;

and for all $\{(x_i, y_i)\}$: $y_i\,(w^T x_i + b) \geq 1$

https://www.youtube.com/watch?v=POqtUhBhiP8



최적화 문제를 사용한 파라미터 계산 (2/3)

Find w and b such that

$\Phi(w) = \boxed{½}\, w^T w$ is minimized ;

and for all $\{(x_i, y_i)\}$: $\boxed{y_i}\,\underline{(w^T x_i + b)} \geq 1$

- 선형 조건에 부합하도록 이차함수를 최적화 시키는 문제
- 이차함수의 최적화 문제는 수학적 프로그래밍 문제에서 잘 알려진 분야로, 해결할 수 있는 많은 알고리즘이 존재함
- Lagrangian multiplier α_i 을 사용하여 다음의 primal과 dual problem으로 변형 가능

Maximize

$L(w,b) = 1/2w^Tw - \Sigma\alpha_i\{y_i(w^Tx_i-b)-1\}$

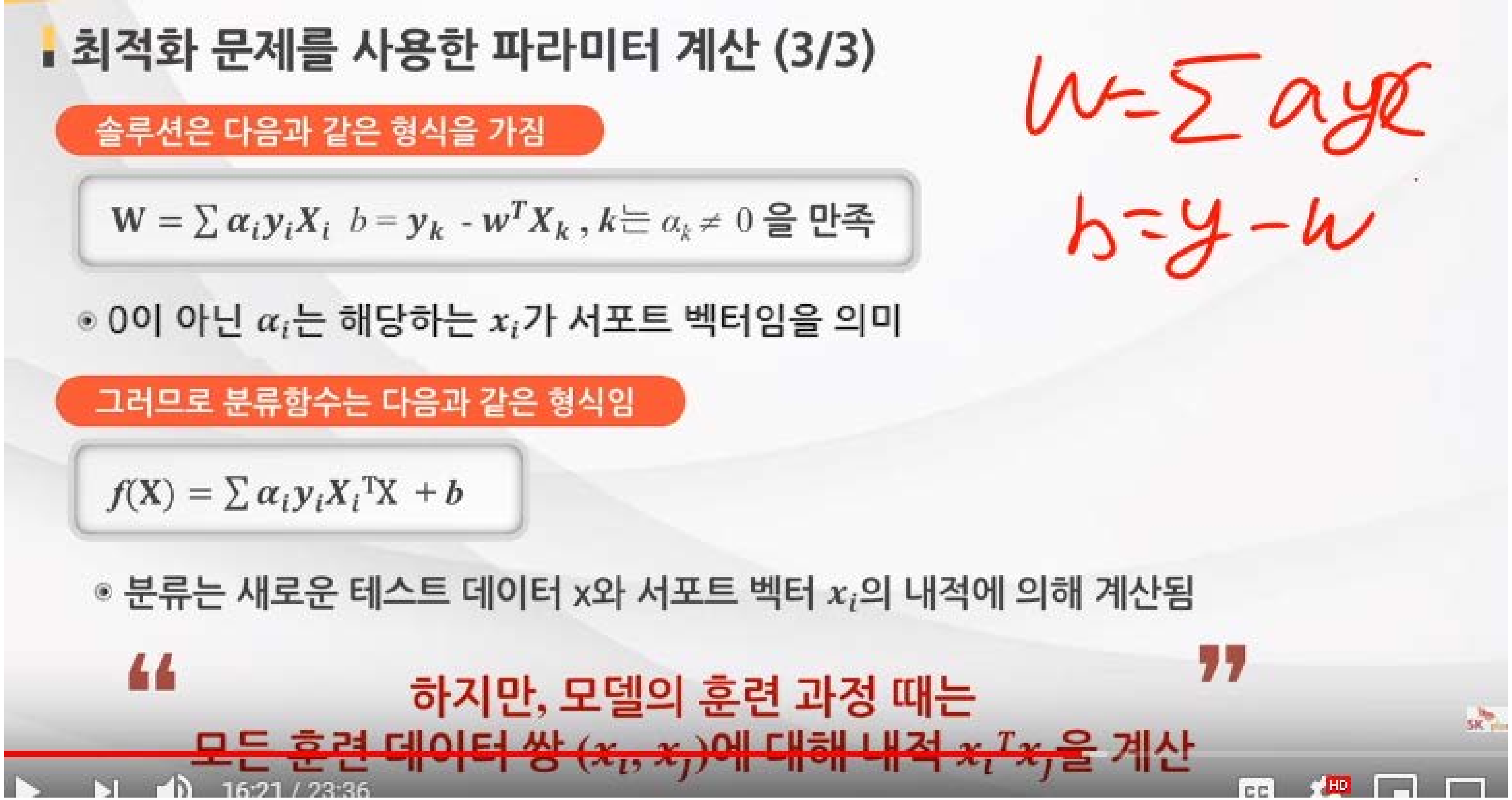
(1) $\alpha_i \geq 0$ for all α_i

Find $\alpha_1 \ldots \alpha_N$ such that

$Q(\alpha) = \Sigma\alpha_i - ½\Sigma\Sigma\alpha_i\alpha_j y_i y_j x_i^T x_j$ is maximized and

(1) $\Sigma\alpha_i y_i = 0$

(2) $\alpha_i \geq 0$ for all α_i



최적화 문제를 사용한 파라미터 계산 (3/3)

솔루션은 다음과 같은 형식을 가짐

$W = \sum \alpha_i y_i X_i \quad b = y_k - w^T X_k$, k는 $\alpha_k \neq 0$ 을 만족

$w = \Sigma\, a y x$

$b = y - w$

- 0이 아닌 α_i는 해당하는 x_i가 서포트 벡터임을 의미

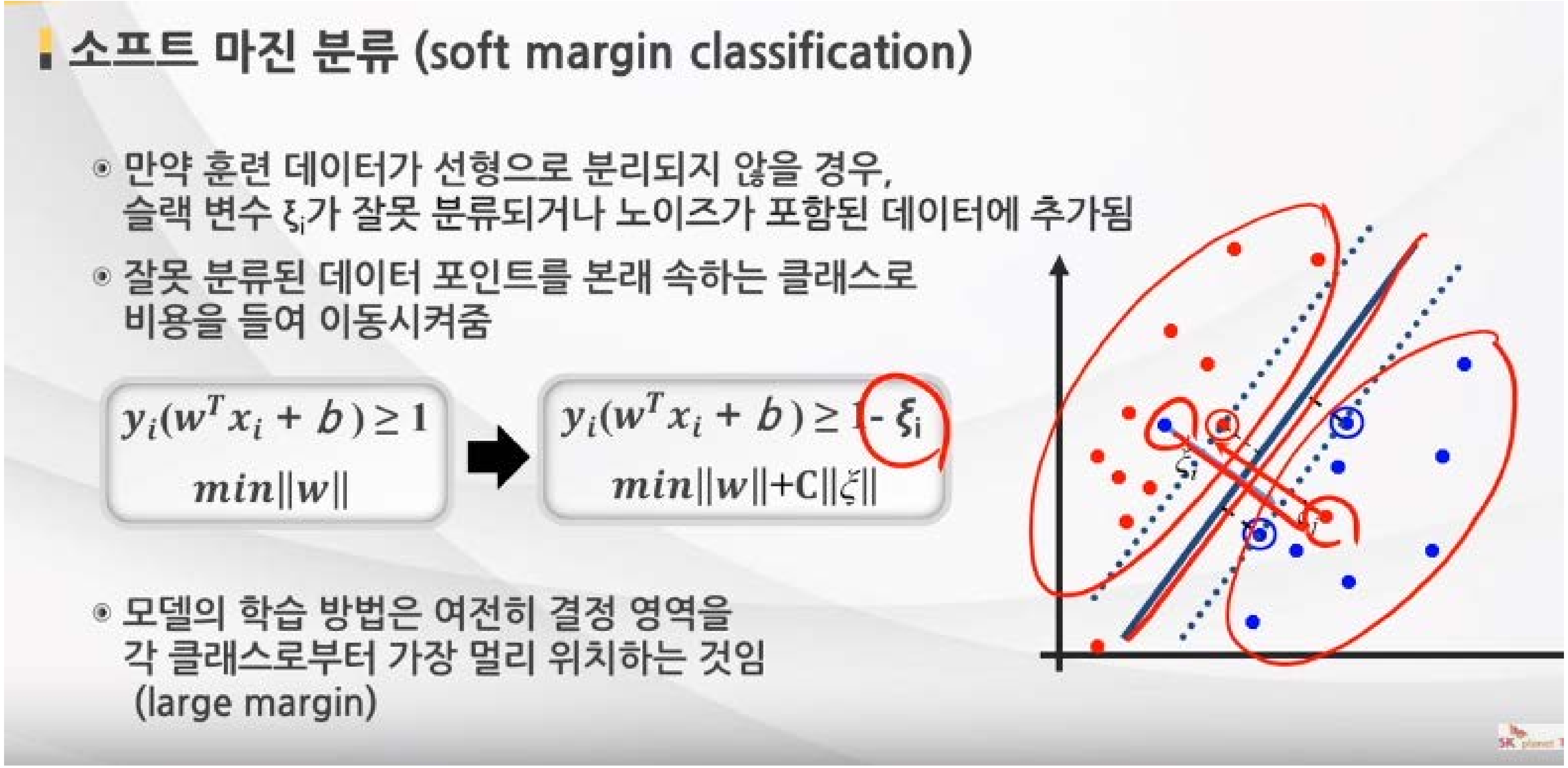
그러므로 분류함수는 다음과 같은 형식임

$f(X) = \sum \alpha_i y_i X_i^T X + b$

- 분류는 새로운 테스트 데이터 x와 서포트 벡터 x_i의 내적에 의해 계산됨

“ 하지만, 모델의 훈련 과정 때는

모든 훈련 데이터 쌍 (x_i, x_j)에 대해 내적 $x_i^T x_j$을 계산 ”



소프트 마진 분류 (soft margin classification)

- 만약 훈련 데이터가 선형으로 분리되지 않을 경우, 슬랙 변수 ξ_i가 잘못 분류되거나 노이즈가 포함된 데이터에 추가됨
- 잘못 분류된 데이터 포인트를 본래 속하는 클래스로 비용을 들여 이동시켜줌

$$\boxed{\begin{array}{c} y_i(w^T x_i + b) \geq 1 \\ min\|w\| \end{array}} \Rightarrow \boxed{\begin{array}{c} y_i(w^T x_i + b) \geq 1 - \xi_i \\ min\|w\| + C\|\xi\| \end{array}}$$

- 모델의 학습 방법은 여전히 결정 영역을 각 클래스로부터 가장 멀리 위치하는 것임 (large margin)