Tutorial on Support Vector Machine (SVM)

CONTENTS

- What is SVM?
- Why SVM?
- Mathematical formulation of SVM
- Kernel Trick
- SVR (Support Vector Regressor)
- Advantage & disadvantage

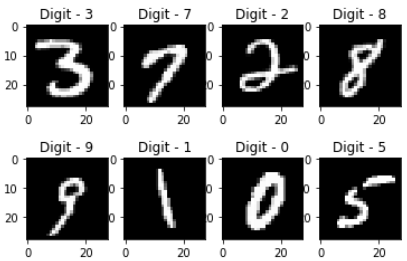
1. What is SVM?

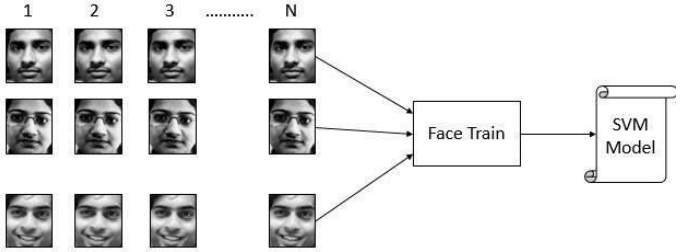
- In 1992, by Boser, Guyon, and Vapnik in COLT-92

Guyon Boser Vapnik

- **Supervised learning** (for **Classification** and **Regression**)
  - learning with label

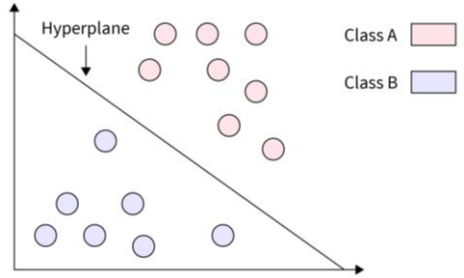
- application areas
  - **handwritten digit recognition**
  - **face analysis**





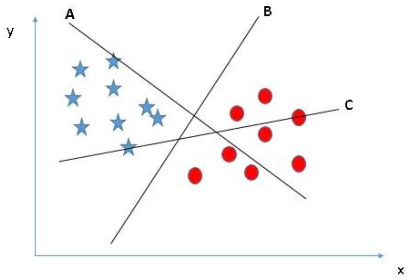
1. What is SVM?

- **"Support Vector machines can be defined as systems which use hypothesis space of a linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory."**



  - **hypothesis space of a linear functions in a high-dimensional feature space**
    - hypothesis space of a linear functions (선형 함수 가설 공간)
      - hypothesis space (가설 공간) : candidate models (in svm, linear functions)
    - high dimensional feature space (고차원 특성 공간)
      - if input space is too low-dimensional, impossible to separate data with many features effectively → Kernel Trick (for converting data into a higher-dimensional space)

  - **learning algorithm from optimization theory that implements a learning bias**
    - learning algorithm from optimization theory (최적화)
      - optimization (optimal classification boundary)
      - maximize margin
    - learning bias (편향)
      - not capture every detail of train data (simplify model)
      - prevent overfitting (과적합 방지)



1. What is SVM?

- **"Generalized linear Classifier"**
  - generalized
    - maximize predictive accuracy & avoid overfitting
  - with statistical learning theory(for optimization): SRM (Structural Risk Minimization)
  - ↔ traditional approach: ERM (Empirical Risk Minimization) (ex. Conventional Neural Network)

- ERM: minimize loss for train data (경험적 위험 최소화)
- SRM: minimize loss for train data + control model complexity (prevent overfitting) (구조적 위험 최소화)

- supervised learning defined in statistical learning theory
  - Minimize

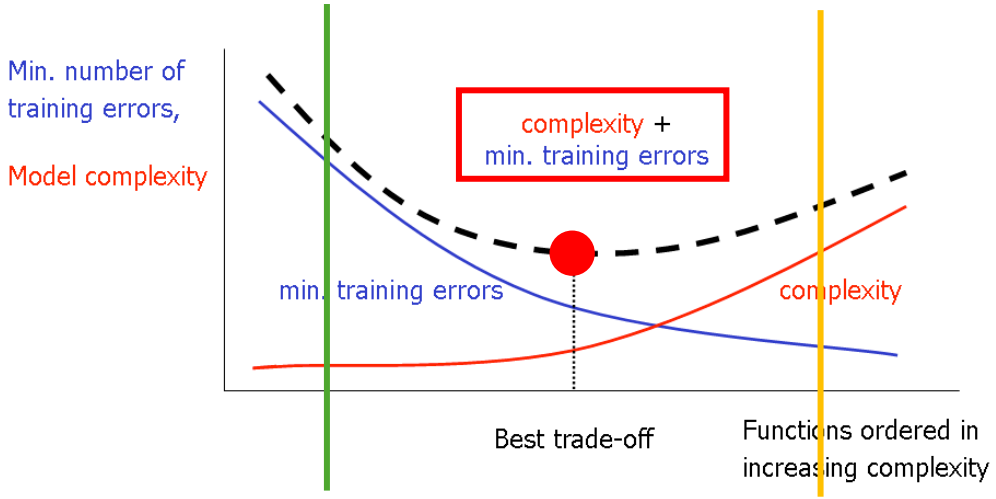
$$\int V\big(y, f(x)\big)P(x, y)\, dx\, dy$$

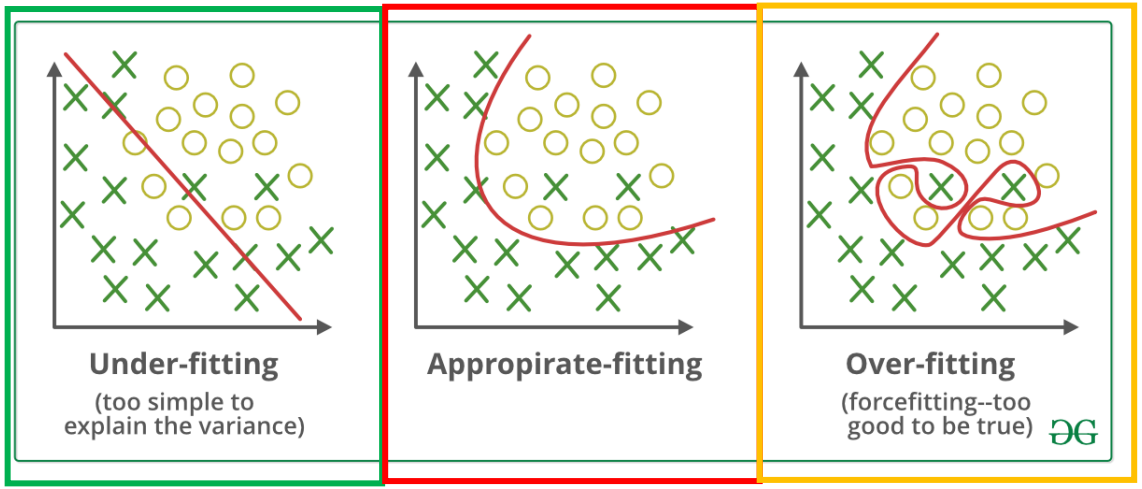
(expectation of the error between real value y and prediction value $f(x)$)

    - Loss function of each data: $V(y, f(x))$
    - entire dataset probability distribution: $P(x, y)$

1. What is SVM?

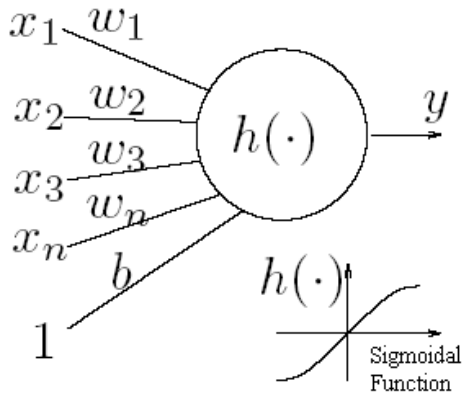
- **Complexity** vs **Training errors**
  - Find best trade-off
  - model complexity ↓, train errors ↑ (Underfitting)
  - model complexity ↑, train errors ↓ test errors ↑ (Overfitting) → cross validation
  - best tradeoff: min(complexity + train errors)



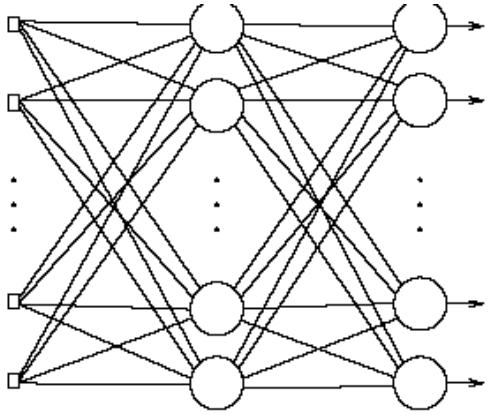


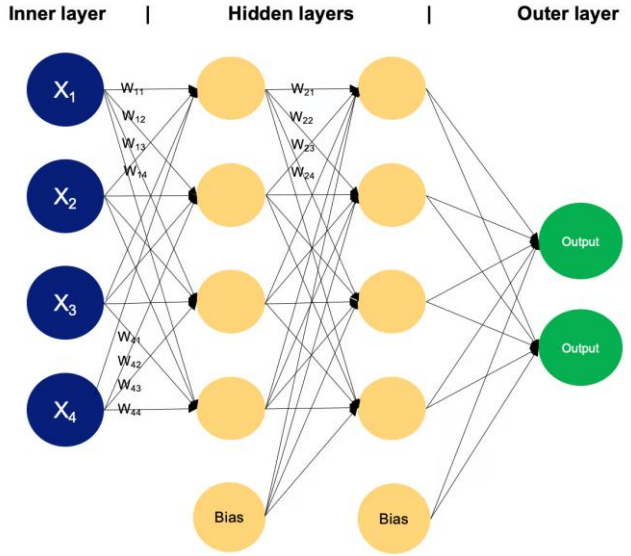
2. Why SVM?

- **Neural Networks (신경망)**
  - ERM
  - good result for supervised and unsupervised learning
  - Single-Layer Perceptron
  - MLP (Multi-Layer Perceptron)
    - with hidden layers
    - feed forward and recurrent network



Single-Layer Perceptron

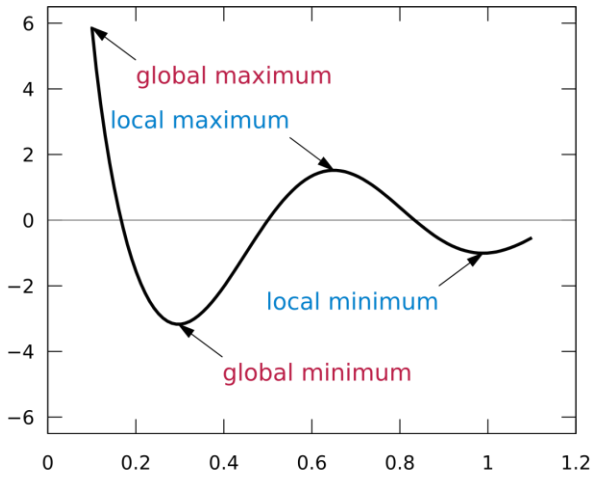


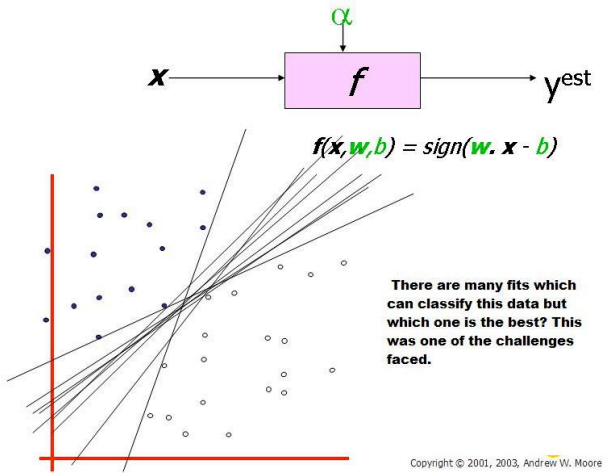


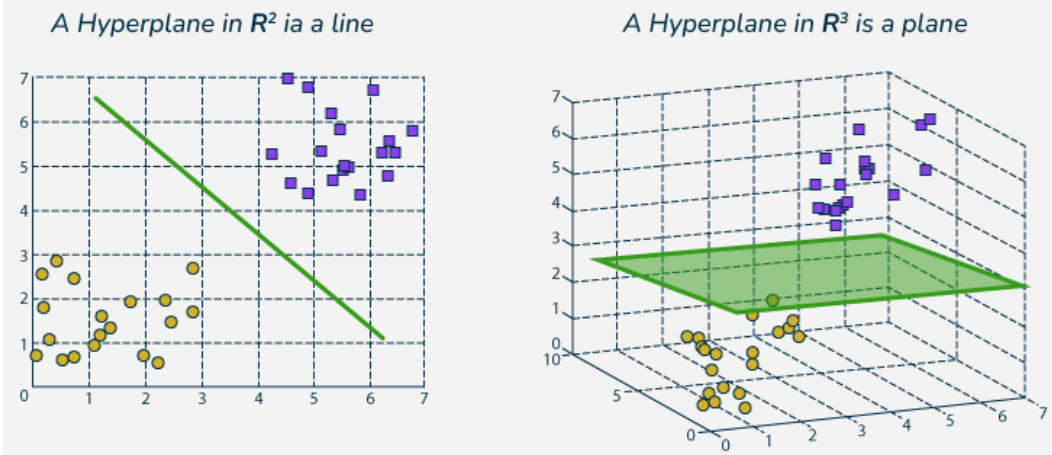
Multi-Layer Perceptron

2. Why SVM?

- Issues of Neural Networks
  - local minima
  - how many neurons for optimization
  - not unique solution (many hyper-planes(linear classifiers))→ which one is the best(**optimal**) solution?
    - hyper-plane (초평면): (n-1)-dim subspace that divides the n-dim space into two parts



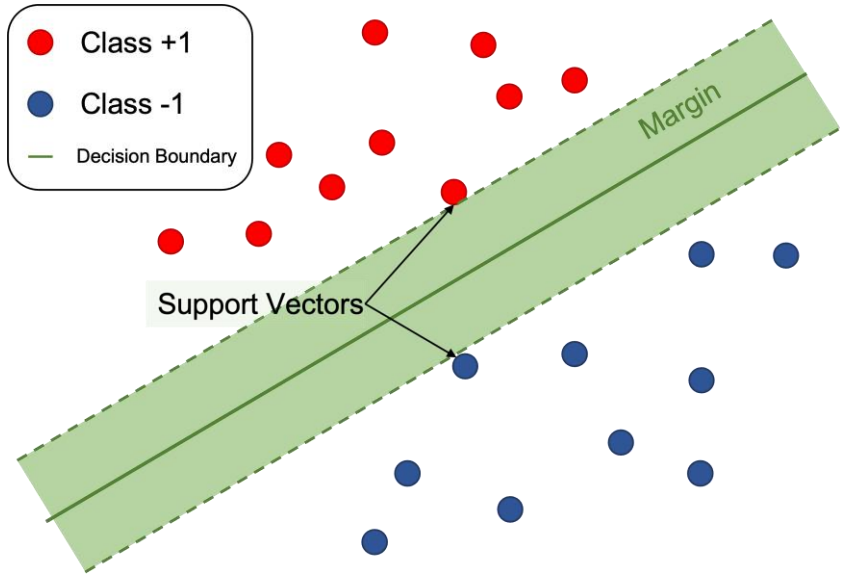




Tutorial on Support Vector Machine (SVM)

2. Why SVM?

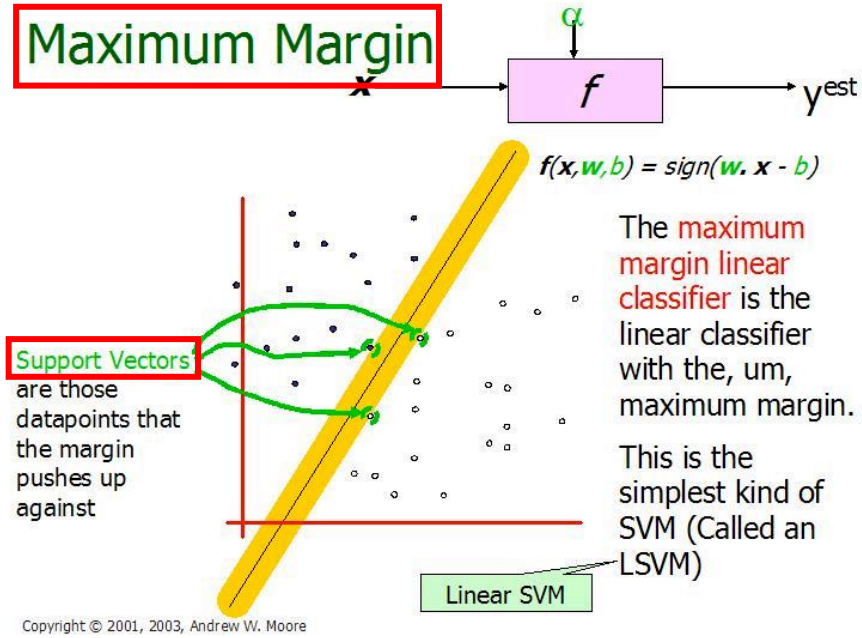
- Maximum margin classifier(hyper plane)
  - Margin: min distance between decision boundary and support vector
  - support vector: the nearest data point from decision boundary
  - **object: Maximize Margin**



2. Why SVM?

- Why max margin?
  - improve empirical performance (generalization)
  - even if small error in the location of the boundary, least chance of misclassification
  - avoid local minima (vs. Neural Net)

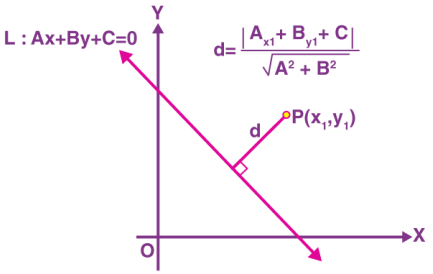
- Ex) Linear SVM (LSVM) (in this thesis..)



Tutorial on Support Vector Machine (SVM)

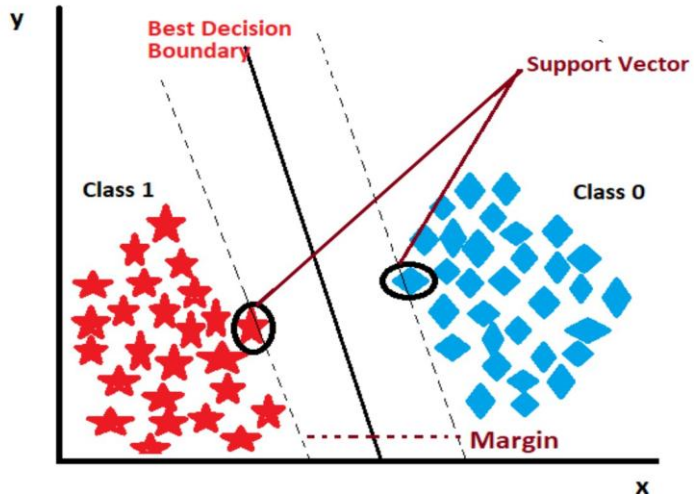
3. Mathematical formulation of SVM

- **How to get Margin**
  - min distance between decision boundary and arbitrary data point
    = distance between decision boundary and support vectors



$$\text{margin} = \arg\min_{x \in D} d(x) = \arg\min_{x \in D} \frac{|\mathbf{x} \cdot \mathbf{w} + b|}{\sqrt{\sum_{i=1}^{d} w_i^2}}$$

- **decision boundary line:** $\boldsymbol{x \cdot w + b = 0}$
- red box means distance between **data point** $\boldsymbol{x}$ and **decision boundary line**



3. Mathematical formulation of SVM

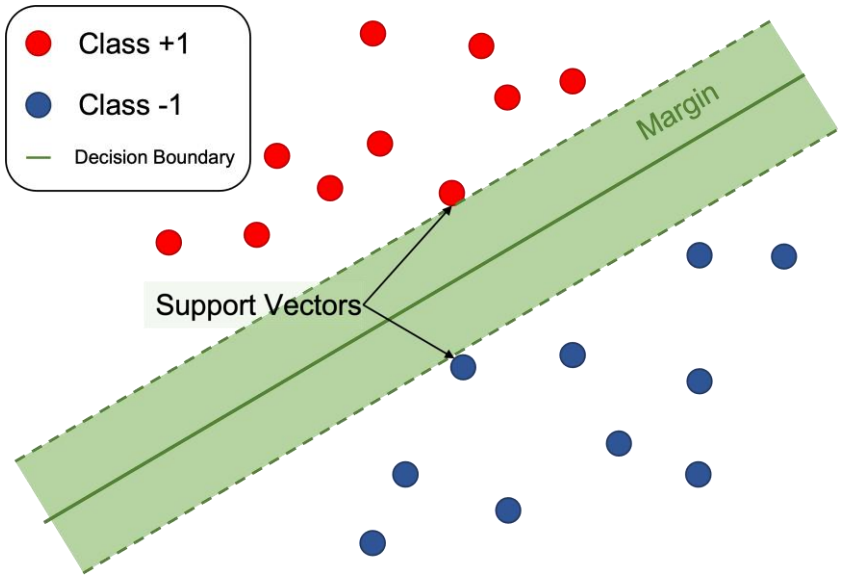
- constraints [a], [b], [c]
  - define two **margin boundaries**: each support vectors on each line

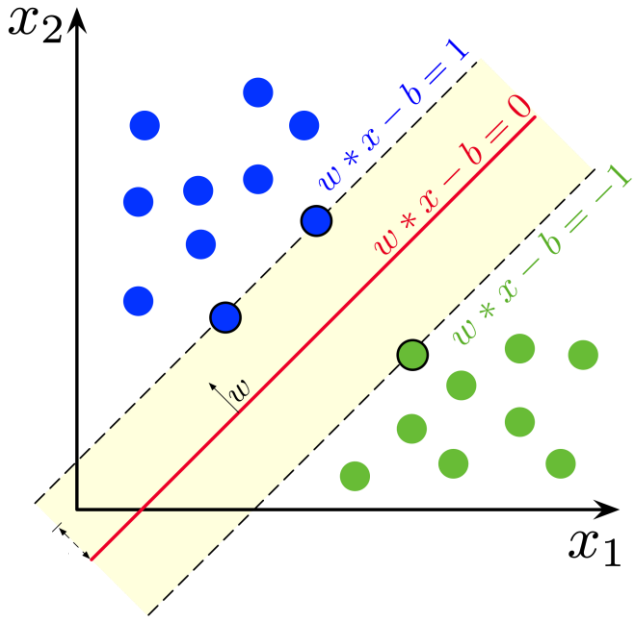
$[a]\ If\ Y_i = +1;\ wx_i + b \geq 1$ class Y_i = +1

$[b]\ If\ Y_i = -1;\ wx_i + b \leq -1$ class Y_i = -1

$\Rightarrow$

$[c]\ For\ all\ i;\ y_i(wx_i + b) \geq 1$



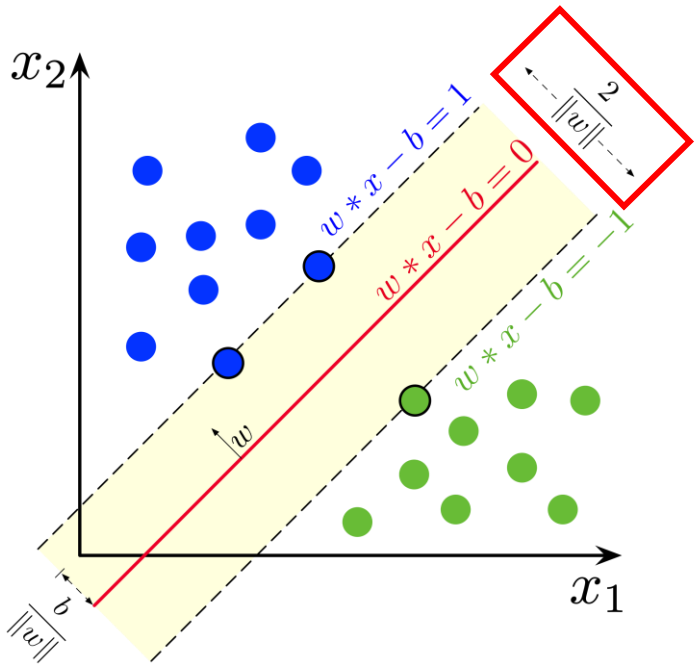


3. Mathematical formulation of SVM

- simplify margin formula (with [c] constraint)
  - distance between $wx + b = 1$ and $wx + b = -1$

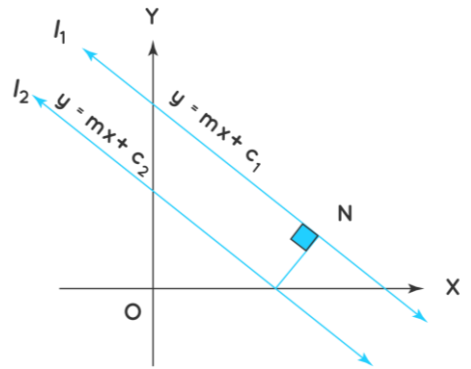
$$\text{margin} = \arg\min_{x \in D} d(x) = \arg\min_{x \in D} \frac{|\mathbf{x} \cdot \mathbf{w} + b|}{\sqrt{\sum_{i=1}^{d} w_i^2}}$$

$$[c]\ For\ all\ i;\ y_i(wx_i + b) \geq 1$$



$$(||w|| = \sqrt{w_1^2 + w_2^2 + \cdots + w_n^2})$$

$$\text{distance} = \frac{|1-(-1)|}{||w||} = \frac{2}{||w||}$$



$$d = \frac{|c_2 - c_1|}{\sqrt{1 + m^2}}$$

3. Mathematical formulation of SVM

- **object** (with simplified margin)
  - **maximize** $\frac{2}{||w||}$

    = minimize $||w||$

    = minimize $\frac{1}{2}||w||^2$ (due to gradient descent for optimization)

$$\Phi(w) = \frac{1}{2}||w||^2$$
$$S.T. \quad y_i(w \cdot x_i + b) \geq 1$$

$\Phi(w)$ = loss function (minimize, optimize)
→ **Quadratic function** (이차 최적화 문제)

- Quadratic function → construct Dual problem (쌍대 문제) & Lagrange multiplier α_i (라그랑즈 승수)

$$\max_a \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$
$$S.T. \quad \alpha_i \geq 0, \quad \sum_i \alpha_i y_i = 0$$

→ optimal α_i

$$w = \sum_i \alpha_i x_i, \quad b = y_k - w \cdot x_k \quad \text{for any } x_k \text{ with } \alpha_k \neq 0$$

  - classification function
    - decide label (f(x) → +1/-1 class)

$$f(x) = w \cdot x + b = \sum_i \alpha_i y_i (x_i \cdot x) + b$$

3. Mathematical formulation of SVM

- Primal (loss function)

$$\min \frac{1}{2}||w||^2$$
$$S.T. \quad y_i(w \cdot x_i + b) \geq 1$$

- Dual (with α_i)

$$\max_a \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$
$$S.T. \quad \alpha_i \geq 0, \quad \sum_i \alpha_i y_i = 0$$

$$w = \sum_i \alpha_i x_i, \quad b = y_k - w \cdot x_k \quad \text{for any } x_k \text{ with } \alpha_k \neq 0$$

- classification function

$$f(x) = w \cdot x + b = \sum_i \alpha_i y_i (x_i \cdot x) + b$$

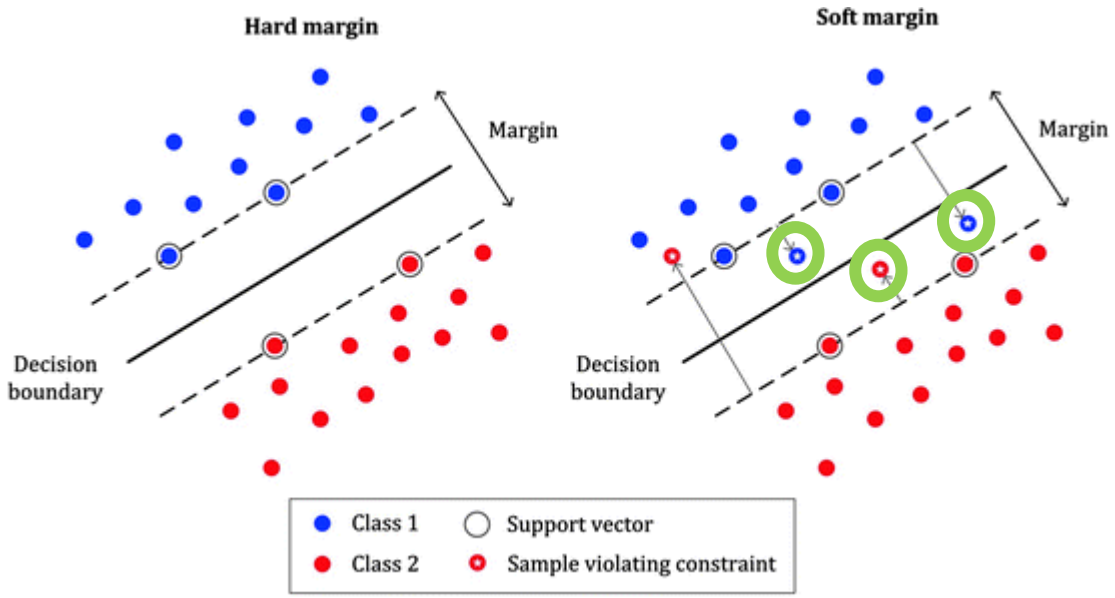
→ **Hard Margin SVM**

- attempt to classify data **with noise** exactly
- **sensitive to noise (overfitting)**

→ **Soft Margin SVM: ignore few data points**

3. Mathematical formulation of SVM

- **Soft Margin SVM**
  - ignore few data points



3. Mathematical formulation of SVM

- Primal (loss function)

$$\min \frac{1}{2}||w||^2$$
$$S.T. \quad y_i(w \cdot x_i + b) \geq 1$$

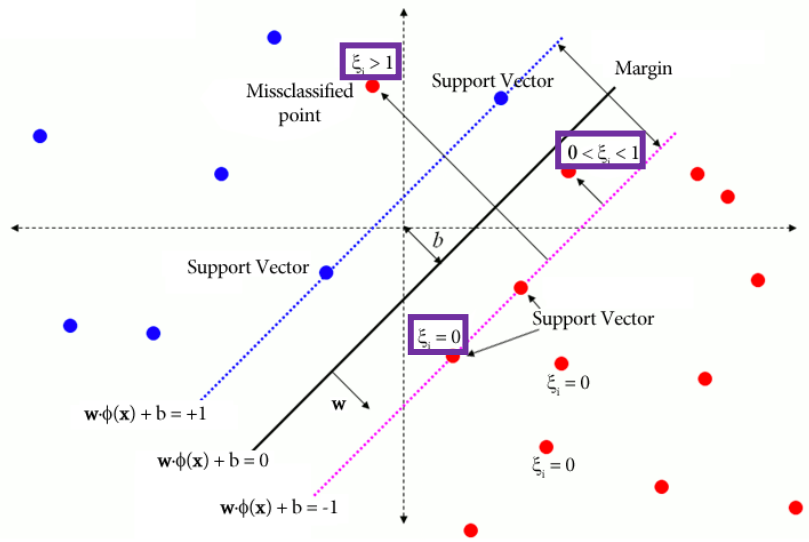
→ **Soft Margin SVM**: with slack variable ξ

$$\min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \xi_i$$
$$S.T. \quad y_i(w \cdot x_i + b) \geq 1 - \xi_i$$

- C is trade-off between maximizing margin and penalizing slack variable

$$\Longrightarrow \min L = \frac{1}{2} w^T w - \sum_k \lambda_k \left(y_k(w^T x_k + b) + s_k - 1 \right) + \alpha \sum_k s_k$$

- slack variable(ξ): how much each point violate the margin requirement (for **noise**)



- ξ ↑, *noise* ↑
- ξ > 1
- 0 < ξ < 1
- ξ = 0 (noise X)

Tutorial on Support Vector Machine (SVM)

3. Mathematical formulation of SVM

- Dual (with α_i)

$$\max_a \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

$$S.T. \quad \alpha_i \geq 0, \quad \sum_i \alpha_i y_i = 0$$

$$w = \sum_i \alpha_i x_i, \quad b = y_k - w \cdot x_k \quad \text{for any } x_k \text{ with } \alpha_k \neq 0$$

→ **Soft Margin SVM**: constraint with C

$$\max_a \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

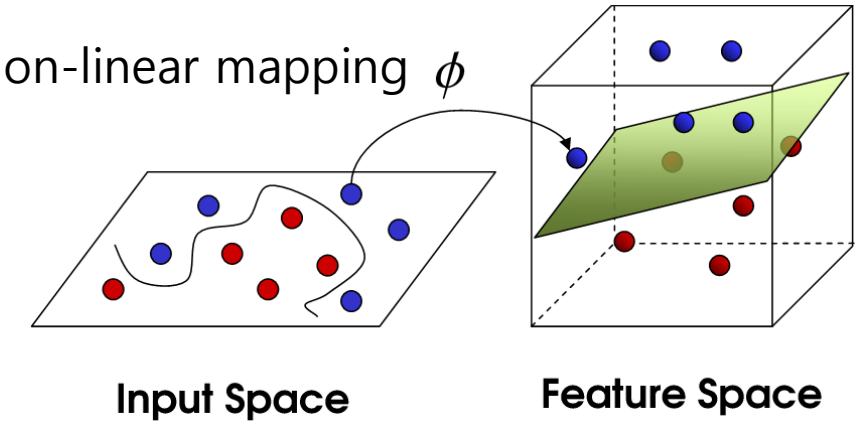
$$S.T. \quad 0 \leq \alpha_i \leq C, \quad \sum_i \alpha_i y_i = 0$$

$$w = \sum_i \alpha_i x_i, \quad b = y_k - w \cdot x_k \quad \text{for any } x_k \text{ with } \alpha_k \neq 0$$

- classification function (same) $f(x) = \sum_i \alpha_i y_i (x_i \cdot x) + b$

4. Kernel Trick

- mapping **non-linear data** to high-dimension → linear separation in high-dim feature space (impossible in original input space)



- Kernel Function: $K(x, y) = \Phi(x) \cdot \Phi(y)$
  - x, y: original (non-linear) input space
  - Φ: mapping x, y to higher-dimension
  - Why dot-product?
    - for similarity measure between two feature vectors
  - based on Reproducing Kernel Hilbert Spaces, RKHS (재생 커널 힐베르트 공간)
  - If K satisfies Mercer's condition($K(x, y) = K(y, x)$) (머서 조건, symmetric) & positive definite($inner\ product\ =\ 0, \forall\ input$ (양의 정부호), kernel represents inner product

4. Kernel Trick

- **Inner product** of Kernel Trick
  - Instead of original variable(w, b) (primal), inner products of data sets (for dual problem)

$$f(x) = w \cdot x + b = \sum_i \alpha_i y_i (x_i \cdot x) + b$$

  - add Pairwise data for interaction between variables
    - $x = [x_1, x_2]$
    - $\Phi(x) = [x_1, x_2, x_1 \cdot x_2]$: add $(x_1 \cdot x_2)$ pairwise
    - Why?
      - new features
  - Inner product computing is so computationally expensive (kernel function)
    - $K(x, y) = \Phi(x) \cdot \Phi(y)$
    - **Ex**. $x = (x_1, x_2), z = (z_1, z_2)$
      - non-kernel (direct inner product computation)
        - $\Phi(x) = (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$
        - $\Phi(z) = (z_1^2, \sqrt{2}\, z_1 z_2, z_2^2)$
        - $\Phi(x) \cdot \Phi(z) = x_1^2 z_1^2 + 2 x_1 x_2 z_1 z_2 + x_2^2 z_2^2$
  - **kernel function** (ex. polynomial)
    - $K(x, z) = (x \cdot z)^2$

    → operation to be performed in input space rather than high-dim feature space

4. Kernel Trick

- Type of Kernel Function

1. **Polynomial**

$K(x, x') = \langle x, x' \rangle^d$

$K(x, x') = (\langle x, x' \rangle + 1)^d$

Non-linear patterns by expanding $\langle x, x' \rangle$ into d-th order polynomial

first formulation(pure polynomial) has Hessian problem (to zero)

2. **Gaussian RBF(Radial Basis Function)**

$$K(x, x') = exp(-\frac{||x - x'||^2}{2\sigma^2})$$

3. **Exponential RBF**

$$K(x, x') = exp(-\frac{||x - x'||}{\sigma})$$

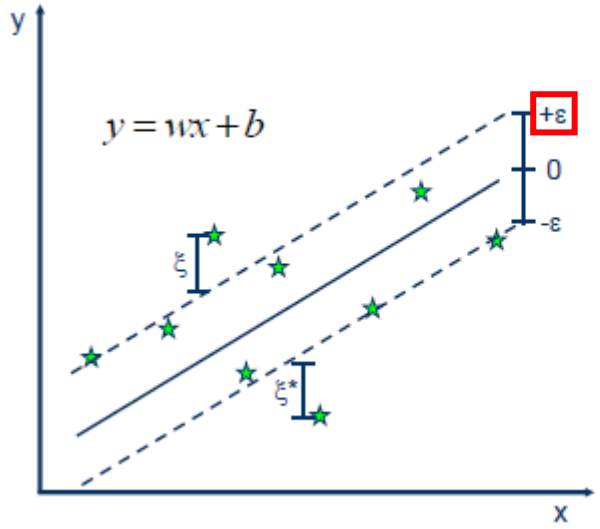
4. **MLP**

$$K(x, x') = \tanh\big(\rho\langle x, x' \rangle + \theta\big)$$

+) Fourier, splines, B-splines, additive kernels and tensor products

5. SVR (Support Vector Regressor)

- SVR
  - ϵ: tolerance range of error between real point and predictive line
  - ξ_i: error when prediction < real
  - ξ_i^*: error when prediction > real



- Minimize:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right)$$

- Constraints:

$$y_i - wx_i - b \le \varepsilon + \xi_i$$

$$wx_i + b - y_i \le \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \ge 0$$

6. Advantage & Disadvantage

- Advantage
  - easy training
  - No local optimal unlike neural network
  - scale relatively well to high-dimensional data
  - trade off between complexity and error explicity

- Disadvantage
  - select appropriate kernel function