Spring 2023

SWCON253: Machine Learning

Lecture 13
Support Vector Machines

Jinwoo Choi
Assistant Professor
CSE, Kyung Hee University

Contents

References

- *기계 학습* by 오일석, *패턴 인식* by 오일석

1. Introduction

1. (Revisit) Linear Decision Boundary
2. Distance from the Decision Boundary
3. Concept of Margin
4. SVM (Support Vector Machine)

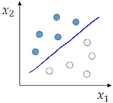
(Recap.) Linear Decision Boundary

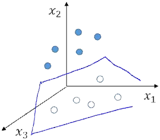
- Equation for *Linear Decision Boundary* (선형 결정 경계)

$$\boxed{d(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + w_0 = 0}$$

★ class 1 if $d(\mathbf{x}) > 0$, class 2 if $d(\mathbf{x}) < 0$

- Two types of vector representation:
  - ★ $\mathbf{x} = [x_0\ x_1\ \ldots x_d], \mathbf{w} = [w_0\ w_1\ \ldots w_d]$: $d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} = 0$
  - ★ $\mathbf{x} = [x_1\ \ldots x_d]$, $\mathbf{w} = [w_1\ \ldots w_d]$: $d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$



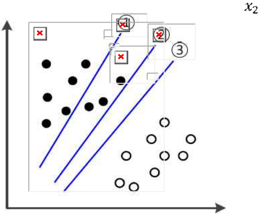


- Geometric Interpretation
  - $d(\mathbf{x}) = 0$ is a *hyperplane* in the feature space.
  -

$d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$

al vector of the hyperplane.

★ b determines the *position* (i.e., the displacement from the origin) of the hyperplane.



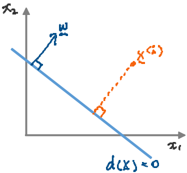
w는 결정경계의 방향을 결정하고, b는 위치를 결정한다.

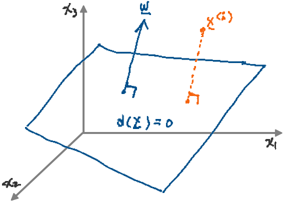
Distance From the Decision Boundary

- Consider
  - a feature vector representation *without* x_0 : $\mathbf{x}$
  - a decision boundary: $d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$
  - a (i^{th}) sample point to be classified: $\mathbf{x}^{(i)}$

- Then, the Euclidean distance from $\mathbf{x}^{(i)}$ to $d(\mathbf{x}) = 0$ is given by

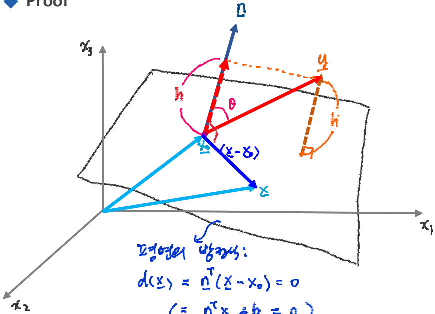
$$h = \frac{|d(\mathbf{x}^{(i)})|}{\|\mathbf{w}\|_2}$$





Distance From the Decision Boundary (cont'd)

- **Proof**



평면의 방정식:

$$d(\underline{x}) = \underline{n}^T(\underline{x} - \underline{x}_0) = 0$$

$$(= \underline{n}^T \underline{x} + b = 0)$$

$$h = \|\underline{y} - \underline{x}_0\|_2 |\cos\theta|$$

$$= \cancel{\|\underline{y} - \underline{x}_0\|_2} \cdot \frac{|\underline{n}^T(\underline{y} - \underline{x}_0)|}{\cancel{\|\underline{y} - \underline{x}_0\|_2} \|\underline{n}\|_2}$$

$$= \frac{|\underline{n}^T(\underline{y} - \underline{x}_0)|}{\|\underline{n}\|_2}$$

Since $d(\underline{x}) = \underline{n}^T(\underline{x} - \underline{x}_0)$,

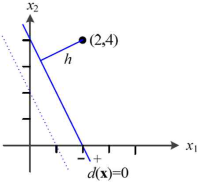
$$|\underline{n}^T(\underline{y} - \underline{x}_0)| = |d(\underline{y})|.$$

$$\therefore\ h = \frac{|d(\underline{y})|}{\|\underline{n}\|_2}$$

Distance From the Decision Boundary (cont'd)

Example: 점과 직선 사이의 거리 구하기

- 특징 공간 상의 한 점: $\mathbf{x}^{(1)} = [2\ 4]^T$
- 결정 직선: $d(\mathbf{x}) = 2x_1 + x_2 - 4 = 0$



- **풀이**
  - ★ 결정 직선을 $d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = 0$ 하면, $\mathbf{w} = [2\ 1]^T,\ b = -4.$
  - ★ 점과 직선 사이의 거리

$$h = \frac{|d(\mathbf{x}^{(1)})|}{\|\mathbf{w}\|_2} = \frac{|2 \cdot 2 + 1 \cdot 4 - 4|}{\sqrt{2^2 + 1^2}} = \frac{4}{\sqrt{5}}$$

Note

- $d_c(\mathbf{x}) = c \times d(\mathbf{x})$라 하면, $d_c(\mathbf{x}) = 0$ 과 $d(\mathbf{x}) = 0$ 은 동일한 결정경계를 나타낸다.
- 따라서, $d_c(\mathbf{x})$를 사용 해도 위 거리 공식은 성립한다.
- 이를 이용하여, $|d_c(\mathbf{x}^{(1)})| = 1$이 되도록 c를 결정할 수 있다. (위의 예에서는 $c = 1/4$)

Concept of Margin

- **The previously learned classifiers try to minimize some "*error*".**
  - Cross entropy loss in logistic regression

$$J(\mathbf{w}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(h(\mathbf{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-h(\mathbf{x}^{(i)})\right)\right]$$

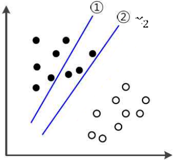
  - Miss-classification cost in perceptron (with ***step*** activation)

$$J(\mathbf{w}) = -\sum_{k\in Y} y^{(k)}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(k)}\right)$$

(Y: 오분류된 샘플의 집합)

  - MSE loss in perceptron or MLP (with ***sigmoid*** activation)

$$J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\left(h(\mathbf{x}^{(i)}) - y^{(i)}\right)^2$$



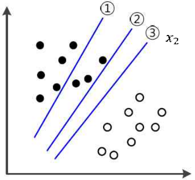
①에서 출발하여
②에 도달하면 멈춤

- ***If all the samples are correctly classified, the optimization will stop.***

Concept of Margin (cont'd)

- **Margin (여백)**
  - 오류가 없으면 최선인가?
  - 결정 초평면으로부터 양쪽 class에 대한 "여백"을 최대화할 수록 분류기의 일반화 능력이 향상될 것이다.

    ★ ② 보다 ③이 양쪽 class에 대해 여백이 커서 일반화 능력이 우월



- **Questions**
  - 여백(margin)이라는 개념을 어떻게 수학적으로 표현할 것인가?

    → *Minimum distance from training sample to the decision boundary !*
  - 여백을 최대로 하는 결정 초평면을 어떻게 찾을 것인가?

    → *Constrained optimization !*

SVM (Support Vector Machine)

SVM

- Classification을 위한 ML Algorithm의 하나로, 여백(margin)을 이용하여 일반화 능력을 향상시킴
- 1990년대 신경망의 성능을 능가하여 인기 있는 모델로 부상
- 2000년대 들어 DL에 밀려 시들한 편
- 최근 SVM과 DL을 결합하는 접근방법이 시도되고 있음

Linear SVM vs. Non-linear SVM

- 선형 SVM은 선형 분류문제를 오류 없이 풀 수 있음
- 비선형 SVM은 비선형 분류문제도 풀 수 있음

Why Are SVMs So Successful?

SVMs	Neural Networks
Convex optimization problem, easy to implement	Training a NN is a *non-convex* problem
Very few variables to tune that can be done by cross validation - C and a kernel parameter (e.g., σ in the Gaussian kernel)	*Lots of parameters to tune* - how many neurons - how many layers, etc.
So simple to run Many efficient algorithms are available	So many tricks are involved Needs a large amount of experience
The kernel framework boosts the potential of the SVM to the non-linear regime, but does *not lead to excessive overfitting*	By now, DNNs are very successful for highly structured data (speech, text, images) but: - they requires *high computational power* - It is needed to prevent *excessive overfitting* - techniques are needed to train *small dataset*
Statistical learning *theory* shows many nice guarantees about the SVM (consistency, etc.)	From theory point of view, *not understood very well* why they actually work

Vladmir Vapnik

Vladimir Vapnik
러시아

Vapnik은 SVM을 창안한 사람으로 유명하다. 현재 SVM은 일반화 능력이 ~~각종~~ 뛰어난 분류기로 인정 받고 있다. 패턴 인식에서 기술 돌파로 breakthrough 평가되고 있는 것이다. 그는 러시아에서 태어났으며 모스코바에 있는 제어 과학 연구원에서 Institute of Control Sciences 통계학으로 박사 학위를 취득하였다. 그 후 이 연구원에서 1990년까지 근무하였고 이후 미국 AT&T로 이적하였고 주로 미국에서 연구 활동을 하였다. 현재는 Columbia 대학과 London 대학의 교수이다. SVM을 포함하여 통계적 학습 이론을 정리한 책이 그의 대표적인 저서 중의 하나이다 [Vapnik98].

[Vapnik98] Vladimir Vapnik, *Statistical Learning Theory*, John Wiley and Sons, 1998.

Contents

References

- *기계 학습* by 오일석, *패턴 인식* by 오일석

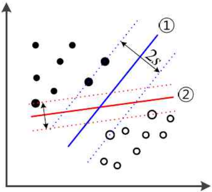
2. Linear SVM with Hard Margin

1. **Support Vector Machine**
2. Linear SVM – *Formulation*
3. Linear SVM – *Lagrange Method*
4. Linear SVM – *Lagrange Dual Problem*
5. Linear SVM – *Illustrative Example*

Support Vector Machine

◆ 선형분리 가능한 분류 문제를 생각하자.

- 먼저 주어진 데이터를 오류없이 분류하도록 **w**를 정한다.
- 정해진 직선의 방향 **w**에 대해, 직선으로부터 가장 가까운 샘플까지의 거리가 같게 되도록 바이어스 b를 정한다.
- 그림의 ①과 ②는 위의 과정을 통해 얻은 결정 직선의 예를 나타낸다.



$|d(\mathbf{x}_i)| \geq 1 \ for \ \forall y_i$

- 이때, 분할 띠의 너비 2s를 **여백**(*margin*)이라 부른다.
- 이때, 분할 띠의 경계에 있는 샘플 벡터를 **서포트 벡터** (*support vector*)라 부른다.

- **SVM은 여백을 최대로 하는 결정 초평면을 구하는 알고리즘이다.**

$$\text{여백} = 2s = \frac{2|d(\mathbf{x})|}{\|\mathbf{w}\|_2} = \frac{2}{\|\mathbf{w}\|_2}$$ (여기서 **x**는 support vector)

★ $d(\mathbf{x})$에 상수를 곱해도 같은 초평면을 나타낸다는 성질을 이용하여 $|d(\mathbf{x})| = 1$ 이 되도록 하였음에 유의하라.

Linear SVM – *Formulation*

Maximize: $J(\mathbf{w}) = \frac{2}{\|\mathbf{w}\|_2}$

Subject to:
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \geq 1, \forall y_i = 1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \leq -1, \forall y_i = -1$$

Minimize: $J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$

Subject to: $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 \geq 0, i = 1,2,\cdots,n$

해의 유일성과 전역성

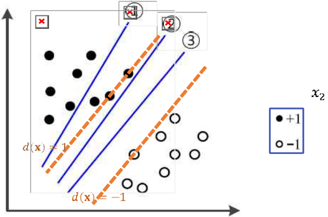
- 선형 부등식 영역 내에서 L_2 Loss는 convex.
- 따라서 해는 유일하며 전역 최적해가 된다.

문제의 난이도

- n개의 선형 부등식을 제약조건으로 가진 2차 함수의 최적화 문제
- 조건부 최적화 문제는 Lagrange Multiplier 방법으로 푼다.

Linear SVM – *Support Vectors*

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



Linear SVM – *Lagrange Method*

minimize $f(x)$
subject to $g(x) \le 0$

$$L(x, \lambda) = f(x) + \lambda g(x)$$

$$\nabla_x L(x, \lambda) = 0$$

- $g(x) = 0,\ \lambda > 0$ (*active*)
- $g(x) < 0,\ \lambda = 0$ (*inactive*)

Lagrangian

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{\|\mathbf{w}\|_2^2}{2} - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1\right)$$

KKT Condition → 결국 α_i와 Support Vector를 구하는 문제

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0} \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

→ α_i와 $\mathbf{x}_i$를 알면 $\mathbf{w}$를 구할 수 있음
($\alpha_i > 0$인 경우의 $\mathbf{x}_i$만 알면 됨)

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

$\alpha_i \ge 0, \quad i = 1, 2, \cdots, n$

- $\alpha_i = 0$ and $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 > 0$ (*inactive*)
- or $\alpha_i > 0$ and $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 = 0$ (*active*)

(이 식을 만족하는 샘플($\mathbf{x}_i$)들이 support vector)

$\mathbf{w}$와 $\mathbf{x}_i$를 알면 b를 구할 수 있음:

$$b = y_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_i$$

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

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b = y_i$$

Linear SVM – *Lagrange Dual Problem*

◆ Wolfe Dual Problem

- Convex 성질을 만족하는 조건부 최적화 문제는 Wolfe Dual로 변형할 수 있다.
- Wolfe Dual로 바꾸면 <u>부등식 조건이 등식 조건으로 </u>바뀌어 풀기에 유리해 진다.

- Original Problem:

Minimize:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{\|\mathbf{w}\|_2^2}{2} - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1\right)$$

Subject to:

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

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

$$\alpha_i \geq 0, \qquad i = 1, 2, \cdots, n$$

Linear SVM – *Lagrange Dual Problem (cont'd)*

Wolfe Dual Problem (cont'd)

- 수식을 간단히 정리하면, 아래와 같은 **Wolfe Dual Problem**을 얻을 수 있다.

Maximize:

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\top} \mathbf{x}_j$$

Subject to:

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

$$\alpha_i \geq 0, \qquad i = 1,2,\cdots,n$$

★ 2차 함수의 최대화 문제이다.

★ **w**와 b가 사라지고, **α**를 찾는 문제가 되었다. (**α**를 찾으면 앞 슬라이드의 수식을 통해 **w**와 b를 구할 수 있다.)

★ 특징 벡터 $\mathbf{x}_i$가 내적 형태로 나타난다. (비선형으로 확장하는 발판)

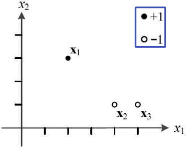
★ 목적 함수의 두번째 ∑항은 n^2개의 항을 갖는다. (여전히 풀기 어려운 문제)

- 예를 들어, 샘플이 6만 개인 MNIST는 36억개 항이 생긴다.
- 따라서 효율적인 최적화 알고리즘이 필요하다.

Linear SVM – *Illustrative Example*

◆ 훈련집합의 샘플 개수가 3개인 경우, Wolfe Dual Problem의 해를 구해 보자.

- **훈련 집합:** $\mathbf{x}_1 = \binom{2}{3},\ \mathbf{x}_2 = \binom{4}{1},\ \mathbf{x}_3 = \binom{5}{1},\ y_1 = 1,\ y_2 = -1,\ y_3 = -1$



- **Wolfe Dual Problem:**

Maximize: $\tilde{\mathcal{L}}(\boldsymbol{\alpha}) = (\alpha_1 + \alpha_2 + \alpha_3)$
$\qquad -\frac{1}{2}(13\alpha_1^2 + 17\alpha_2^2 + 26\alpha_3^2 - 22\alpha_1\alpha_2 - 26\alpha_1\alpha_3 + 42\alpha_2\alpha_3)$

Subject to: $\alpha_1 - \alpha_2 - \alpha_3 = 0$
$\qquad 0 \le \alpha_1, 0 \le \alpha_2, 0 \le \alpha_3$

- **풀이:**

★ 구해야 하는 미지수가 3개뿐이므로 경우를 일일이 나열하여 풀 수 있다.

★ Class별로 Support Vector가 하나 이상 있어야 하므로, 다음의 세 가지 경우만 가능하다.

(1) $\alpha_1 \ne 0,\ \alpha_2 = 0,\ \alpha_3 \ne 0$

(2) $\alpha_1 \ne 0,\ \alpha_2 \ne 0,\ \alpha_3 = 0$

(3) $\alpha_1 \ne 0,\ \alpha_2 \ne 0,\ \alpha_3 \ne 0$

Linear SVM – *Illustrative Example (cont'd)*

(1) $\alpha_1 \neq 0,\ \alpha_2 = 0,\ \alpha_3 \neq 0$

조건 $\alpha_1 - \alpha_2 - \alpha_3 = 0$으로부터 $\alpha_1 = \alpha_3$이다. 이것을 $\tilde{L}(\boldsymbol{\alpha})$에 대입하여 정리하면 다음 식을 얻는데, 이 식은 $\alpha_1 = \frac{2}{13}$에서 최댓값을 가진다. 따라서 답은 $\alpha_1 = \frac{2}{13},\ \alpha_2 = 0,\ \alpha_3 = \frac{2}{13}$이다.

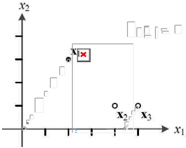
$$\tilde{L}(\boldsymbol{\alpha}) = 2\alpha_1 - \frac{13}{2}\alpha_1^2 = -\frac{13}{2}\left(\left(\alpha_1 - \frac{2}{13}\right)^2 - \frac{4}{169}\right)$$

i=1과 3에 대해, $y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 = 0$ 이므로, $\mathbf{w} = \left(-\frac{6}{13}, \frac{4}{13}\right)^T$, $b = 1$

$d(\mathbf{x}) = -\frac{6}{13}x_1 + \frac{4}{13}x_2 + 1$ 이므로,

$d(\mathbf{x}_1) = 1,\ d(\mathbf{x}_2) = -\frac{7}{13},\ d(\mathbf{x}_3) = -1$

확인해 보면 $\mathbf{x}_2$는 분할 띠 안에 있다.



x1과 x3가 support vector라는 조건 하에 문제를 푸는 것이므로 x2는 분할띠 바깥에 있어야함. 따라서 y2 * d(x2)가 1보다 커야 되는 데, 이 조건을 만족하지 못하므로 해가 되지 못함.

Linear SVM – *Illustrative Example (cont'd)*

(2) $\alpha_1 \neq 0,\ \alpha_2 \neq 0,\ \alpha_3 = 0$

(1)과 마찬가지 방법으로 풀면,

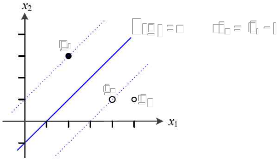
$$\alpha_1 = \frac{1}{4},\ \alpha_2 = \frac{1}{4},\ \alpha_3 = 0$$

$\mathbf{w} = \left(-\frac{1}{2}, \frac{1}{2}\right)^{\mathrm{T}}$ 이고 $b = \frac{1}{2}$

$$d(\mathbf{x}) = -\frac{1}{2}x_1 + \frac{1}{2}x_2 + \frac{1}{2}$$

$|d(\mathbf{x}_1)| = |d(\mathbf{x}_2)| = 1$이므로 $\mathbf{x}_1$과 $\mathbf{x}_2$는 서포트 벡터

$d(\mathbf{x}_3) < -1$이므로 $\mathbf{x}_3$는 분할 띠 바깥

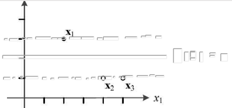


(b) SVM으로 구한 결정 직선

x1과 x2가 support vector라는 조건 하에 문제를 푸는 것이므로 x3가 분할띠 바깥에 있어야 함. 즉 y3 * d(y3)가 1보다 커야 되는데, 이 조건을 만족하므로 해가 될 수 있음

(3) $\alpha_1 \neq 0,\ \alpha_2 \neq 0,\ \alpha_3 \neq 0$

(1)과 마찬가지 방법으로 풀면, ... (연습해 보세요!)

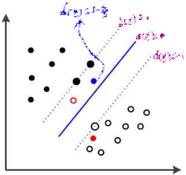


3. Linear SVM with Soft Margin

1. Hard vs. Soft Margin
2. Linear SVM with Soft Margin
3. Illustrative Example

Hard vs. Soft Margin

- 앞의 Linear SVM은 ***Hard Margin***을 사용
  - Hard Margin 내에는 샘플이 존재하지 않는다.
  - Hard Margin을 사용하면 선형분리가 불가능한 상황에서는 해를 구할 수 없다.



- ***Soft Margin & Slack Variable***
  - Soft Margin이란 분할 띠 안에도 샘플을 허용하는 것을 말한다.
  - 샘플 $(\mathbf{x}_i, y_i)$는 다음 세 가지 case 중 하나에 속하며,
    슬랙 변수(*Slack Variable*) ξ를 도입하여 세 case를 하나로 표현할 수 있다.

$$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \ge 1 - \xi_i$$

Case	분류 결과	샘플 위치	그림 표시	$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)$	슬랙 변수
1	옳게 분류	분할 띠 바깥	● o	$1 \le y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)$	$\xi_i = 0$
2	옳게 분류	분할 띠 안쪽	●	$0 \le y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 1$	$0 < \xi_i \le 1$
3	틀리게 분류	결정 경계 넘음	● o	$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 0$	$1 < \xi_i$

Linear SVM with Soft Margin – *Formulation*

◆ Problem Definition

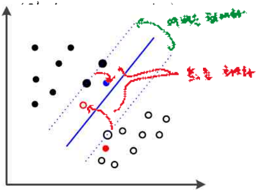
여백을 될 수 있는 한 크게 하면서(목적 1), $0 < \xi$인 (즉, 경우 2와 경우 3에 속하는) 샘플의 수를 될 수 있는 한 적게 하는(목적 2) 결정 초평면의 방향 $\mathbf{w}$를 찾아라.

Minimize: $J(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n} \xi_i$

Subject to: $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, i = 1,2,\cdots,n$

$0 \leq \xi_i, i = 1,2,\cdots,n$

- C를 작게 하면 여백의 비중이 커진다. (여백 최대화)
- C를 크게 하면 ξ_i의 비중이 커진다. (분할 띠 안쪽 샘플 수 최소화)



Linear SVM with Soft Margin – *Lagrange Method*

- **Lagrangian**

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \left(\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i\right) - \left(\sum_{i=1}^{n}\alpha_i\left(y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 + \xi_i\right) + \sum_{i=1}^{n}\beta_i\xi_i\right)$$

- **KKT Condition** → 결국 α_i, β_i를 구하는 문제

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{0} \implies \mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i$$

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

$$\frac{\partial L}{\partial b} = 0 \implies C = \alpha_i + \beta_i, i = 1,2,\cdots,n$$

$$0 \le \alpha_i, i = 1,2,\cdots,n$$

$$0 \le \beta_i, i = 1,2,\cdots,n$$

$$\alpha_i\left(y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 + \xi_i\right) = 0, i = 1,2,\cdots,n$$

$$\beta_i\xi_i = 0, i = 1,2,\cdots,n$$

Linear SVM with Soft Margin – *Lagrange Dual Problem*

Wolfe Dual Problem

- Original Problem

Minimize:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \left(\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i\right) - \left(\sum_{i=1}^{n}\alpha_i\left(y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 + \xi_i\right) + \sum_{i=1}^{n}\beta_i\xi_i\right)$$

Subject to:

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

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

$$C = \alpha_i + \beta_i, i = 1,2,\cdots,n$$

$$0 \le \alpha_i, i = 1,2,\cdots,n$$

$$0 \le \beta_i, i = 1,2,\cdots,n$$

- Wolfe Dual Problem

Maximize:

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

Subject to:

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

$$0 \le \alpha_i \le C, \quad i = 1,2,\cdots,n$$

Linear SVM with Soft Margin – *Illustrative Example*

◆ 선형분리가 불가능한 경우, Wolfe Dual Problem의 해를 구해 보자.

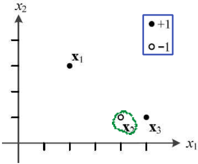
- **훈련 집합:** $\mathbf{x}_1 = \begin{pmatrix}2\\3\end{pmatrix},\ \mathbf{x}_2 = \begin{pmatrix}4\\1\end{pmatrix},\ \mathbf{x}_3 = \begin{pmatrix}5\\1\end{pmatrix},\ y_1 = 1,\ y_2 = -1,\ y_3 = 1$

- **Wolfe Dual Problem:**

Maximize: $\tilde{\mathcal{L}}(\boldsymbol{\alpha}) = (\alpha_1 + \alpha_2 + \alpha_3) - \frac{1}{2}(13\alpha_1^2 + 17\alpha_2^2 + 26\alpha_3^2 - 22\alpha_1\alpha_2 + 26\alpha_1\alpha_3 - 42\alpha_2\alpha_3)$

Subject to: $\alpha_1 - \alpha_2 + \alpha_3 = 0$

$0 \le \alpha_1 \le C, 0 \le \alpha_2 \le C, 0 \le \alpha_3 \le C$



- **풀이:**

★ Class별로 Support Vector가 하나 이상 있어야 하므로,
다음의 세 가지 경우만 가능하다.

(1) $\alpha_1 = 0, \alpha_2 \ne 0, \alpha_3 \ne 0$

(2) $\alpha_1 \ne 0, \alpha_2 \ne 0, \alpha_3 = 0$

(3) $\alpha_1 \ne 0, \alpha_2 \ne 0, \alpha_3 \ne 0$

Linear SVM with Soft Margin – *Illustrative Example (cont'd)*

(1) $\alpha_1 = 0,\ \alpha_2 \neq 0,\ \alpha_3 \neq 0$

$\alpha_1 - \alpha_2 + \alpha_3 = 0$으로부터 $\alpha_2 = \alpha_3$

$$\tilde{L}(\boldsymbol{\alpha}) = 2\alpha_2 - \frac{1}{2}\alpha_2^2 = -\frac{1}{2}((\alpha_2 - 2)^2 - 4)$$

$\Longrightarrow \quad \alpha_1 = 0,\ \alpha_2 = 2,\ \alpha_3 = 2$

$\mathbf{w} = (2,0)^{\mathrm{T}}, b = -9$

$d(\mathbf{x}) = 2x_1 - 9$

$d(\mathbf{x}_1) = -5, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = 1$

(2) $\alpha_1 \neq 0,\ \alpha_2 \neq 0,\ \alpha_3 = 0$

$\alpha_1 - \alpha_2 + \alpha_3 = 0$으로부터 $\alpha_1 = \alpha_2$

$$\tilde{L}(\boldsymbol{\alpha}) = 2\alpha_1 - 4\alpha_1^2 = -4\left(\left(\alpha_1 - \frac{1}{4}\right)^2 - \frac{1}{16}\right)$$

$\Longrightarrow \quad \alpha_1 = \frac{1}{4},\ \alpha_2 = \frac{1}{4},\ \alpha_3 = 0$

$\mathbf{w} = \left(-\frac{1}{2}, \frac{1}{2}\right)^{\mathrm{T}}, b = \frac{1}{2}$

$d(\mathbf{x}) = -\frac{1}{2}x_1 + \frac{1}{2}x_2 + \frac{1}{2}$

$d(\mathbf{x}_1) = 1, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = -\frac{3}{2}$

(3) $\alpha_1 \neq 0,\ \alpha_2 \neq 0,\ \alpha_3 \neq 0$

$\alpha_1 - \alpha_2 + \alpha_3 = 0$으로부터 $\alpha_1 = \alpha_2 - \alpha_3$

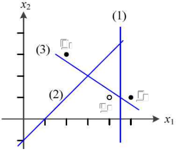
$\frac{\partial \tilde{L}(\boldsymbol{\alpha})}{\partial \alpha_2} = 0$과 $\frac{\partial \tilde{L}(\boldsymbol{\alpha})}{\partial \alpha_3} = 0$을 풀면

$\Longrightarrow \quad \alpha_1 = \frac{3}{2},\ \alpha_2 = \frac{13}{2},\ \alpha_3 = 5$

$\mathbf{w} = (2,3)^{\mathrm{T}}, b = -12$

$d(\mathbf{x}) = 2x_1 + 3x_2 - 12$

$d(\mathbf{x}_1) = 1, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = 1$



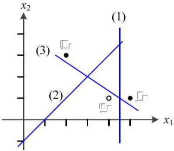
Linear SVM with Soft Margin – *Illustrative Example (cont'd)*

◆ SVM은 (1)~(3) 중 어느 결정 직선을 선택하게 될까? → C에 따라 다름

(1) $\alpha_1 = 0,\ \alpha_2 = 2,\ \alpha_3 = 2$ $\quad d(\mathbf{x}_1) = -5, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = 1$

(2) $\alpha_1 = \frac{1}{4},\ \alpha_2 = \frac{1}{4},\ \alpha_3 = 0$ $\quad d(\mathbf{x}_1) = 1, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = -\frac{3}{2}$

(3) $\alpha_1 = \frac{3}{2},\ \alpha_2 = \frac{13}{2},\ \alpha_3 = 5$ $\quad d(\mathbf{x}_1) = 1, d(\mathbf{x}_2) = -1, d(\mathbf{x}_3) = 1$



● C에 따른 유효성:

$$0 \le \alpha_i \le C, \qquad i = 1,2,\cdots,n$$

★ $C < 2$: 결정 직선 (2)만 유효

★ $2 \le C < 6.5$: 결정 직선 (1)과 (2)가 유효

★ $6.5 \le C$: 결정 직선 (1)~(3)이 모두 유효

★ [참고] 여러 결정 직선이 유효한 경우에는, $\tilde{L}(\boldsymbol{\alpha})$ 가 큰 걸 선택함

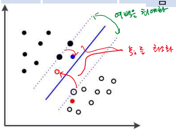
Case	분류 결과	샘플 위치	슬랙 변수
1	옳게 분류	분할 띠 바깥	$\xi = 0$
2	옳게 분류	분할 띠 안쪽	$0 < \xi \le 1$
3	틀리게 분류	결정 경계 넘음	$1 < \xi$

● C의 크기와 여백

$$J(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i \qquad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \ge 1 - \xi_i$$

★ C를 작게 하면 여백의 비중이 커진다. (여백 최대화)

★ C를 크게 하면 ξ_i의 비중이 커진다. (분할 띠 안쪽 샘플 수 최소화)



Contents

References

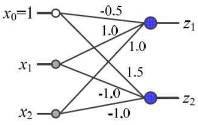
- *기계 학습* by 오일석, *패턴 인식* by 오일석

4. Nonlinear SVM (Kernel SVM)

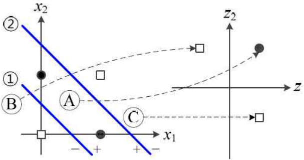
1. Feature Space Conversion
2. Kernel Trick
3. Kernel Function
4. Nonlinear SVM

Feature Space Conversion

- **공간 변환**은 기계 학습의 핵심 연산
  - 원래 특징 공간을 목적 달성에 더 유리한 새로운 공간으로 변환하는 작업
  - 앞에서 공부한 사례
    - ★ MLP: 은닉층을 이용한 특징 공간 변환을 통해 XOR 문제 해결



(a) 두 퍼셉트론을 병렬로 결합

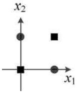


(b) 원래 특징 공간 x를 새로운 특징 공간 z로 변환

Feature Space Conversion – *Examples*

◆ Training Set

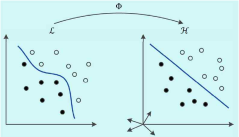
$$\mathbf{x}_1 = \begin{pmatrix}0\\0\end{pmatrix}, y_1 = -1, \mathbf{x}_2 = \begin{pmatrix}1\\0\end{pmatrix}, y_2 = 1, \mathbf{x}_3 = \begin{pmatrix}0\\1\end{pmatrix}, y_3 = 1, \mathbf{x}_4 = \begin{pmatrix}1\\1\end{pmatrix}, y_4 = -1$$



◆ 공간 변환의 목표

- **선형 분리 불가능한 입력 데이터를 다른 공간으로 매핑하여 선형 분리 가능하도록 만들자.**

$$\mathbf{\Phi}(\mathbf{x}) = \mathbf{\Phi}((x_1, x_2, \cdots, x_d)^{\mathrm{T}}) = \left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \cdots, \phi_q(\mathbf{x})\right)^{\mathrm{T}}$$



◆ 매핑 함수의 예

1) Perceptron (step activation)을 이용한 2차원 to 2차원 매핑 (d=q=2)
2) Gaussian function (RBF)을 이용한 2차원 to 2차원 매핑 (d=q=2)
3) 2원에서 3차원(고차원) 공간으로의 매핑 (d=2, q=3)

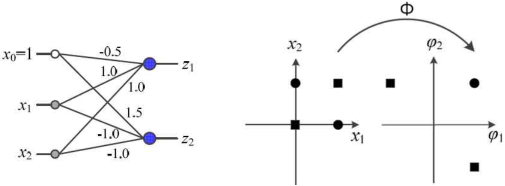
Feature Space Conversion – *Examples* (cont'd)

1) **Perceptron (step activation)을 이용한 2차원 to 2차원 매핑 (*d*=*q*=2)**

- Perceptron with step activation :

$$\phi(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x})$$

$$\boldsymbol{\Phi}(\mathbf{x}) = \left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x})\right)^{\mathrm{T}} = (\mathrm{sign}(x_1 + x_2 - 0.5), \mathrm{sign}(-x_1 - x_2 + 1.5))^{\mathrm{T}}$$



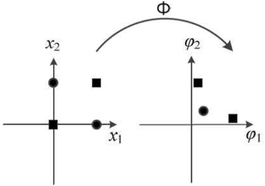
Feature Space Conversion – *Examples* (cont'd)

2) Gaussian function (RBF)을 이용한 2차원 to 2차원 매핑 (d=q=2)

- *RBF (Radial Basis Function)* : a real valued function φ whose value *depends only on the distance* between the input $\mathbf{x}$ and some fixed point $\mathbf{c}$ so that $\phi(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{c}\|)$.
- *Gaussian function* (one of an RBF): $\phi(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}\|_2^2}{2\sigma^2}\right)$

$$\boldsymbol{\Phi}(\mathbf{x}) = \left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x})\right)^{\mathrm{T}}$$

$$= \left(\exp\left(-\left\|\mathbf{x} - \begin{pmatrix}1\\1\end{pmatrix}\right\|_2^2\right),\ \exp\left(-\left\|\mathbf{x} - \begin{pmatrix}0\\0\end{pmatrix}\right\|_2^2\right)\right)^{\mathrm{T}}$$



Feature Space Conversion – *Examples* (cont'd)

3) 2원에서 3차원(고차원) 공간으로의 매핑 (d=2, q=3)

$$\mathbf{\Phi}(\mathbf{x}) = \left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \phi_3(\mathbf{x})\right)^{\mathrm{T}} \quad = \left(x_1^2,\ \sqrt{2}\,x_1 x_2,\ x_2^2\right)^{\mathrm{T}}$$

$\hat{\mathbf{a}} = ($

Feature Space Conversion + Linear SVM

$$\tilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\top}\mathbf{x}_j$$

Maximize:

Subject to:

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

$$0 \le \alpha_i, \qquad i = 1,2,\cdots,n$$

$$\tilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Kernel Trick

- $\mathbf{\Phi}(\mathbf{x})$로 변환한 $\mathcal{H}$ 공간에서 내적 연산을 원래 특징 공간 $\mathcal{L}$에서 커널함수 계산으로 대치
- 원래 특징 공간에서 쉽게 계산하지만 선형 분리 가능이라는 고차원 공간의 좋은 특성을 이용하는 셈 → 수학적 트릭으로 차원의 저주를 피함
- 제약 사항: $\mathcal{H}$ 공간에서의 연산이 내적으로 표현되어야 함 → 쌍대성duality을 이용하여 내적 표현 유도

Kernel Function

원래 특징 공간 $\mathcal{L}$에 정의된 두 특징 벡터 $\mathbf{x}$와 $\mathbf{z}$에 대해 $K(\mathbf{x}, \mathbf{z}) = \mathbf{\Phi}(\mathbf{x}) \cdot \mathbf{\Phi}(\mathbf{z})$인 변환함수 $\mathbf{\Phi}$가 존재하면 $K(\mathbf{x}, \mathbf{z})$를 커널함수라 부른다.

Kernel Function (cont'd)

Formal Definition

Kernel function — definition

Let $\mathcal{X}$ be any space. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a kernel function if for all $n \geq 1$, $x_1, x_2, ..., x_n \in \mathcal{X}$ and $c_1, ..., c_n \in \mathbb{R}$ we have

$$\sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) \geq 0.$$

Given a set of points $x_1, ..., x_n \in \mathcal{X}$, we define the corresponding kernel matrix as the matrix K with entries $k_{ij} = k(x_i, x_j)$.

The condition above is equivalent to saying that $c' K c \geq 0$ for all $c \in \mathbb{R}^n$.

Reference: "Statistical Machine Learning" lecture by Ulrike von Luxburg @Tubingen Univ.

Kernel Function (cont'd)

Theorem: *Kernel Implies Embedding*

A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel if and only if there exists a Hilbert space $\mathcal{H}$ and a map $\Phi : \mathcal{X} \to \mathcal{H}$ such that $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$.

If you have never heard of Hilbert spaces, just think of the space $\mathbb{R}^d$. The crucial properties are:

- $\mathcal{H}$ is a vector space with a scalar product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$
- Space is complete (all Cauchy sequences converge)
- Scalar product gives rise to a norm: $\|x\|_{\mathcal{H}} := \langle x, x \rangle_{\mathcal{H}}$

Reference: "Statistical Machine Learning" lecture by Ulrike von Luxburg @Tubingen Univ.

Cf.) Wikipedia:

- A **Hilbert space** is a vector space equipped with an inner product which defines a distance function for which it is a complete metric space.
- Hilbert spaces allow generalizing the methods of linear algebra and calculus from finite-dimensional Euclidean vector spaces to spaces that may be infinite-dimensional.
- Hilbert spaces arise naturally and frequently in mathematics and physics, typically as function spaces.

Kernel Function (cont'd)

- **Example**: $K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z})^2$ *is a kernel function?*
  - *d*=2, *q*=3 인 경우의 증명:

$$
\begin{aligned}
K(\mathbf{x}, \mathbf{z}) &= (\mathbf{x} \cdot \mathbf{z})^2 \\
&= x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\
&= \left(x_1^2, x_2^2, \sqrt{2}x_1 x_2\right) \cdot \left(z_1^2, z_2^2, \sqrt{2}z_1 z_2\right) \\
&= \mathbf{\Phi}(\mathbf{x}) \cdot \mathbf{\Phi}(\mathbf{z})
\end{aligned}
$$

Kernel Function (cont'd)

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + 1)^p$$

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(\frac{-\|\mathbf{x}-\mathbf{z}\|_2^2}{2\sigma^2}\right)$$

$$K(\mathbf{x}, \mathbf{z}) = \tanh(\alpha\mathbf{x} \cdot \mathbf{z} + \beta)$$

Kernel Function (cont'd)

- **Kernel Function은 어떻게 선택해야 하나?**
  - In general, it is really difficult to prove that a certain function K is indeed a kernel.
  - In practice, it usually does not work to come up with a nice *similarity function* and "hope" that it is a kernel.

- **There are *some simple rules* that can help to transform and combine elementary kernels:**

Assume that $k_1, k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ are kernel functions. Then:

- $\tilde{k} = \alpha \cdot k_1$ for some constant $\alpha > 0$ is a kernel.
- $\tilde{k} = k_1 + k_2$ is a kernel
- $\tilde{k} = k_1 \cdot k_2$ is a kernel
- The pointwise limit of a sequence of kernels is a kernel.
- For any function $f : \mathcal{X} \to \mathbb{R}$, the expression $\tilde{k}(x, y) := f(x)k(x, y)f(y)$ defines a kernel.

  In particular, $\tilde{k}(x, y) = f(x)f(y)$ is a kernel.

Reference: "Statistical Machine Learning" lecture by Ulrike von Luxburg @Tubingen Univ.

Nonlinear SVM (*Kernel SVM*)

◆ Wolfe Dual (Soft-Margin Linear SVM)

Maximize:

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

Subject to:

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

$$0 \le \alpha_i \le C, \qquad i = 1,2,\cdots,n$$

◆ Kernel Trick

$$\tilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

◆ 커널 함수에 대응하는 **매핑 함수는 몰라도 된다**. 단지 존재한다는 사실만 알면 된다.

- 왜? 실재 계산은 벡터의 내적을 커널 함수로 대치하여 하면 되기 때문

5. SVM Implementation

1. SVM Learning
2. SVM Prediction
3. Open Source SVMs

SVM Learning

Nonlinear SVM Learning Problem (Finding α_i's)

Maximize: $$\tilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Subject to: $$\sum_{i=1}^{n} \alpha_i y_i = 0$$

- 목적함수 $\tilde{L}(\boldsymbol{\alpha})$는 1차 항이 n개, 2차 항이 n^2개인 아주 복잡한 식
- 등식조건 1개와 부등식 조건 n개 포함

- Linear SVM은 $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ 인 경우로 볼 수 있음
- **n이 작을 때:** 해석적으로(analytically) 해를 구한다. (앞 강의 example들 참조)
- **n이 클 때:** 수치 최적화로(numerical optimization) 해를 구한다. (α_i 초기화 → iterative update)

SVM Learning (cont'd)

◆ SVM 학습 전략

- 원래 문제를 다룰 수 있을 정도의 **작은 문제로 분해**해 보자.
- 먼저, n개의 Lagrange Multiplier α_i를 ***active set* Y**와 ***inactive set* Z**로 나눈다.
- Z에 속한 α_i는 상수로 간주하고, Y에 대해 최적화를 수행한다.
- 이 과정을 전체 최적화 조건을 만족할 때까지 반복한다.

알고리즘 11-1 SVM 학습

입력: 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, $\mathbb{Y} = \{y_1, y_2, \cdots, y_n\}$
선형과 비선형 선택, 비선형을 선택한 경우 커널함수
출력: 라그랑주 승수 $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_n)^{\mathrm{T}}$

$\alpha_1, \alpha_2, \cdots, \alpha_n$을 초기화한다.
repeat
    q개 라그랑주 승수를 선택하여 집합 Y에 넣고, 나머지는 Z에 넣는다.
    Z에 있는 승수는 상수, Y에 있는 승수는 변수로 취급한다.
    if(선형) [문제 11-5]를 분해한다.
    else [문제 11-10]을 분해한다.
    분해된 문제를 풀어 Y에 있는 승수를 새로운 값으로 변경한다.
until (멈춤 조건)

SVM Learning (cont'd)

◆ 대표적 SVM 최적화 알고리즘

- **SMO** (Sequential Minimal Optimization)
  - ★ q=2를 사용
  - ★ 두 개의 Lagrange Multiplier만 구하면 됨 (해석적으로 풀 수 있음)
- **Cutting-Plane** Algorithm [Joachims 06]
  - ★ KDD 국제학회 최고 논문상
  - ★ SMO보다 빠름

- SVM 구현은 여전히 중요한 연구주제

SVM Prediction

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \, y_i \mathbf{x}_i = \sum_{\alpha_k \neq 0} \alpha_k \, y_k \mathbf{x}_k$$

$$y_i(\mathbf{w}^T\mathbf{x}_i + b)$$

$$\mathbf{w}^T\mathbf{x}_i + b = y_i$$

$$b = y_i - \mathbf{w}^T\mathbf{x}_i$$

$$d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b \gtrless 0 \quad \begin{cases} out = 1 \\ out = -1 \end{cases}$$

SVM Prediction (cont'd)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \, y_i \mathbf{x}_i = \sum_{\alpha_k \neq 0} \alpha_k \, y_k \mathbf{x}_k$$

$$b = y_i - \mathbf{w}^{\mathrm{T}} \mathbf{x}_i$$

$$b_K = y_i - \sum_{k=1}^{n} \alpha_k \, y_k K(\mathbf{x}_k, \mathbf{x}_i)$$

$$\begin{aligned} \mathbf{w}^{\mathrm{T}} \mathbf{x}_i &= \langle \mathbf{w}, \mathbf{x}_i \rangle \\ &= \left\langle \sum_{k=1}^{n} \alpha_k \, y_k \mathbf{x}_k, \mathbf{x}_i \right\rangle \\ &= \sum_{k=1}^{n} \alpha_k \, y_k \langle \mathbf{x}_k, \mathbf{x}_i \rangle \end{aligned}$$

$$d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \mathbf{x} + b$$

$$d_K(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, y_i K(\mathbf{x}_i, \mathbf{x}) + b \gtrless 0 \begin{cases} out = 1 \\ out = -1 \end{cases}$$

즉 Kernel 함수만 바꿔주면 완성!

Open Source SVMs

- SVMLight library
  - http://svmlight.joachim.org
  - Osuna Algorithm 사용 [Joachims 1999]

- LIBSVM library ← *Recommended*
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm
  - 개선된 SMO Algorithm 사용 [Fan 2005]

- Others
  - SVMJS (데모)
  - R, Matlab, SAS, Python 언어
  - OpenCV
  - Scikit Learn

SVM song

- https://www.youtube.com/watch?v=g15bqtyidZs

SVM Demo

- https://cs.stanford.edu/people/karpathy/svmjs/demo/