

## [ML-12] Constrained Optimization

## Convex Set &amp; Convex function.

Constrained optimization의 경우 convex를 전제로 한다.

Convex set.

$\Rightarrow$  for any  $x, y \in C$ , any scalar  $0 \leq \theta \leq 1$

$$\theta x + (1-\theta)y \in C$$

Convex Function.

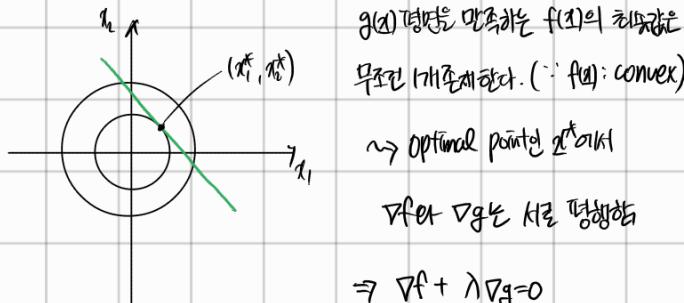
$$\Rightarrow f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$

$\hookrightarrow$  convex combination 한 것의 합집합이

각각의 합집합을 convex combination 한 것과 항상 같다.

## Lagrange Multiplier for Equality Constraints

$\Rightarrow$  minimize  $f(x)$ , subject to  $g(x)=0$



여기서 함수  $L$ 을 다음과 같이 정의할 수 있다.

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

$$\Rightarrow 1) \nabla_x L(x, \lambda) = 0 \quad \Rightarrow \nabla_x L(x, \lambda) = 0$$

$$2) \quad g(x) = 0 \quad \Rightarrow \nabla_\lambda L(x, \lambda) = 0$$

Ex)

$$f(x_1, x_2) = x_1^2 + x_2^2 - 1, \quad g(x_1, x_2) = x_1 + x_2 - 1$$

$$L(x, \lambda) = f(x) + \lambda g(x) = x_1^2 + x_2^2 - 1 + \lambda(x_1 + x_2 - 1)$$

$$\nabla_{x_1} L = 2x_1 + \lambda = 0 \quad \Rightarrow \quad (x_1^*, x_2^*) = (0.5, 0.5)$$

$$\nabla_{x_2} L = 2x_2 + \lambda = 0$$

$$\nabla_\lambda L = x_1 + x_2 - 1 = 0$$

## Lagrange Multiplier for Inequality Constraints

$\rightarrow$  minimize  $f(x)$ , subject to  $g(x) \leq 0$

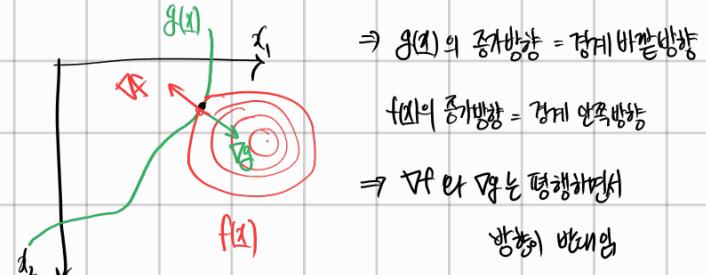
Inequality constraint의 경우 constraint가

active한 경우와 inactive한 경우로 나누어 볼 수 있다.

i) active case:  $g(x) = 0$

$\rightarrow$  active라는 말은 constraint 조건이滿足되었다는 것이다.

즉  $f(x)$ 의 원래 최대값과  $g(x)$ 의 경계값에 있다는 것이다.



$$\Rightarrow \boxed{\nabla_x f(x) + \lambda \nabla_x g(x) = 0, \lambda \geq 0}$$

ii) Inactive case:  $g(x) < 0$

$\rightarrow$  Inactive라는 말은 constraint가  $g(x) \leq 0$ 을 만족하지 않는 것이다.

$\hookrightarrow$   $f(x)$ 의 원래 최대값과 이미  $g(x)$ 의 경계안쪽에 존재한다는 것이다.

$\therefore$  이 경우에는 unconstrained optimization을 풀어야 한다.

$$\Rightarrow \boxed{\nabla_x f(x) = 0}$$

$\Rightarrow$  KKT condition.

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

$$\textcircled{1} \quad \nabla_x L(x, \lambda) = 0$$

$$\textcircled{2} \quad g(x) \leq 0$$

$$\textcircled{3} \quad \lambda \geq 0$$

$$\textcircled{4} \quad \lambda g(x) = 0$$

$\rightarrow$  active:  $g(x) = 0, \lambda \geq 0$

inactive:  $g(x) < 0, \lambda = 0$

$\Rightarrow$  active ( $g(x) = 0$ ):  $\nabla_x L(x, \lambda) = 0$  for  $\lambda \geq 0$ ,  $\nabla_\lambda L(x, \lambda) = 0$

inactive ( $g(x) < 0$ ):  $\nabla_x L(x, \lambda) = 0$  for  $\lambda = 0$

Ex)

$$J(\theta) = \theta_1^2 + 2\theta_2^2$$

s.t.  $\underbrace{2\theta_1 + \theta_2 \geq 1}_{\hookrightarrow g(\theta) = 1 - 2\theta_1 - \theta_2}$

$$L(\theta, \lambda) = \theta_1^2 + 2\theta_2^2 + \lambda(1 - 2\theta_1 - \theta_2)$$

- active:  $(\nabla_{\theta} L(\theta, \lambda)) = 0$  for  $\lambda > 0$ ,  $\nabla_{\lambda} L(\theta, \lambda) = 0$

$$\nabla_{\theta_1} L(\theta, \lambda) = 2\theta_1 - 2\lambda = 0 \quad \Rightarrow \theta_1 = \lambda, \theta_2 = \lambda/4$$

$$\nabla_{\theta_2} L(\theta, \lambda) = 4\theta_2 - \lambda = 0 \quad | -2\lambda - \lambda/4 = 0$$

$$\nabla_{\lambda} L(\theta, \lambda) = 1 - 2\theta_1 - \theta_2 = 0 \quad \boxed{\lambda = \frac{4}{5}}, \theta_2 = \frac{1}{5}$$

- inactive ( $\nabla_{\theta} L(\theta, \lambda)$  for  $\lambda = 0$ ,  $g(\theta) < 0$ )

$$\nabla_{\theta_1} L(\theta, \lambda) = 2\theta_1 - 2\lambda = 2\theta_1 = 0 \quad \boxed{\theta_1 = 0}$$

$$\nabla_{\theta_2} L(\theta, \lambda) = 4\theta_2 - \lambda = 4\theta_2 = 0 \quad \Rightarrow \theta_2 = 0$$

$$\Rightarrow g(\theta) = 1 > 0$$

$\Rightarrow g(\theta) < 0$ 이어야 하는 경우.

## Lagrangian Duality - Linear Programming

$$\min_{x \in \mathbb{R}^n} [C^T x] = f(x) \quad \text{subject to } [Ax \leq b] \rightarrow g(x) = Ax - b \leq 0$$

$$L(x, \lambda) = f(x) + \lambda^T g(x) = C^T x + \lambda^T (Ax - b)$$

$$= (C + A^T \lambda)^T x - \lambda^T b$$

$$\nabla_x L(x, \lambda) = C + A^T \lambda = 0 \Rightarrow L(x, \lambda) = -\lambda^T b$$

$$\hookrightarrow D(\lambda) = -\lambda^T b \Rightarrow \max_{\lambda \in \mathbb{R}^m} -\lambda^T b$$

$$\text{subject to } C + A^T \lambda = 0 \\ \lambda \geq 0$$

$\Rightarrow$  만약  $m$  (제약조건 수)와  $d$  (식의 차원)을 비교하여

더 개방성이 좋은 방향을 선택해 solving 하자.

## Lagrange Multiplier for Multiple Constraints

$\rightarrow \min_{x \in \mathbb{R}^n} f_0(x)$ , subject to  $f_i(x) \leq 0, h_j(x) = 0$

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{j=1}^k \nu_j h_j(x) \leq f_0(x)$$

## Lagrangian Duality

$\rightarrow$  primal problem  $\neq$  not Lagrangian dual problem  $\neq$  같은 결과.

Primal Problem

$$\min_x f(x) \quad \text{subject to } g_i(x) \leq 0 \quad i=1, \dots, m$$

## Lagrangian Dual Problem

$$\max_{\lambda \in \mathbb{R}^m} D(\lambda) = \min_{x \in \mathbb{R}^n} L(x, \lambda) \quad \text{subject to } \lambda \geq 0$$

$\downarrow$

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

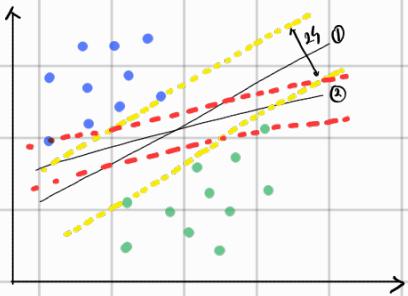
$$\nabla_x L(x, \lambda) = 0 \Rightarrow x^* = \dots$$

$$\therefore D(\lambda) = L(x^*, \lambda)$$

## [ML-13] Support Vector Machines

### Linear SVM with Hard Margin

먼저 Linear SVM의 대장치는 선형분류가 가능한 문제에 대해 생각한다는 것이다. 그렇다면 다음과 같이 선형분류가 가능한 문제를 생각해보자.



위와 같은 문제에서 선형분류가 가능한 결정계선은 여러개가 나올 수 있다.

→ 결정계선 정하는 방향: 임의의 직선방향  $w$ 에 대해 가장 가까운 샘플까지의 거리가 최대로  $b$ 를 정한다.

그리고 샘플과의 거리를 여백(margin)이라고 한다.

SVM은 이렇게 여러개의 결정계면 중에서 여백이 가장 큰 초평면을 찾는 것이다.

$$\text{Margin} = 2b = \frac{2|d(w)|}{\|w\|_2} = \frac{2}{\|w\|_2}$$

### Problem Definition

Training set:  $D = \{(x^1, y^1), \dots, (x^n, y^n)\}$

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

상동분류기의 관점에서

$$\text{Subject to: } \boxed{w^T x^i + b} \geq 1, \quad y^i = 1$$

$d(x^i) \rightarrow |d(x^i)| = 1$  이 정도로

$$w^T x^i + b \leq -1, \quad y^i = -1$$

$y^i (w^T x^i + b - 1) = 0$   
을 만족하는 데이터  
 $\Downarrow$

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

$$\text{Subject to: } y^i (w^T x^i + b) - 1 \geq 0$$

### Lagrangian

$$L(w, b, \alpha) = \frac{\|w\|_2^2}{2} - \sum_{i=1}^n [\alpha^i (y^i (w^T x^i + b) - 1)]$$

$$\Rightarrow f(\alpha) = \frac{\|w\|_2^2}{2}, \quad g(\alpha) = \sum_{i=1}^n \alpha^i (y^i (w^T x^i + b) - 1)$$

↓ KKT condition.

$$\frac{\partial L(w, b, \alpha)}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^n \alpha^i y^i x^i$$

$$\frac{\partial L(w, b, \alpha)}{\partial b} = 0 \Rightarrow \sum_{i=1}^n \alpha^i y^i = 0$$

$$\alpha^i \geq 0$$

$$\Rightarrow \text{active } (g(\alpha) < 0, \lambda > 0)$$

$$: y^i (w^T x^i + b) - 1 = 0, \quad \alpha^i > 0 \rightarrow \text{이걸 만족하는 샘플} \\ = \text{support vector}$$

$$\text{inactive } (g(\alpha) > 0, \lambda = 0)$$

$$: y^i (w^T x^i + b) - 1 > 0, \quad \alpha^i = 0$$

### Wolfe Dual Problem

$$\text{Minimize: } L(w, b, \alpha) = \frac{\|w\|_2^2}{2} - \sum_{i=1}^n \alpha^i (y^i (w^T x^i + b) - 1)$$

$$\text{Subject to: } w = \sum_{i=1}^n (\alpha^i y^i x^i)$$

$$\sum_{i=1}^n \alpha^i y^i = 0$$

$$\alpha^i \geq 0$$

||

$$\text{Maximize: } L(\alpha) = \sum_{i=1}^n \alpha^i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha^i \alpha^j y^i y^j x^i x^j$$

$$\text{Subject to: } \sum_{i=1}^n \alpha^i y^i = 0,$$

$$\alpha^i \geq 0$$

## Linear SVM with Soft Margin

### Soft Margin & slack Variables

- Soft Margin은 hard Margin과는 다르게 분할 때 안에 샘플을 허용하는 것을 말한다. 이때 샘플은 다음 3가지 케이스 중 하나에 속하며 slack variables  $\xi_i$ 를 통해 하나로 표현한다.

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

$$\Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq -\xi_i$$

### Problem Definition

→ 예측을 가능한 케이스에서(목표1), 더 많은 샘플수를 가능한  
정제 하는(목표2) 결정 초평면의 쓰기를 하는 것.

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

$$\text{subject to: } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq -\xi_i, \\ 0 \leq \xi_i$$

- $\Rightarrow$  ① C를 크게 하면 예측의 범위가 커진다.  $\rightarrow$  예측 정밀화  
② C를 크게 하면  $\xi_i$ 의 범위가 커진다.  $\rightarrow$  분할 띠 안쪽 샘플수  
최대화.

### Lagrangian

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

↓ KKT Condition

$$\begin{aligned} ① \frac{\partial L}{\partial \mathbf{w}} = 0 &\Rightarrow \mathbf{w} = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i & ④ \alpha_i \geq 0 \\ ② \frac{\partial L}{\partial b} = 0 &\Rightarrow \sum_{i=1}^n \alpha_i y_i = 0 & ⑥ \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i) = 0 \\ ③ \frac{\partial L}{\partial \xi_i} = 0 &\Rightarrow C = \alpha_i + \beta_i & ⑦ \beta_i \xi_i = 0 \end{aligned}$$

### Wolfe Dual Program

Minimize:

$$L(\mathbf{w}, \mathbf{b}, \alpha, \beta) = \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^n \xi_i$$

$$- \left( \sum_{i=1}^n \alpha_i y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right) + \sum_{i=1}^n \beta_i \xi_i$$

subject to:

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

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

Maximize:

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

subject to:

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

↳ hard margin / soft margin  
추가한 부분

## Nonlinear SVM (kernel SVM)

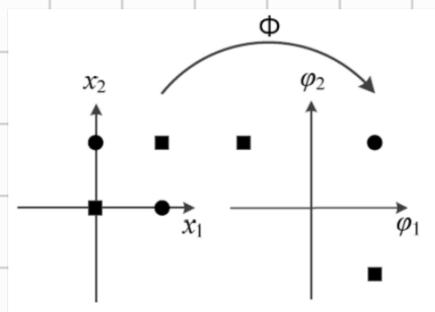
### Feature Space Conversion

공간변환의 목표: 선형 분류가 불가능한 데이터를,  
다른 공간으로 대체하여 선형분리 가능하도록 만든다.

### Ex) Perceptron

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

$$\phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}))^T = (\text{Sign}(x_1 + x_2 - 0.5), \text{Sign}(-x_1 - x_2 + 1.5))^T$$

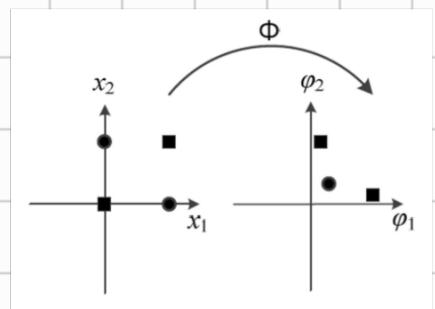


### Ex) Gaussian

$$\phi(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}\|_2^2}{2\sigma^2}\right)$$

$$\phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}))$$

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



### Kernel Trick

→ 공간변환의 경우 모든 데이터셋이 적용하기 힘들다.

데이터셋이 굉장히 넓을 때는 계산량이 굉장히 많다.

→ 계산트릭은 데이터셋을 계산함수로 대체하여 계산하기에

계산복잡도를 줄일 수 있다

→ 대용량기반 방식으로 학습이 끝나면 훈련과정은 대용량에 저장하고

있다가 예측에 사용함

## kernel Function

특정 공간 \$V\$에 정의된 두 특성벡터 \$1\$와 \$2\$에 대해 \$k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{z})\$

변환함수 \$\Phi\$가 존재하면 \$k(\mathbf{x}, \mathbf{z})\$를 계산할수라고 한다.

넓이 \$N\$에는 계산함수

- polynomial kernel : \$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + 1)^P\$

- Gaussian (RBF) kernel : \$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2}\right)\$

- Hyperbolic tangent kernel : \$k(\mathbf{x}, \mathbf{z}) = \tanh(\alpha \mathbf{x} \cdot \mathbf{z} + \beta)\$

→ 각각의 함수는 hyper parameter를 가짐, tuning이 필요하다.

tuning을 할때는 validation set을 이용

### kernel function property

$$\tilde{k} = \alpha k_1$$

$$\left\{ k_n \right\}_{n=1,2,\dots} \xrightarrow{n \rightarrow \infty} k_n \text{ is a kernel}$$

$$\tilde{k} = k_1 k_2$$

$$\tilde{k}(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) k(\mathbf{x}, \mathbf{y}) f(\mathbf{y})$$

## Wolfe Dual (Soft-Margin Linear SVM)

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

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

↓  
kernel Trick

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

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

## SVM Implementation

### SVM Learning

→ 노이즈 없는 (샘플이 지울 때)

: 핵기법으로 해를 구한다.

노이즈 있는 (샘플이 많을 때)

: 소거 최적화로 해를 구한다.

↳ ① 원래 문제를 다룰 수 있을 정도의 작은 문제로 분해

② 각각의  $\alpha^i$  를 activation set  $\mathcal{Y}$  or inactivation set  $\mathcal{Z}$

나눈다.

③  $\mathcal{Z}$ 에 속한  $\alpha^i$  를 살피고, 그에 대해 최적화를

수행한다.

④ 이 과정을 전체 최적화 조건을 만족할 때까지 반복한다.

### SVM Prediction

#### ① Linear SVM Prediction

- weight 벡터:  $w = \sum_{i=1}^n \alpha^i y^i \underline{x}^i = \sum_{\alpha \neq 0} \alpha^k y^k \underline{x}^k$

- Bias 벡터:

$$SV(\underline{x}^i) \text{ 를 } 1 \text{ 으로 대체, } b = y^i - w^T \underline{x}^i$$

- Output 벡터:

$$d(\underline{x}) = w^T \underline{x} + b \quad \begin{cases} > 0 \rightarrow \text{out} = 1 \\ < 0 \rightarrow \text{out} = -1 \end{cases}$$

#### ② Nonlinear SVM Prediction

- weight 벡터:  $w = \sum_{i=1}^n \alpha^i y^i \underline{x}^i = \sum_{\alpha \neq 0} \alpha^k y^k \underline{x}^k$

- Bias 벡터:

$$SV(\underline{x}^i) \text{ 를 } 1 \text{ 으로 대체, } b_k = y^i - \sum_{k=1}^n \alpha^k y^k K(\underline{x}^k, \underline{x}^i)$$

- Output 벡터:  $\rightarrow w^T \underline{x}^i = \sum_{k=1}^n \alpha^k y^k K(\underline{x}^k, \underline{x}^i)$

$$d_K(\underline{x}) = \sum_{i=1}^n \alpha^i y^i K(\underline{x}^i, \underline{x}) + b \quad \begin{cases} > 0 \rightarrow \text{out} = 1 \\ < 0 \rightarrow \text{out} = -1 \end{cases}$$