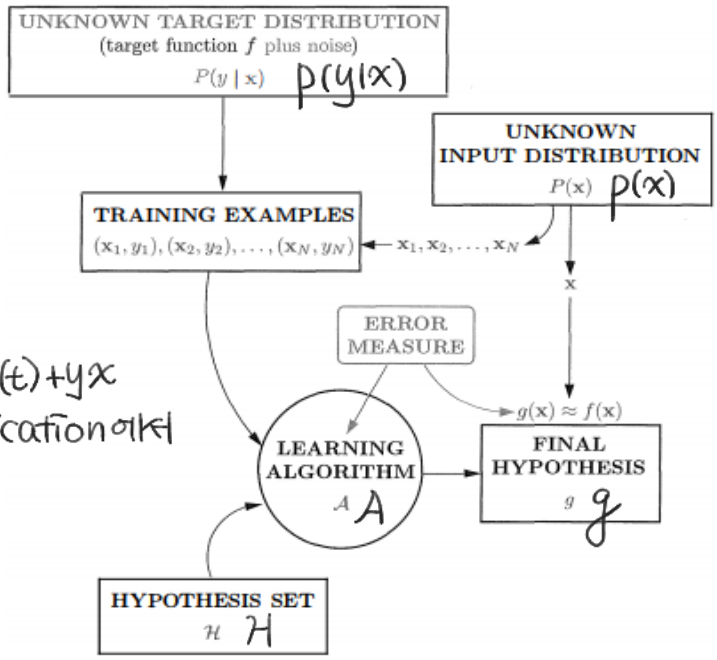
Lecture1. Learning from data



- Learning from data
  1) data
  2) pattern
  3) 수학적인 분석 불가능
- Perceptron
  h(x) = sign (wTx)
- => 우리는 supervised learning

※ PLA
$w(t+1) = w(t) + yx$
classification에서

Lecture2. Feasibility of learning

- Hoeffding inequality
  $\mathbb{P}[|\hat{\mu} - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$
- Multiple hypothesis : H (hypothesis set)에 여러 개의 h 가 있는 경우
  $\mathbb{P}[|\mathrm{E_{in}}(g) - \mathrm{E_{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$: M 배 looser, M must be finite
- Feasibility of learning
  1. Eout(g) ~ Ein(g) 라고 할 수 있는가?
  2. Ein(g)를 충분히 작게 만들 수 있는가?

문제의 Ein, Eout

- Error and Noise (다르다)

→ $E_{out}(h) = \mathbb{E}_x[e(h(x), f(x))]$
noisy하면 $E_{out}(h) = \mathbb{E}_{x,y}[e(h(x), y)]$

f가 p(y|x)

Lecture3. Linear classification and regression

small VC dim → generalization 퍼포먼스 좋아.

Linear classification

use signal : $w^T x$,
이때 $x \in \mathbb{R}^d$, $w \in \mathbb{R}^d$

- Linear model : 1) Ein ~ Eout
  VC dimension, dvc : d+1
  VC generalization bound : (m: effective 한 hypothesis set 의 크기)

$E_{out} \leq E_{in} + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$, $m_H(N) \leq N^{d_{vc}} + 1$

probability $\geq 1 - \delta$

따라서, 높은 확률로 $\mathrm{E_{out}}(g) = \mathrm{E_{in}}(g) + O\left(\sqrt{\frac{d}{N} \ln N}\right)$ 이므로, large N 에서 Ein~Eout

- Linearly in-separable data 1. 전체적으로 linearly separable 하지만, outlier(noise)가 존재
  minimum Ein 을 가지도록 문제를 풀자.

ex2) digit recognition
raw input space -> feature input space
feature space : reduce dimension, better handle curse of dimensionality,
but can cause loss of information
이후, PLA 나 pocket algorithm 으로 해결)

① 우리가 고려해야 하는것 자유적으로 증가한다.
② 전체에서 우리가 아는 부분이 자유적으로 감소한다.

· linearly separable
: Ein(w*)=0 만족하는 w*가 언제나 존재
→ PLA 등
· separable x
: Ein 작게 만들수 있다.
→ Pocket, SVM 등

Linear regression : y is continuous

- assumption : homoscedasticity : (same finite variance for each value of x) — same bell shape
  여러 평가방법 1) least squares, 2) maximum likelihood
- OLS solution w* : minimizing Ein(w)

OLS definition :

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n)-y_n)^2 = \frac{1}{N}\|Xw-y\|^2$$

linear regression 에서, h=wTx, (wTx called signal)

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}_{N}, \quad w \in \mathbb{R}^d$$

이때, Ein(w) is continuous, differentiable, convex, ∇Ein(w)=0

$$\nabla E_{in}(w) = \frac{2}{N}(X^TXw - X^Ty) = 0 \text{ 을 만족하는 } w.$$

즉 normal equation(XTXw=XTy)을 만족하는 w를 찾아야 한다.

$X^TXw = X^Ty$

1) X^TX is invertible, $w = (X^TX)^{-1}X^Ty = X^+y$ pseudo-inverse of X), most cases → 1 step
2) X^TX is not invertible, X# still defined, but no unique solution
   invertible 하지 않으면 SVD 등 다양한 방법이 존재

- Hat matrix H (projection matrix)

H 는 Ein 과 Eout 의 관계를 나타낸다.

observed y / fitted ŷ

in-sample error 에 의해 $y \neq \hat{y}$

ŷ는 y를 X의 column space로 orthogonal projection.

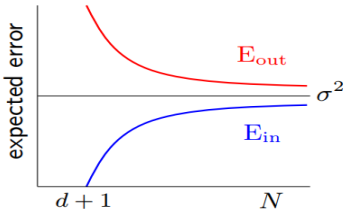
$$\text{fitted } \hat{y} = Xw^* = X(X^TX)^{-1}X^Ty = Hy, \quad H = X(X^TX)^{-1}X^T$$

- generalization bound : Eout = Ein + O(d/N),

if assume $y = f(x) + e$, 0 mean, $0 < \infty$

$$E_{in} = \sigma^2\left(1 - \frac{d+1}{N}\right) < \sigma^2$$

$$E_{out} = \sigma^2\left(1 + \frac{d+1}{N}\right) > \sigma^2$$



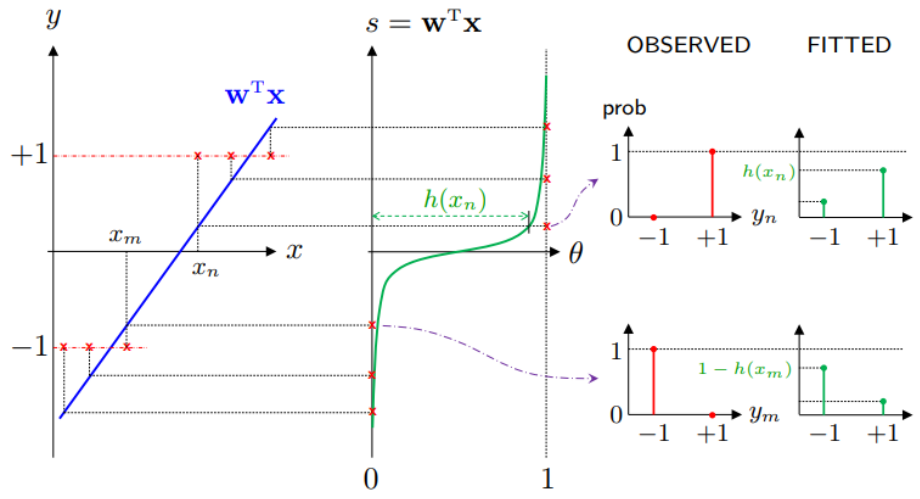
Lecture4. Logistic regression

- Logistic regression

input : h(x) = θ(wTx), θ is sigmoid, θ 의 특징 : 1- θ(s) = θ(-s)

$$\theta(w^Tx) = \frac{1}{1+e^{-w^Tx}} : \text{sigmoid}$$

- Big picture



$$\mathbb{P}[y=+1|\mathbf{x}] = f(\mathbf{x}) \sim h(\mathbf{x}) = \theta(\mathbf{w}^T\mathbf{x})$$

$$P(y|\mathbf{x}) = h(\mathbf{x})^{[\![y=+1]\!]}(1-h(\mathbf{x}))^{[\![y=-1]\!]} = \theta(y\mathbf{w}^T\mathbf{x})$$

* $h(x) = \theta(w^Tx) = \frac{1}{1+e^{-w^Tx}}$

$$P(y|x) = \theta(yw^Tx)$$

$$= \begin{cases} \theta(w^Tx) & \text{if } y=1 \\ \theta(-w^Tx) & \text{if } y=-1 \\ \downarrow \\ 1-\theta(w^Tx) \end{cases}$$

$$= h(x)^{[y=1]} + (1-h(x))^{[y=-1]}$$

Logistic regression 의 Error ⟶ 유도 과정에서 θ(-s)=1-θ(s) 임을 이용.

- Learning target (target: noisy)
  f(x) = P(y|x) = (h(x) for y=+1, 1-h(x) for y=-1)
- Error measure : simple 한것 말고, cross-entropy 사용
  2) cross-entropy measure $\mathrm{E}_{\mathrm{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1+e^{-y_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)$ (error property and convex)

$$\ln \frac{1}{P(y_n|x_n)} = [y=+1] \ln \frac{1}{h(x_n)} + [y=-1] \ln \frac{1}{1-h(x_n)}$$

(observed, fitted)

- Error measure 유도 방법 1: likelyhood
  likelihood : P(y|x) = θ(ywTx)
  (이 때, 1-θ(x)= θ(-x) 임을 이용)

$$P(y_1|\mathbf{x}_1)P(y_2|\mathbf{x}_2)\cdots P(y_N|\mathbf{x}_N) = \prod_{n=1}^{N} P(y_n|\mathbf{x}_n)$$

  likelihood of training data :
  maximum likelihood h 를 select
  수식을 변형 : minimizing $-\frac{1}{N}\ln\left(\prod_{n=1}^{N} P(y_n|\mathbf{x}_n)\right) = \frac{1}{N}\sum_{n=1}^{N}\ln\frac{1}{P(y_n|\mathbf{x}_n)}$,wrt w
  결론 : in-sample error : $\mathrm{E}_{\mathrm{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1+e^{-y_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)$
  P(y|x) = θ(ywTx)를 대입하고, sigmoid 함수를 대입해 정리한 식
  implied point-wise error : ln(1+e^(-ywTx))

- Error measure 유도 방법 2: cross-entropy
  cross entropy : (p)log(1/q)+(1-p)log(1/1-q)
  : 'error' for 'observed' pmf {p, 1-p} by 'fitted' pmf {q, 1-q}

$$\begin{aligned} NLL(\mathbf{w}) &\triangleq -\log\left\{\prod_{n=1}^{N} P(y_n|\mathbf{x}_n)\right\} \\ &= -\log\left\{\prod_{n=1}^{N} h(\mathbf{x}_n)^{[\![y_n=+1]\!]}(1-h(\mathbf{x}_n))^{[\![y_n=-1]\!]}\right\} \\ &= \sum_{n=1}^{N}\left\{[\![y_n=+1]\!]\log\frac{1}{h(\mathbf{x}_n)} + [\![y_n=-1]\!]\log\frac{1}{1-h(\mathbf{x}_n)}\right\} \end{aligned}$$

: cross entropy error function

Logistic regression 의 optimization : iterative (e.g. gradient descent)

Iteratively solve unconstrained optimization

- set ∇Ein=0 (in error surface)

$$\nabla_{\mathbf{w}}\mathrm{E}_{\mathrm{in}} = -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n x_n}{1+e^{y_n \mathbf{w}^{\mathrm{T}} x_n}} = \frac{1}{N}\sum_{n=1}^{N} -y_n x_n \theta(-y_n \mathbf{w}^{\mathrm{T}} x_n)$$

no analitic solution : iteratively 풀자.
어디로? −∇Ein 방향.
얼만큼? 가만큼. |∇Ein| 만큼.
언제 끝날까? 그래디언트 디센트
① iteration 수로 bound
② gradient 작고
③ error itself 작고

- Gradient descent
  - Algorithm : gradient descent
    for (interatively)
    compute ∇Ein(w(t))
    set v^ = negative gradient = −∇Ein(w(t)),
    - v^is due to cancellation of |∇Ein|
    update weights w(t+1) = w(t) - η∇Ein(w(t))

매번 모든 n에 Ein을 계산,
w(t+1) = w(t) − η∇Ein(w(t))
모든 n이 아니라 subset, 또 1개의 point로 update 하는 방법 사용하기도.

  - ∇Ein 의 계산이 O(N) time if batch-> 다 볼 필요 없는 mini batch, stochastic 사용하기도

Linear model review

	linear classification	linear regression	logistic regression (= probability estimation)
$\mathcal{Y}$	$\{-1,+1\}$	$\mathbb{R}$	$\{-1,+1\}$
$\hat{y}=h(\mathbf{x})$	$\text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$	$\mathbf{w}^{\mathrm{T}}\mathbf{x}$	$\theta^{\star}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$
$e(\hat{y},y)$	0-1 loss $[\![\hat{y}\neq y]\!]$	squared error $(\hat{y}-y)^2$	cross-entropy error $[\![y=+1]\!]\ln\frac{1}{\hat{y}}+[\![y=-1]\!]\ln\frac{1}{1-\hat{y}}$
$\mathrm{E}_{\text{in}}(h)$	$\frac{1}{N}\sum_{n=1}^{N}[\![h(\mathbf{x}_n)\neq y_n]\!]$	$\frac{1}{N}\sum_{n=1}^{N}(h(\mathbf{x}_n)-y_n)^2$	$\frac{1}{N}\sum_{n=1}^{N}\ln\left(1+e^{-y_n\mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)$
opt.	combinatorial optimization (NP-hard) PLA.	set $\nabla\mathrm{E}_{\text{in}}(\mathbf{w})=0$ (closed-form solution exists) X+.	set $\nabla\mathrm{E}_{\text{in}}(\mathbf{w})=0$ iterative optimization (*e.g.* gradient descent)

* VC generalization bound

① linear classification

$d_{VC}=d+1$, $m_H(N)\le N^{d_{VC}+1}$

$\therefore E_{out}(g)=E_{in}(g)+O\left(\sqrt{\frac{d}{N}\ln N}\right)$

② linear regression

$\therefore E_{out}(g)=E_{in}(g)+O\left(\frac{d}{N}\right)$

→ linear는 first try로 좋아. simple, robust, work well, generalize well

Lecture5. Artificial neural networks

- 분포를 구현하기 위한 것이 함수

	K class	Binary
함수	Softmax function	Logistic function
1 번의 event	멀티눌리 distribution = categorical distribution (주사위 한번 던지는 것)	베르눌리 distribution (동전 한번 던지는 것)
N 번의 event	Multinomial distribution	Binomial distribution

softmax function $\sigma:\mathbb{R}^K\mapsto\mathbb{R}^K$

$$\sigma(\mathbf{h})_j=\frac{e^{h_j}}{\sum_{k=1}^{K}e^{h_k}}$$

- Error measures

1) e_k : 하나의 원소에 대해 : $e_k=h_k-y_k$

2) E_n : 하나의 vector에 대해 : $E_n=\frac{1}{2}\sum_{k=1}^{d}e_{k,n}^2$

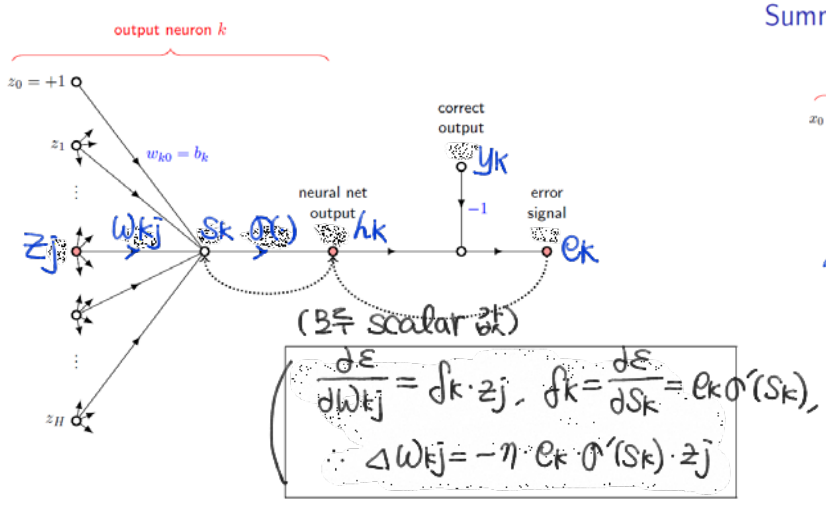
3) E_D : 전체 vector set, data에 대해 : $E_D=\frac{1}{2N}\sum_{n=1}^{N}\sum_{k=1}^{d}e_{k,n}^2$

→ 우리는 E_D를 최소화하는 w 찾고 싶다.

→ back prop. : first order method

Back propagation in output layer

Summary: output layer



(모두 scalar 값)

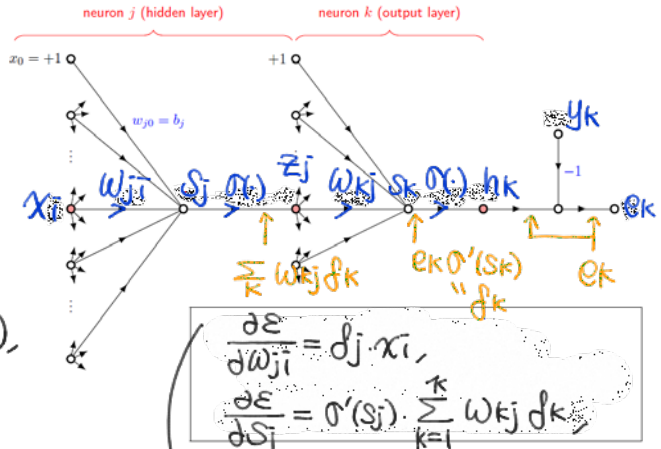
$$\frac{\partial E}{\partial w_{kj}}=\delta_k\cdot z_j,\quad \delta_k=\frac{\partial E}{\partial s_k}=e_k g'(s_k)$$

$$\therefore \Delta w_{kj}=-\eta\cdot e_k\, g'(s_k)\cdot z_j$$

(이때, error, input 이 클수록 w 를 많이 변화시킨다)

Back propagation in input layer

Summary: hidden layer



$$\frac{\partial E}{\partial w_{ji}}=\delta_j\cdot x_i,$$

$$\frac{\partial E}{\partial s_j}=g'(s_j)\cdot\sum_{k=1}^{K}w_{kj}\delta_k,$$

$$\therefore \Delta w_{ji}=-\eta\cdot\delta_j\cdot x_i$$

※ 이때, x_i와 같은 layer들은 fan out, ∴ back에서는 sum을 해주는것 유의!

- Back prop Algorithm (다음을 repeat)

  pick n (하나의 x 를 pick)

  forward : compute all z, h

  backward : compute all delta

  update weights

  until it is time to stop, update final weights

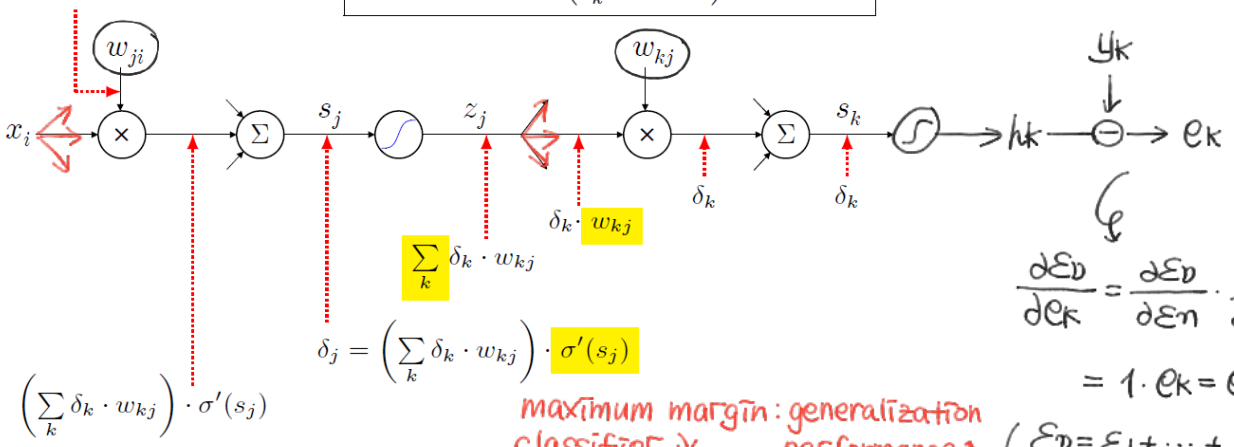
$\frac{\partial E}{\partial w}$를 모두 구해, 모든 w를 update. → 한번에 하나의 n pick 하지 않고 subset을 pick 해서 평균으로 update 할수도 있다.

Back prop 간단한 계산

- 간단한 계산 공식
  1) out = f(in) : f `(in)을 곱한다.
  2) out = in1 * in2 : in1 에 대한 식을 구하는 경우 in2 를 곱한다.
  3) out = sum(in) : 그대로 fanout 한다.
  4) out = max(in) : ini 가 max 이면 fanout, 아니면 0 이다.
  5) out = in ,(fanout) : sum 으로 모두 더한다.

- computing $\frac{\partial \mathcal{E}}{\partial w_{ji}}$

$$\left(\sum_k \delta_k \cdot w_{kj}\right) \cdot \sigma'(s_j) \cdot x_i \implies \boxed{\frac{\partial \mathcal{E}}{\partial w_{ji}} = \delta_j \cdot x_i = \left(\sum_k \delta_k \cdot w_{kj}\right) \cdot \sigma'(s_j) \cdot x_i}$$



$$\frac{\partial \mathcal{E}_D}{\partial e_K} = \frac{\partial \mathcal{E}_D}{\partial \mathcal{E}_n} \cdot \frac{\partial \mathcal{E}_n}{\partial e_K}$$

$$= 1 \cdot e_K = e_K$$

$$\mathcal{E}_D = \mathcal{E}_1 + \cdots + \mathcal{E}_N$$

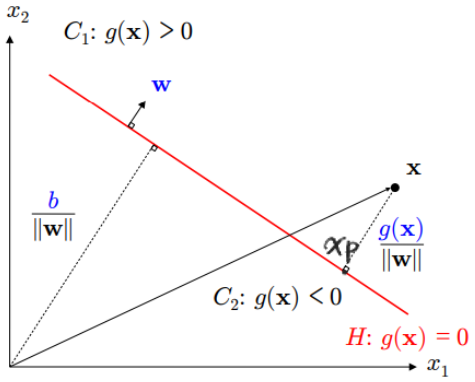
$$\mathcal{E}_n = \frac{1}{2}(e_1^2 + \cdots + e_N^2)$$

maximum margin : generalization performance ↑
classifier ↓

Lecture6. Support vector machines : SVM

- Linear discriminant function : g(x) = wTx + b
  이제는 b 를 따로 생각한다.

- big picture:



최적화에 대한 내용

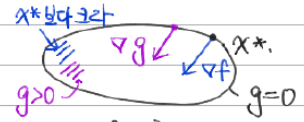
풀고자하는 문제 : constraint optimization
(max, min f) (g≤0, g≥0). → $\mathcal{L} = f \pm \lambda g$ 이용.

→ $\mathcal{L} = f \pm \lambda g$: ① $\lambda \geq 0$ ② 문제 g≥0. V g≤0
③ $\nabla_x \mathcal{L} = \nabla_x f \pm \lambda \nabla_x g = 0$
④ $\lambda g = 0$ (둘다 0은 아니다)

4) 4가지 case.
(f가 max문제, min 문제
g가 g≥0, g≤0.

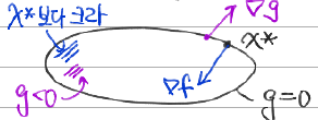
* $\lambda g = 0$ ← 둘다 0은 아니다.

case A) min f, g≥0.



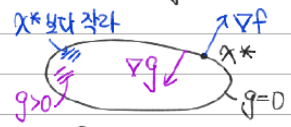
∴ $\nabla f - \lambda \nabla g = 0$

case B) min f, g≤0



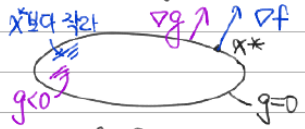
∴ $\nabla f + \lambda \nabla g = 0$

case C) max f, g≥0.



∴ $\nabla f + \lambda \nabla g = 0$.

case D) max f, g≤0



∴ $\nabla f - \lambda \nabla g = 0$

case 1) $\lambda = 0$, $g \neq 0$

g: inactive constraint

case 2) $\lambda > 0$, $g = 0$

g: active constraint

Primal form : 이해하기 좋다. But 의미를 파악하기 힘들다.

- SVM primal (linearly classified)

SVM (primal) optimization problem

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{subject to} \quad & y_t(\mathbf{w}^\top \mathbf{x}_t + b) \geq 1, \ \forall t \in [1, N] \end{aligned}$$

→ w에게 convex

→ w, b에게 linear

- QP (Quadratic programming)

% QP

$$\begin{aligned} \underset{\mathbf{z}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{z}^\top Q\mathbf{z} + \mathbf{c}^\top \mathbf{z} \\ \text{subject to} \quad & A\mathbf{z} \geq \mathbf{a} \end{aligned}$$

solution : $\mathbf{z}^* \leftarrow \text{QP}(Q, \mathbf{c}, A, \mathbf{a})$

$z = \begin{bmatrix} b \\ w \end{bmatrix}$ $Q = \begin{bmatrix} 0 & 0 \\ 0 & I_d \end{bmatrix}$ $c = 0_{d+1}$

$A = \begin{bmatrix} y_1 & y_1 x_1^\top \\ y_N & y_N x_N^\top \end{bmatrix}_N$ $a = 1_N$

d+1

or

$z = \begin{bmatrix} w \\ b \end{bmatrix}$ $Q = \begin{bmatrix} I_d & 0 \\ 0 & 0 \end{bmatrix}$ $c = 0_{d+1}$

$A = \begin{bmatrix} y_1 x_1^\top & y_1 \\ y_N x_N^\top & y_N \end{bmatrix}_N$ $a = 1_N$

d+1

Dual form : support vector, kernel trick 에 대한 이해 가능

- Dual formulation

  if cost, constraint functions are strictly convex, dual form is exist

  and solving dual form is equivalent to solving primal

*우리의 문제.

min f. $g_i \geq 0$ 문제.

$\begin{pmatrix} \nabla f = \lambda_1 \nabla g_1 \\ \vdots \\ \nabla f = \lambda_N \nabla g_N \end{pmatrix}$

$\therefore \mathcal{L} = f - \sum \frac{\lambda_i}{N} g_i$ 라 하고

primal w, b를 미분해 0.

$\frac{\lambda_i}{N} = \alpha_i$, $\alpha_i g_i = 0$ 등을 해석함.

- Step 1 : compute Lagrangian

  위의 primal form 을 constraint 가 없는 Lagrangian optimization 문제로 변환 :

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{t=1}^{N} \alpha_t [y_t(\mathbf{w}^\top \mathbf{x}_t + b) - 1]$$

  a(알파) : Lagrangian multipliers = dual variables, 0 이상의 값을 가진다.

  -> solution 은 saddle point for Lagrangian

  L 은 w,b 에 대해서는 min, a 에 대해서는 max : saddle point

Lagrangian duality

: 'constrained min' of convex can also be obtained as 'max' task to Lagrangian

- Step 2 : KKT conditions

condition 1: $\frac{\partial \mathcal{L}(\mathbf{w}^*, b^*, \boldsymbol{\alpha}^*)}{\partial \mathbf{w}} = \mathbf{0}$

condition 2: $\frac{\partial \mathcal{L}(\mathbf{w}^*, b^*, \boldsymbol{\alpha}^*)}{\partial b} = 0$

미분해서 0 Stationarity $\nabla f - \lambda \nabla g = 0$

$\rightarrow \frac{\partial \mathcal{L}}{\partial w} = w - \sum_{t=1}^{N} \alpha_t y_t x_t = 0$

$\rightarrow \frac{\partial \mathcal{L}}{\partial b} = \sum_{t=1}^{N} \alpha_t y_t = 0$

condition 3: $\alpha_t^* [y_t(\mathbf{w}^{*\top} \mathbf{x}_t + b^*) - 1] = 0, \ \forall t$

$\alpha_t^* = 0$ 또는 $= 0$

$\lambda g = 0$

g가 힘을 써서 즉. boundary 에 존재해. → support vector

active constraint 가 S.V.

$\rightarrow w = \sum_{t=1}^{N} \alpha_t y_t x_t = \sum_{t=1}^{Ns \ll N} \alpha_t y_t x_t$

w 구할때 적은 수의 S.V. 로 구할수. S.V. 의 intution 이 살아남 in dual.

- Step 3 : formulate dual problem

  기존의 Lagrangian 에서 식을 변형하면

$$\mathcal{L} = \frac{1}{2}||\mathbf{w}||^2 - \sum_{t=1}^{N} \alpha_t y_t \mathbf{w}^\top \mathbf{x}_t - b \sum_{t=1}^{N} \alpha_t y_t + \sum_{t=1}^{N} \alpha_t$$

  primal 이 w, dual 은 a 에 대한 식

  inner products XTX -> can use kernel trick

  따라서, nonlinearly separable case 도 고려할 수.

SVM (dual) optimization problem

dual a에 대한 max

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{maximize}} \quad & \sum_{t=1}^{N} \alpha_t - \frac{1}{2} \sum_{s=1}^{N} \sum_{t=1}^{N} \alpha_s \alpha_t y_s y_t \mathbf{x}_s^\top \mathbf{x}_t \\ \text{subject to} \quad & \sum_{t=0}^{N} \alpha_t y_t = 0 \\ & \alpha_t \geq 0, \ \forall t \in [1, N] \end{aligned}$$

오직 N에 scale, not d

kernel trick (inner product)

dual a에 대한 N개 조건추가됨.

- Step 4 : solve dual problem as QP

- QP solver (등호조건 취해진 QP)

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \mathbf{c}^\top \mathbf{x} \quad (\alpha \text{가 } X) \\ \text{subject to} \quad & A\mathbf{x} \le \mathbf{b} \\ & E\mathbf{x} = \mathbf{d} \end{aligned}$$

max → 부호 바꿔서 min α로

- Dual form 을 QP 에 적용하면

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{\alpha}^\top Q\boldsymbol{\alpha} - \mathbf{1}^\top \boldsymbol{\alpha} \\ \text{subject to} \quad & -\boldsymbol{\alpha} \le \mathbf{0} \\ & \mathbf{y}^\top \boldsymbol{\alpha} = \mathbf{0} \end{aligned}$$

※ Q 매우 dense 한 matrix.
α 매우 sparse 한 vector.

$X = [\alpha_1 \cdots \alpha_N] = \alpha$

$C = -1_N$ $\quad x_i^T x_j$

$Q = [y_i y_j \langle x_i, x_j \rangle]_{N \times N}$

$A = -I_N$, $b = 0_N$

$E = [y_1 \cdots y_N]$, $d = 0$

- Step 5 : complete

SVM classifier g(x) : linearly separable case

$$g(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{N} \alpha_t^* y_t \mathbf{x}_t^\top \mathbf{x} + b^*\right)$$

$= \text{sign}(w^{*T}x + b^*)$, $w^* = \sum \alpha_t^* y_t x_t$

generalization bound of SVM (Vapnik, 1995):

$$\mathbb{E}[E_{out}] \le \frac{\mathbb{E}[\#\text{ of support vectors}]}{N-1}$$ (매우크다)

E_{out} 매우 좋다 ※ S.V.만 depend

Lecture7. Support vector machines (linearly non-separable)

Linearly non-separable case1) kernel trick 으로 해결

- Non-linear transform

kernel trick, inner product.

$z_s^T z_t = K(x_s, x_t)$

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{t=1}^{N} \alpha_t - \frac{1}{2}\sum_{s=1}^{N}\sum_{t=1}^{N} \alpha_s \alpha_t y_s y_t \mathbf{x}_s^\top \mathbf{x}_t \quad \Rightarrow \quad \mathcal{L}(\boldsymbol{\alpha}) = \sum_{t=1}^{N} \alpha_t - \frac{1}{2}\sum_{s=1}^{N}\sum_{t=1}^{N} \alpha_s \alpha_t y_s y_t \mathbf{z}_s^\top \mathbf{z}_t$$

- Kernel : non-linear similarity measure

- polynomials (of degree $q > 0$):

$$K(\mathbf{x}, \mathbf{x}') = (a\mathbf{x}^\top \mathbf{x}' + b)^q$$

- radial-basis functions ($\gamma > 0$):

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

- Gaussian RBF:

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

- hyperbolic tangent (aka sigmoid, multilayer perceptron kernel):

$$K(\mathbf{x}, \mathbf{x}') = \tanh(\beta \mathbf{x}^\top \mathbf{x}' + \gamma)$$

- multiquadric:

$$K(\mathbf{x}, \mathbf{x}') = \sqrt{||\mathbf{x} - \mathbf{x}'||^2 + c^2}$$

- wave:

$$K(\mathbf{x}, \mathbf{x}') = \frac{\theta}{||\mathbf{x} - \mathbf{x}'||} \sin \frac{||\mathbf{x} - \mathbf{x}'||}{\theta}$$

- equivalent kernel :
  poly에서 a=b=0
- linear kernel : poly에서 q=1
- RBF kernel :
  infinite feature dim,
  r이 크면 over fitting.

- Valid kernel 인가? : Mercer's theorem

Let function $K : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ be given. For K to be a valid (Mercer) kernel, it is necessary and sufficient that, for any finite set of instances, the corresponding *kernel matrix* is

$K_{ij} = K(x_i, x_j)$ 가 어떤 finite set 에 대해서도 성립.

: 1) symmetric 2) positive semi-definite

kernel matrix : K = (m,m) matrix, Kij = K(xi,xj)

positive semi-definite : PSD

all eigen values, determinant is non-negative, real symmetric K 가 tTKt 는 0 이상 for all vector

Linearly non-separable case2) soft-margin SVM : C가 클수록 hard margin에 가까워

- Reformulating SVM problem : primal form → 원래 primal에서 +C Σξt, 조건식 변형과 추가.

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{t=1}^{N}\xi_t \\ \text{subject to} \quad & y_t(\mathbf{w}^T\mathbf{x}_t + b) \geq 1-\xi_t, \quad \forall t \\ & \xi_t \geq 0, \quad \forall t \end{aligned}$$

: primal value : w. b. ξt ∀t.

- Reformulating SVM problem : dual form
  - Lagrangian $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$

$$= \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{t=1}^{N}\xi_t - \sum_{t=1}^{N}\alpha_t(y_t(\mathbf{w}^\top\mathbf{x}_t + b) - 1 + \xi_t) - \sum_{t=1}^{N}\beta_t\xi_t$$

: dual value : αt. βt ∀t.

    - minimize wrt $\mathbf{w}$, b, and $\boldsymbol{\xi}$
    - maximize wrt each $\alpha_t \geq 0$ and $\beta_t \geq 0$
  - KKT conditions are obtained as follows:

$$\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{t=1}^{N}\alpha_t y_t \mathbf{x}_t = \mathbf{0} \quad \Rightarrow \quad \mathbf{w} = \sum_{t=1}^{N}\alpha_t y_t \mathbf{x}_t$$

$$\frac{\partial\mathcal{L}}{\partial b} = -\sum_{t=1}^{N}\alpha_t y_t = 0 \quad \Rightarrow \quad \sum_{t=1}^{N}\alpha_t y_t = 0$$

$$\frac{\partial\mathcal{L}}{\partial \xi_t} = C - \alpha_t - \beta_t = 0 \quad \Rightarrow \quad \alpha_t + \beta_t = C$$

$$\alpha_t[y_t(\mathbf{w}^\top\mathbf{x}_t + b) - 1 + \xi_t] = 0 \quad \Rightarrow \quad \alpha_t = 0 \vee y_t(\mathbf{w}^\top\mathbf{x}_t + b) = 1 - \xi_t$$

$$\beta_t\xi_t = 0 \quad \Rightarrow \quad \beta_t = 0 \vee \xi_t = 0$$

Final form, 아래의 식을 QP에 적용

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & \sum_{t=1}^{N}\alpha_t - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\top\mathbf{x}_j \\ \text{subject to} \quad & \sum_{t=1}^{N}\alpha_t y_t = 0 \\ & 0 \leq \alpha_t \leq C, \quad \forall t \end{aligned}$$

only change

final solution

Soft-margin SVM classifier

$$g(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\top\mathbf{x} + b\right)$$

where

$$\mathbf{w} = \sum_{0<\alpha_t\leq C}\alpha_t y_t \mathbf{x}_t \quad , \; w^* = \sum_{0<\alpha_t\leq C}\alpha_t y_t x_t$$

$$b = y_{sv} - \sum_{0<\alpha_t\leq C}\alpha_t y_t \mathbf{x}_t^\top\mathbf{x}_{sv} \quad , \; b = y_{s.v} - w^* x_{s.v}$$

w = sum (ayx)에서 a=C 이므로 w에서 비중이 크다.

α=C인 S.V들

- 모두 정리

	non S.V.	S.V.		
	모범생	원래 S.V	군사분계선	간첩
$y_t(\mathbf{w}^\top\mathbf{x}_t + b)$	y(wTx+b)>1	y(wTx+b)=1	0≤y(wTx+b)<1	y(wTx+b)<0
ξ_t	E=0	E=0	0<E≤1	1<E
a (alpha)	a=0	0<a≤C	a=C	a=C
b (beta)	b≠0	b≠0	b=0	b=0
Vector x lies	Out of margin	On margin	In margin	The other area
분류	right	right	right	wrong