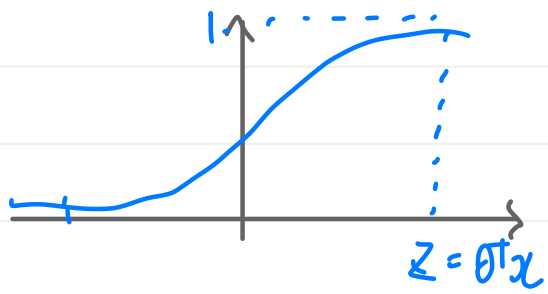
12. Support Vector Machines

12.1 Optimization objective

① Alternative view of logistic regression

$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$



If $y=1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$

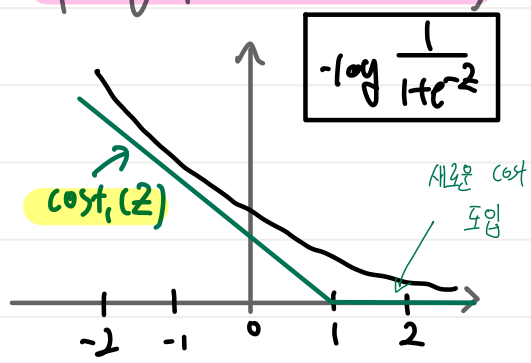
If $y=0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

$\Rightarrow$ 한 개의 example의 cost:

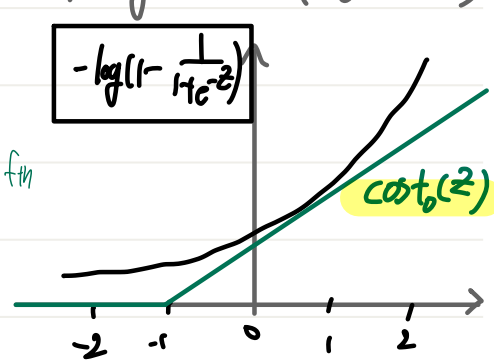
$-(y \log h_\theta(x) + (1-y) \log(1-h_\theta(x)))$

$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1-y) \log\left(1 - \frac{1}{1+e^{-\theta^T x}}\right)$

If $y=1$ (want $\theta^T x \gg 0$)



If $y=0$ (want $\theta^T x \ll 0$)



② Support vector machine

Logistic regression :

$$\min_\theta \frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)})) + (1-y^{(i)})(-\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Support vector machine :

$$\min_\theta \frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\,\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\,\text{cost}_0(\theta^T x^{(i)})\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

편의를 위해 추가한 것임
→ 삭제해도 θ의 최적값 찾는데 영향X

$A + \lambda B$ → λ 값 높이면 B에 큰 가중치

$\Rightarrow CA + B \left(C = \frac{1}{\lambda}\right)$ → λ 값 높이면 A 가중치 줄여 B에 큰 가중치

⇒ 두 식이 같은 최적의 θ 찾을 수 있음

$$\Rightarrow \min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2$$

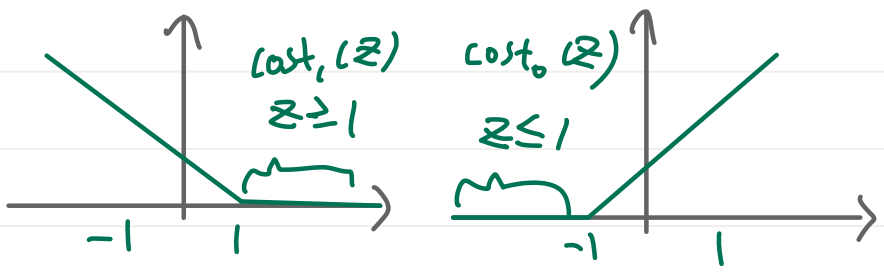
hypothesis : $h_\theta(x) \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{o.w} \end{cases}$ → SVM의 hypothesis는 확률 아니고 output 자체가 1 또는 0 이산.

12. Support Vector Machines

12.2 Large Margin Intuition

① SVM Decision Boundary

$$\min_\theta \; C\sum_{i=1}^{m}\left[y^{(i)}\,\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\,\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$



If $y=1$, we want $\theta^T x \geq 1$ (not ≥ 0)

If $y=0$, we want $\theta^T x \leq -1$ (not < 0)

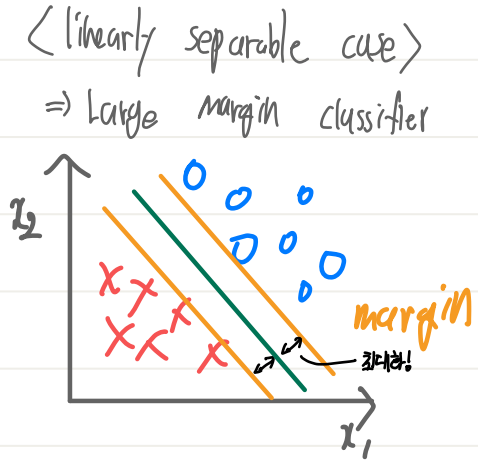
whenever $y^{(i)}=1$: $\theta^T x^{(i)} \geq 1$

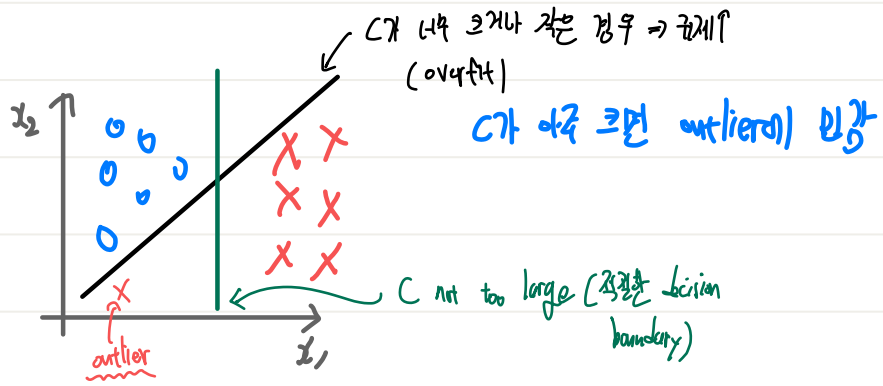
whenever $y^{(i)}=0$: $\theta^T x^{(i)} \leq -1$

$\Rightarrow$ 충분조건 $\min_\theta \; \cancel{C\times 0} + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$

↳ zero가 됨

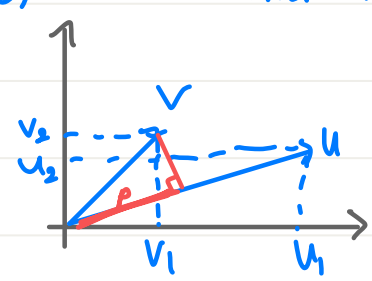
$$\Rightarrow \boxed{\min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 \quad \begin{pmatrix} \theta^T x^{(i)} \geq 1 & \text{if } y^{(i)}=0 \\ \theta^T x^{(i)} \leq -1 & \text{if } y^{(i)}=0 \end{pmatrix}}$$





12.3 The mathematics behind large margin classification

① Vector Inner Product



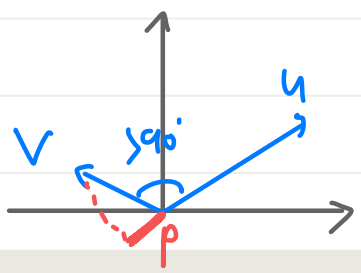
$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$

$\|u\|$ = length of vector u

$= \sqrt{u_1^2 + u_2^2} \in \mathbb{R}$

p = length of projection of v on u

$u^T v = p \cdot \|u\| = u_1 v_1 + u_2 v_2 = v^T u$



→ $u^T v = p \cdot \|u\|$ $(p<0)$

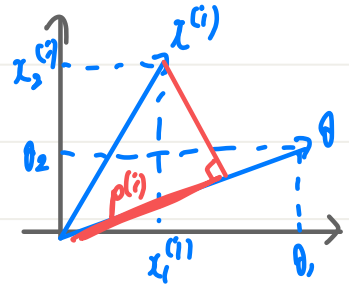
12. Support Vector Machines

② SVM Decision Boundary

$\min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}(\theta_1^2+\theta_2^2) = \frac{1}{2}\left(\sqrt{\theta_1^2+\theta_2^2}\right)^2 = \frac{1}{2}\|\theta\|^2$

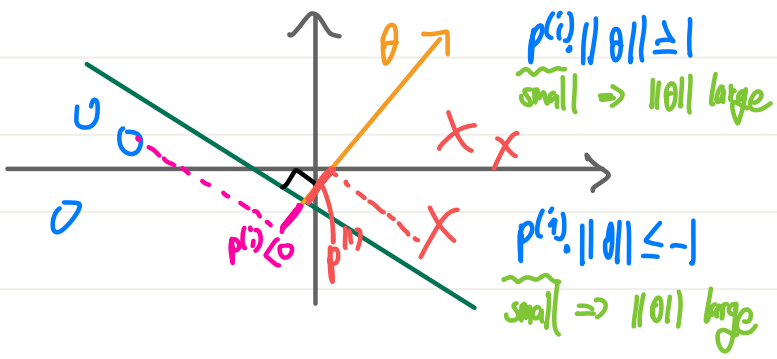
Set. $\theta^T x^{(i)} \geq 1$ if $y^{(i)}=1$ (간단하게 $\theta_0=0$, $n=2$)

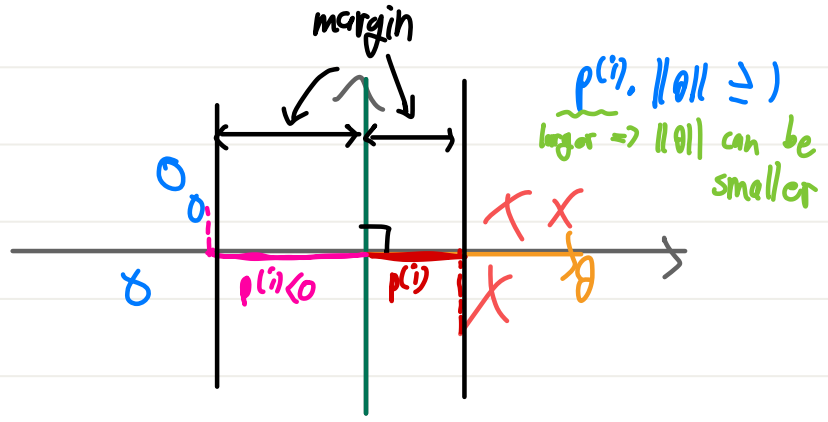
$\theta^T x^{(i)} \leq -1$ if $y^{(i)}=0$ ($p^{(i)}$는 $x^{(i)}$를 θ에 정사영 시킨 것)



$\theta^T x^{(i)} = p^{(i)} \cdot \|\theta\|$

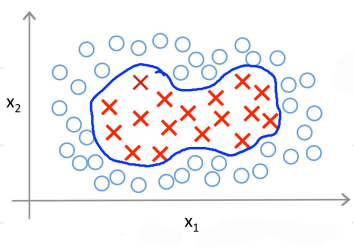
$= \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)}$





12.4 Kernels 1

① Non-linear Decision Boundary



predict $y=1$ if

$\theta_0+\theta_1 x_1+\theta_2 x_2+\theta_3 x_1 x_2+\theta_4 x_1^2+\theta_5 x_2^2+\cdots \geq 0$

$h_\theta(x)$ { 1 if $\theta_0+\theta_1 x_1+\cdots \geq 0$; 0 o.w

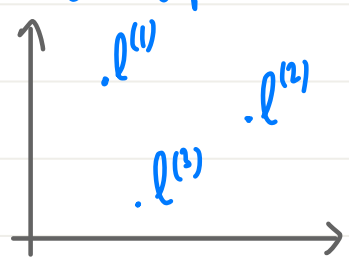
$\Rightarrow f_1=x_1,\ f_2=x_2,\ f_3=x_1x_2$

$f_4=x_1^2,\ f_5=x_2^2,\ \ldots$

② Kernel

주어진 x에서 새로운 feature를 계산한다.

landmarks $l^{(1)}, l^{(2)}, l^{(3)}$의 근접성에 따라서 결정



Given x: $f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right)$

$f_2 = \text{similarity}(x, l^{(2)}) = \exp\left(-\frac{\|x-l^{(2)}\|^2}{2\sigma^2}\right)$

$f_3 = \text{similarity}(x, l^{(3)}) = \exp(\ldots)$

↖ kernel (Gaussian kernel)

ex) $f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right) = \exp\left(\frac{-\sum_{i=1}^{n}(x_i-l_i^{(1)})^2}{2\sigma^2}\right)$

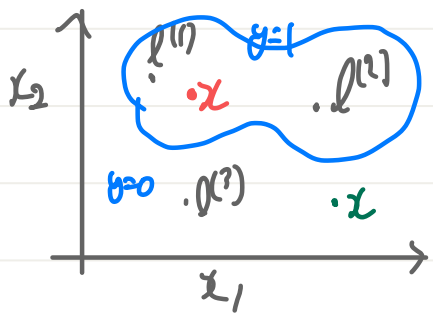
If $x \approx l^{(1)}$: $f_1 \approx \exp\left(-\frac{0^2}{2\sigma^2}\right) \approx 1$

If x far from $l^{(1)}$: $f_1 \approx \exp\left(-\frac{(\text{large num})^2}{2\sigma^2}\right) \approx 0$

* Gaussian 함수는 정규분포와 같이 bell-curve 형태로 landmark에 가까울수록 큰 값, 멀수록 작은 값을 출력

12. Support Vector Machines

③ Example



predict "y" when

$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Set, $\theta_0 = -0.5$, $\theta_1 = 1$, $\theta_2 = 1$, $\theta_3 = 0$

x가 $l^{(1)}$에 가까울 때 $f_1 \approx 1$, $f_2 \approx 0$, $f_3 \approx 0$

$\theta_0 + \theta_1 x_1 + \theta_2 x_0 + \theta_3 x_0$

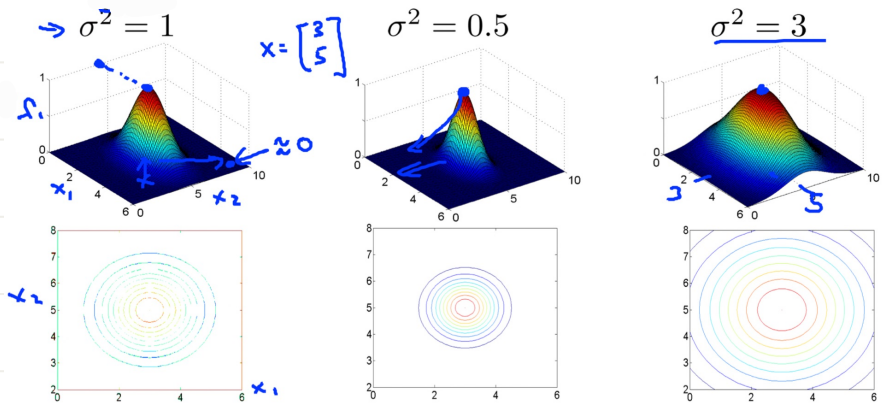
$= -0.5 + 1 = 0.5 \geq 0$ ⇒ 큰 $y=1$ 영역에 표함

x가 큰 landmark에서 멀 때 $f_1, f_2, f_3 \approx 0$

$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 = -0.5 < 0$ ⇒ x는 $y=0$ 영역에 포함

④ σ^2 영향

ex) $l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$ $\quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$



σ^2 커질수록 그래프의 기울기 낮아진다!

12.5 Kernels 2

① choosing the landmarks

⇒ 한 가지 방법은 landmark는 training example과 같은 위치에 놓는 것

Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})$

choose $l^{(1)} = x^{(1)}$, $l^{(2)} = x^{(2)}$, $\cdots$, $l^{(m)} = x^{(m)}$

Given example x:

$f_1 = \text{similarity}(x, l^{(1)} = x^{(1)})$

$f_2 = \text{similarity}(x, l^{(2)} = x^{(2)})$

$\vdots$

For training example $(x^{(i)}, y^{(i)})$:

$f_1^{(i)} = \text{sim}(x^{(i)}, l^{(1)})$

$f_2^{(i)} = \text{sim}(x^{(i)}, l^{(2)})$

$\vdots$ ← 대각선에 $f_i^{(i)} = \text{sim}(x^{(i)}, l^{(i)} = x^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1$

$f_m^{(i)} = \text{sim}(x^{(i)}, l^{(m)})$

$f^{(i)} = \begin{bmatrix} f_0^{(i)} \\ f_1^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix}$, $f_0^{(i)} = 1$

12. Support Vector Machines

② SVM with Kernels

Hypothesis : Given x, compute features $f \in \mathbb{R}^{m+1}$

→ predict "$y=1$" if $\theta^T f \geq 0$

Training :

$$\min_\theta \; C \sum_{i=1}^{m} \left[y^{(i)} \text{cost}_1(\theta^T f^{(i)}) + (1-y^{(i)}) \text{cost}_0(\theta^T f^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{m} \theta_j^2$$

($x^{(i)}$ 대신 $f^{(i)}$)

※ 꼭 SVM이 아니더라도 kernel 개념 도입할 수 있지만 SVM에서의 classification 시는 연산속도 매우 느려지기에 ⇒ kernel은 주로 SVM에서만 사용

③ SVM parameters

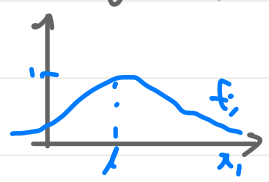
$C \left(= \frac{1}{\lambda}\right)$

- Large C : Lower bias, high variance (small λ) overfit
- Small C : Higher bias, low variance (large λ) underfit

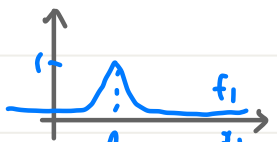
σ^2

$\exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right)$

- Large σ^2 : features f_i vary some smoothly ⇒ Higher bias, lower variance



- Small σ^2 : features f_i vary less smoothly. ⇒ Lower bias, higher variance



12.6 Using an SVM

- SVM은 잘 알려지고, 잘 구현된 library가 많아 직접 구축하기 보다 잘 만들어진 package 이용하는 편이 낫다 eg) liblinear, libsvm
- C, kernel의 종류 명시해 주어야 함

ex) No kernel (linear kernel):

Predict "$y=1$" if $\theta^T x \geq 0$

$\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

→ n large, m small $x \in \mathbb{R}^{n+1}$

ex) Gaussian kernel :

$f_i = \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$

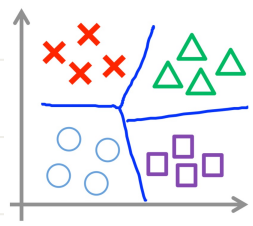
Need to choose σ^2

Note) feature scaling 필요 (가우시안 kernel 사용전에)

Note) 모든 similarity function "$\text{sim}(x, l)$"을 kernel로 사용할 수 있는 것은 아님. "Mercer Theorem"이라는 조건 만족해야 SVM package의 optimization 과정이 발산하지 않고 제대로 동작함

12. Support Vector Machines

① Multi-class classification



$y \in \{1, 2, 3, \cdots, k\}$

많은 SVM package들이 이미 multi-class classification을 가지고 있다.

"one vs all" method

K개의 SVM train 한다. 구분된 $y = i$에 대해 $(i = 1, 2, \cdots, k)$

$\Rightarrow \theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(k)}$ 을 얻어 largest $(\theta^{(i)})^T x$ 를 만족하는 class i 선택

($\theta^{(1)}$: $y=1$, $\theta^{(2)}$: $y=2$, $\theta^{(k)}$: $y=k$)

② Logistic regression vs. SVM

n = # features ($x \in \mathbb{R}^{n+1}$)

m = # training examples

① if n is large (relative to m) $n \geq m$ ex) $n = 10,000$, $m = 10 \sim 1000$

$\Rightarrow$ use logistic regression or SVM without a kernel (linear kernel)

② If n is small, m is intermediate $n = 1 \sim 1000$, $m = 10 \sim 10,000$

$\Rightarrow$ Use SVM with Gaussian kernel

③ If n is small, m is large $n = 1 \sim 1000$, $m = 50,000^+$

$\Rightarrow$ Create/add more features, then use logistic regression or SVM w/o kernel

④ Neural Network likely to work well for most of these settings, but may be slower to train