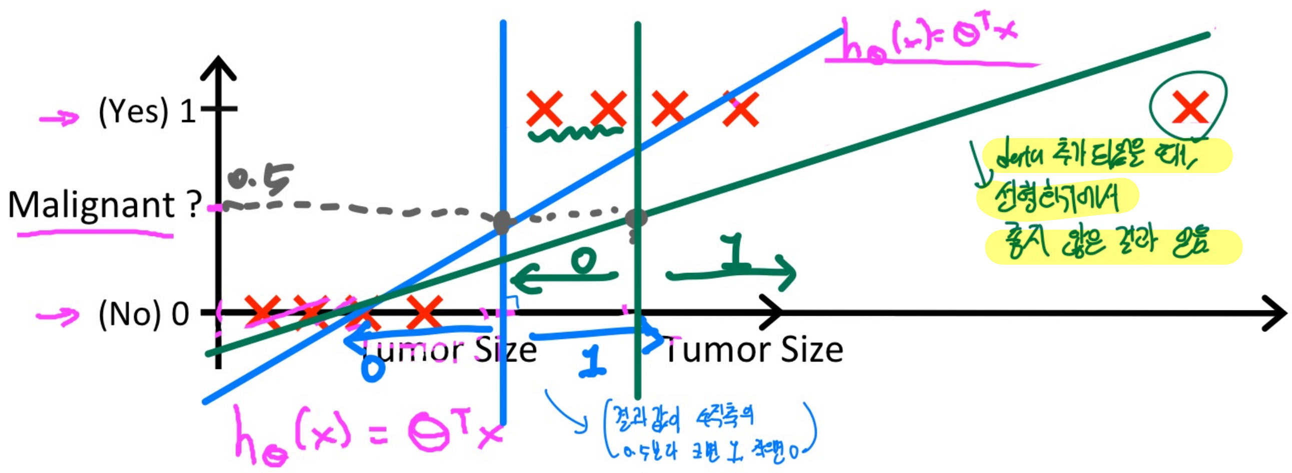
6. Logistic Regression

6.1 Classification (분류)

ex) Email (spam / not spam)

Tumor (Malignant / Benign)



⇒ linear regression은 classification에 적용하면 잘 되지 않은 경우 있음

→ Threshold classifier output $h_\theta(x)$ at 0.5:

→ If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

linear regression Classification 에서 $h_\theta(x)$는 >1 or <0 만족할 수 있다.

Logistic regression classification 에서는 $0 \le h_\theta(x) \le 1$ 의 값만 갖는다.

6.2 Hypothesis Representation

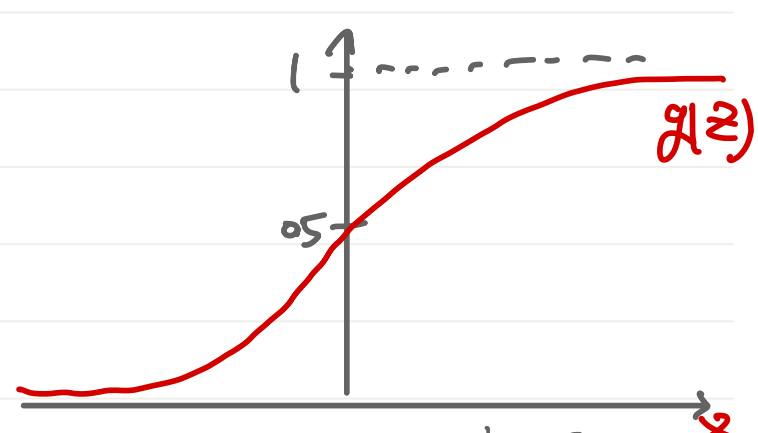
we want $0 \le h_\theta(x) \le 1$

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

= sigmoid fn

= Logistic fn

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



함수 값을 0~1사이로 만들어 줌!

6. Logistic Regression

※ 가설 출력($h_\theta(x)$)의 해석

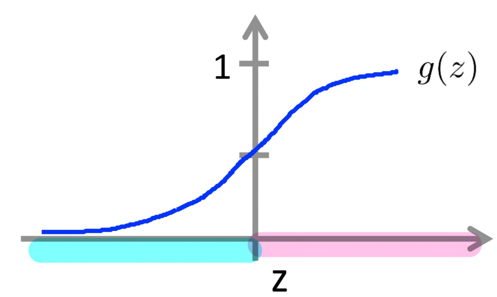
: $h_\theta(x)$의 출력값은 주어진 feature가 x라는 값을 가질 때 class 1에 들어갈 확률 (= y=1)

$$\boxed{h_\theta(x) = P(y=1 \mid x;\theta) \quad (y=0 \text{ or } 1)}$$

ex) $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumor size} \end{bmatrix}$

$h_\theta(x) = 0.7$ (악성 종양(y=1)일 확률이 70%)

참고) $P(y=0 \mid x;\theta) = 1 - P(y=1 \mid x;\theta)$



6.3 Decision boundary

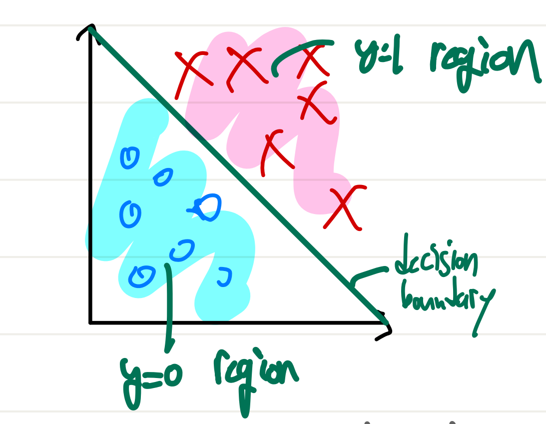
$h_\theta(x) = g(\theta^T x)$

$g(z) = \dfrac{1}{1+e^{-z}}$ ⇒

$h_\theta(x) \ge 0.5$ 이면	$y=1$	→ $\theta^T x \ge 0$, $g(\theta^T x) \ge 0.5$
$h_\theta(x) < 0.5$ 이면	$y=0$	→ $\theta^T x < 0$, $g(\theta^T x) < 0.5$

· Decision boundary는 y=0 과 y=1을 가르는 경계선, $h_\theta(x)$에 의해 결정됨

└ θ에 의해 결정됨 (training data(x)는 parameter(θ)를 결정하는데 이용될 뿐, decision boundary에 직접적 영향X)

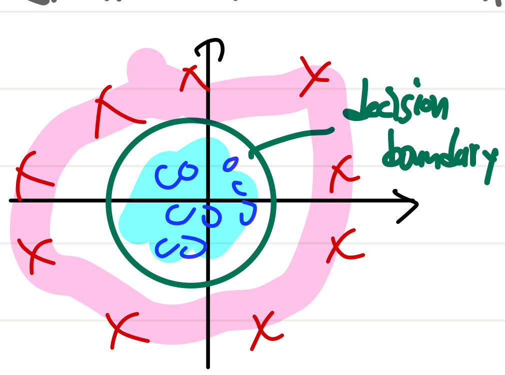


$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

If $\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

→ y=1이다. $\boxed{\text{if } -3 + x_1 + x_2 \ge 0}$

<Non-linear decision boundaries>



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

if $\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$ → y=1 $\boxed{\text{if } -1 + x_1^2 + x_2^2 \ge 0}$

6. Logistic Regression

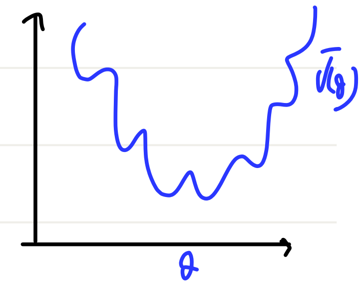
6.4 Cost function

linear regression 에서 $J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

↓

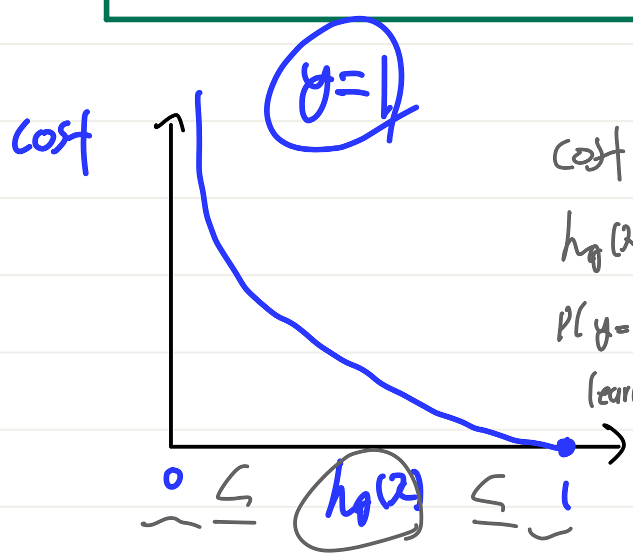
logistic에 적용하면 non-convex cost fn이 된다. (볼록함수 X)

⇒ local optima에 빠질 수 있음



↓

$$\boxed{\mathrm{cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases}}$$

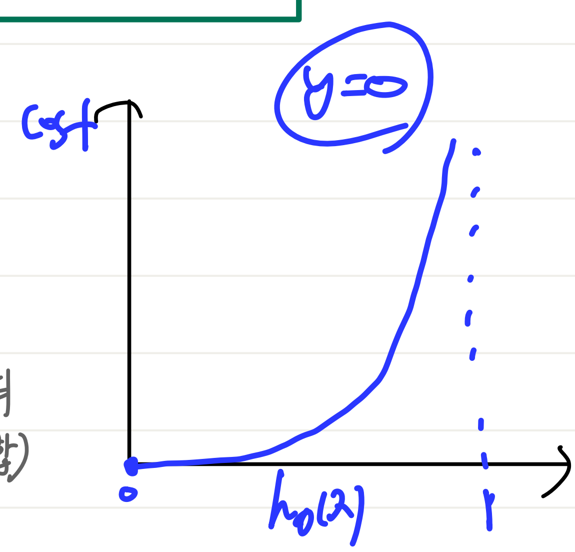


cost $=0$ if $y=1$, $h_\theta(x)=1$

$h_\theta(x) \to 0 \Rightarrow$ cost $\to \infty$

$P(y=1 \mid x;\theta) = 0$ 일 때,

learning algorithm에 큰 패널티 줘야 함 (직관에 부합)



6.5 Simplified cost function and gradient descent

(참고 maximum likelihood estimation 적용하여 구함)

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathrm{cost}(h_\theta(x^{(i)}), y^{(i)})$$

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

if $y=1$: $\mathrm{cost}(h_\theta(x), y) = -\log h_\theta(x)$

if $y=0$; $\mathrm{cost}(h_\theta(x), y) = -\log(1-h_\theta(x))$

6. Logistic Regression

<Gradient Descent>

want $\min J(\theta)$

→ Repeat $\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta)$ (simultaneously update all θ_j)

$$:= \boxed{\theta_j - \alpha \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}}$$

수식만 보면 linear regression과 동일

but $h_\theta(x) = \theta^T x \rightarrow h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ 로 바뀜 (같지 않음)

6.6 Advanced optimization

optimization algorithms

- Gradient descent
- Conjugate gradient
- BFGS
- L-BFGS

디테일한 설명 X

Advantages
- α 자동으로 골라줌
- gradient descent 보다 빠름

DisAdvantages
- 더 복잡함

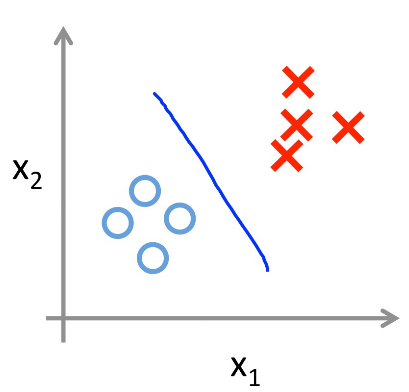
→ 직접 계산 X, 언어 내장 library 사용

6.7 Multi-class classification: One-vs-all

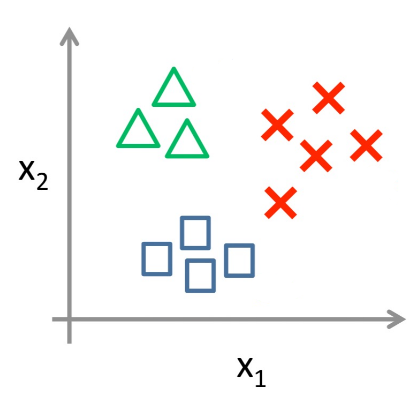
ex) Email foldering / tagging : Work, Friends, Family, hobby
($y=1$, $y=2$, $y=3$, $y=4$)

Weather : Sunny, Cloudy, Rain, Snow
($y=1$, $y=2$, $y=3$, $y=4$)

Binary classification:

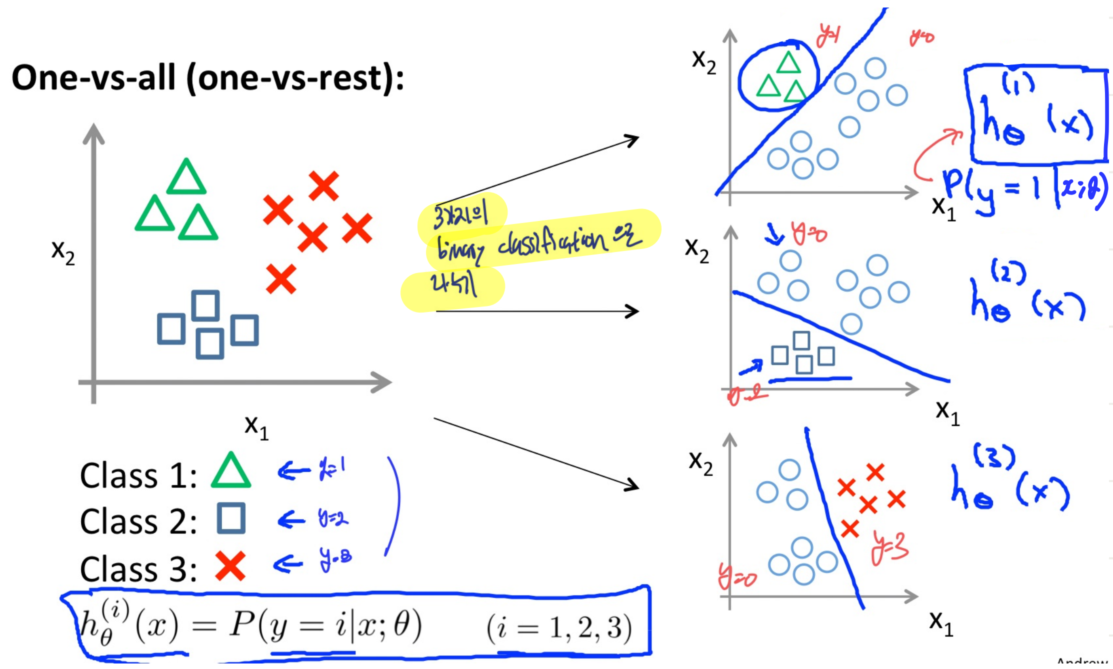


Multi-class classification:



6. Logistic Regression

<one-vs-all (one-vs-rest)>



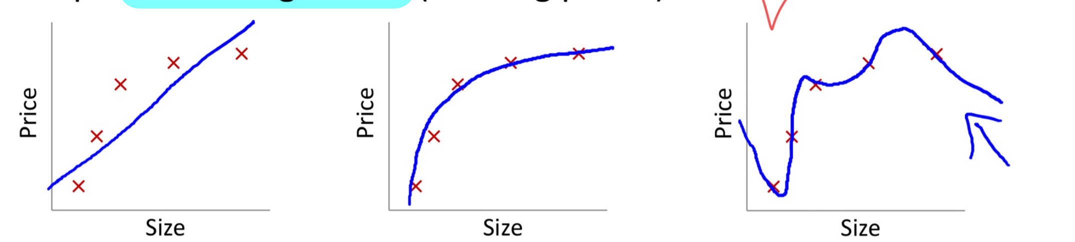
1. $P(y=i)$ 구하기 위해 각 class i 에 대해 logistic regression classifier $h_\theta^{(i)}(x)$ 를 train 한다.

2. 새로운 data x 받으면 $\max_i h_\theta^{(i)}(x)$ 인 class i를 선택 (가장 높은 확률을 주는 i 값이면 y가 그 값이라고 예상할 수 있음)

7. Regularization

7.1 The problem of overfitting

Example: Linear regression (housing prices)



→ $\theta_0 + \theta_1 x$ "Underfit" "High bias"

→ $\theta_0 + \theta_1 x + \theta_2 x^2$ "Just right"

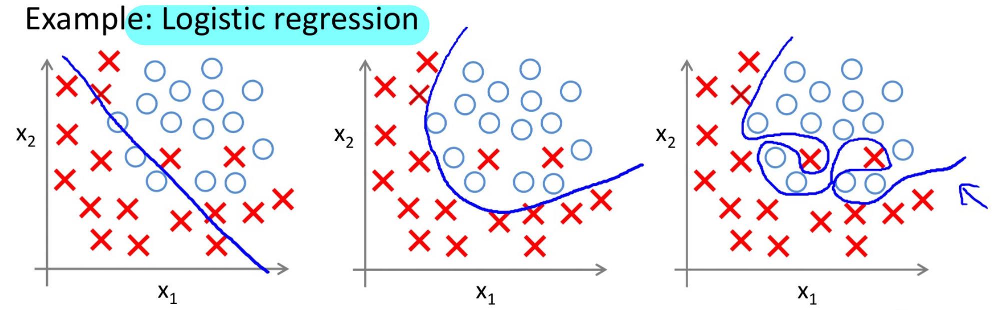
→ $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ "Overfit" "High variance"

overfitting

: training set에 매우 잘 맞음

△ but, 일반화 실패 (새로운 data 예측 X)

Example: Logistic regression



→ $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

"Underfit" high bias

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$

"Over-fit" high variance

<overfitting 다루기>

1. feature 수 줄이기
   - feature 수동으로 고르기
   - model selection algorithm (나중에 배움)

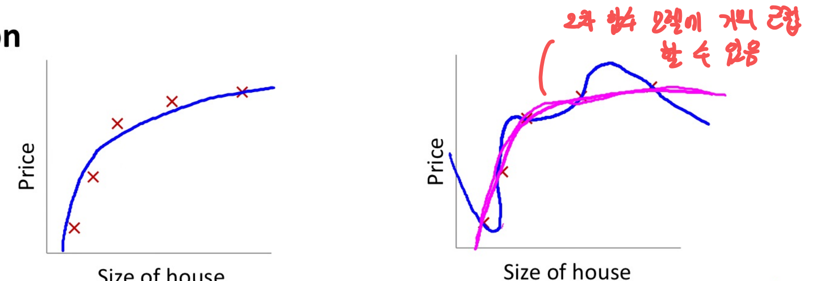
   몇몇 정보를 함께 버린다는 단점..

2. 정규화 (regularization)
   - 모든 feature 유지
     but, parameter θ_j 크기 줄이기
   - 많은 feature들 각각이 y 예측하는데 기여하면 잘 작동

7.2 Cost function

Intuition

4차 함수 모델 사용하더라도 2차 함수 모델에 거의 근접 할 수 있음



$\theta_0 + \theta_1 x + \theta_2 x^2$

$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

불리하게 만든다

Suppose we penalize and make θ_3, θ_4 really small.

→ $\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$

$\theta_3 \approx 0$ $\theta_4 \approx 0$

궁금한 점 1000 곱하면 θ_3, θ_4 작아지나?

→ $+1000\theta_3^2 + 1000\theta_4^2$ 해주면 minimize 값 찾아주기 위해 $\theta_3 \to 0$, $\theta_4 \to 0$ 으로?

<Regularization>

parameters $(\theta_0, \dots, \theta_n)$ 들이 작은 값 갖게 함

→ 단순한 $h_\theta(x)$ 얻음

→ overfitting 가능성 줄여 줌

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

너무 크면 underfitting

⇒ 적절한 λ값 정하는 것 중요

regularization parameter

: trade-off를 control

① training data 잘 맞게하기

② parameter 작게 유지

7. Regularization

7.3 Regularized linear regression

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

최적의 parameter θ 찾기 위한 방법 2가지 (gradient descent / normal equation)

① Gradient descent

Repeat {

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\boxed{\sum_{i=1}^{m} h_\theta(x^{(i)}) - y^{(i)})\, x_0^{(i)}} \quad \frac{\partial}{\partial\theta_0}J(\theta)$$

$$\theta_j := \theta_j - \alpha\boxed{\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]} \quad \frac{\partial}{\partial\theta_j}J(\theta)$$

}

$j = \cancel{0}, 1, 2, \dots, n$

$$\theta_j := \boxed{\theta_j(1 - \alpha\frac{\lambda}{m})} - \boxed{\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}}$$

< 1

ex) $\theta_j \times 0.99$ 처럼 더 작아짐

지금까지 봤던 gradient descent 와 똑같음.

② Normal equation

$$X = \begin{bmatrix}(x^{(1)})^T \\ \vdots \\ (x^{(m)})^T\end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)}\end{bmatrix}$$

$m \times (n+1)$ $\qquad$ $\mathbb{R}^m$

$$\theta = \left(X^TX + \lambda\underbrace{\begin{bmatrix}0 & & & 0\\ & 1 & & \\ & & 1 & \\ 0 & & & 1\end{bmatrix}}_{(n+1)\times(n+1)}\right)^{-1} X^Ty$$

ex) $n=2$ $\begin{bmatrix}0&0&0\\0&1&0\\0&0&1\end{bmatrix}$

※ feature 개수에 비해 data 수가 부족할 때 발생할 수 있는 non-invertibility 문제 해결에 도움

suppose) $m \le n$

※ examples $\qquad$ ※ features

if $\lambda > 0$

$$\theta = \underbrace{\left(X^TX + \lambda\begin{bmatrix}0 & & \\ & 1 & \\ & & \ddots \\ & & & 1\end{bmatrix}\right)}_{\text{invertable!}}{}^{-1} X^Ty$$

7.4 Regularized logistic regression

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

⟨gradient descent⟩

Repeat {

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)} \quad - \frac{\partial}{\partial\theta_0}J(\theta)$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right] \frac{\partial}{\partial\theta_j}J(\theta)$$

$(j = \cancel{0}, 1, \ldots, n)$

}