Supervised Learning

prediction

$h: X \longrightarrow Y$

Images	Cat
TEXT	Is hate speech?
House data	Price

Given Training Set

$\{(x^{(1)}, y^{(1)}), \dots (x^{(n)}, y^{(n)})\}$

$x^{(i)} \in X \cdot y^{(i)} \in Y$

Do: find good $h: X \to Y$ (hypothesis)

start with how accurate it is -

We care about new X values not in training set

if y is discrete ⇒ "classification"

continuous ⇒ "Regression"

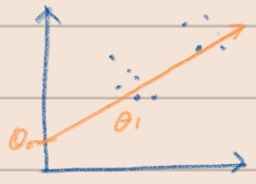
First plot it, from the data set

Sales price	Lot in sqft
000 ~	0000

$\longrightarrow$ h: Lot area → price

How do we represent the h?

$h(x) = \theta_0 x_0^{(i)=1} + \theta_1 x_1$



	Size	bedroom ...	(y) price
$x^{(1)}$	2104 ($x_1^{(1)}$)	4 ($x_2^{(1)}$)	400
$x^{(2)}$	2500	3	900

$h(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d$

$= \sum_{j=0}^{d} \theta_j \cdot x_j$ where $x_0 = 1$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ \cdot \end{bmatrix}$$

parameters — features

$y^{(i)}$ is price

$(x^{(i)}, y^{(i)})$ ← training graph

$$X = \begin{bmatrix} — x^{(1)} — \\ \\ — x^{(n)} — \end{bmatrix} \in R^{n \times (d+1)}$$

matrix

$h_\theta(x) = \sum_{j=0}^{d} \theta_j x_j$ $\quad h_\theta(x) \simeq y$

성형회귀 손실함수 Mean Squared Error

IDEA $J(\theta) = \frac{1}{2} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2$ ⇒ "optimization"

Minimize

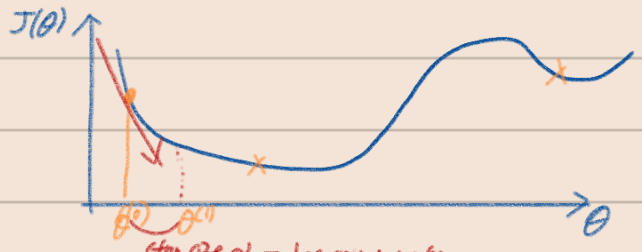
선형회귀, 경사하강법 초기값에서의 손실함수는 convex라서 + bounded

every local min = global min

→ 경사하강법 같은 최적화 알고리즘이 어디에 시작하든 최적해에 도달할 수 있다는 안정성

good clean model = bounded + convex

Gradient Descent



step size α = learning rate

$\theta^0 = 0$

$\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \frac{\partial}{\partial \theta_j} J(\theta^{(t)})$ $\quad j = 0 \dots d$ } iteration

partial derivative w/ respect to theta

too big: go too far

too small: not enough progress

$$\frac{\partial}{\partial \theta_j} J(\theta) = \sum_{i=1}^{n} \frac{1}{2} \frac{\partial}{\partial \theta_j} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$= \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) \cdot \frac{\partial}{\partial \theta_j} h_\theta(x^{(i)})$$

$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d$

$\frac{\partial}{\partial \theta_j} h_\theta(x) = x_j$

$\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \cdot \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Vector Equation ⇒ $\theta^{(t+1)} = \theta^{(t)} - \alpha \cdot \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$

정확식

수렴할 때까지 하거나, 정해진 수만큼 진행

convex함수이면 gradient가 정확 작아짐

but, for harder functions, we really don't know

(non-deterministic)

Batch vs. Stochastic mini batch

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \cdot \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)}$$

mini batch B (at random) $\quad B \ll N$

$$\theta^{(t+1)} = \theta^{(t)} - \alpha_B \sum_{i \in B}^{n} (h_\theta(x^{i}) - y^{i})\, x^{(i)}$$

come back when we look at different loss functions

Normal Equations ← 사실 안외워도 되고 그냥 끝남...!