- Deep Learning : find a model & parameter

- Linear Regression

Sol 1) Gradient Descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \rightarrow \theta_j := \theta_j + \alpha (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$$

Sol 2) Normal Equations

$$\nabla_\theta J(\theta) = 0 \Rightarrow \theta = (X^T X)^{-1} X^T \vec{y}.$$

Sol 3) Probabilistic Interpolation

$$\ell(\theta) = \mathrm{Log}\, L(\theta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2$$

- Locally Weighted Regression

: fit θ to minimize $\sum_i w^{(i)} (y^{(i)} - \theta^T x^{(i)})^2$. $\left(w^{(i)} = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)\right)$

- Logistic Regression

$$P(y \mid x; \theta) = (h_\theta(x))^y (1 - h_\theta(x))^{1-y}$$

$$L(\theta) = \prod_{i=1}^{n} (h_\theta(x^{(i)}))^{y_i} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}}$$

$$\ell(\theta) = \mathrm{Log}\, L(\theta) = \sum_{i=1}^{n} y^{(i)} \log(h(x^{(i)})) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))$$

- General Linear Model

Design choice (Linear Model)

$$X \rightarrow \boxed{\theta^T x} \xrightarrow{\eta} \text{Exponential training}$$

$\rightarrow \max_\theta \log P(y^{(i)}, \theta^T x^{(i)})$ (Train)

$\rightarrow \mathbb{E}[y; \eta] = h_\theta(x)$ (Test)

Real — Gaussian
Binary — Bernoulli
count — Poisson
R^+ — Gamma, Exponential
Dist^n — Beta

Basic Model

→ data X $x_j^{(i)}$ (i) th example ($1 \sim n$) jth feature ($1 \sim d$).

→ Linear function, Score function

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d = \sum_{i=0}^{d} \theta_i x_i$$

$$= \theta^T x = Wx + b$$

→ Cost. Loss function

Full Loss. $L(\theta) = \frac{1}{n} \sum_{i=1}^{n} L_i(f(x_i, W), y_i) + \lambda \underbrace{R(W)}_{\text{regularization}}$

1) L1 Reg $\Sigma_{k,l} |W_{k,l}|$.
2) L2 Reg $\Sigma_{k,l} W_{k,l}^2$.
3) Elastic
4) Max Norm

Loss function (Softmax)

1) Regression loss
2) Hinge loss $L_i = \sum \max(0, S_j - S_{y_i} + 1)$
3) Cross-entropy loss $L_i = -\sum (z_{ij} \log p_j + (1 - z_{ij}) \log(1 - p_j))$
4) Log likelihood loss $L_i = -\log(p_j)$

* Probability : Softmax $\frac{e^j}{\Sigma e^j}$

ex. Softmax + Log likelihood
Softmax + Cross-entropy

Activation function (0∼1로 Mapping)

1) tanh, ReLU, Leaky ReLU, MaxOut, ELU.

(2) Logistic / Sigmoid) $\frac{1}{1+e^{-x}}$

Option
- Batch Gradient Descent
- Stochastic Gradient Descent.

→ Optimization : Back Propagation 역전파
: Gradient Descent (어떻게 역전)

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Chain Rule : [local gradient] × [upstream gradient]

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