DL: part 1

linear regression

① hypothesis : $H(x) = Wx + b$

② cost func : $cost(W,b) = \| H(x) - y \|^2$

$$= \frac{1}{M} \sum_{i=1}^{M} \left[H(x_i) - y_i \right]^2$$

③ minimization : gradient descent

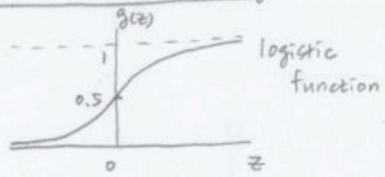
$$H(x) = Wx$$

$$W \triangleq W - \alpha \cdot \frac{1}{M} \sum_{i=1}^{M} (Wx_i - y_i) \cdot x_i$$

logistic (regression) classification : binary

$$g(z) = \frac{1}{1+e^{-z}}$$

$$z = Wx$$



① hypothesis : $H(X) = \frac{1}{1+e^{-W^T X}}$

② cost func : $cost = \frac{1}{M} \sum c(H(x), y)$

$$c(H(x), y) = -y \cdot \log H(x) - (1-y) \cdot \log (1 - H(x))$$

③ minimization : gradient descent

Softmax regression : multinomial

① hypothesis

$$\begin{bmatrix} W_{A1} & W_{A2} & W_{A3} \\ W_{B1} & W_{B2} & W_{B3} \\ W_{C1} & W_{C2} & W_{C3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \bar{y}_A \\ \bar{y}_B \\ \bar{y}_C \end{bmatrix} \begin{matrix} 2.0 \\ 1.0 \\ 0.1 \end{matrix} \rightarrow S(y_i) = \frac{e^{y_i}}{\sum e^{y_i}} \begin{matrix} 0.7 \\ 0.2 \\ 0.1 \end{matrix} \rightarrow \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

W X Y (score) Softmax prob argmax (one-hot encoding)

② cost function

$$D(S, \bar{y}) \quad \bar{y}: y(\text{true})$$

$$D(S, L) = -\sum_i L_i \cdot \log(S_i) : \text{cross-entropy}$$

$$L = \frac{1}{N} \sum_i D\{ S(WX_i + b), L_i \}$$

③ minimization

$$-\alpha \cdot \Delta L(S, L) : \text{gradient descent}$$

ML application

① learning rate
- large → overshooting
- small → local minimum

② data pre-processing : original data → zero-centered

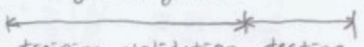
$$x_i' = \frac{x_i - \mu}{\sigma} : \text{standardization}$$

③ overfitting
- more training data
- reduce # of features → valuable features
- regularization : too big numbers in the weight

* drop-out

$$L = \frac{1}{N} \sum D(S_i, L_i) + \lambda \cdot \sum W^2$$

④ training/testing data



α, λ tuning

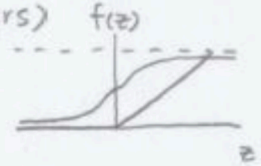
DL Basics

① activation function

Sigmoid : vanishing gradient (many layers) *

↓

ReLU (Rectified Linear Unit)



② initial value on weights

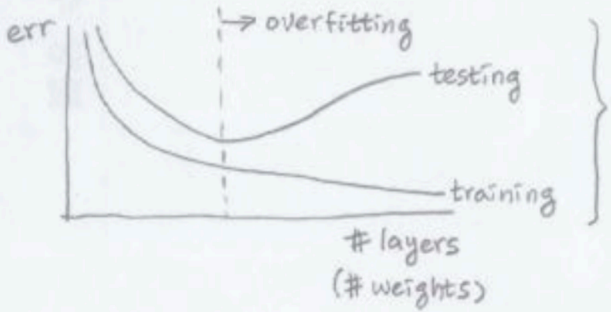
- not all 0's
- restricted Boltzman machine (RBM) → Deep Belief Nets

forward — backward (encode) (decode)

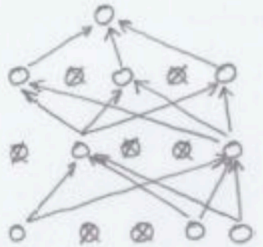
pre-training → fine tuning (x) (x, y)

- Xavier init / He init

③ drop-out



regularization "drop-out"



training : dropout = 0.5

testing : dropout = 1.0