Multilayer Perceptron

ML Lab

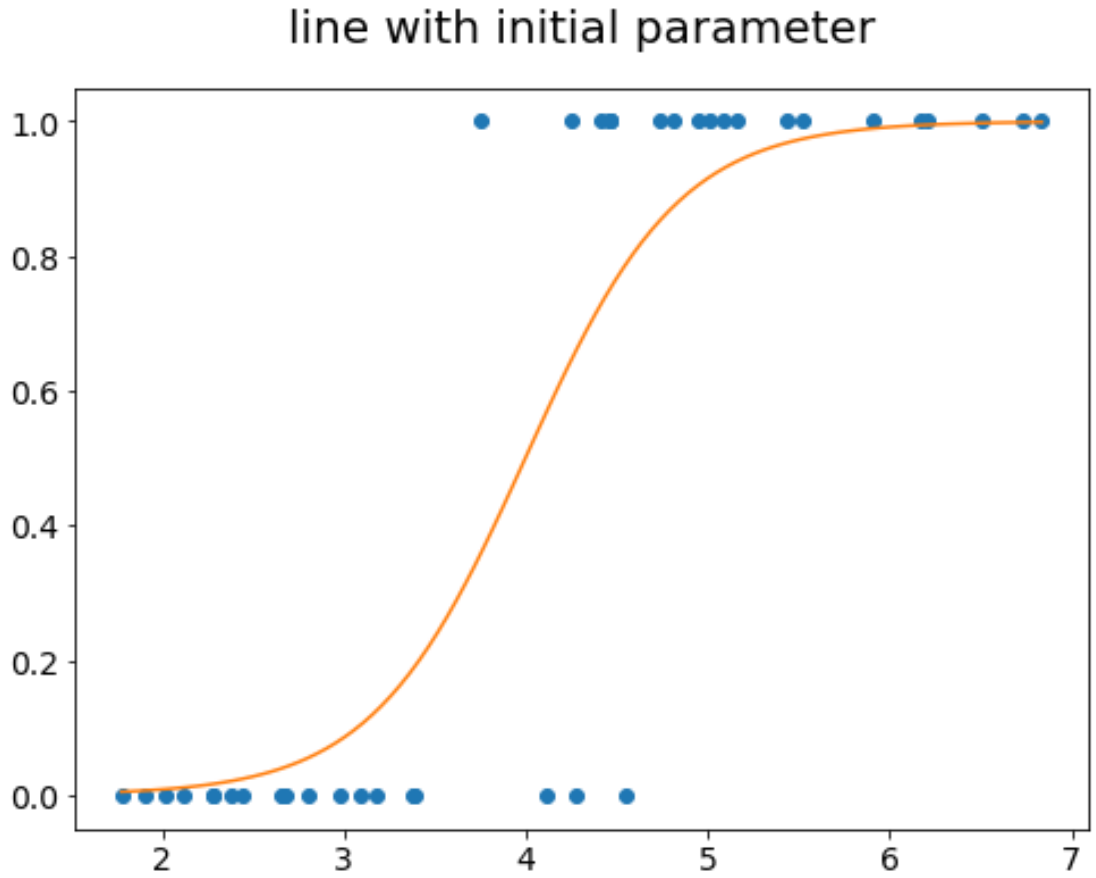
Machine Learning Laboratory
POSTECH

Table of Contents

Table of Contents

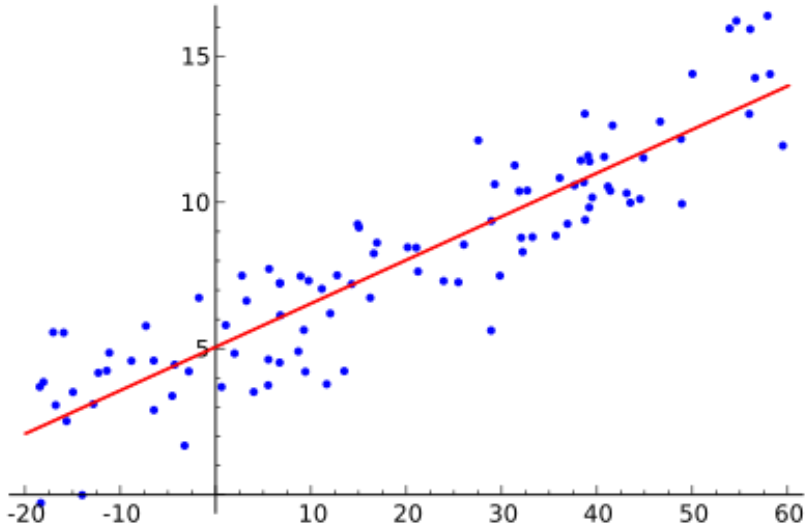
Review: Linear Regression Model

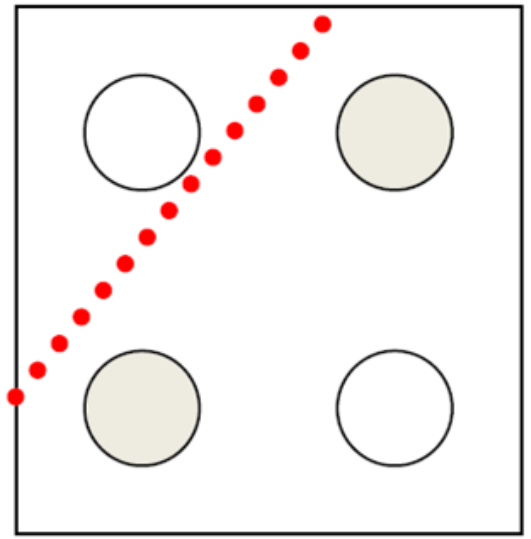
- 지난 시간에는 linear regression 모델을 가지고 학습을 진행하였다.
  - $f(x) = Wx^T + b.$

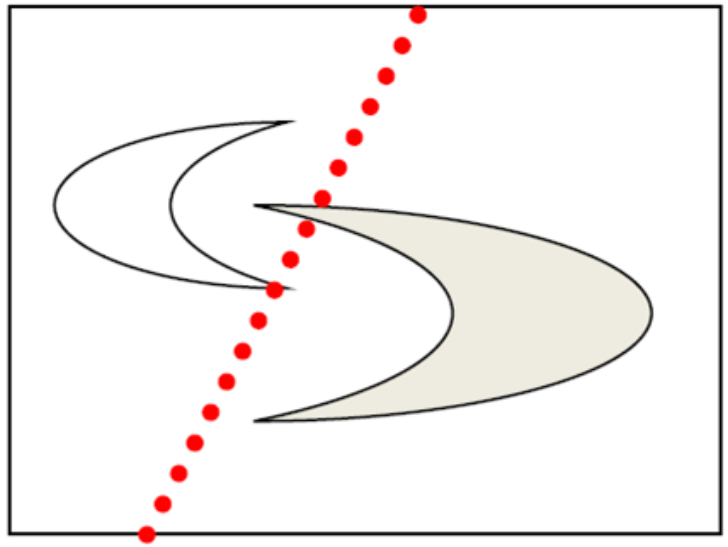


Linear Regression Model: Problem

- 하지만 linear regression model의 가장 큰 단점 중 하나는 선형 data만 처리할 수 있다.

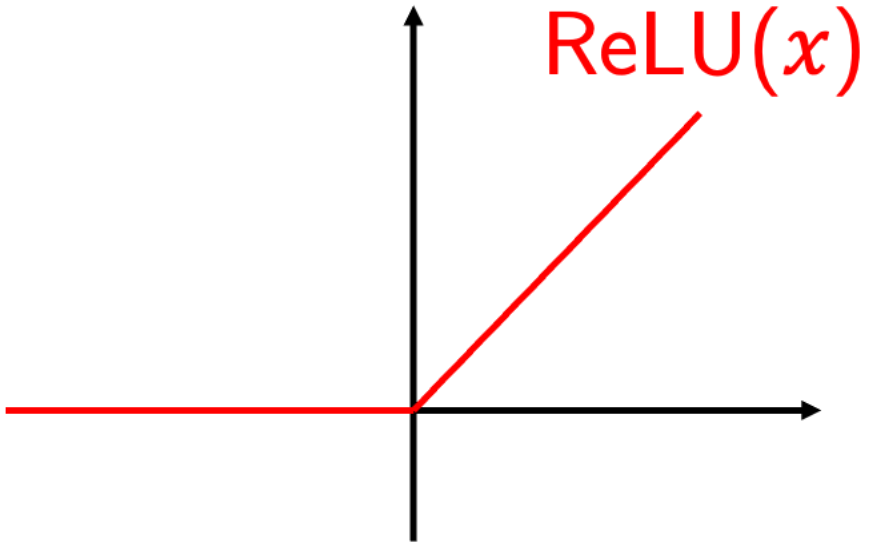


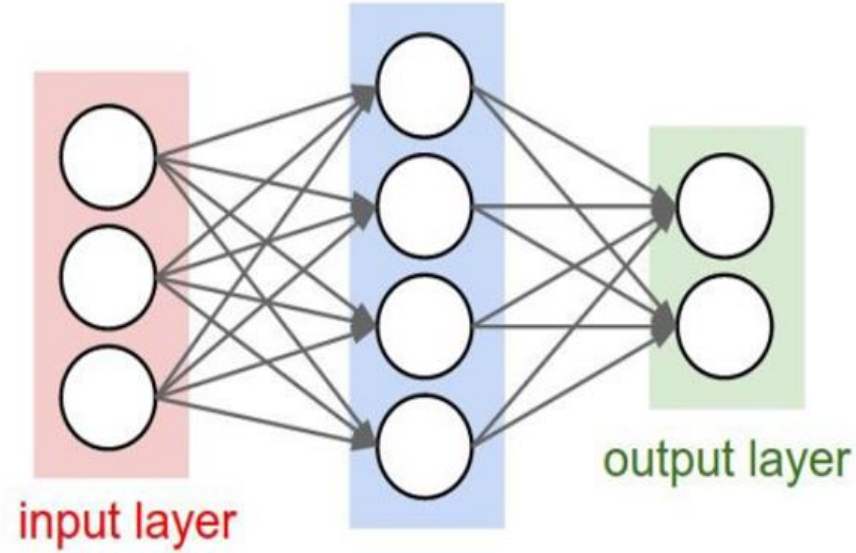




Multi-layer Perceptron (MLP)

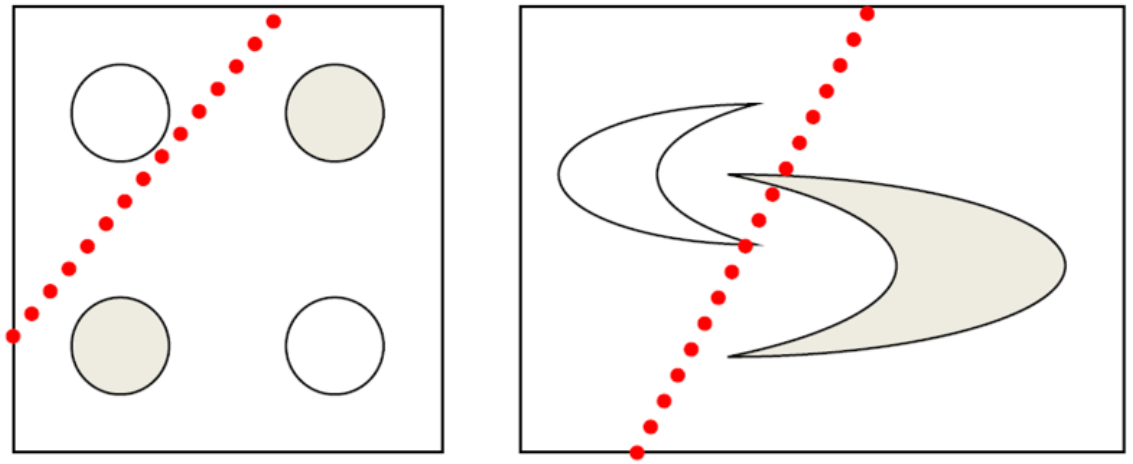
- MLP는 linear regression model을 다음과 같이 확장한 모델이다:
  - $f(x) = W_2 ReLU(W_1 x^T + b_1) + b_2$.
  - 여기서 $ReLU(x)$는 비선형(non-linear) 함수로, activation이라고 부른다.





Why Non-linear Activation Function? (1)

- 데이터가 복잡해지고, feature들의 차원이 증가하면서 데이터의 분포가 **선형적으로 표현되는 경우가 매우 적음**
- 선형적이지 않은 데이터는 선형적인 boundary로는 표현이 불가능하므로 **비선형 boundary가 필요함**



Why Non-linear Activation Function? (2)

- 만약, activation function을 사용하지 않는다면?

$$\mathbf{f} = \mathbf{W}_2(\mathbf{W}_1\mathbf{x} + \mathbf{b})$$
$$\mathbf{f} = \mathbf{W}_2\mathbf{W}_1\mathbf{x} + \mathbf{W}_2\mathbf{b}$$
$$\mathbf{f} = \mathbf{W}'\mathbf{x} + \mathbf{b}'$$

- 서로 다른 linear regression model을 많이 쌓더라도 하나의 linear regression과 같은 표현력을 가지게 됨.

MLP notation

- Single layer NN

$$f = \sigma(W_1 x + b_1)$$

- 2-layer NN

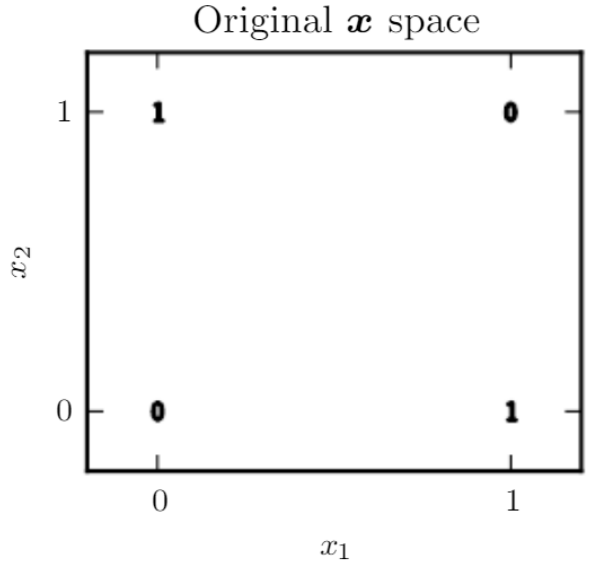
$$f = W_2 \sigma(W_1 x + b_1)$$

- 3-layer NN

$$f = W_3 \sigma(W_2 \sigma(W_1 x + b_1) + b_2)$$

MLP on XOR Problem (1)

- 다음과 같은 X, y가 data로 주어졌다고 하자.
  - 이를 2-layer MLP로 풀어보자!
  - $f(x) = w^T \max(0, XW + c) + b$.



$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

MLP on XOR Problem (2)

- Parameter들의 값을 다음과 같이 설정해보자:

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

MLP on XOR Problem (3)

$$f(x) = w^T \max(0, XW + c) + b$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$XW + C = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix} \qquad \max\{0, XW + C\} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix} \qquad w^T \max\{0, XW + C\} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$w^T \max\{0, XW + C\} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

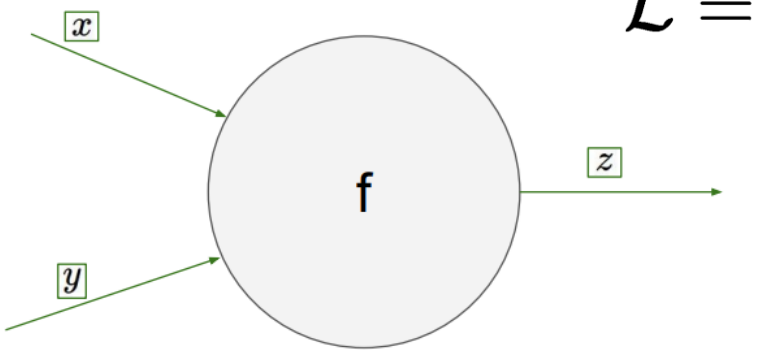
MLP: Summary

- MLP가 XOR과 같은 linear regression 모델은 못 푸는 비선형 data를 잘 푸는 것을 확인했다.
  - Q) 학습은 어떻게 할 것인가? 각 parameter들에 대한 gradient를 쉽게 계산하는 방법은 없을까?

Table of Contents

Backpropagation

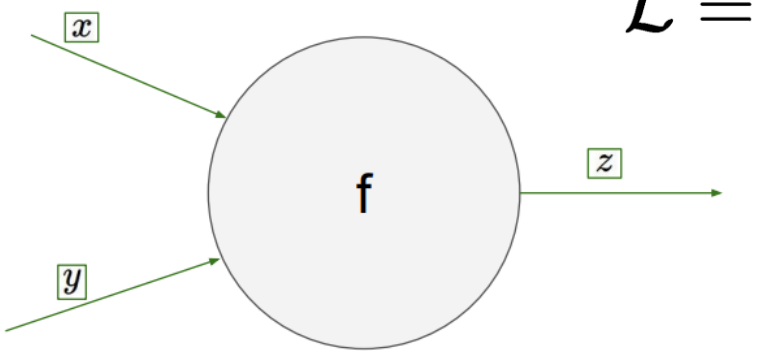
$$\mathcal{L} = (t - z)^2,\ t : label$$



- 다음과 같이 Neural Net이 있고, loss function은 L2-norm을 사용할 때,
  - $\frac{\partial \mathcal{L}}{\partial z}$는 쉽게 구할 수 있음
  - Backpropagation에서는 $\frac{\partial \mathcal{L}}{\partial z}$ 를 이용하여 $\frac{\partial \mathcal{L}}{\partial x}, \frac{\partial \mathcal{L}}{\partial y}$ 를 구함

Backpropagation

$$\mathcal{L} = (t - z)^2,\ t : label$$



- Chain Rule $\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial z}\frac{\partial z}{\partial x}$

$$z = f(x, y) = x^2 - xy \quad x = 3,\ y = 1,\ t = 5$$

$$z = 6,\ \mathcal{L} = 1,\ \frac{\partial \mathcal{L}}{\partial z} = -2 * (5 - 6) = 2$$

$$\frac{\partial z}{\partial x} = 2 * 3 - 1 = 5$$

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial z}\frac{\partial z}{\partial x} = 2 * 5 = 10$$

Connect with Neural Net

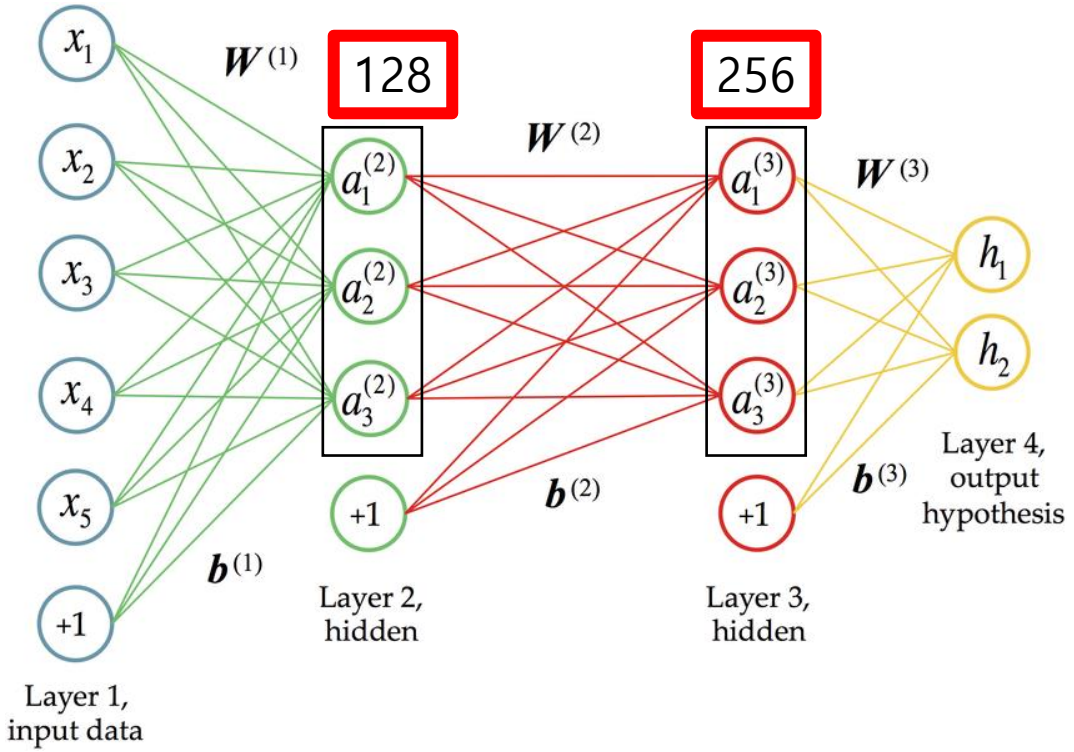
- $\ell = 1, \ldots, N$ layer를 갖는 neural net
  - L번째 hidden layer를 다음과 같이 표현

$$h_L = W_L \sigma(h_{L-1}) + b_L$$

- $\frac{\partial \mathcal{L}}{\partial W_L} = \frac{\partial \mathcal{L}}{\partial h_L} \frac{\partial h_L}{\partial W_L}$ $\frac{\partial \mathcal{L}}{\partial W_{L-1}} = \frac{\partial \mathcal{L}}{\partial h_L} \frac{\partial h_L}{\partial W_{L-1}}$ $\frac{\partial \mathcal{L}}{\partial h_L} \frac{\partial h_L}{\partial h_{L-1}} \frac{\partial h_{L-1}}{\partial W_{L-1}}$
  - 주황색 박스부분은 이미 이전에 계산된 부분을 재사용
  - 연두색 박스부분 역시 계산이 가능
  - 위 과정을 반복하면 모든 weight, bias에 대해서 gradient를 계산 가능

Practice: MLP on MNIST

- MNIST data를 분류하는 MLP 구현하기
  - 2 hidden layer (1st hidden layer : 128, 2nd hidden layer : 256)



Thank You :)