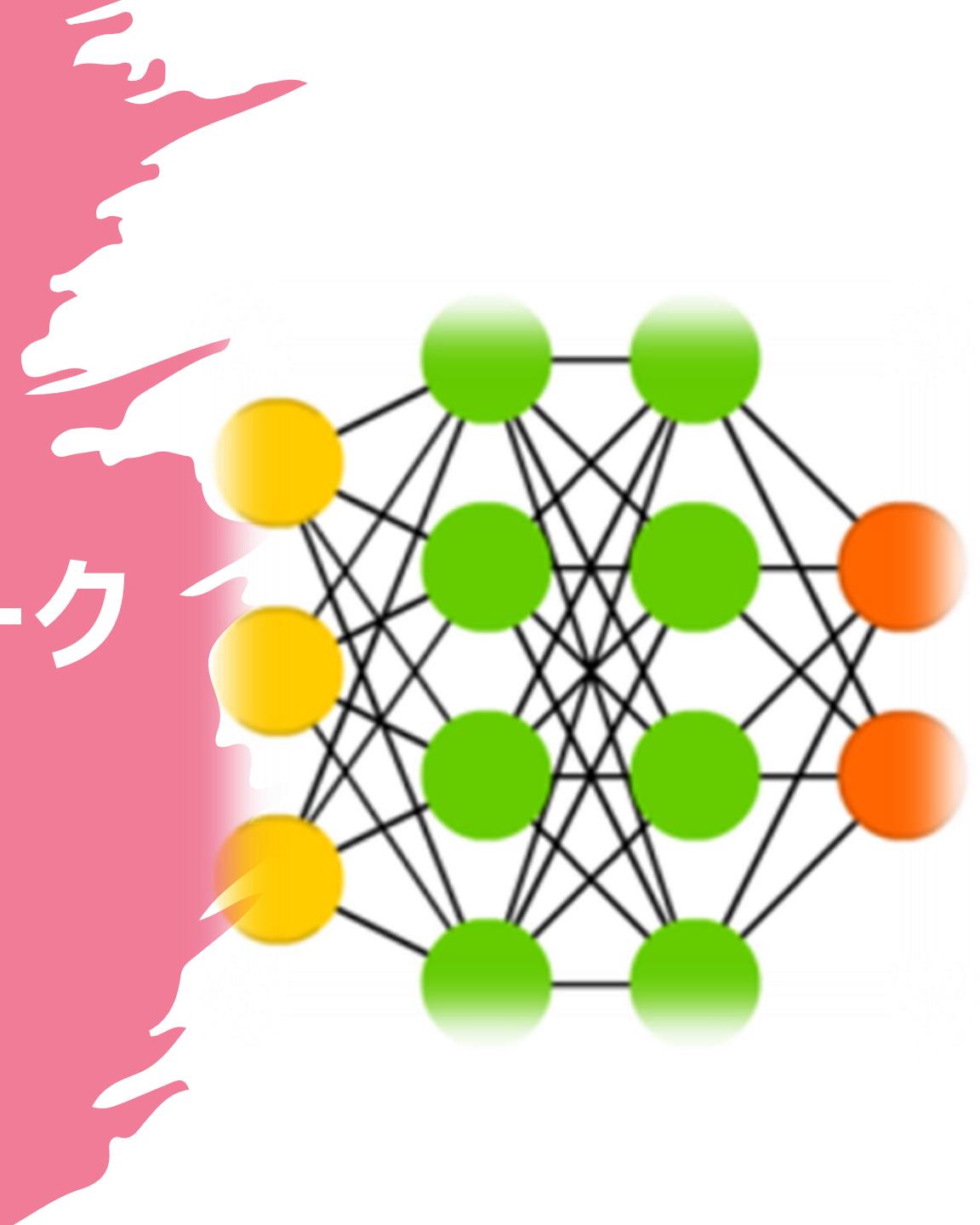


手計算 ニューラルネットワーク 入門 ①順伝播編





- 第1回 理論：順伝播
- 第2回 理論：逆伝播
- 第3回 実装
- 第4回 実験
- 第5回

ニューラルネットワーク
を完全に理解したい



pythonのライブラリ使うだけ

なんとなく理論は知ってるけど
結局呪文を呪文になってしまってる

引数やパラメータの説明を聞いても
多分理解できない（自分が）

```
from tensorflow.python.keras.models import Sequential  
from tensorflow.python.keras.layers import Dense  
  
model = Sequential()  
model.add(Dense(...))
```



手計算ニューラルネットワーク

完全に理解したい

⇒手計算できるレベルで理解する

⇒for文if文で実装ができる

最終目標（最低限）

ニューラルネットワークを完全に理解する

C++でニューラルネットワークを実装し
何かしらの問題を解く

最終目標（理想）

Kaggle Titanic in C++



KAGGLE · GETTING STARTED PREDICTION COMPETITION · ONGOING

Submit Prediction

...

Titanic - Machine Learning from Disaster

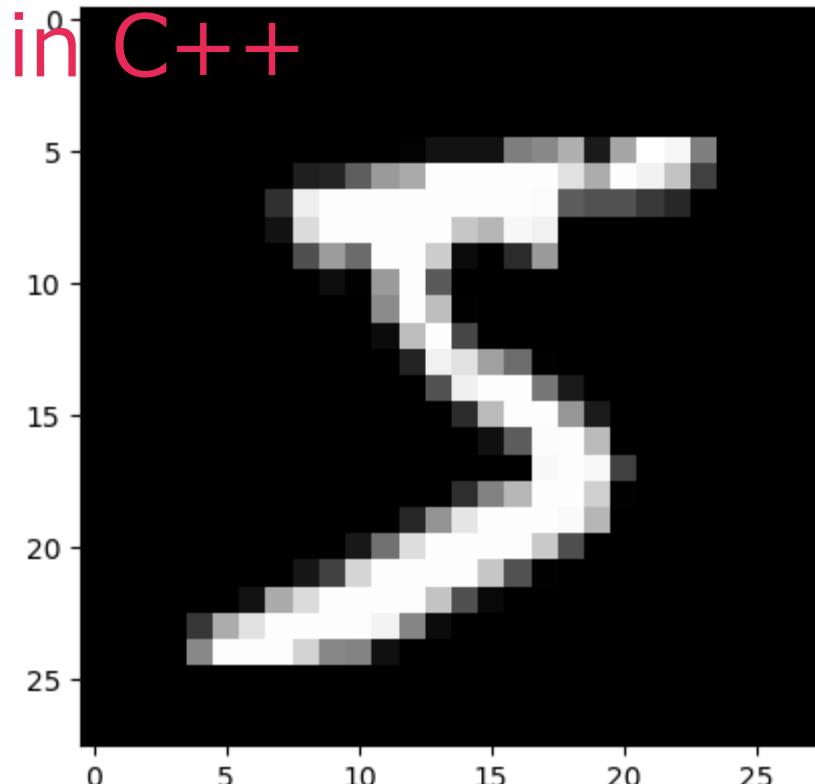
Start here! Predict survival on the Titanic and get familiar with ML basics



最終目標（理想）

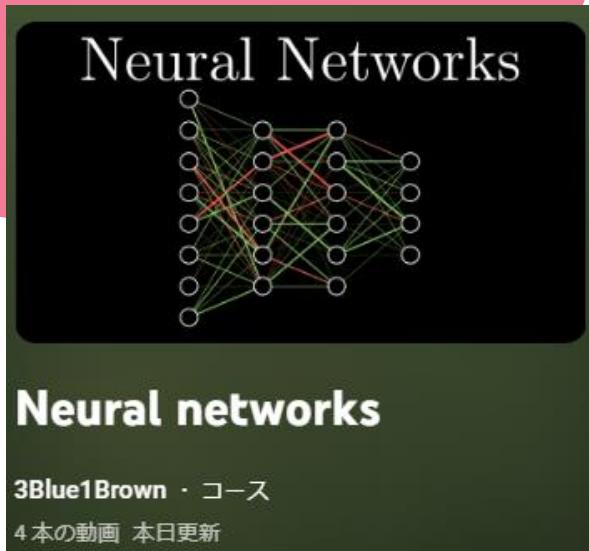
MNISTデータセットの手書き文字認識 in C++

0	4	1	9	2	1	3	1	4	3
5	3	6	1	7	2	8	6	9	4
0	9	1	1	2	4	3	2	7	3
8	6	9	0	5	6	0	7	6	1
8	7	9	3	9	8	5	9	3	3
0	7	4	9	8	0	9	4	1	4
4	6	0	4	5	6	1	0	0	1
7	1	6	3	0	2	1	1	7	9
0	2	6	7	8	3	9	0	4	6
7	4	6	8	0	7	8	3	1	5



28×28×256

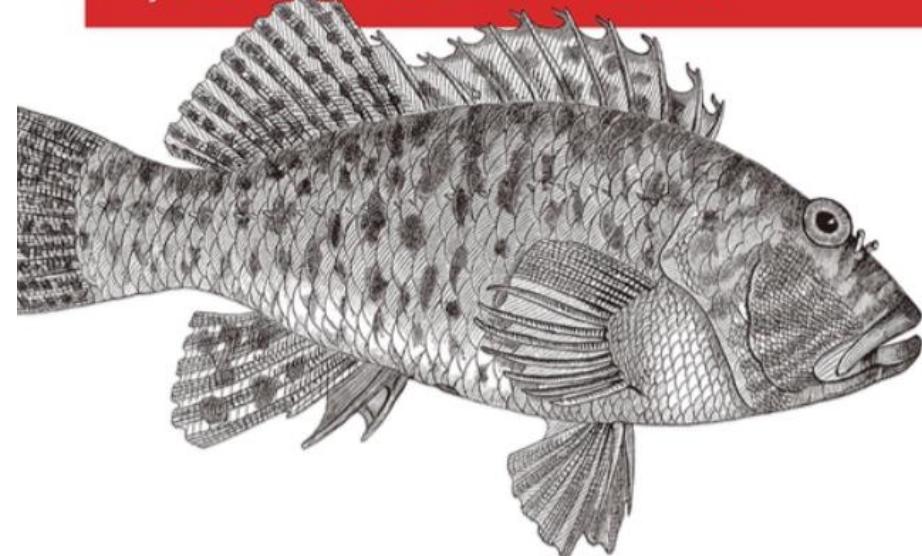
参考文献



ゼロから作る

Deep Learning

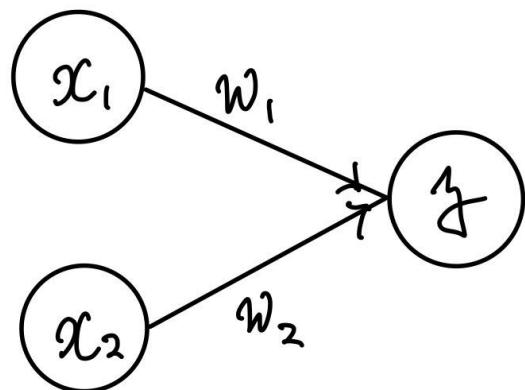
Pythonで学ぶディープラーニングの理論と実装



パーセプトロン
iPadのご準備を



パーセプトロン



ニューロン
w : 重み
θ : 閾値

$$y = \begin{cases} 1 & , w_1x_1 + w_2x_2 > \theta \\ 0 & , w_1x_1 + w_2x_2 \leq \theta \end{cases}$$

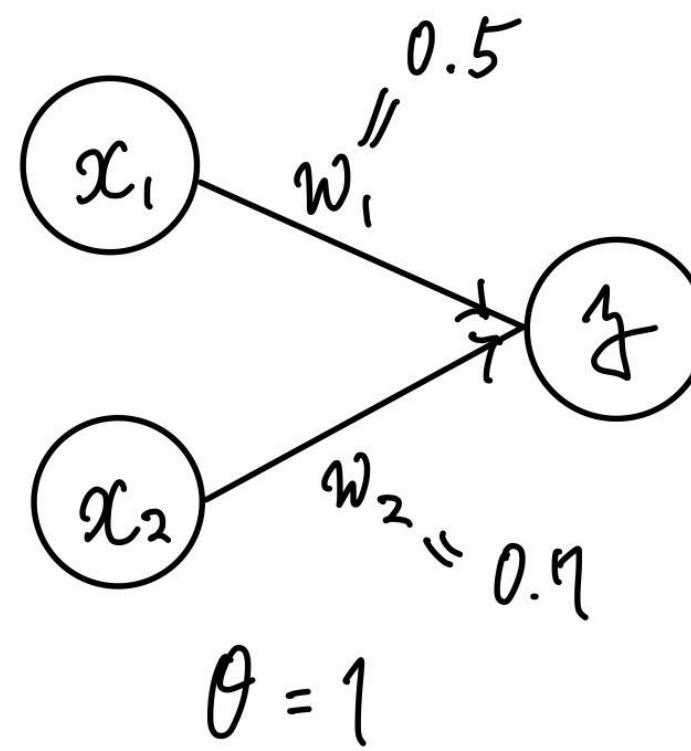
$y=1 \rightarrow$ ニューロンが発火する

2入力パーセプトロン

パーセプトロン-簡単な関数の設計

AND

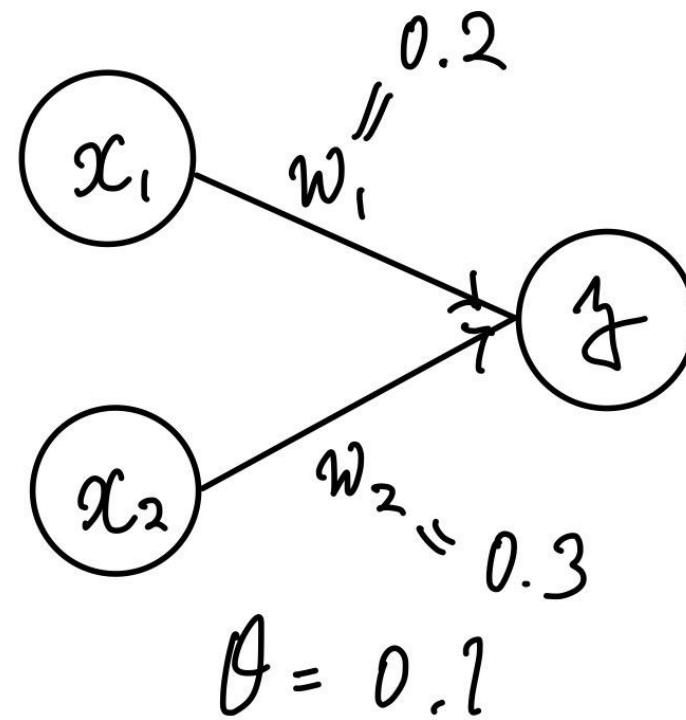
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



パーセプトロン-簡単な関数の設計

OR

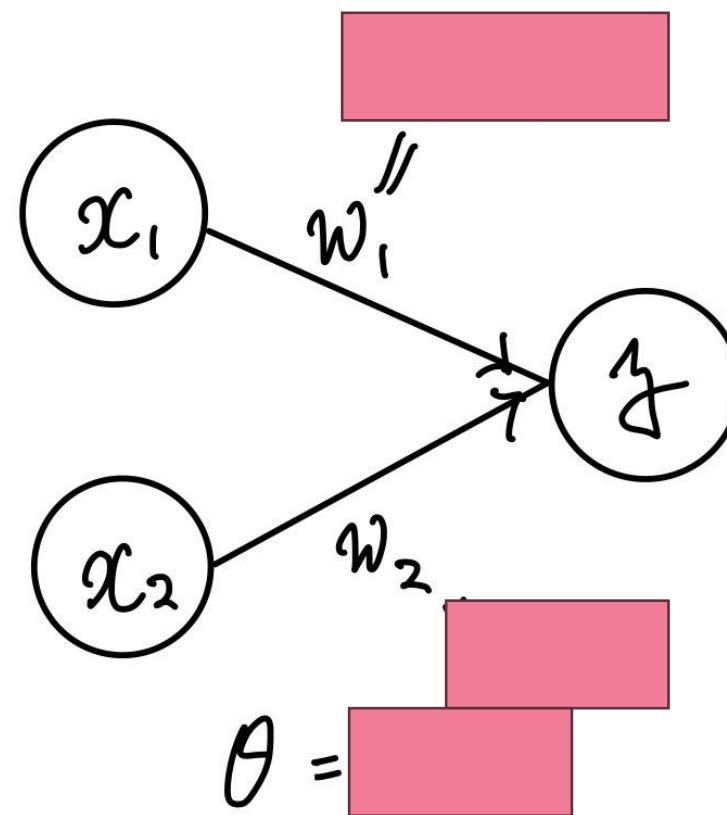
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



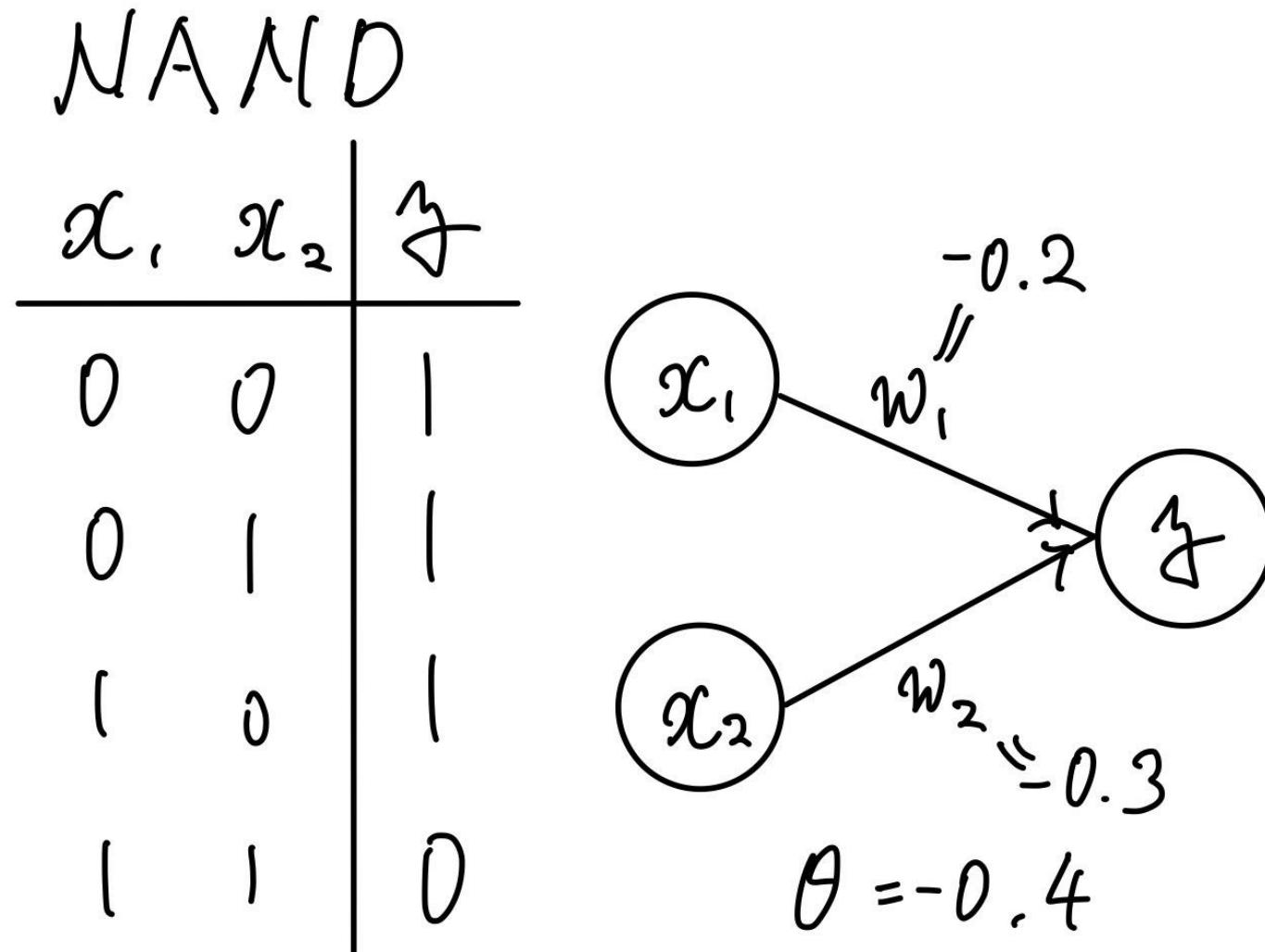
パーセプトロン-簡単な関数の設計 問題

NAND

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

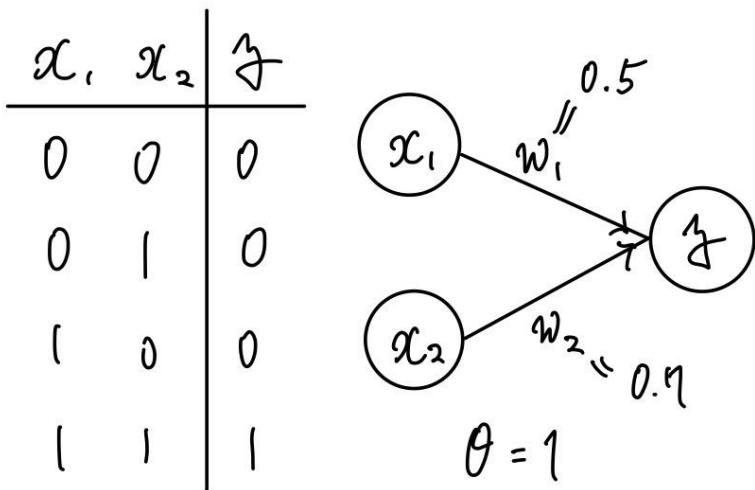


パーセプトロン-簡単な関数の設計 解答

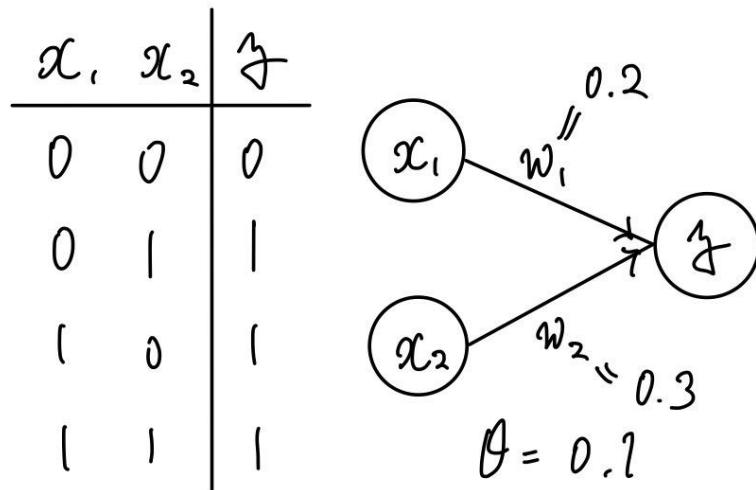


パーセプトロン

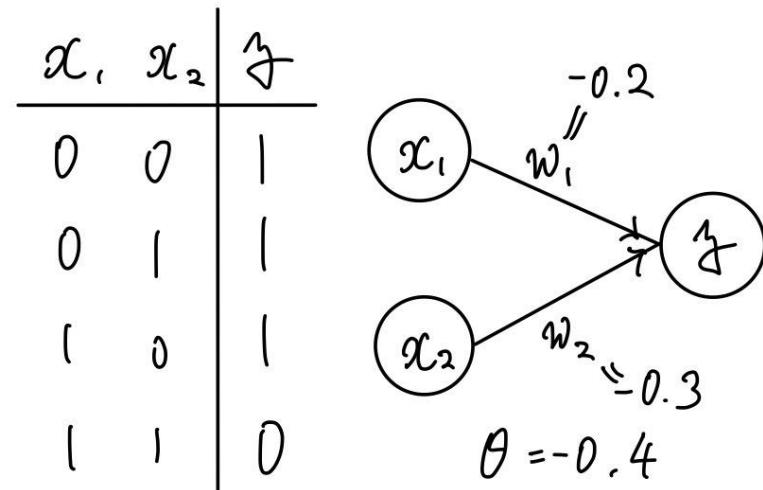
AND



OR



NAND



この関数もパーセプトロンの構造は同じ
違うのはパラメータ (w, θ) だけ

パーセプトロン-バイアスの導入

$$\hat{y} = \begin{cases} 1 & , w_1x_1 + w_2x_2 > \theta \\ 0 & , w_1x_1 + w_2x_2 \leq \theta \end{cases}$$

$$\Leftrightarrow \hat{y} = \begin{cases} 1 & , w_1x_1 + w_2x_2 + b \cdot 1 > 0 \\ 0 & , w_1x_1 + w_2x_2 + b \cdot 1 \leq 0 \end{cases}, b := -\theta \quad \text{バイアス}$$

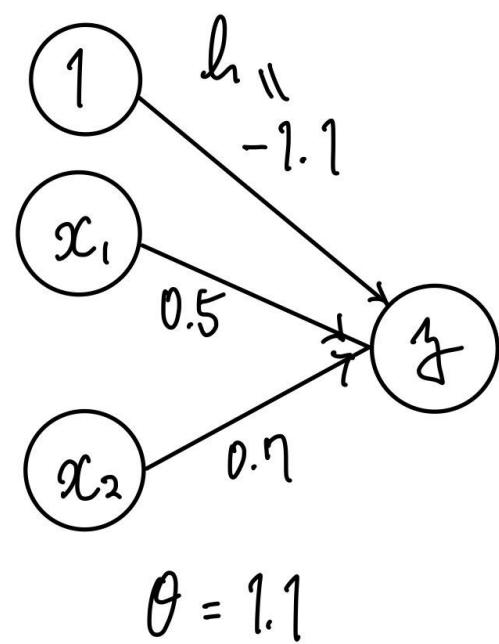
パーセプトロン-バイアスの導入

$$\hat{y} = \begin{cases} 1, & w_1x_1 + w_2x_2 > \theta \\ 0, & w_1x_1 + w_2x_2 \leq \theta \end{cases}$$

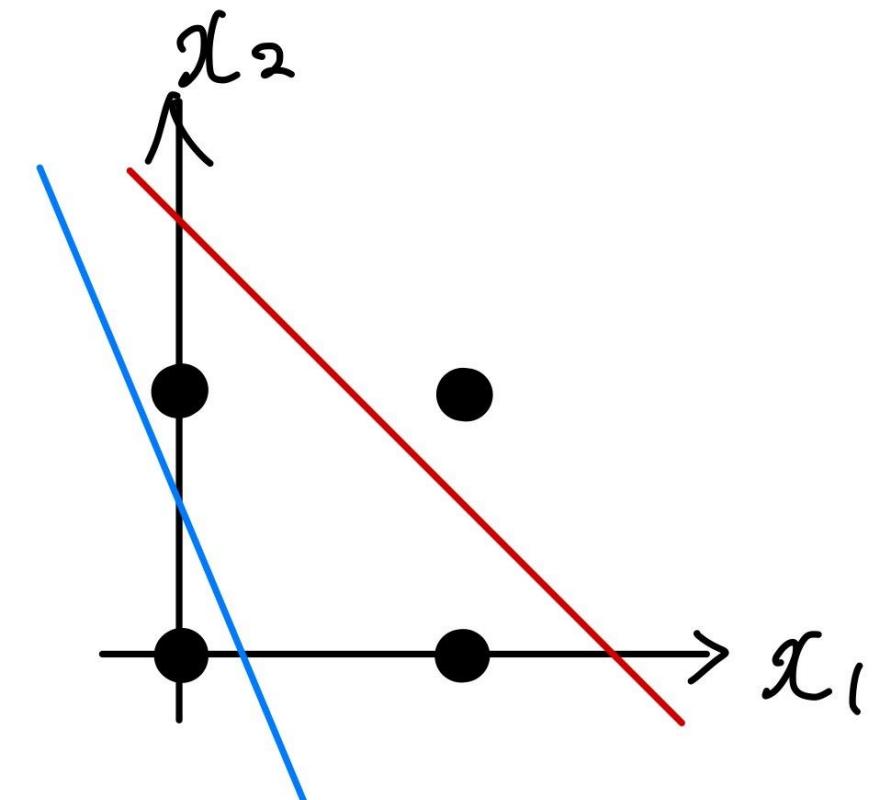
$$\Leftrightarrow \hat{y} = \begin{cases} 1, & w_1x_1 + w_2x_2 + b \cdot 1 > 0, \text{ バイアス } \\ 0, & w_1x_1 + w_2x_2 + b \cdot 1 \leq 0, \quad b := -\theta \end{cases}$$

		\hat{y}
x_1	x_2	
0	0	0
0	1	0
1	0	0
1	1	1

AND



パーセプトロン-線形分離

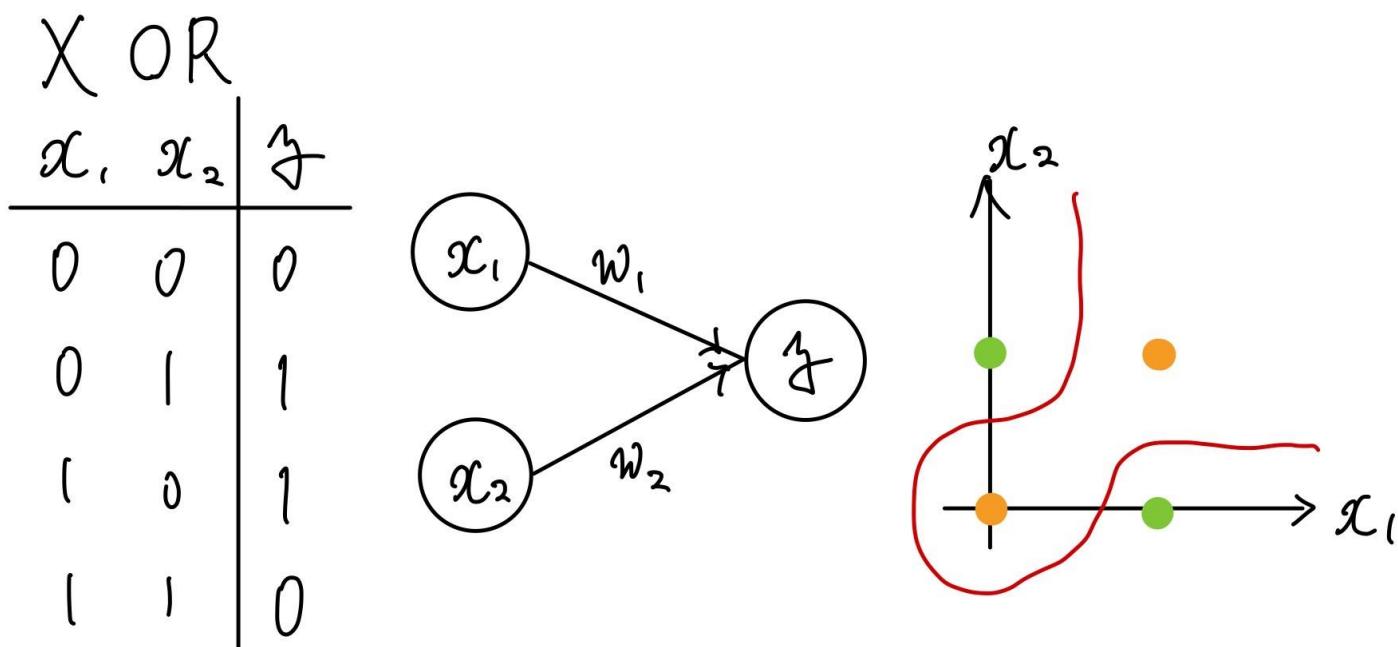


$$w_1 x_1 + w_2 x_2 + b > 0 \Rightarrow z = 1$$

$$\Leftrightarrow x_2 > -\frac{w_1}{w_2} x_1 - \frac{b}{w_2} \Rightarrow z = 1$$

\Leftrightarrow 1次関数の上 $\Rightarrow z = 1$

パーセプトロン-限界 非線形分離



1 次関数で領域を
分割できない。



パーセプトロンの限界

パーセプトロン-非線形分離 多層パーセプトロン

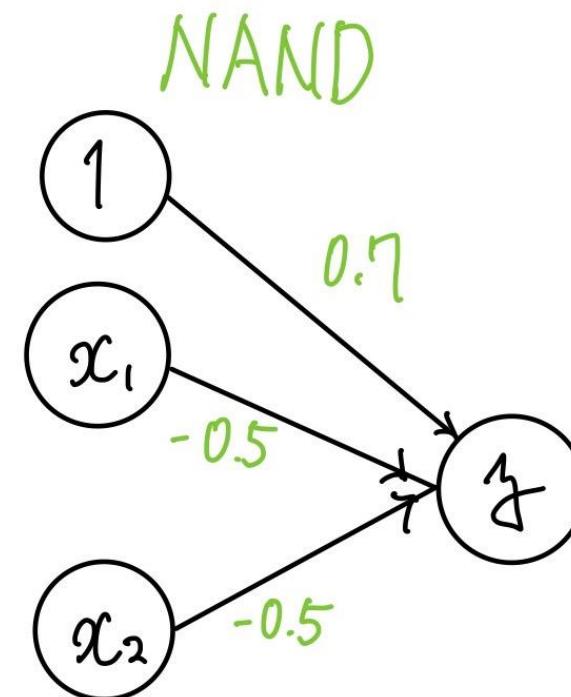
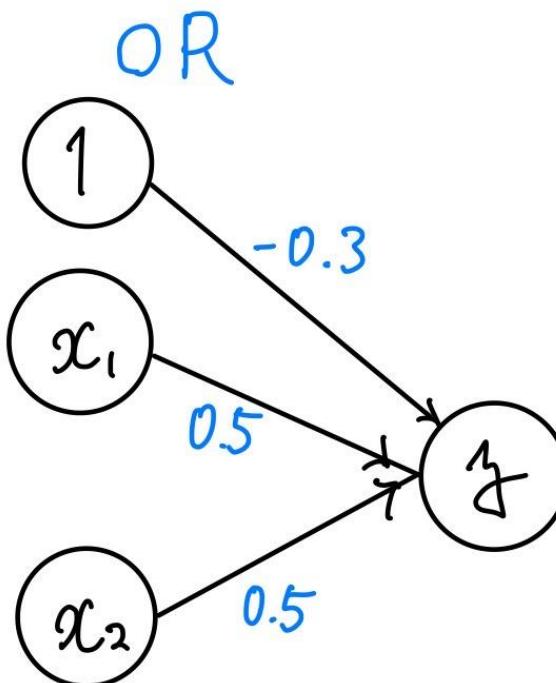
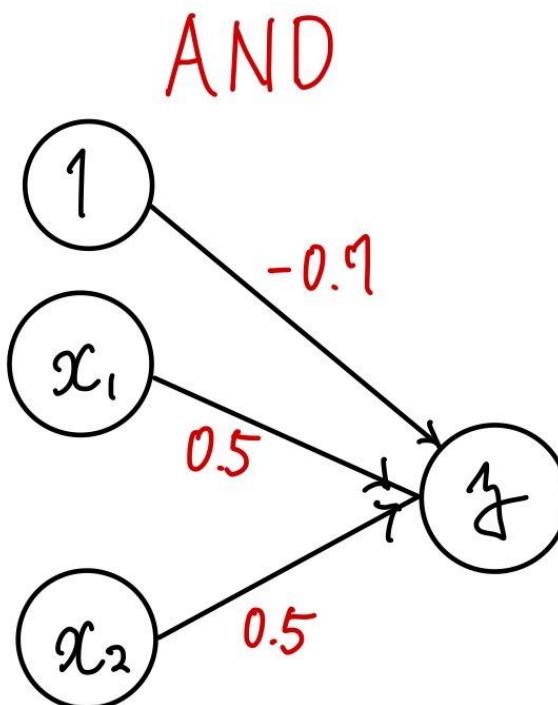
論理数学において

$$\begin{aligned}f_{\text{OR}}(A, B) &= A \bar{B} + \bar{A} B \\&= A \bar{A} + A \bar{B} + \bar{A} B + B \bar{B} \\&= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B}) \\&= (A + B)(\bar{A} + \bar{B}) \\&= (A + B) \overline{AB} \\&= f_{\text{AND}}(f_{\text{OR}}(A, B), f_{\text{NAND}}(A, B))\end{aligned}$$

関数を複数回適用する
さっきの線形で分離できなかったやつ
→ 多層パーセプトロンは
非線形領域を表現可能

パーセプトロン-多層パーセプトロンでXORの設計

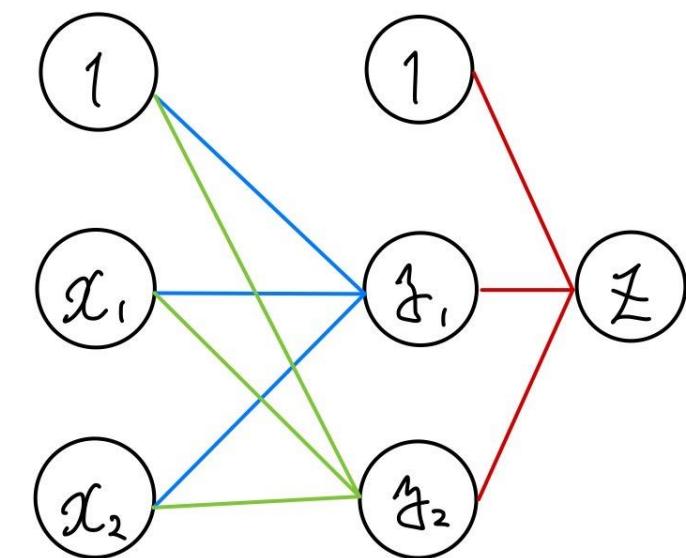
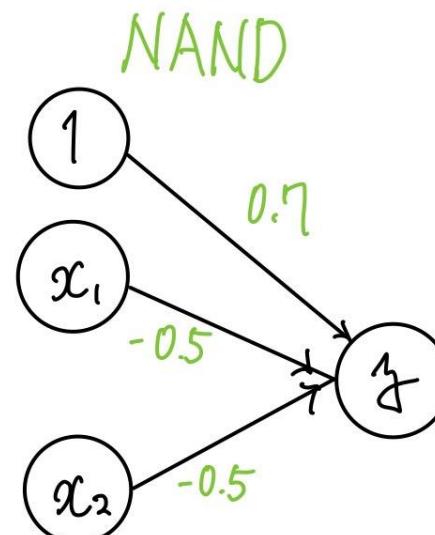
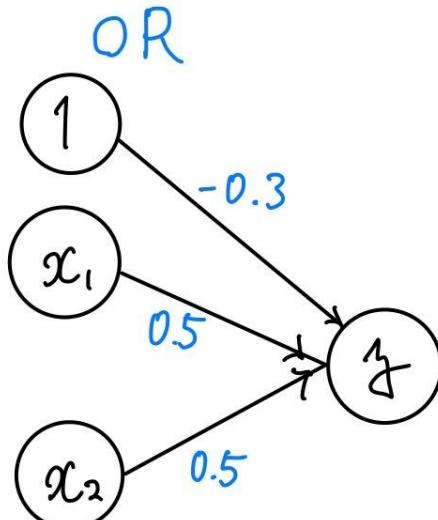
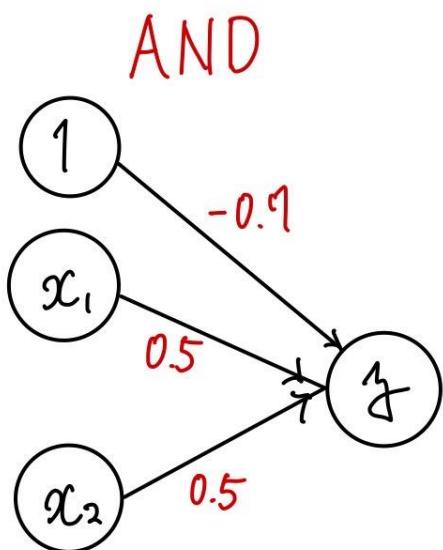
$$f_{\text{XOR}}(A, B) = f_{\text{AND}}(f_{\text{OR}}(A, B), f_{\text{NAND}}(A, B))$$



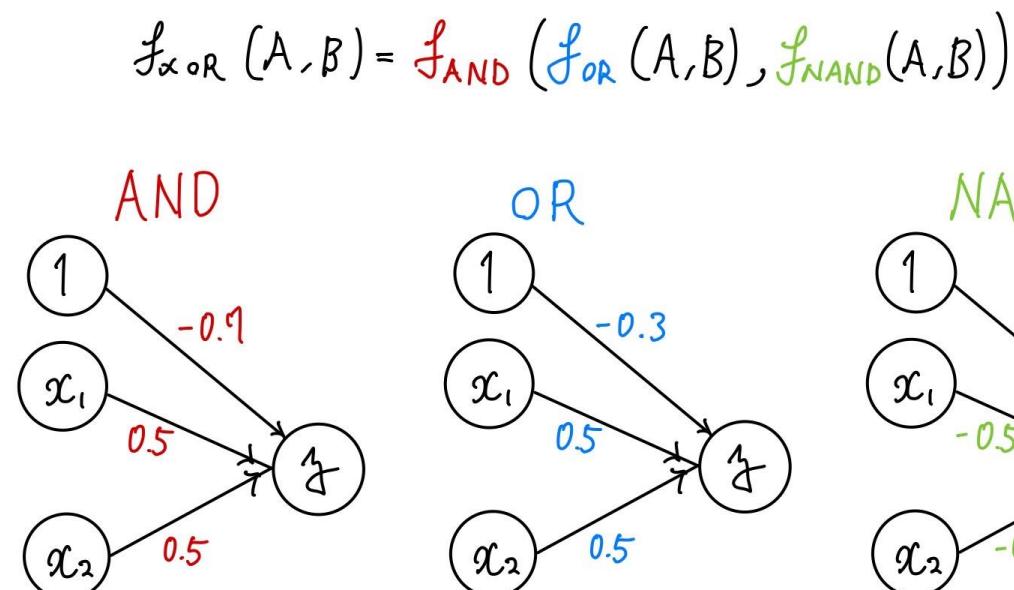
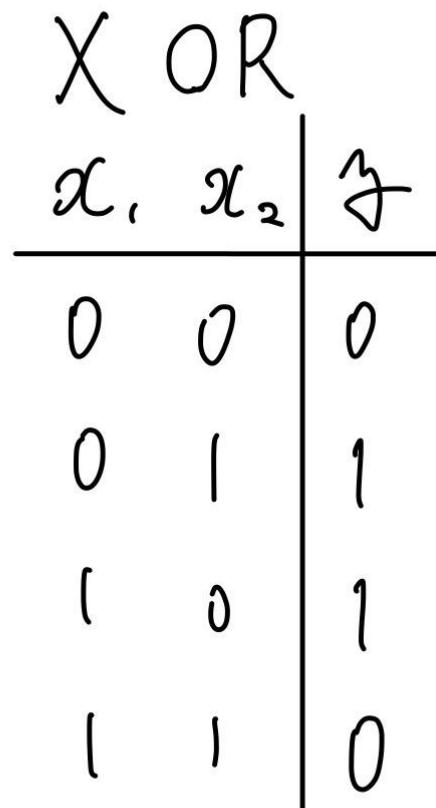
パーセプトロン-多層パーセプトロンでXORの設計

$$f_{\text{XOR}}(A, B) = f_{\text{AND}}(f_{\text{OR}}(A, B), f_{\text{NAND}}(A, B))$$

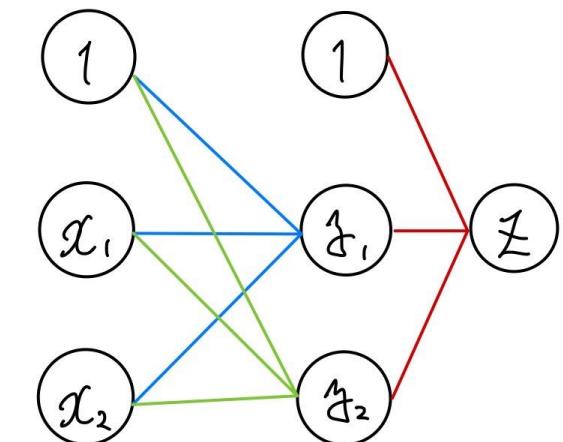
2入力2(多)層 パーセプトロン



パーセプトロン-多層パーセプトロンでXORの設計

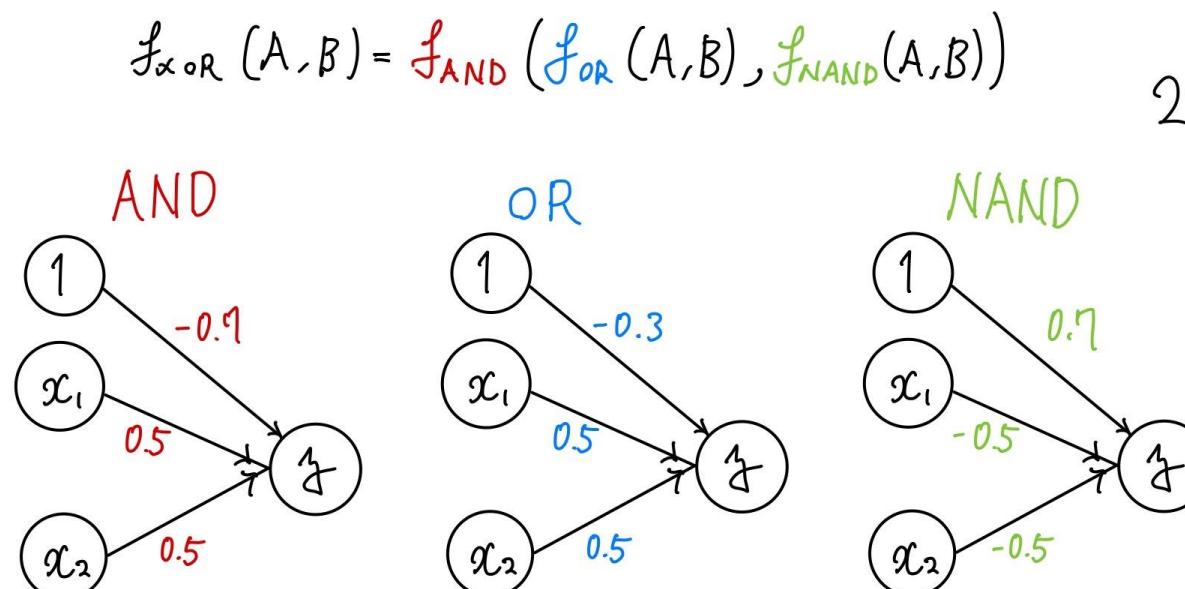


2入力2(多)層パーセプトロン

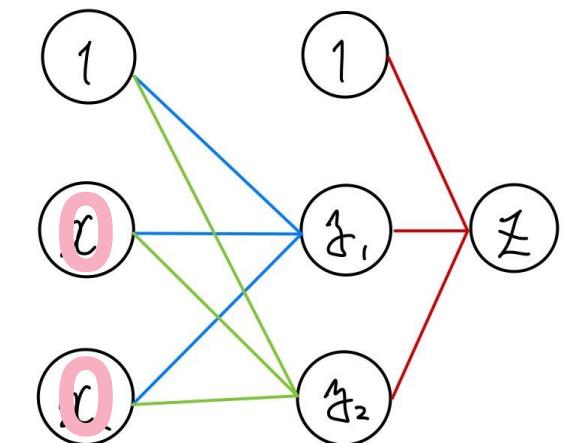


パーセプトロン-XOR確認①

X OR		γ
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

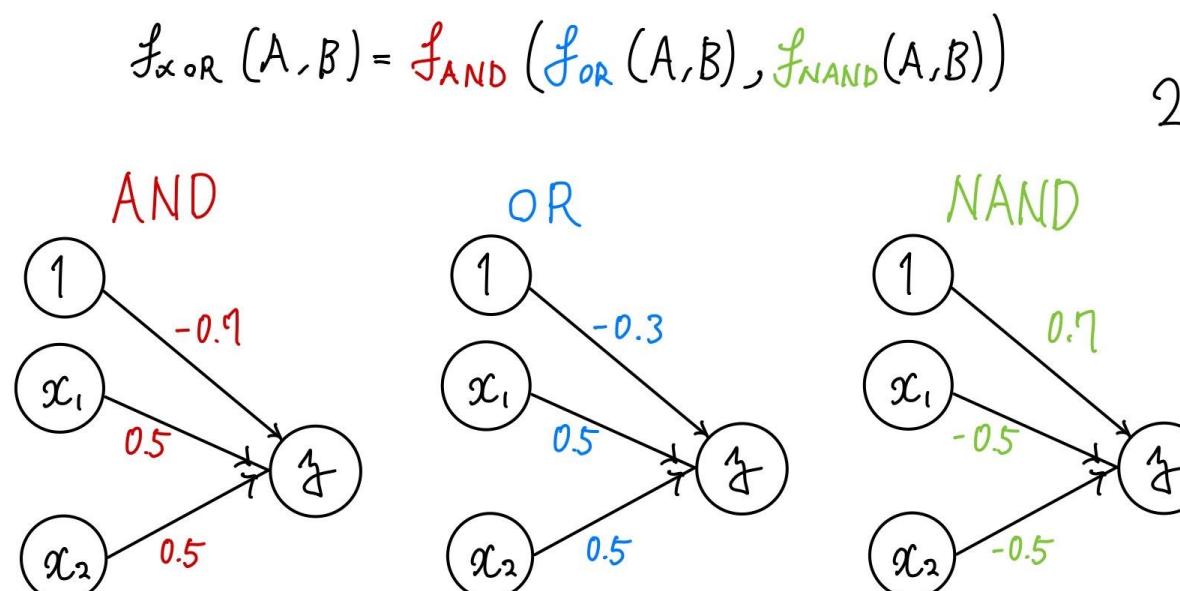


2入力2(多)層パーセプトロン

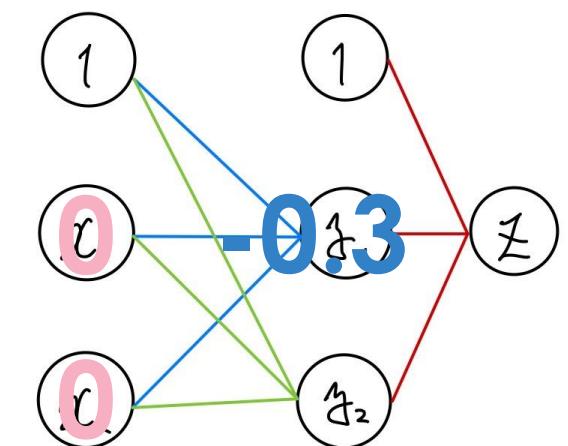


パーセプトロン-XOR確認①

X OR		γ
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

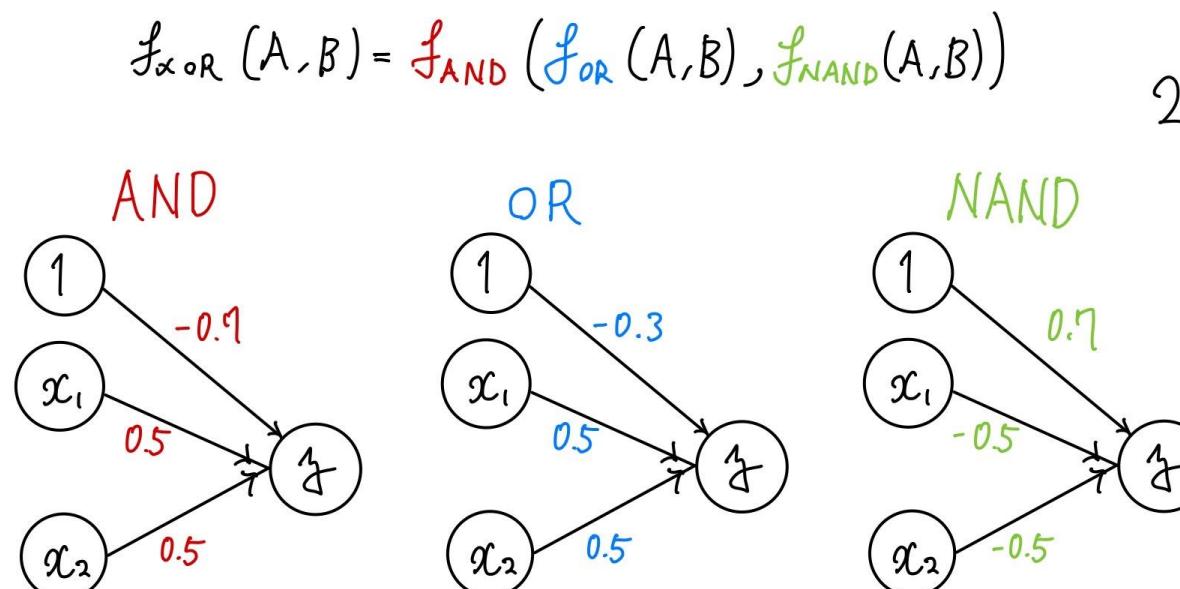


2入力2(多)層パーセプトロン

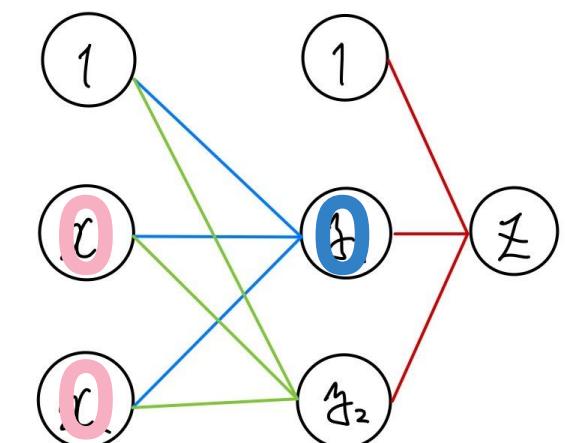


パーセプトロン-XOR確認①

X OR		y
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

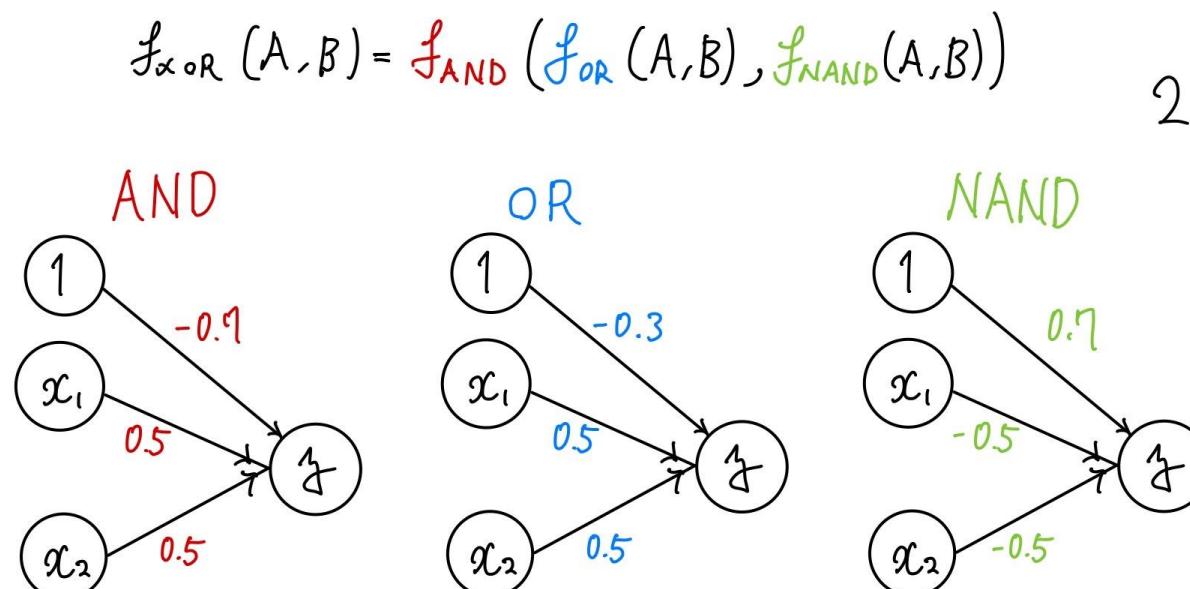


2入力2(多)層パーセプトロン

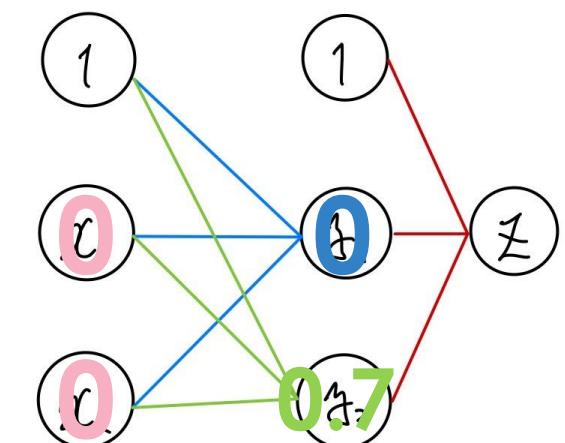


パーセプトロン-XOR確認①

X OR		γ
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

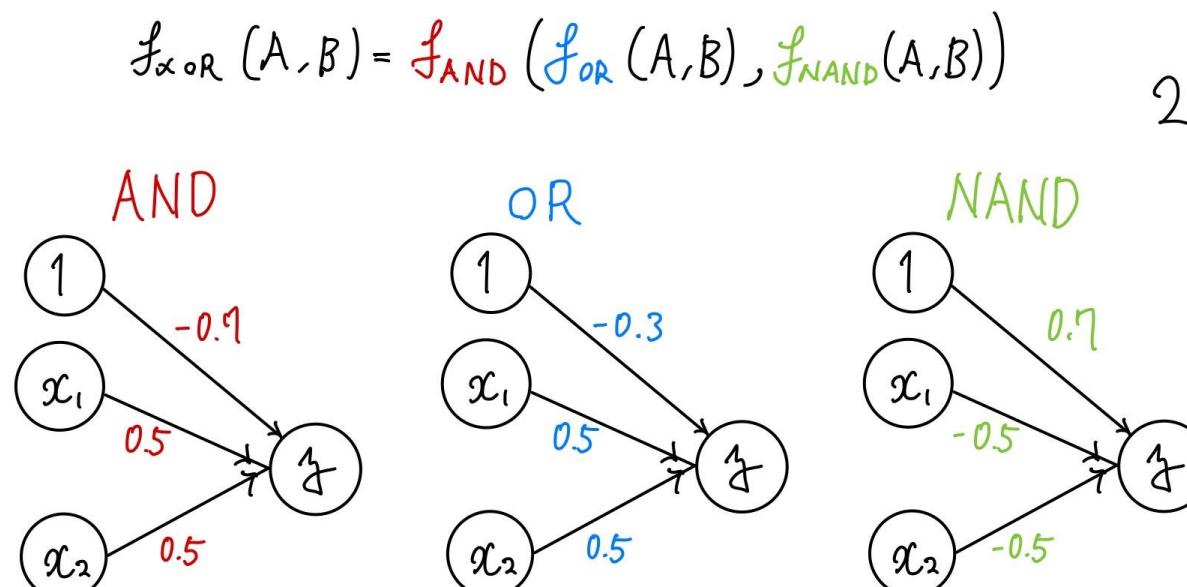


2入力2(多)層パーセプトロン

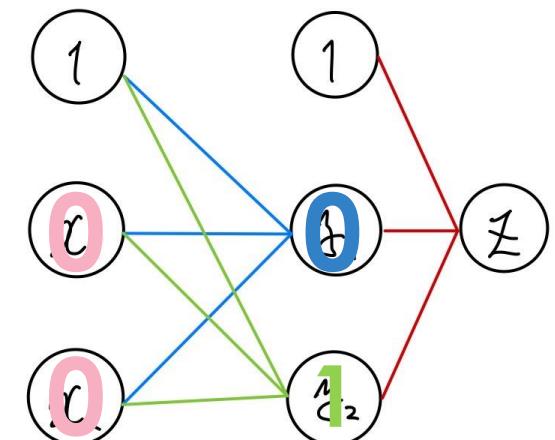


パーセプトロン-XOR確認①

X OR		γ
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

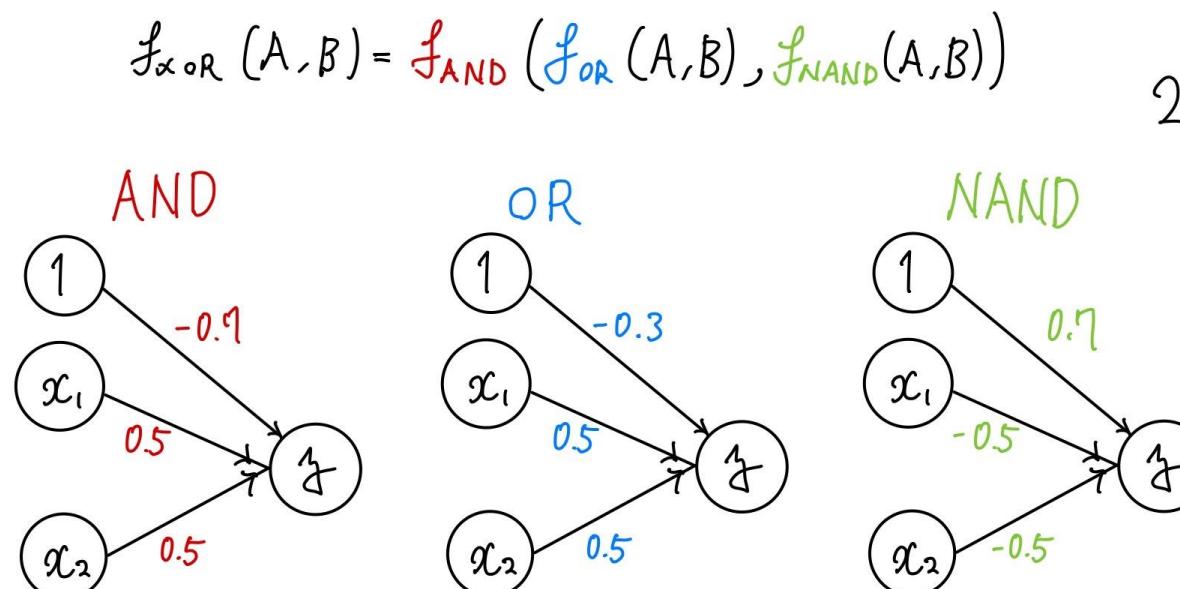


2入力2(多)層パーセプトロン

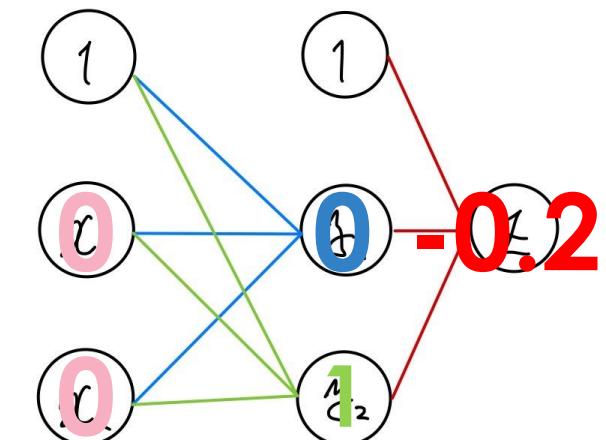


パーセプトロン-XOR確認①

X OR		y
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0

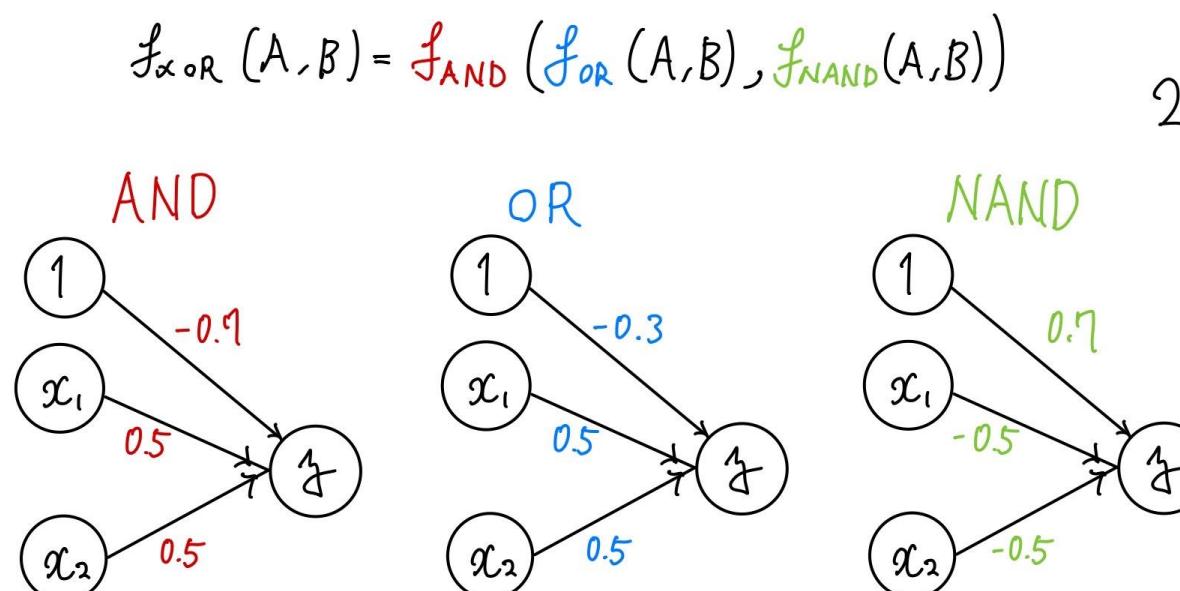


2入力2(多)層パーセプトロン

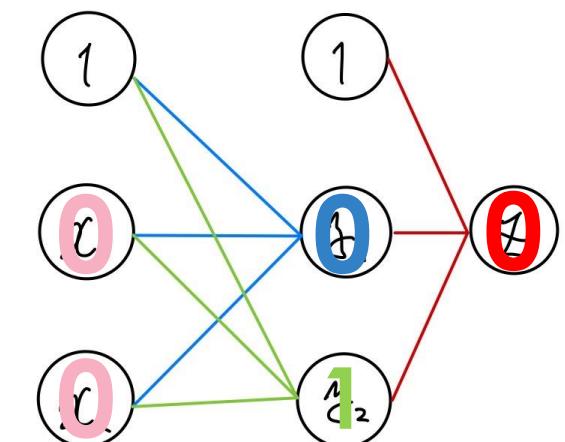


パーセプトロン-XOR確認①

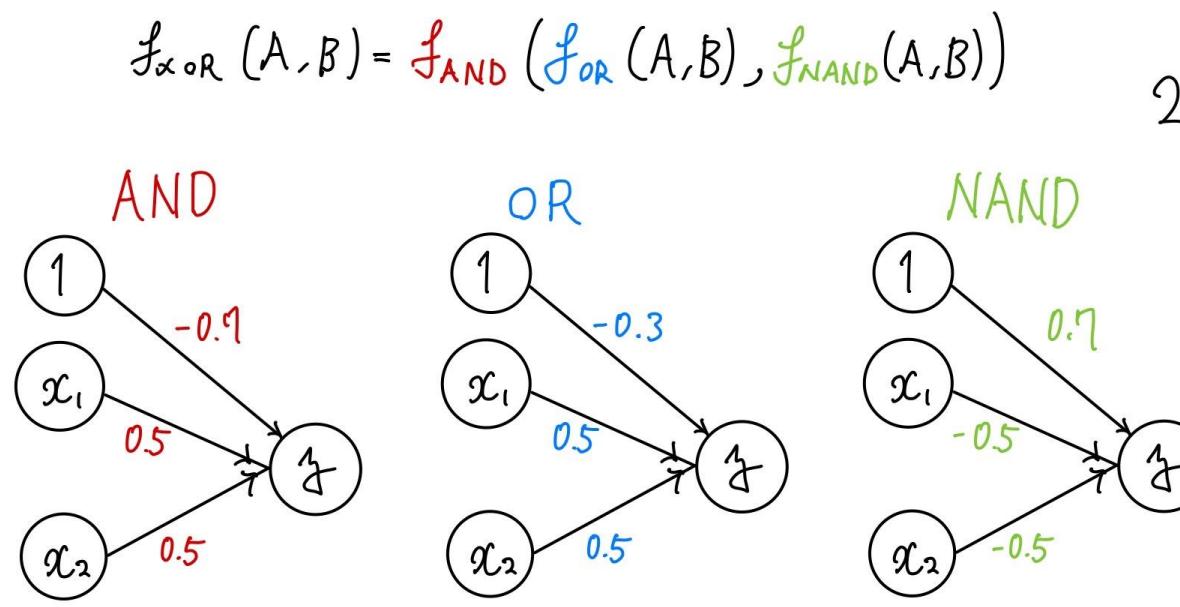
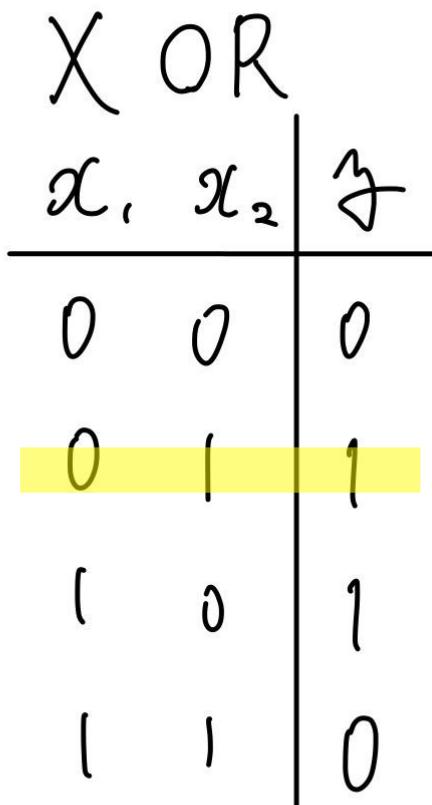
X OR		y
x_1	x_2	
0	0	0
0	1	1
1	0	1
1	1	0



2入力2(多)層パーセプトロン

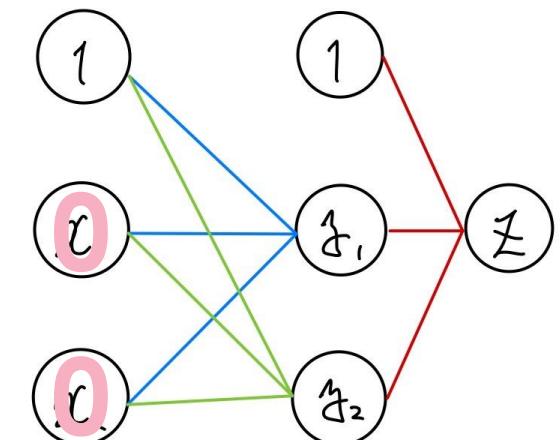


パーセプトロン-XOR確認②

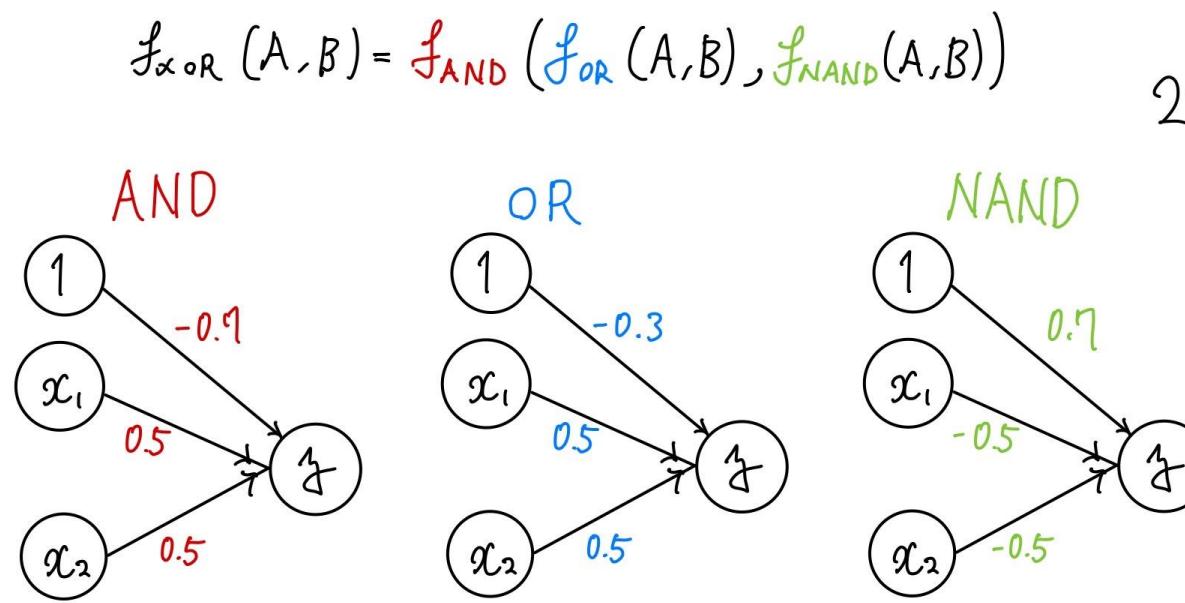
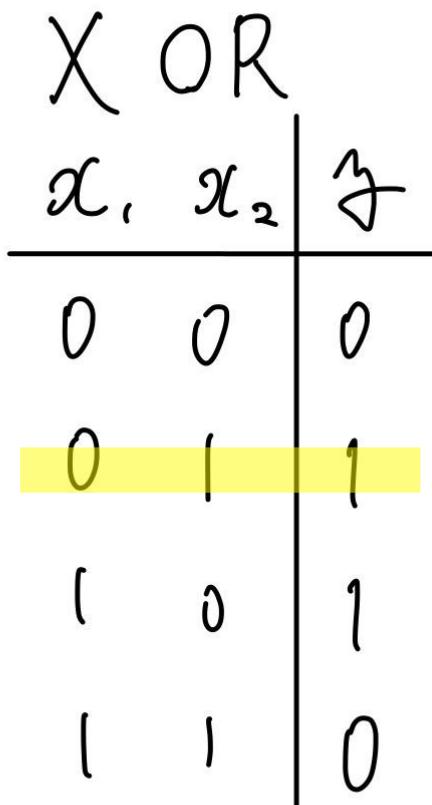


XORを構成する各関数が
 x_1, x_2 に関して対称なので
(1,0)と(0,1)の結果は等しい

2入力2(多)層パーセプトロン

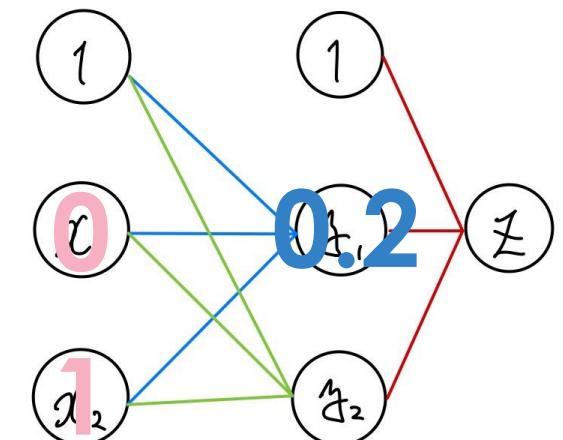


パーセプトロン-XOR確認②

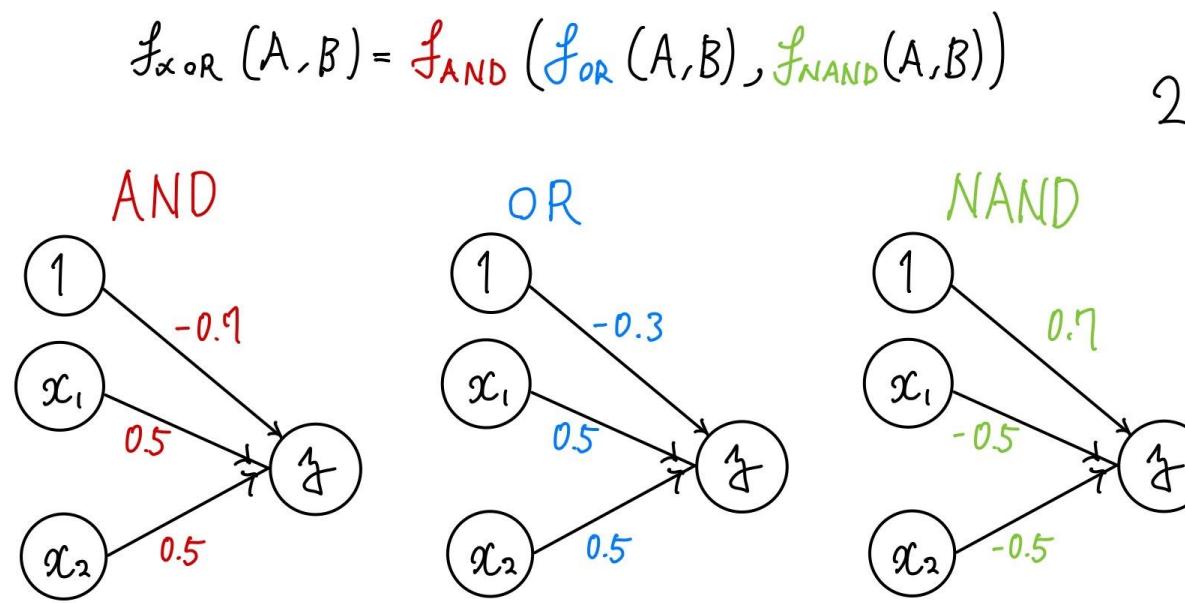
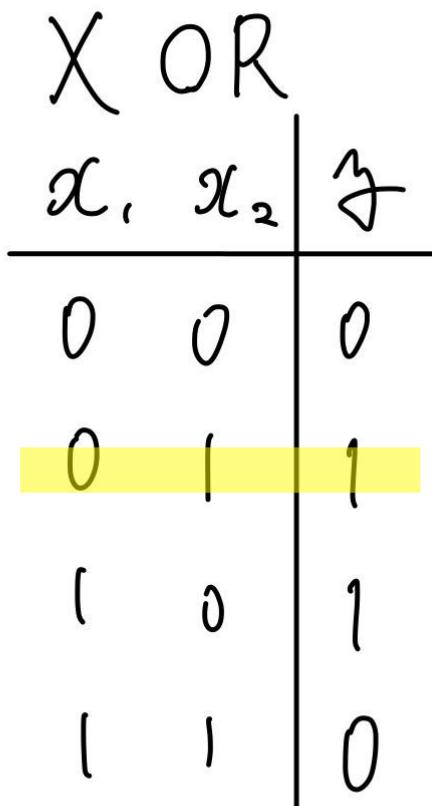


XORを構成する各関数が
 x_1, x_2 に関して対称なので
(1,0)と(0,1)の結果は等しい

2入力2(多)層パーセプトロン

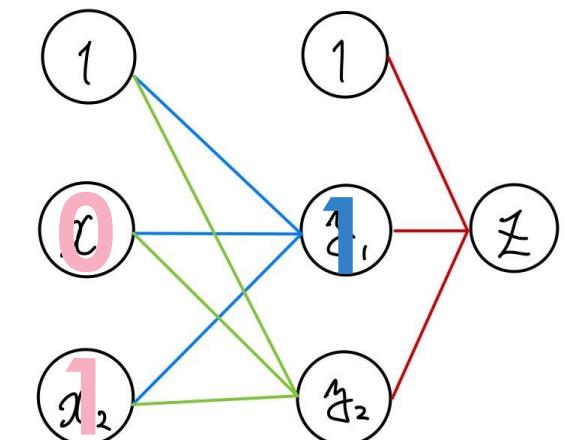


パーセプトロン-XOR確認②

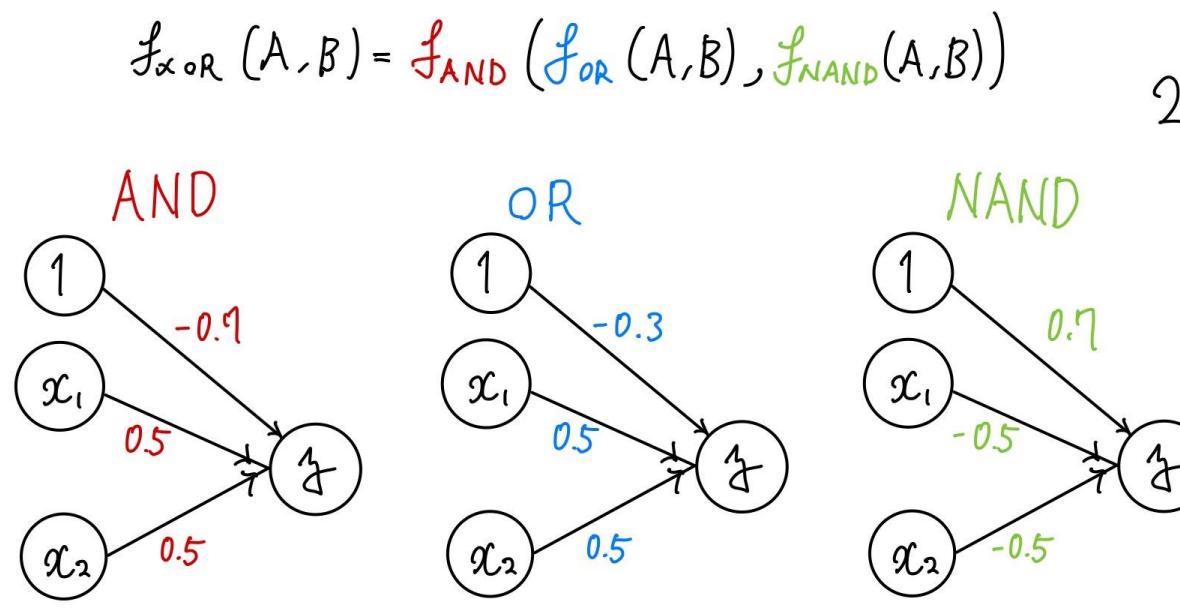
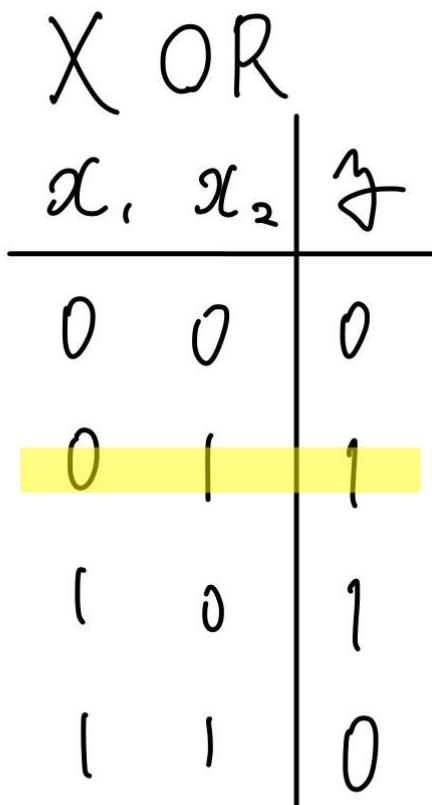


XORを構成する各関数が
 x_1, x_2 に関して対称なので
(1,0)と(0,1)の結果は等しい

2入力2(多)層パーセプトロン

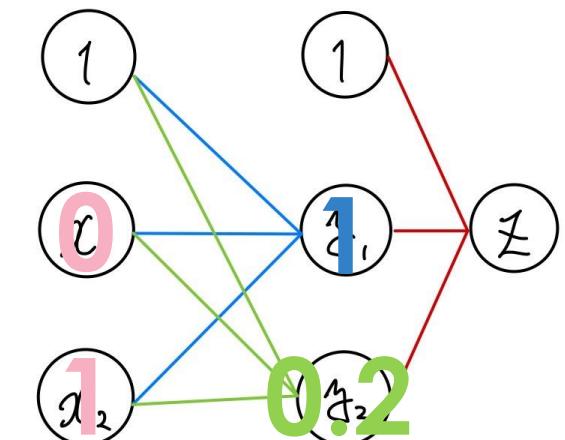


パーセプトロン-XOR確認②

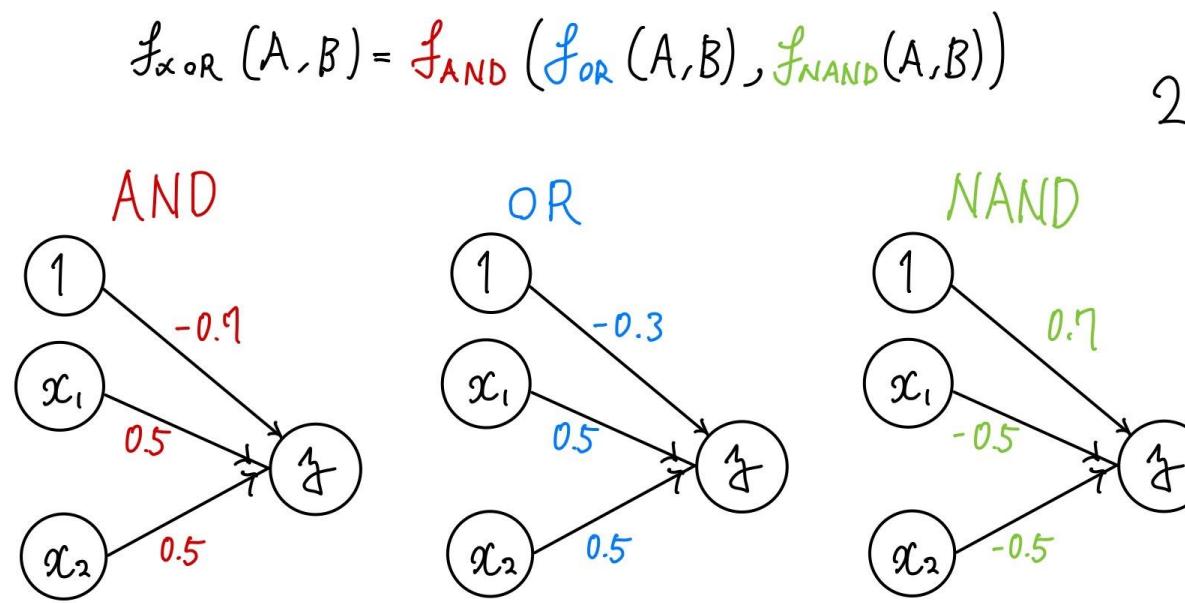
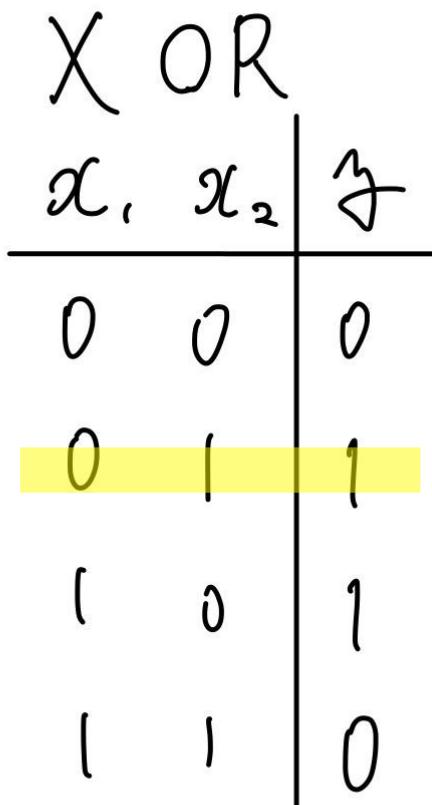


XORを構成する各関数が
x1,x2に関して対称なので
(1,0)と(0,1)の結果は等しい

2入力2(多)層パーセプトロン

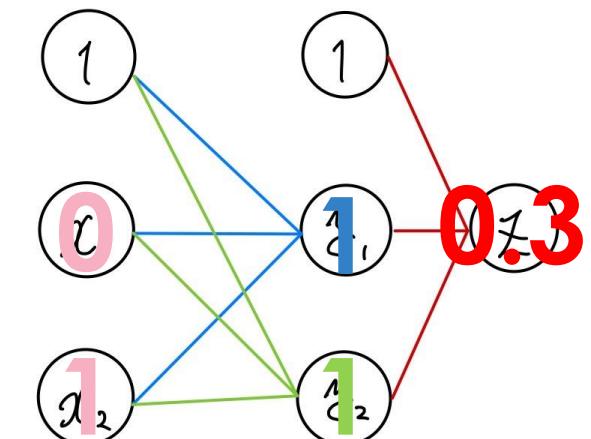


パーセプトロン-XOR確認②

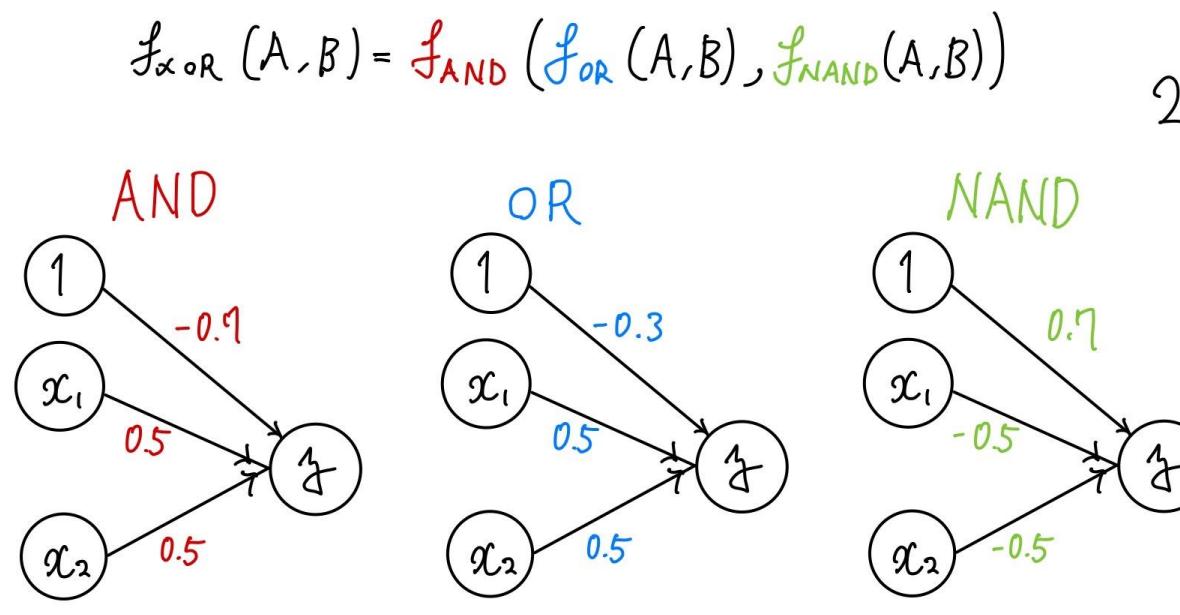
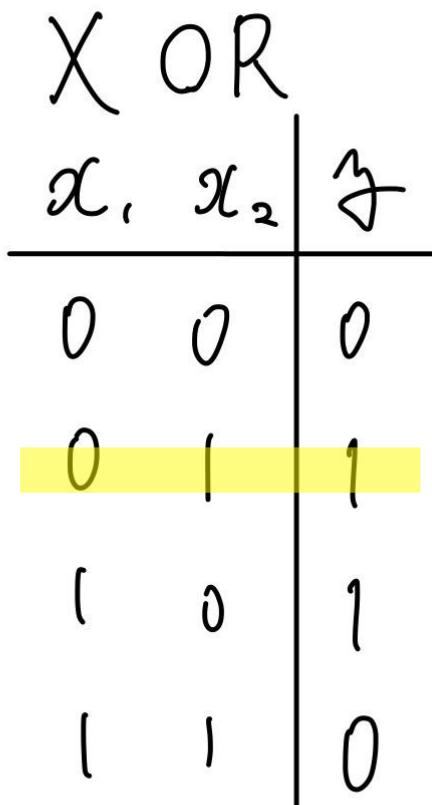


XORを構成する各関数が
 x_1, x_2 に関して対称なので
(1,0)と(0,1)の結果は等しい

2入力2(多)層パーセプトロン

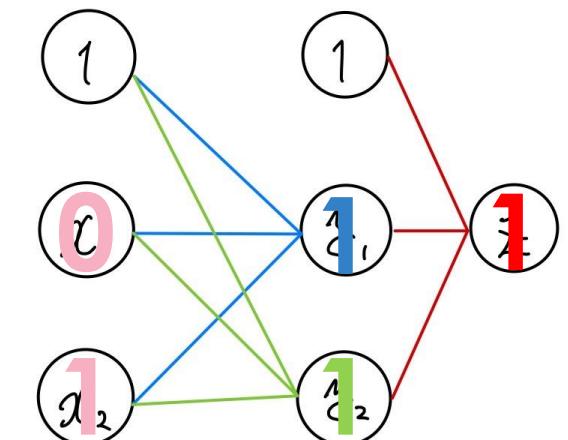


パーセプトロン-XOR確認②

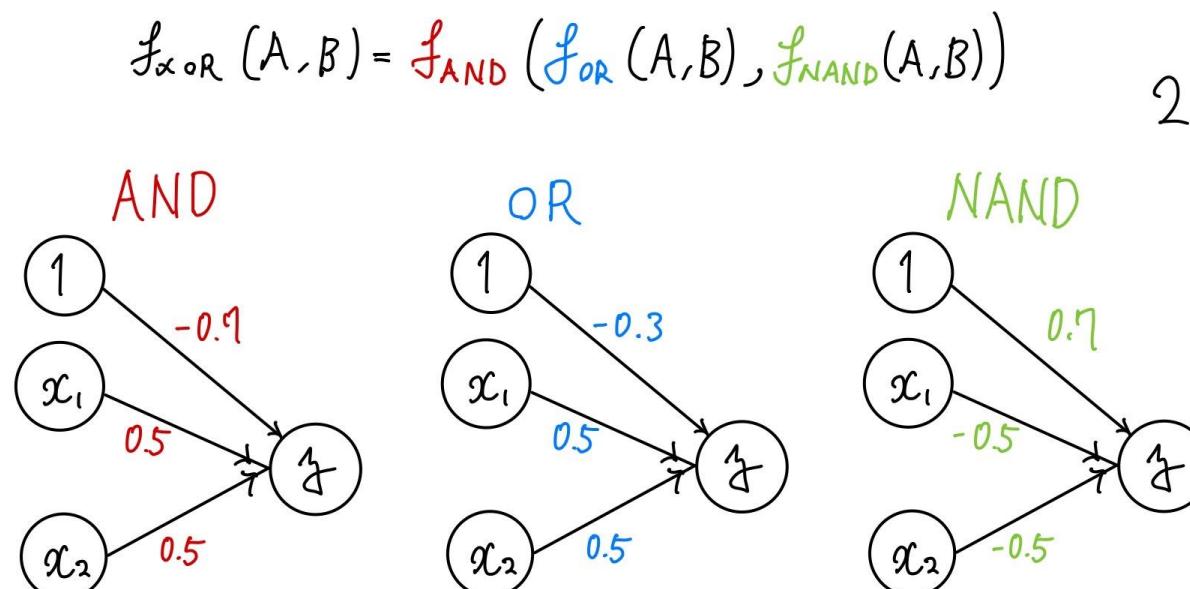
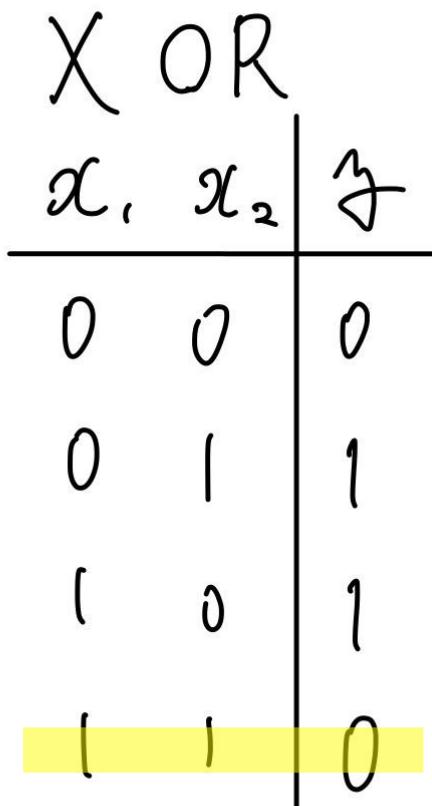


XORを構成する各関数が
 x_1, x_2 に関して対称なので
(1,0)と(0,1)の結果は等しい

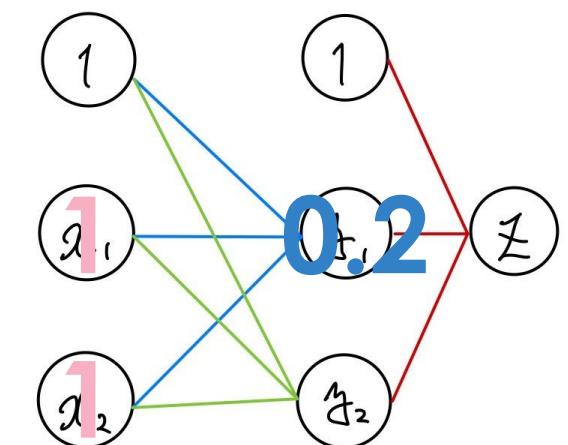
2入力2(多)層パーセプトロン



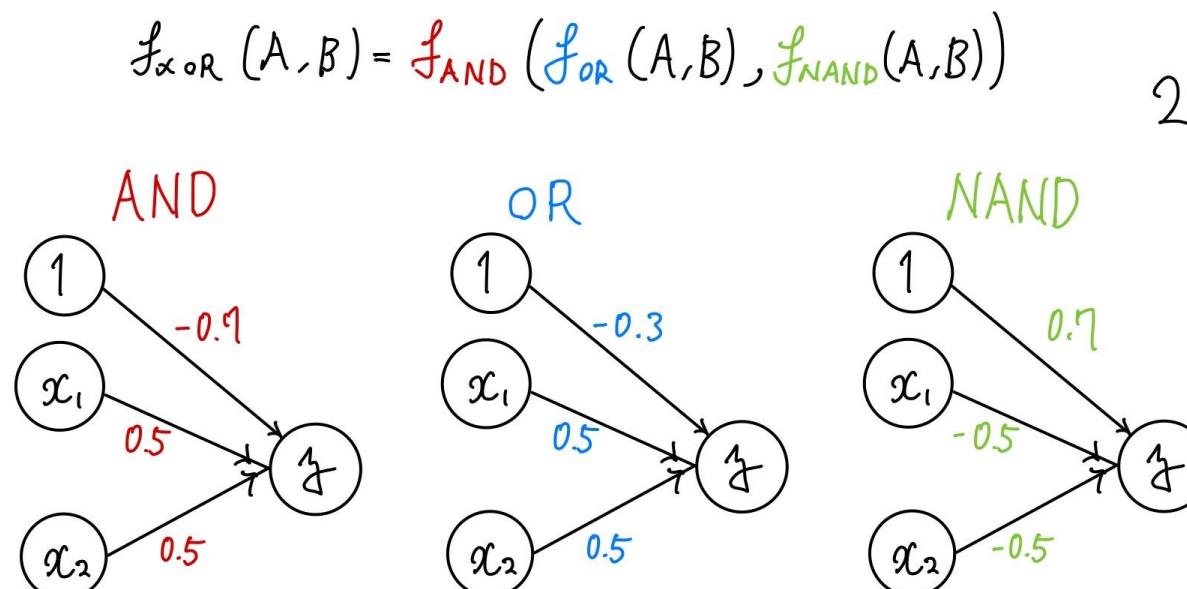
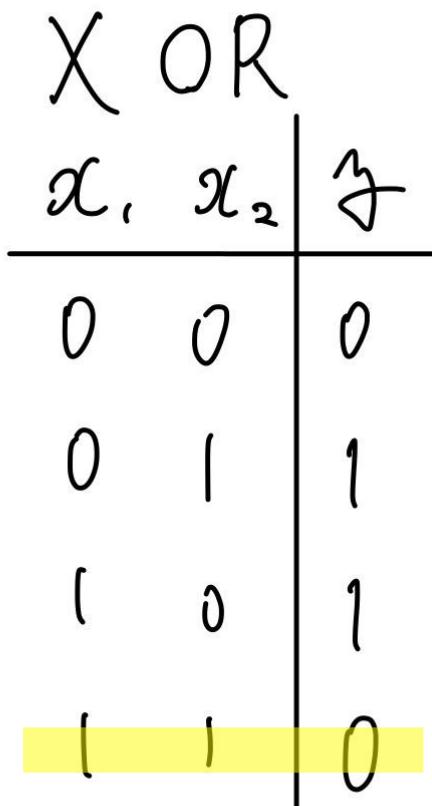
パーセプトロン-XOR確認③



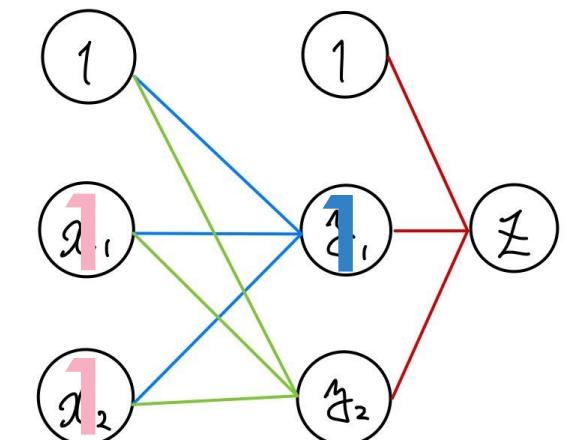
2入力2(多)層パーセプトロン



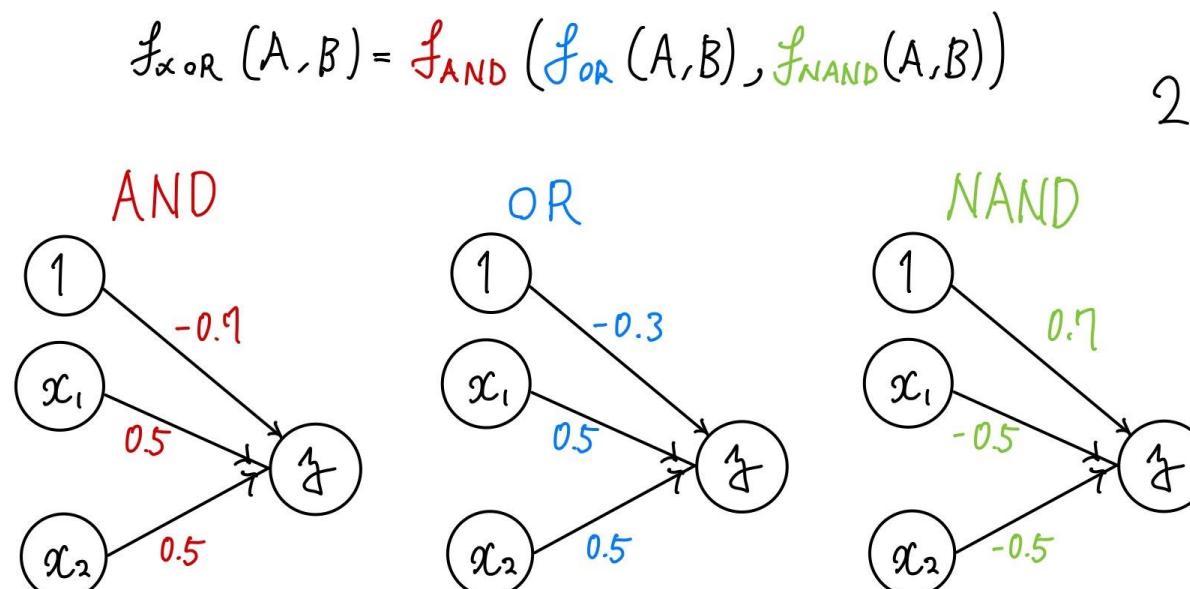
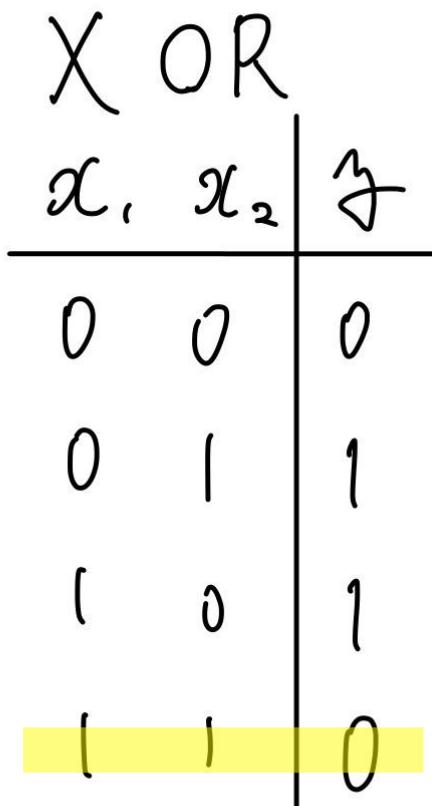
パーセプトロン-XOR確認③



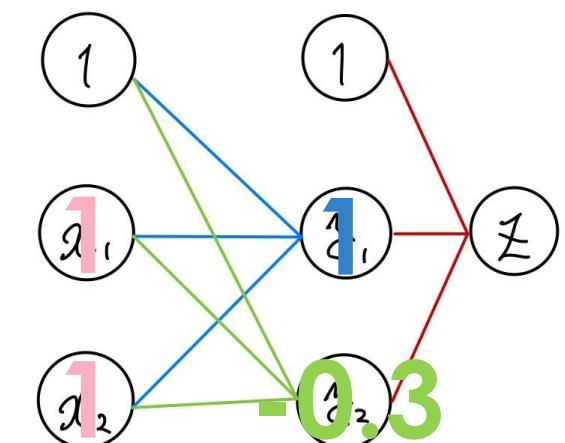
2入力2(多)層パーセプトロン



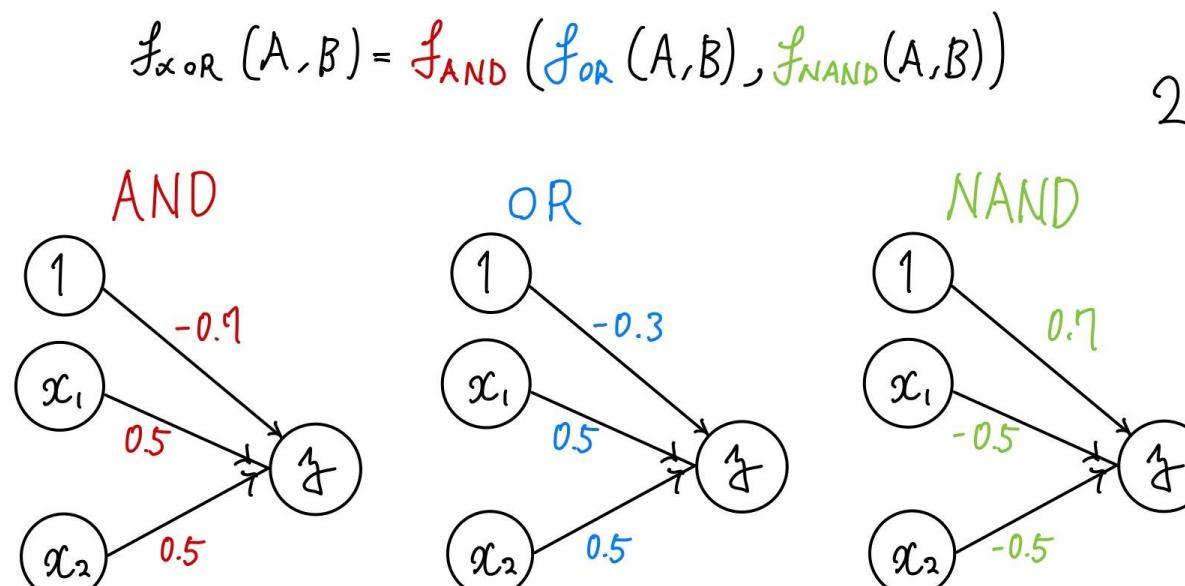
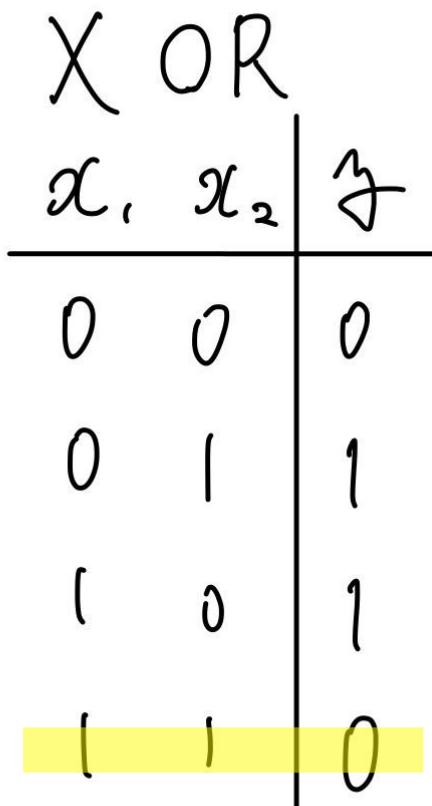
パーセプトロン-XOR確認③



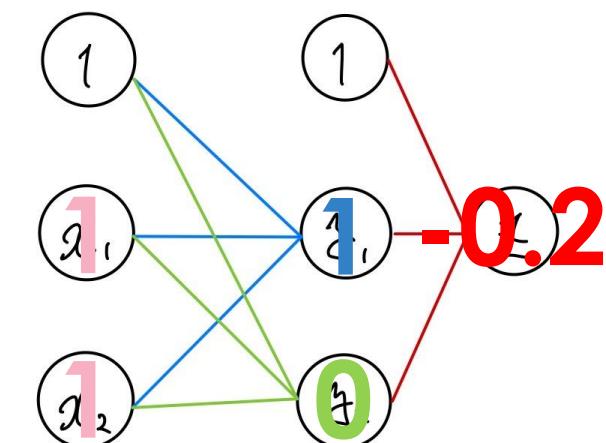
2入力2(多)層パーセプトロン



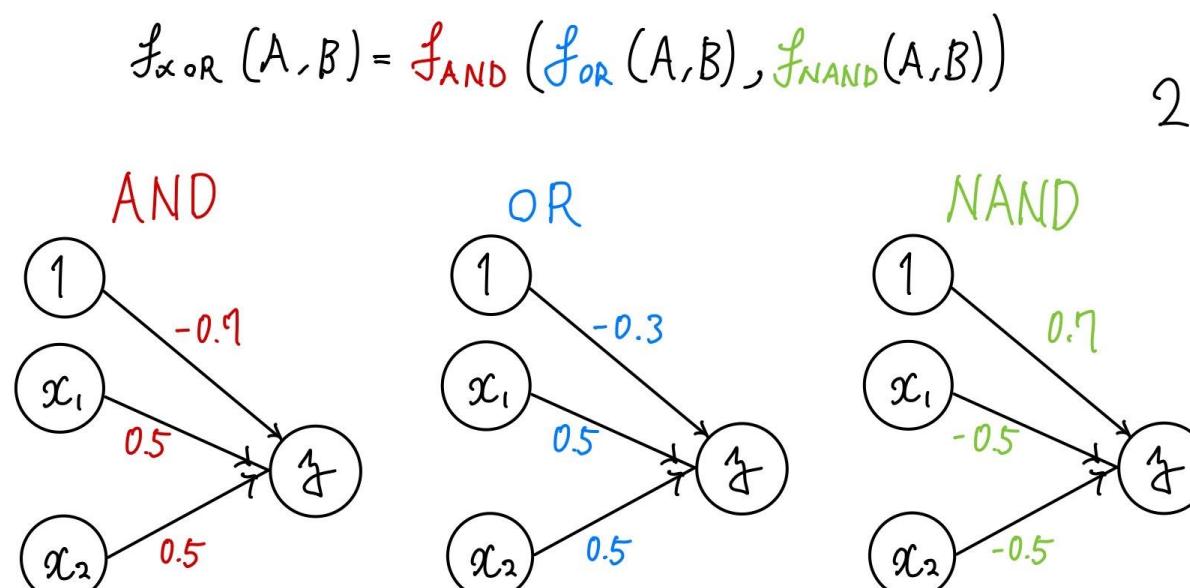
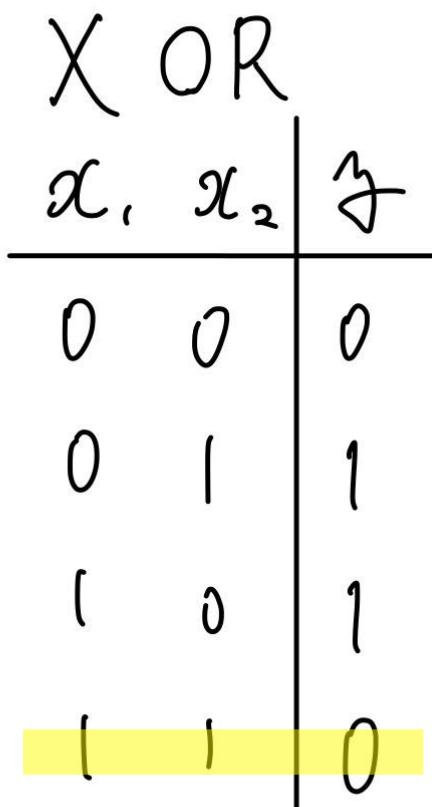
パーセプトロン-XOR確認③



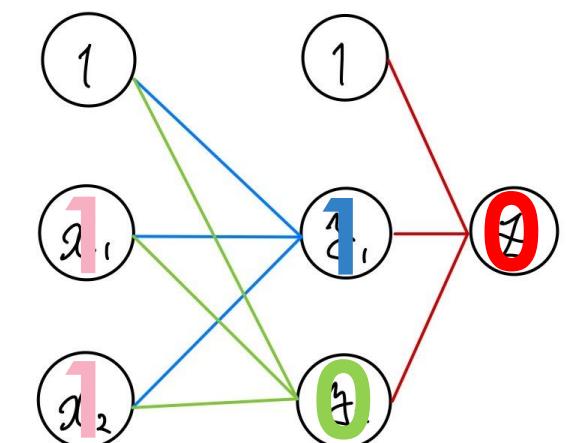
2入力2(多)層パーセプトロン



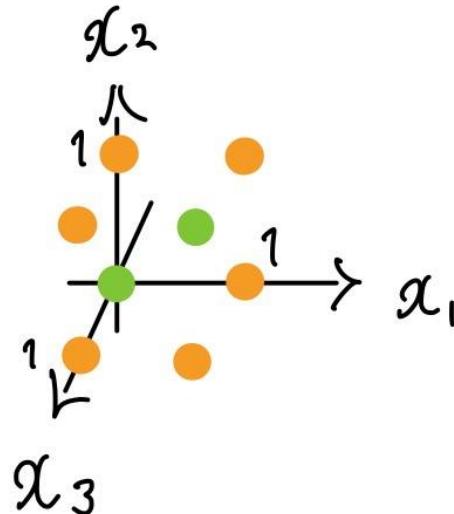
パーセプトロン-XOR確認③



2入力2(多)層パーセプトロン



パーセプトロン



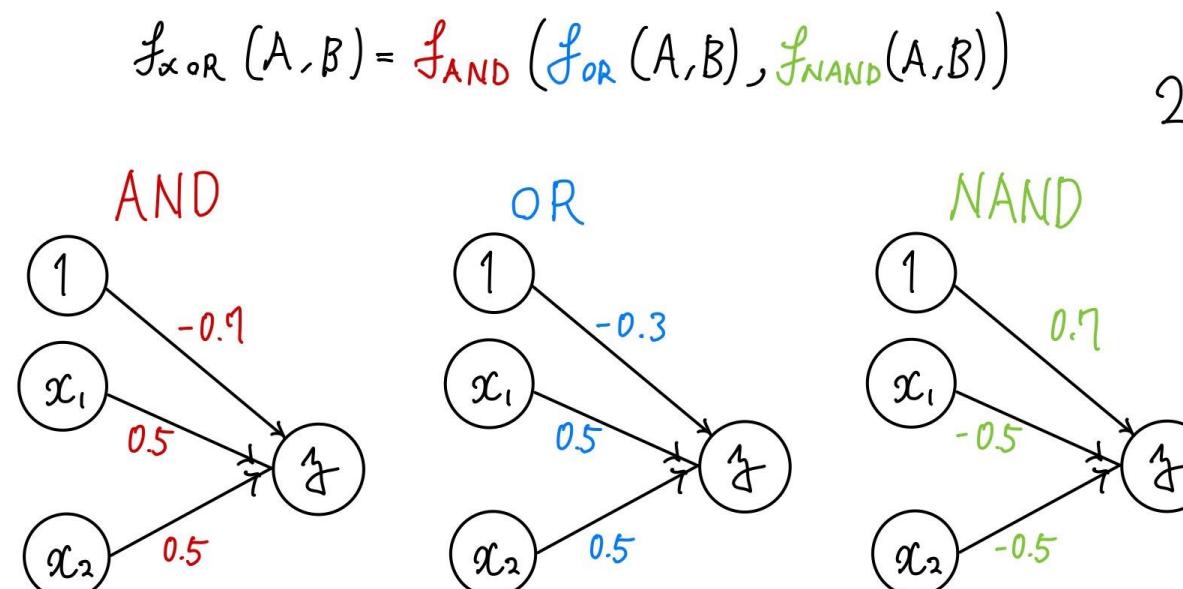
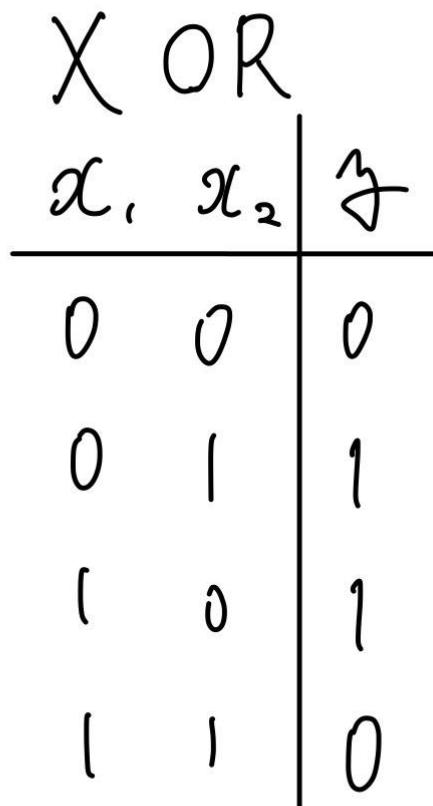
XOR ₃			y
x ₁	x ₂	x ₃	
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

問題

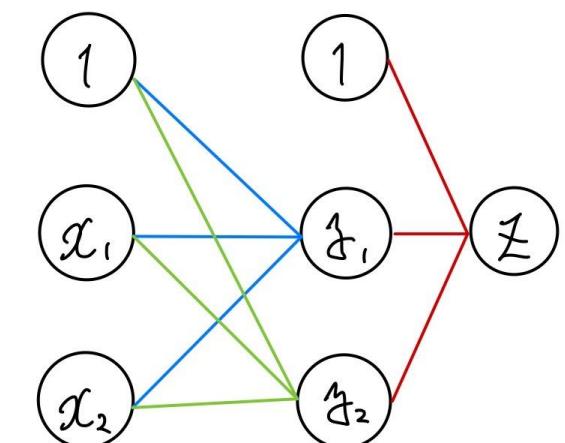
次の点群を
非線形分離する
多層パーセプトロンを設計せよ
(重みもバイアスも
層数もノード数も自由)

パーセプトロン-多層パーセプトロンでXORの設計

再掲

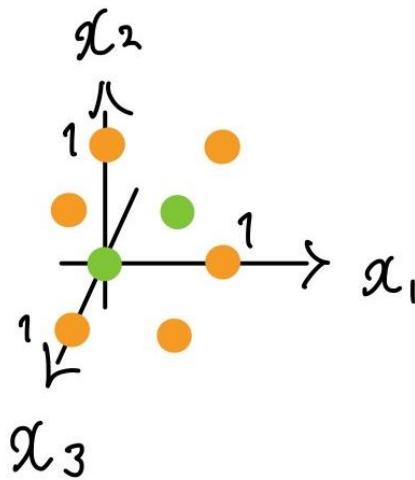


2入力2(多)層パーセプトロン



パーセプトロン-解答

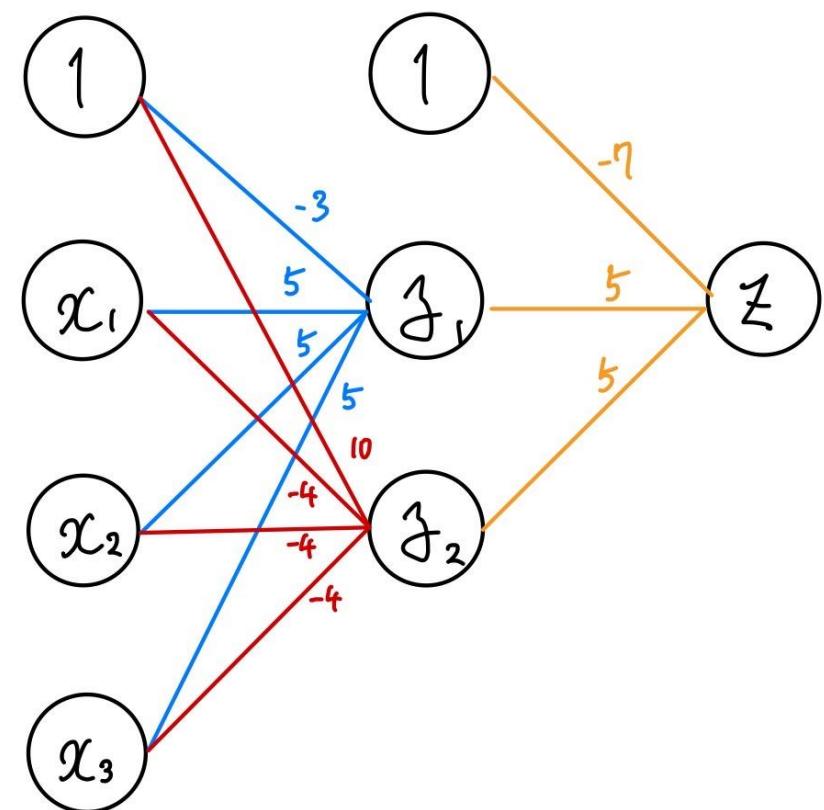
$$\begin{aligned}
 f &= \overline{\bar{x}_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3} \\
 &= \overline{x_1 + x_2 + x_3} + x_1 x_2 x_3 \\
 &= (x_1 + x_2 + x_3) \cdot \overline{x_1 x_2 x_3} \\
 &= f_{AND_2} (f_{OR_3}(x_1, x_2, x_3), f_{NAND_3}(x_1, x_2, x_3))
 \end{aligned}$$



XOR ₃			
x ₁	x ₂	x ₃	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

パーセプトロン

$$\begin{aligned}
 f &= \overline{\bar{x}_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3} \\
 &= \overline{x_1 + x_2 + x_3} + x_1 x_2 x_3 \\
 &= (x_1 + x_2 + x_3) \cdot \overline{x_1 x_2 x_3} \\
 &= f_{\text{AND}_2} (f_{\text{OR}_3}(x_1, x_2, x_3), f_{\text{NAND}_3}(x_1, x_2, x_3))
 \end{aligned}$$





パーセプトロン 完



ニューラルネットワーク 活性化関数-表記

ニューラルネットワーク-活性化関数

$$y = \begin{cases} 1, & w_1x_1 + w_2x_2 + b \cdot 1 > 0 \\ 0, & w_1x_1 + w_2x_2 + b \cdot 1 \leq 0 \end{cases}$$

は $\text{step}(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$ を用いて

$$y = \underbrace{\text{step}}_{\downarrow}(w_1x_1 + w_2x_2 + b \cdot 1)$$
 と書ける

活性化関数

ニューラルネットワーク-活性化関数

ペーセプトロンでは活性化関数に step 関数を使,
ANN では様々な活性化関数が使われ
ANN は活性化関数に 線形関数 を用いてはいけない
 \therefore 各層にしても 同値な $f(x) = cx$
1 層 ANN が存在するから

ニューラルネットワーク-活性化関数

· Step $f(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$

· シグモイド関数 $f(x) = \frac{1}{1 + e^{-x}}$

$$\lim_{x \rightarrow -\infty} f(x) = 0, \quad \lim_{x \rightarrow \infty} f(x) = 1$$

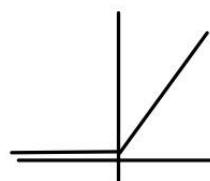
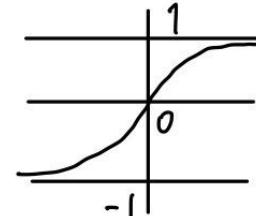
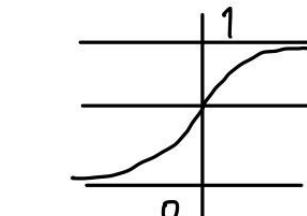
$\tanh(x) = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

$$\lim_{x \rightarrow -\infty} f(x) = -1, \quad \lim_{x \rightarrow \infty} f(x) = 1$$

· ReLU 関数

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

主にこれを使っていく



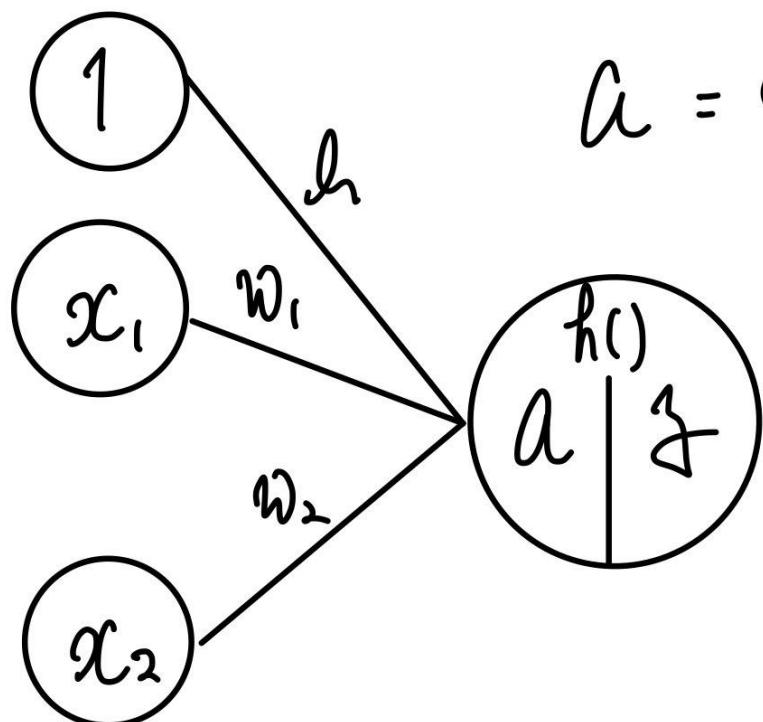
最近の DNN では
勾配消失問題

によりあまり使われない

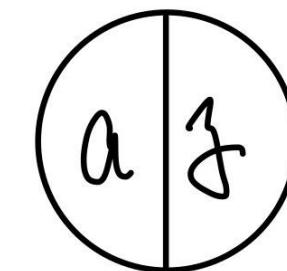


これを解決

ニューラルネットワーク-活性化関数 表記の導入

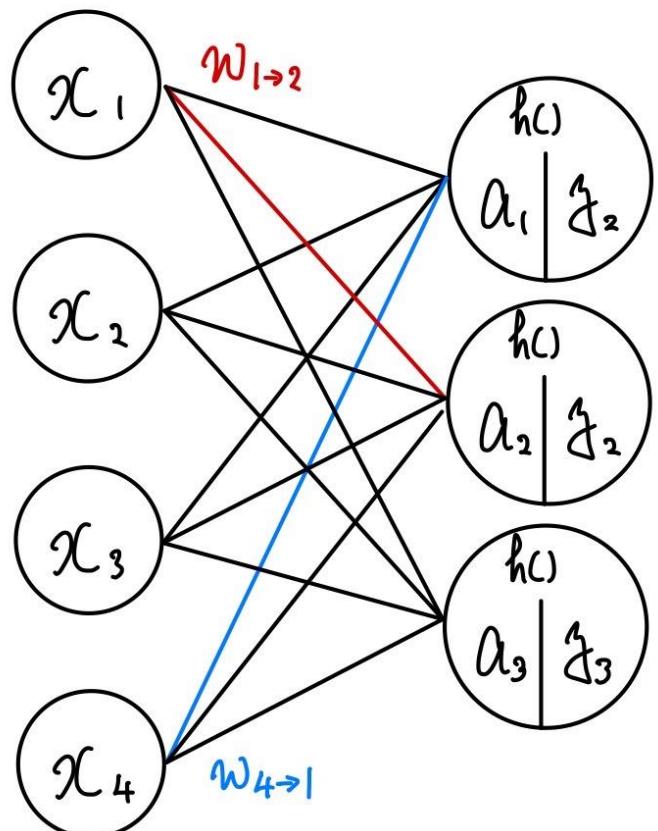


$$z = h(a)$$



$h()$ を表記上省略

ニューラルネットワーク-表記の導入



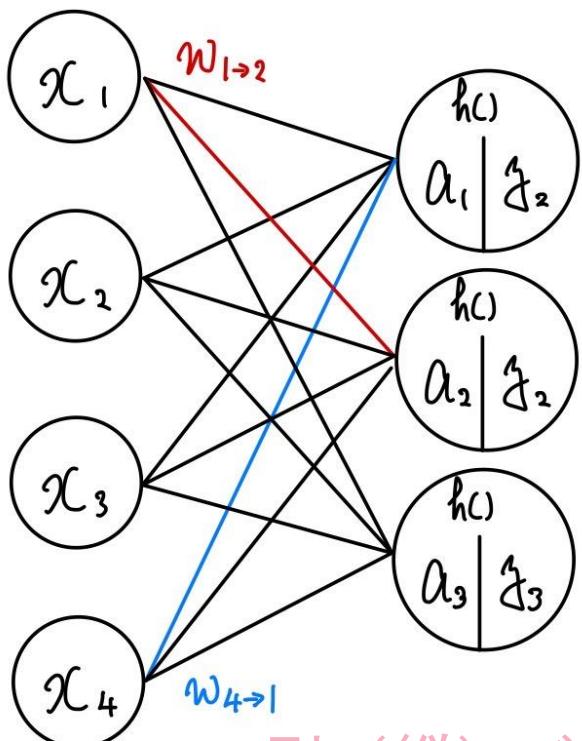
w_{ij} で i 個目のノードから
 j 個目のノードへの重みを表す
イメージ : $w_{i \rightarrow j}$

$$a_1 = w_{11} x_1 + w_{21} x_2 + w_{31} x_3 + w_{41} x_4 + b_1, \quad f_1 = h(a_1)$$

$$a_2 = w_{12} x_1 + w_{22} x_2 + w_{32} x_3 + w_{42} x_4 + b_2, \quad f_2 = h(a_2)$$

$$a_3 = w_{13} x_1 + w_{23} x_2 + w_{33} x_3 + w_{43} x_4 + b_3, \quad f_3 = h(a_3)$$

ニューラルネットワーク-表記の導入



列（縦）ベクトルでなく
行（横）ベクトルなことに注意
そのありがたみは後述

$$a_1 = w_{11} x_1 + w_{21} x_2 + w_{31} x_3 + w_{41} x_4 + b_1, \quad f_1 = h(a_1)$$

$$a_2 = w_{12} x_1 + w_{22} x_2 + w_{32} x_3 + w_{42} x_4 + b_2, \quad f_2 = h(a_2)$$

$$a_3 = w_{13} x_1 + w_{23} x_2 + w_{33} x_3 + w_{43} x_4 + b_3, \quad f_3 = h(a_3)$$

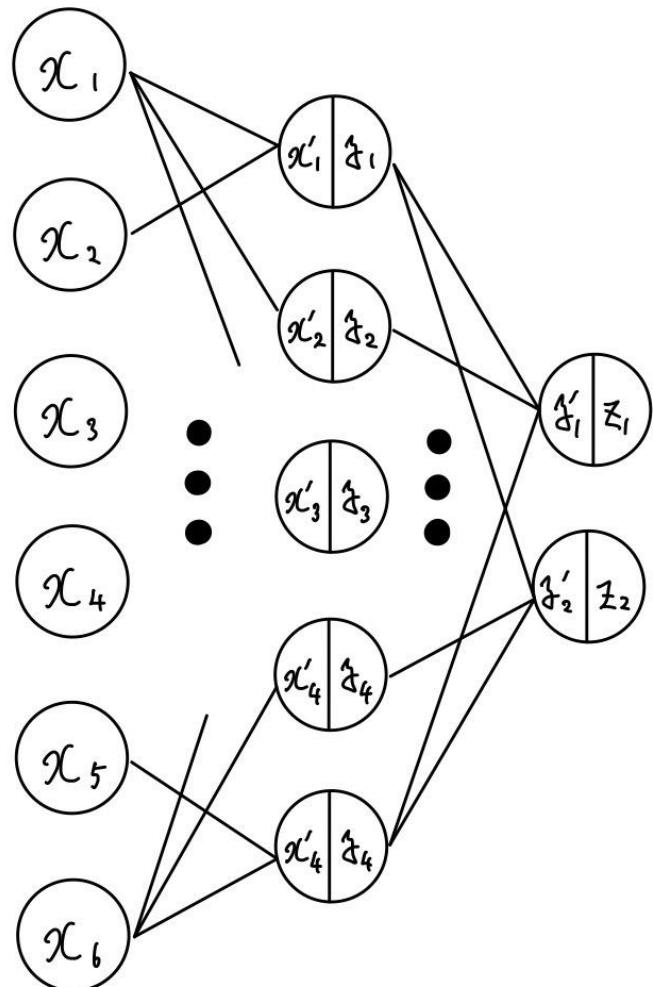
添え字がいい感じに並んでるのも利点

$$(a_1, a_2, a_3) = (x_1, x_2, x_3, x_4) \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix} + (b_1, b_2, b_3)$$

サイズ (1×3) (1×4) (4×3) (1×3)

$$A = XW + B$$

ニューラルネットワーク-データの流れ確認問題

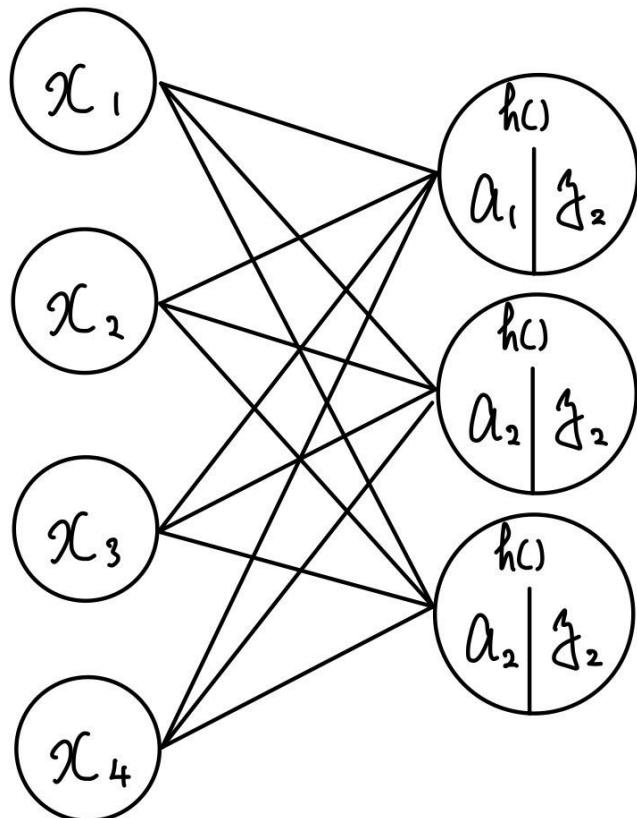


$$X' = X W_1 + B_1, Y = h(X')$$
$$Y' = Y W_2 + B_2, Z = h(Y')$$

以下の行列の形状を求めよ。

$$W_1, B_1, X' Y W_2 B_2 Y' Z$$

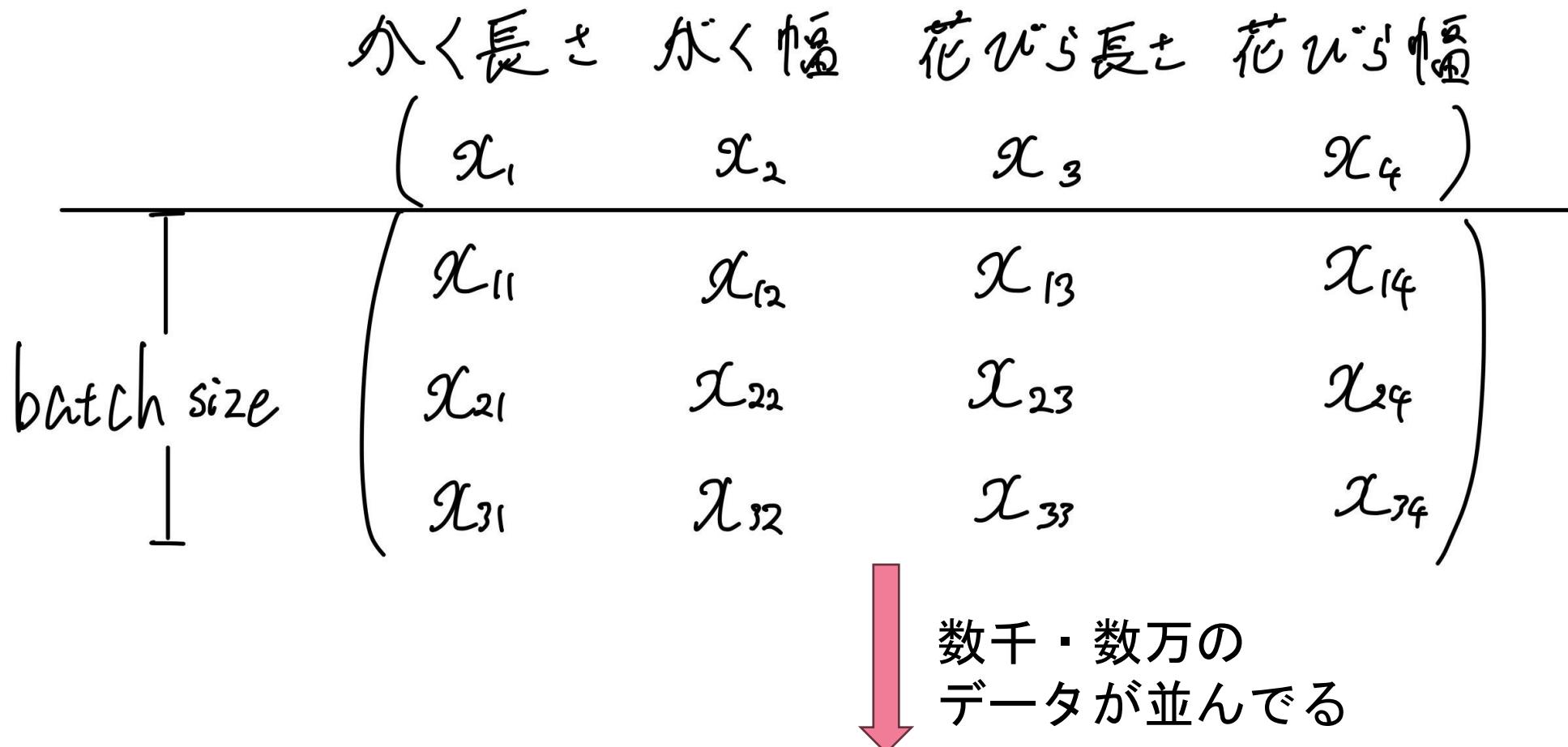
ニューラルネットワーク-行ベクトルの理由



$$(a_1 \ a_2 \ a_3) = (x_1 \ x_2 \ x_3 \ x_4) \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix} + (h_1 \ h_2 \ h_3)$$

1つのデータ組に対して計算して
る
実際コーディングするときは一気に複数の
データを計算して順伝播させたい

ニューラルネットワーク-行ベクトルの理由



ニューラルネットワーク-行ベクトルの理由

$$\begin{pmatrix}
 x_{11} & x_{12} & x_{13} \\
 x_{21} & x_{22} & x_{23} \\
 x_{31} & x_{32} & x_{33}
 \end{pmatrix}
 \begin{pmatrix}
 x_{14} \\
 x_{24} \\
 x_{34}
 \end{pmatrix}
 \begin{pmatrix}
 w_{11} & w_{12} & w_{13} \\
 w_{21} & w_{22} & w_{23} \\
 w_{31} & w_{32} & w_{33} \\
 w_{41} & w_{42} & w_{43}
 \end{pmatrix}
 +
 \begin{pmatrix}
 h_1 & h_2 & h_3 \\
 h_1 & h_2 & h_3 \\
 h_1 & h_2 & h_3
 \end{pmatrix}
 \Bigg|$$

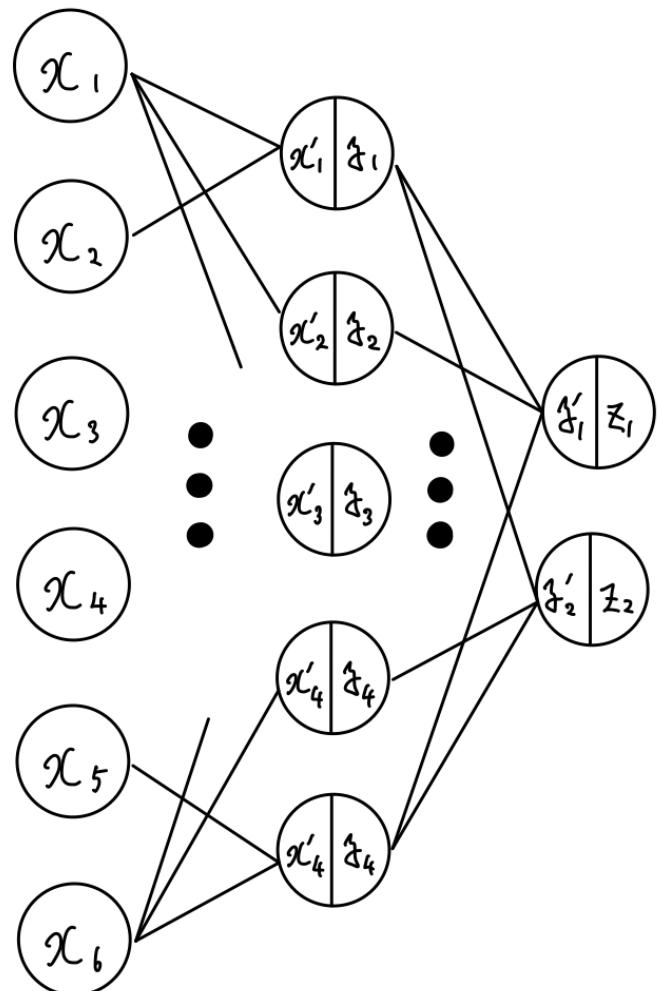
バッチサイズ
N=3

(N×3) (N×4) (4×3) (N×3)

バッチサイズ分
同じものをコピーする

$$A = X W + B$$

ニューラルネットワーク-データの流れ確認問題



$$X' = X W_1 + B_1, \quad Y = h(X')$$
$$Y' = Y W_2 + B_2, \quad Z = h(Y')$$

以下の行列の形状を求めよ。

$$W_1, B_1, X' \quad Y \quad W_2, B_2, Y' \quad Z$$



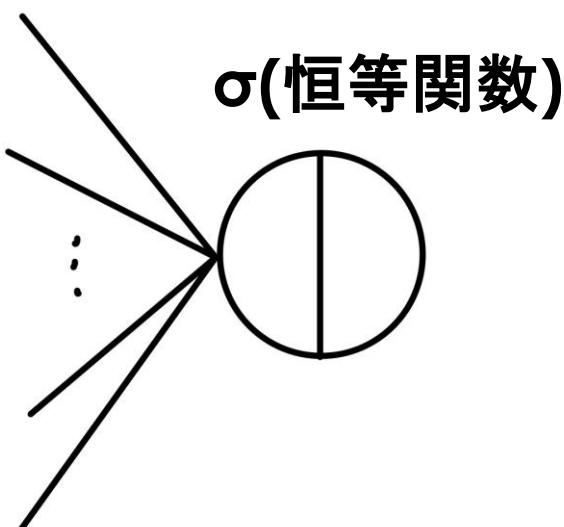
ニューラルネットワーク 活性化関数-表記 完



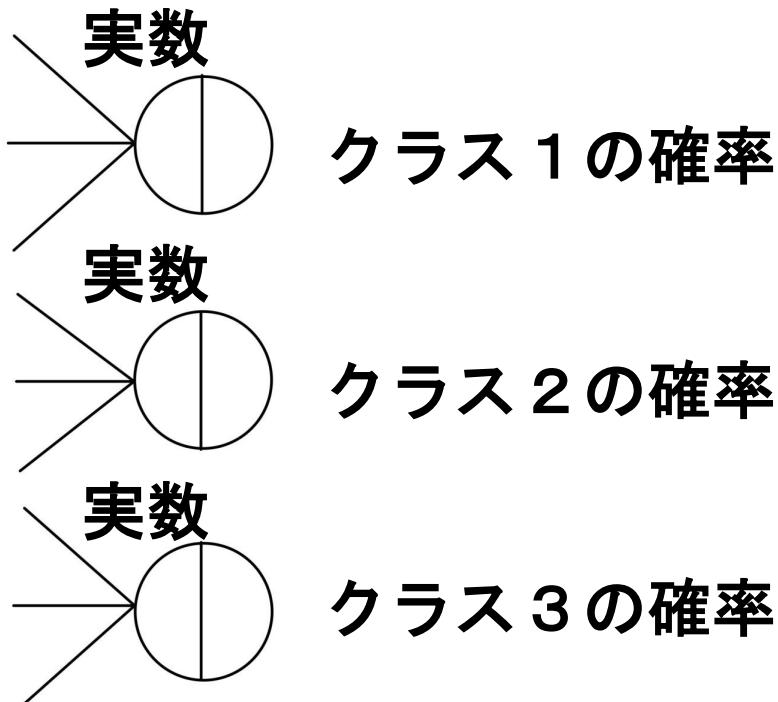
ニューラルネットワーク 学習

ニューラルネットワーク-回帰と分類

回帰
数値の予測

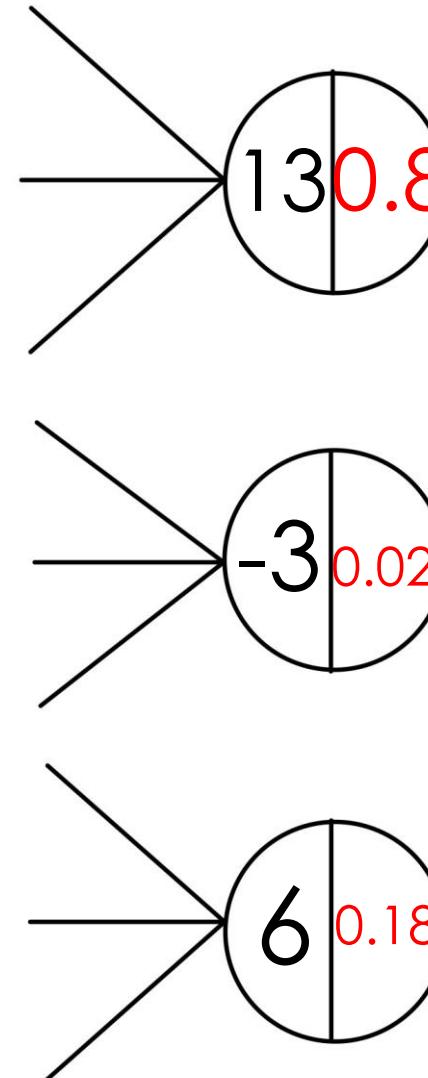
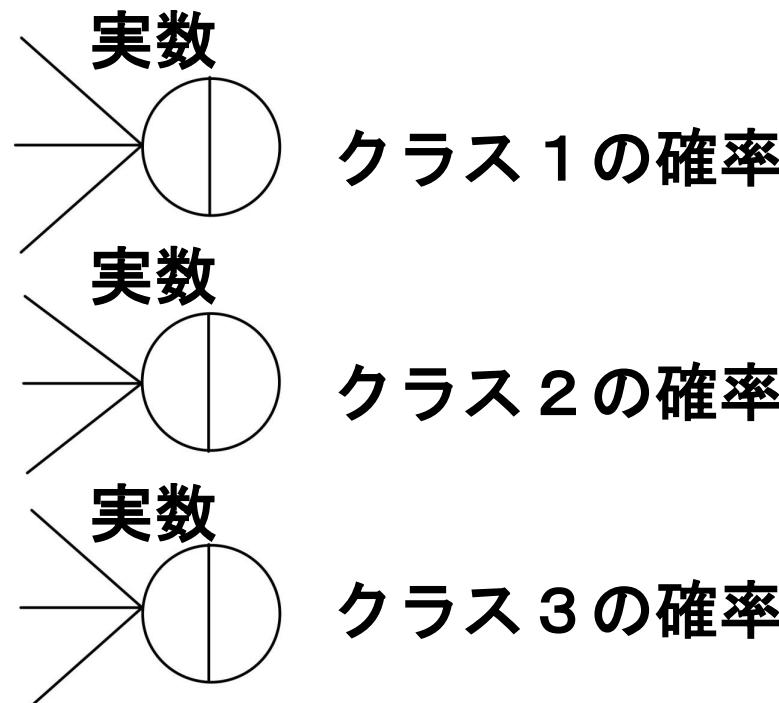


(nクラス)分類
どのクラスに含まれるか

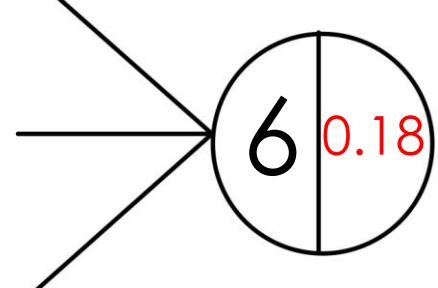
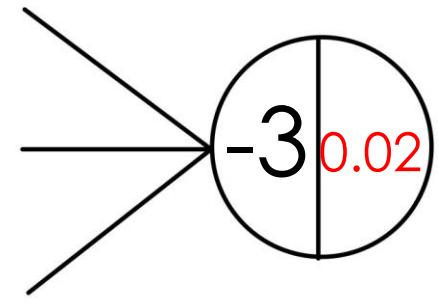
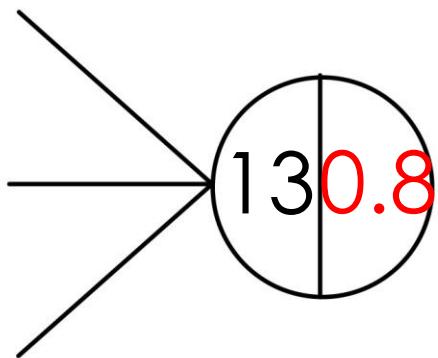


ニューラルネットワーク-分類 出力層の活性化関数

(nクラス)分類
どのクラスに含まれるか



ニューラルネットワーク-softmax



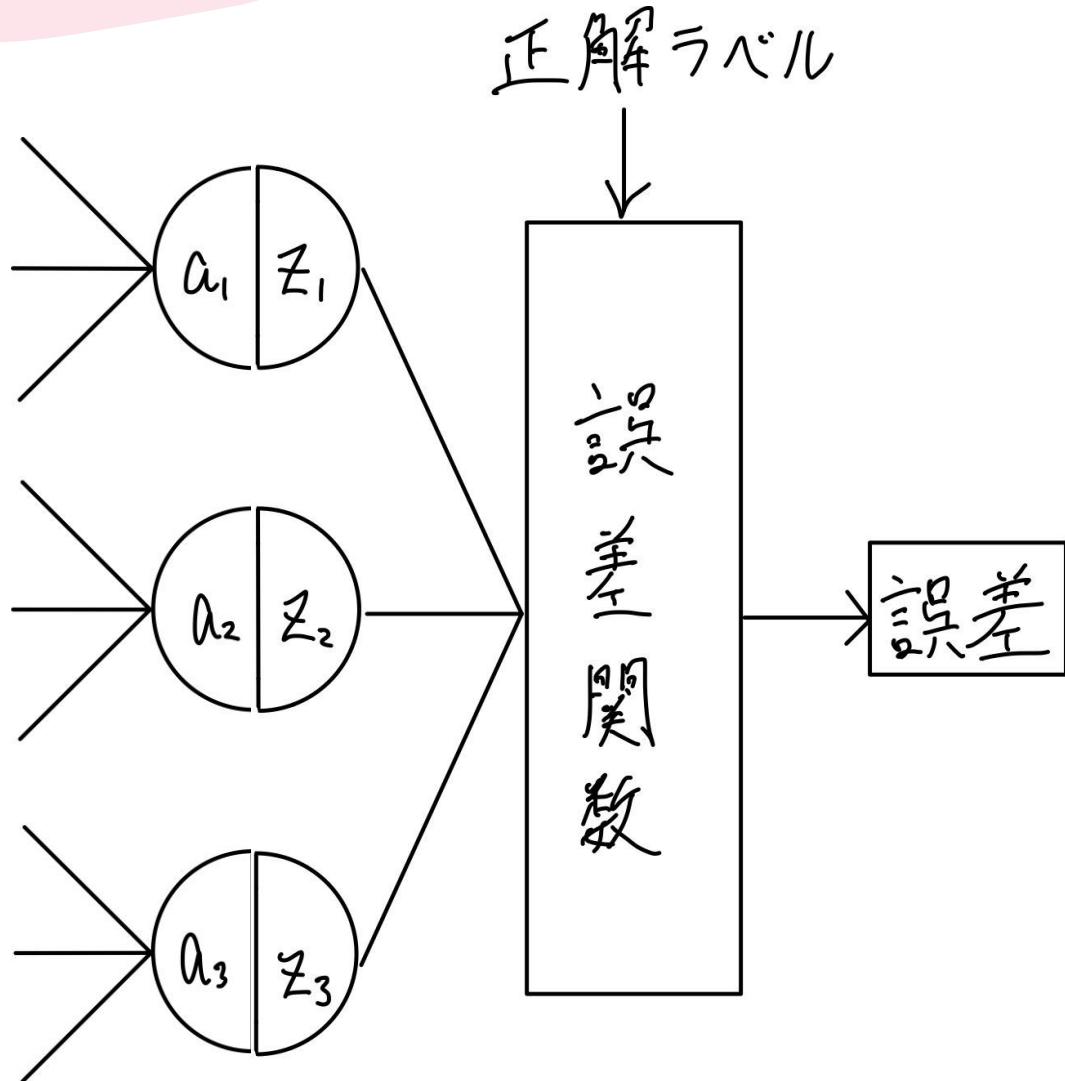
ソフトマックス関数

$$z_k = \frac{e^{a_k}}{\sum_{i=1}^n e^{a_i}}$$

たしかに

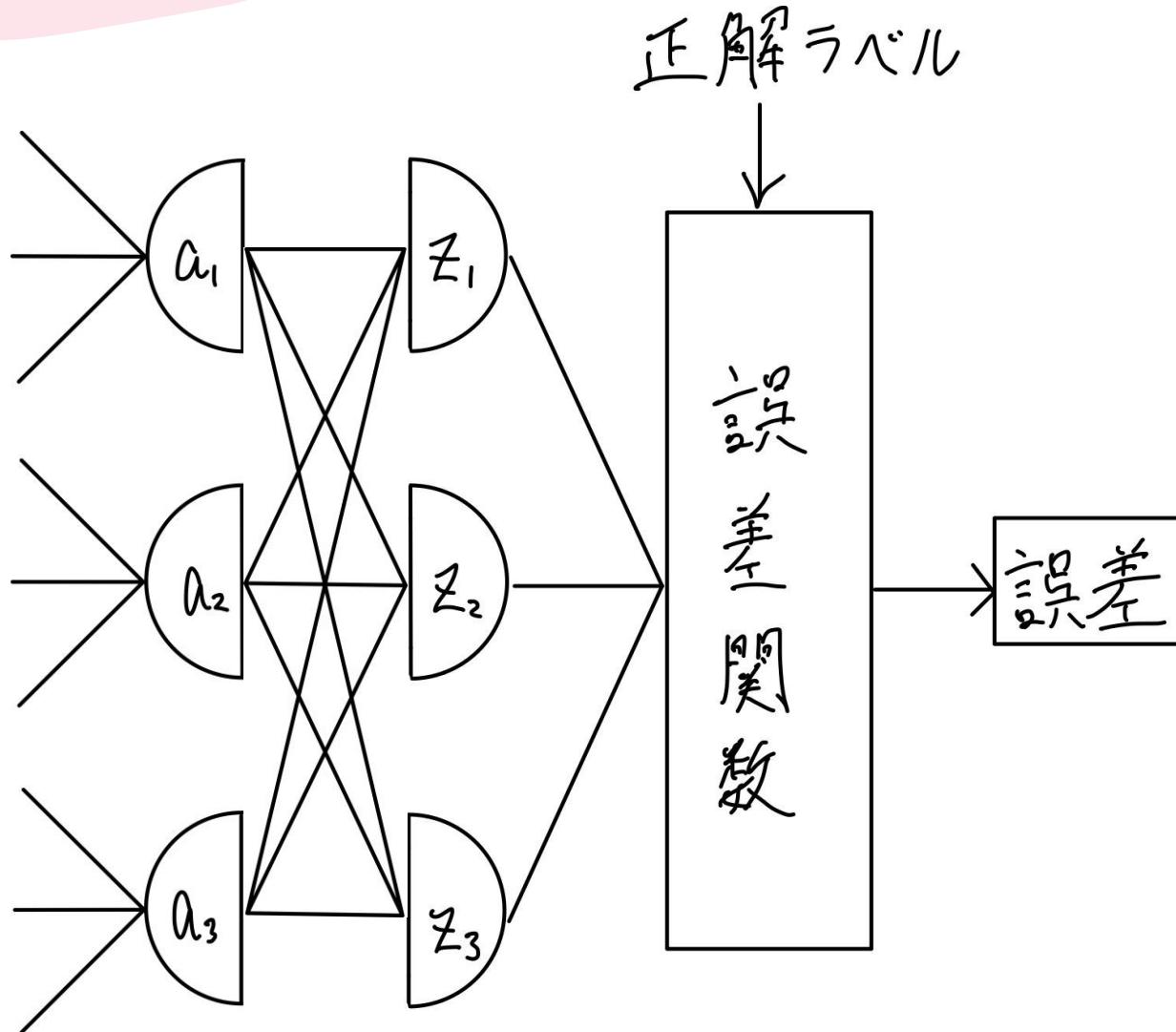
$$\sum z_k = 1$$

ニューラルネットワーク-softmax



$$z_k = \frac{e^{a_k}}{\sum_{i=1}^n e^{a_i}}$$

ニューラルネットワーク-softmax



$$z_k = \frac{e^{a_k}}{\sum_{j=1}^n e^{a_j}}$$

ニューラルネットワーク-softmax 実装上の注意

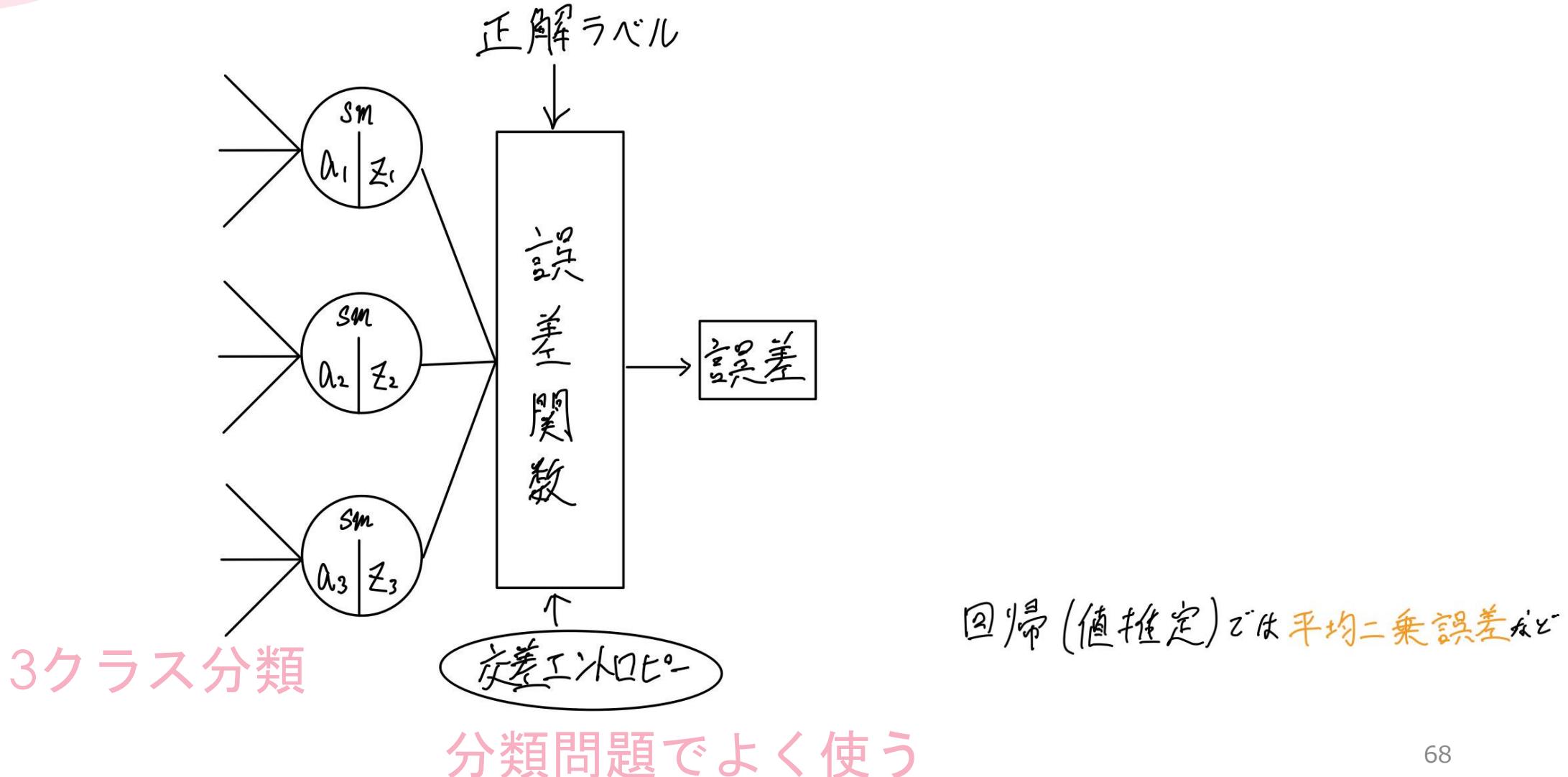
e^x は大きく誤差がうまる,
double も表現できない

$$z_k = \frac{e^{a_i}}{\sum_{j=1}^n e^{a_j}} = \frac{C e^{a_i}}{C \sum_{j=1}^n e^{a_j}}$$

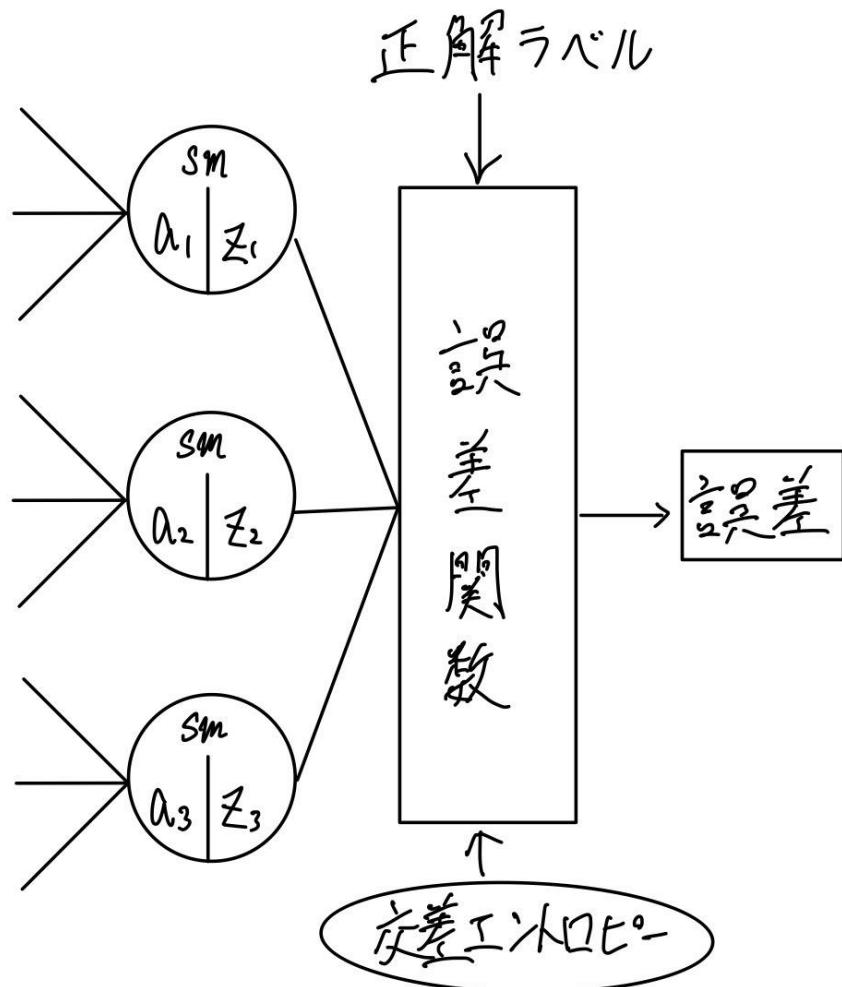
$$= \frac{e^{a_i + \ln C}}{\sum_{j=1}^n e^{a_j + \ln C}} = \frac{e^{a_i + C'}}{\sum_{j=1}^n e^{a_j + C'}}$$

$$\therefore C' = -\max[a_i] \approx 3$$

ニューラルネットワーク-誤差関数



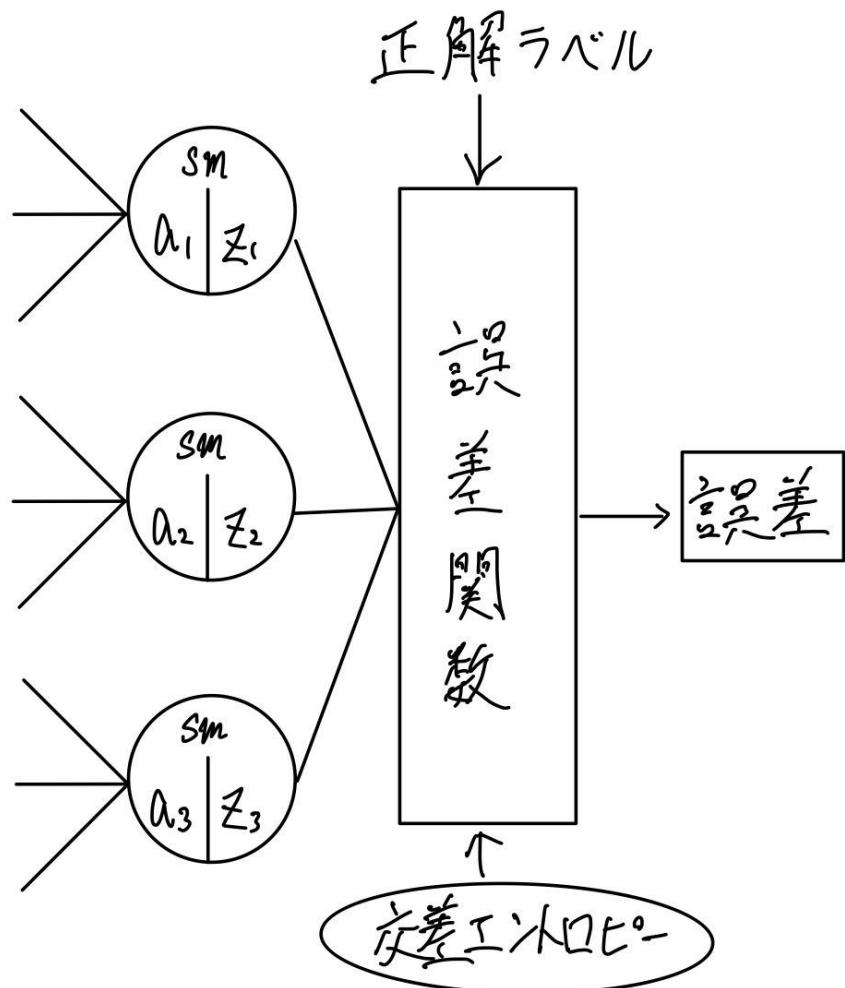
ニューラルネットワーク-交差エントロピー



正解確率 推定確率

$$E = - \sum_{k=1}^K y_k \ln \hat{y}_k$$

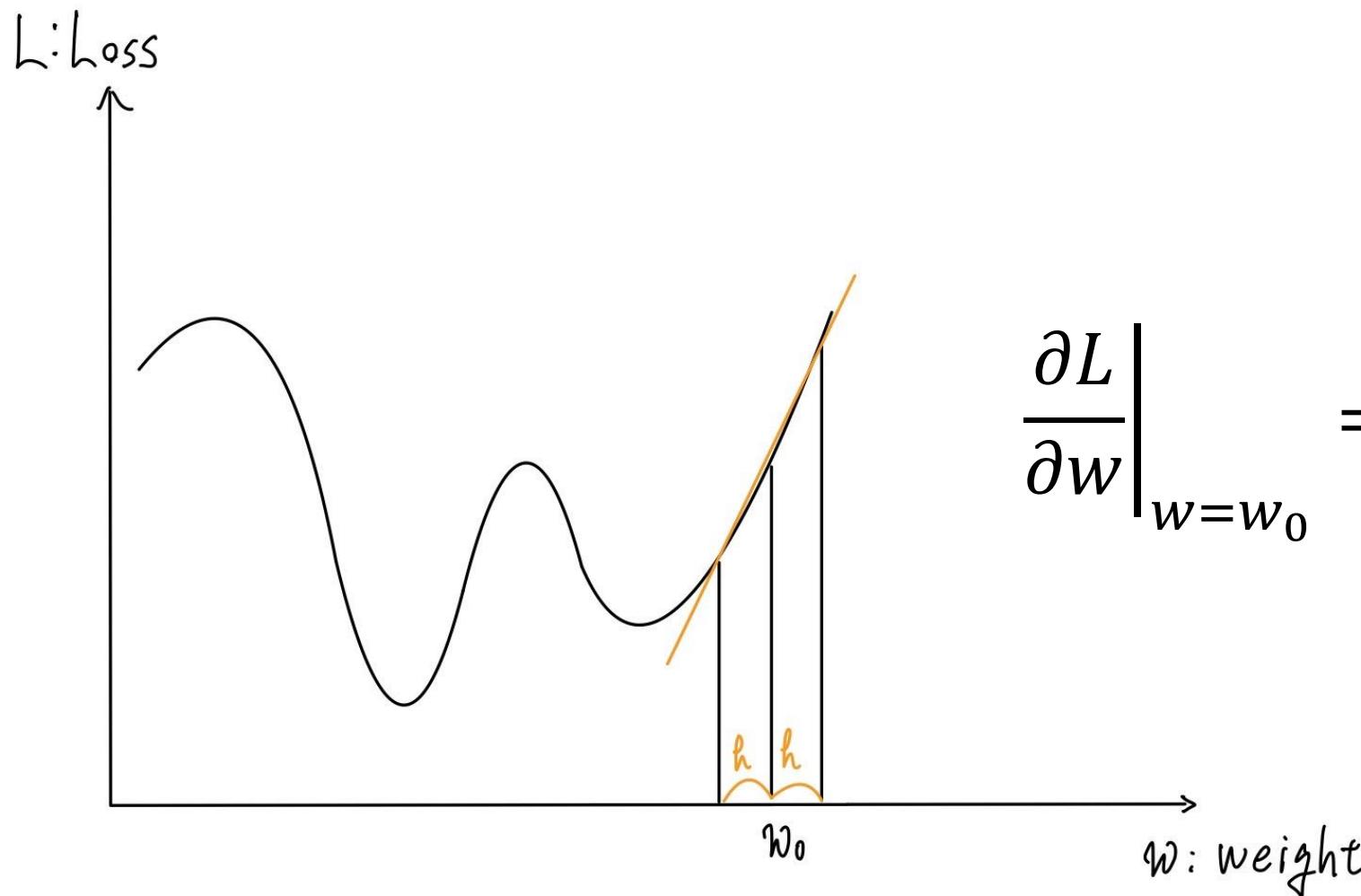
ニューラルネットワーク-交差エントロピー- batch ver



$$Z = \begin{pmatrix} z_{11} & z_{12} & z_{13} \\ \vdots & & \end{pmatrix} \quad T = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ \vdots & & \end{pmatrix}$$

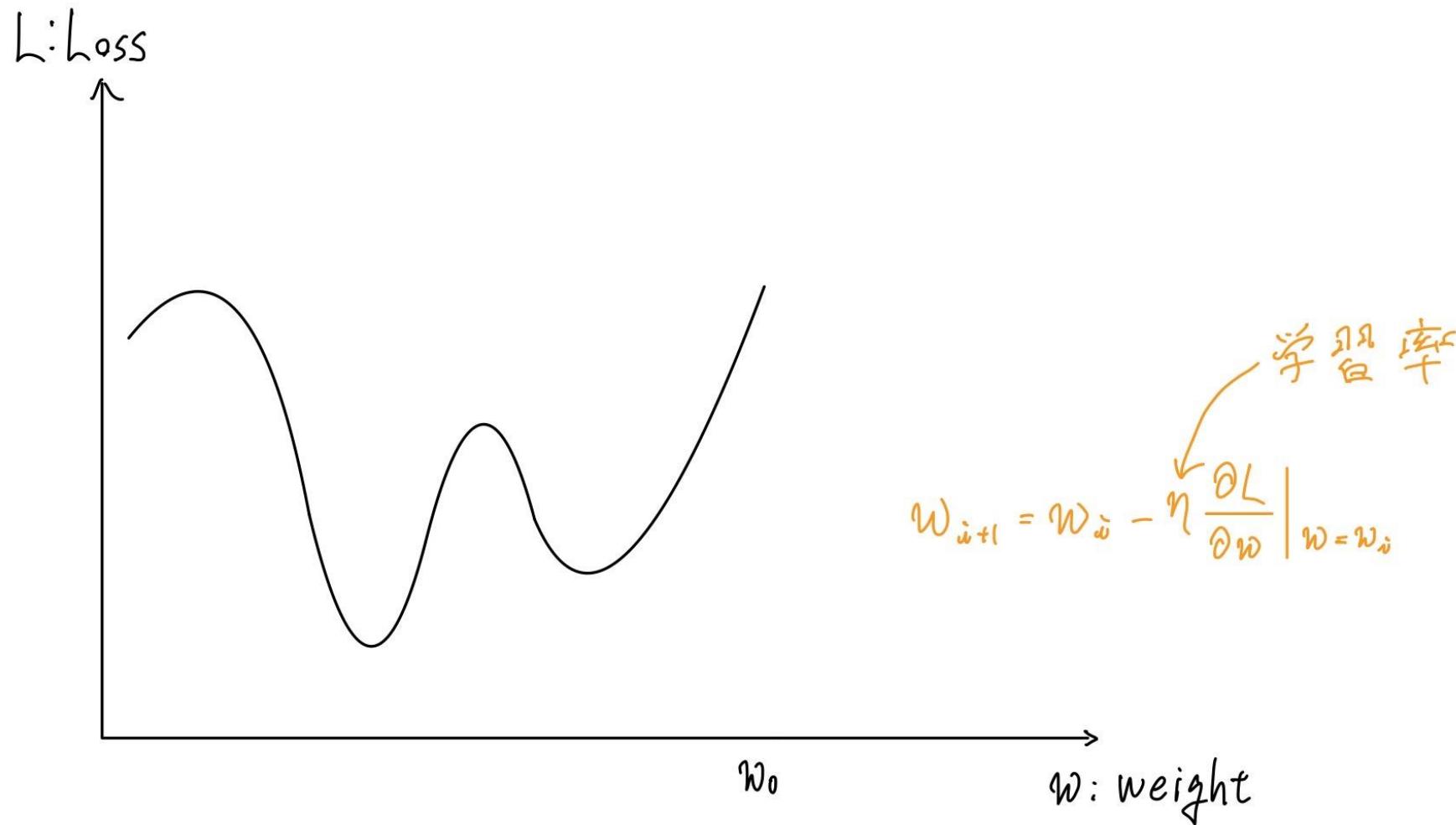
$$\begin{aligned} E &= -\frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K \hat{y}_{nk} \ln \hat{y}_{nk} \\ &= -\frac{1}{N} \sum_{n=1}^N \sum_{i=1}^3 t_{ni} \ln z_{ni} \end{aligned}$$

ニューラルネットワーク-数値微分

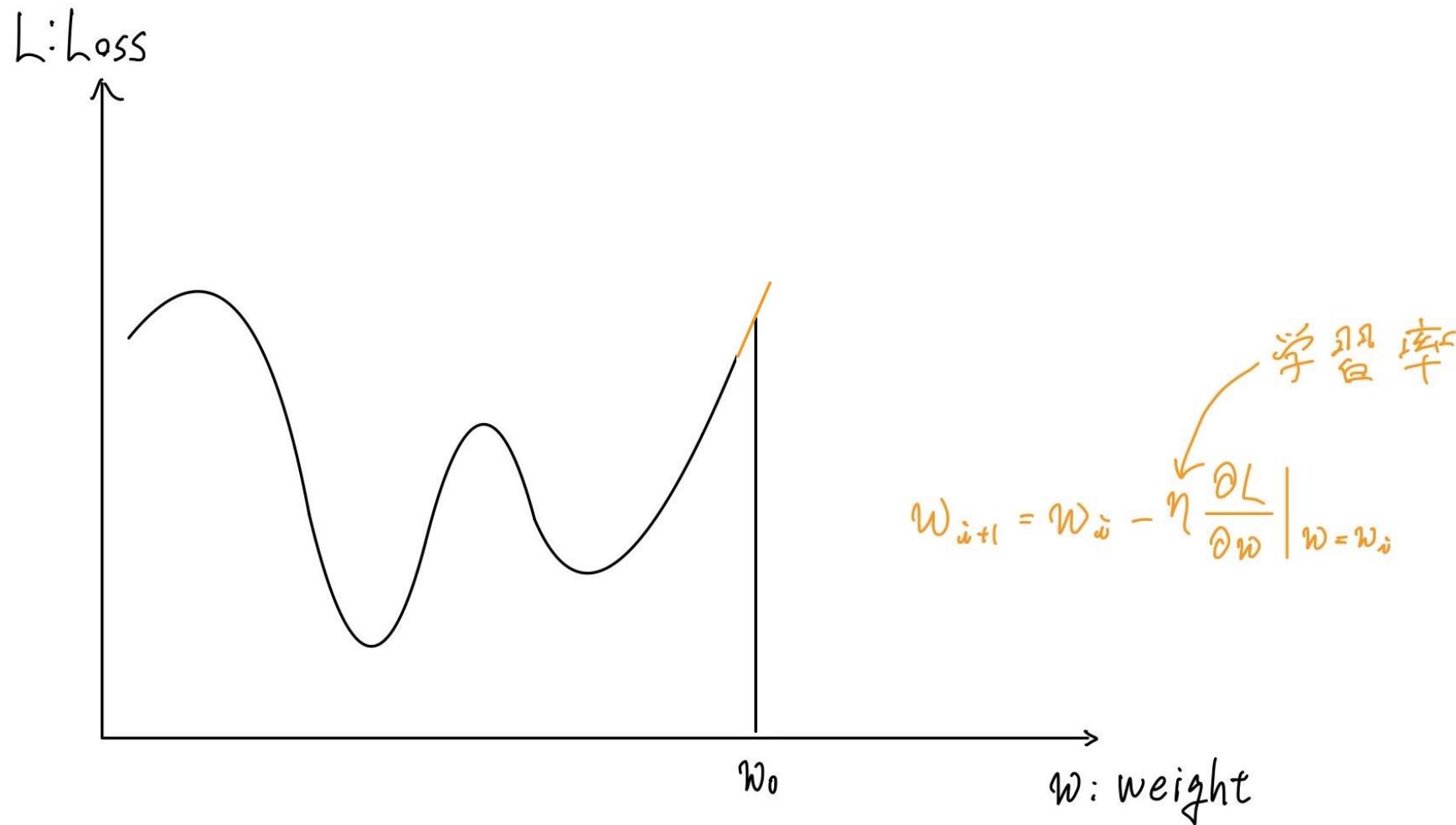


$$\left. \frac{\partial L}{\partial w} \right|_{w=w_0} = \frac{f(w_0 + h) - f(w_0 - h)}{2h}$$

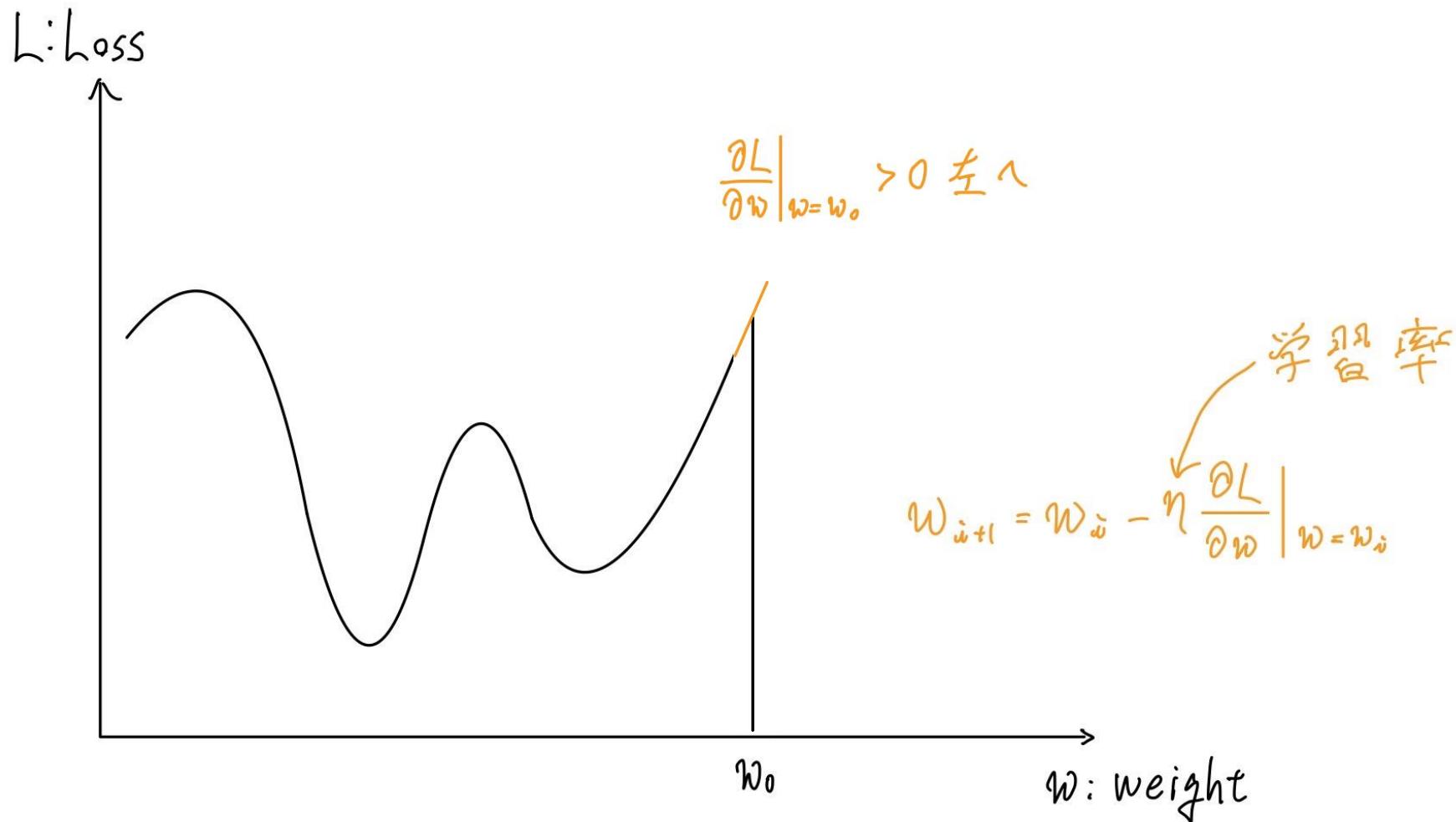
ニューラルネットワーク-勾配法



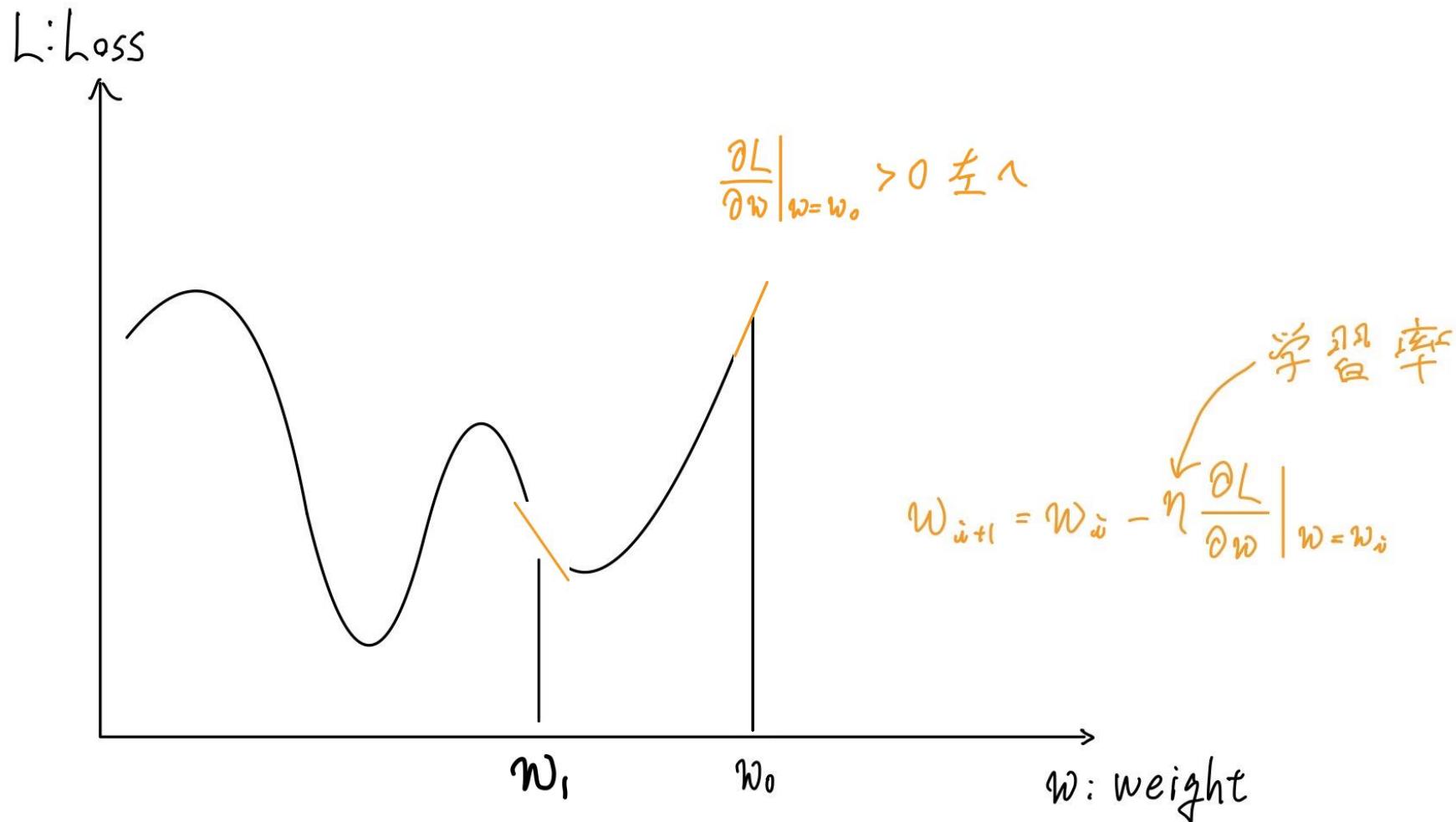
ニューラルネットワーク-勾配法



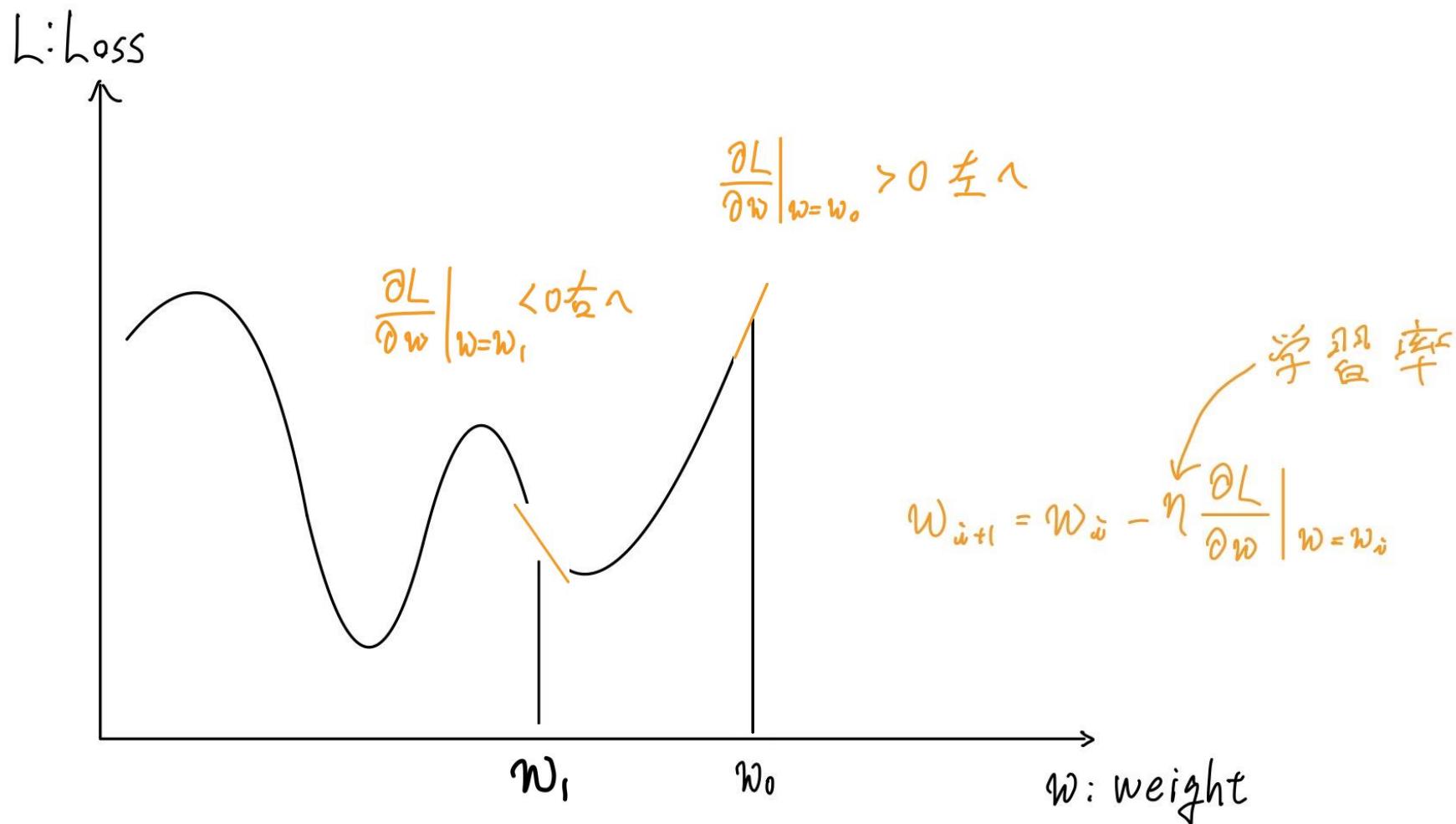
ニューラルネットワーク-勾配法



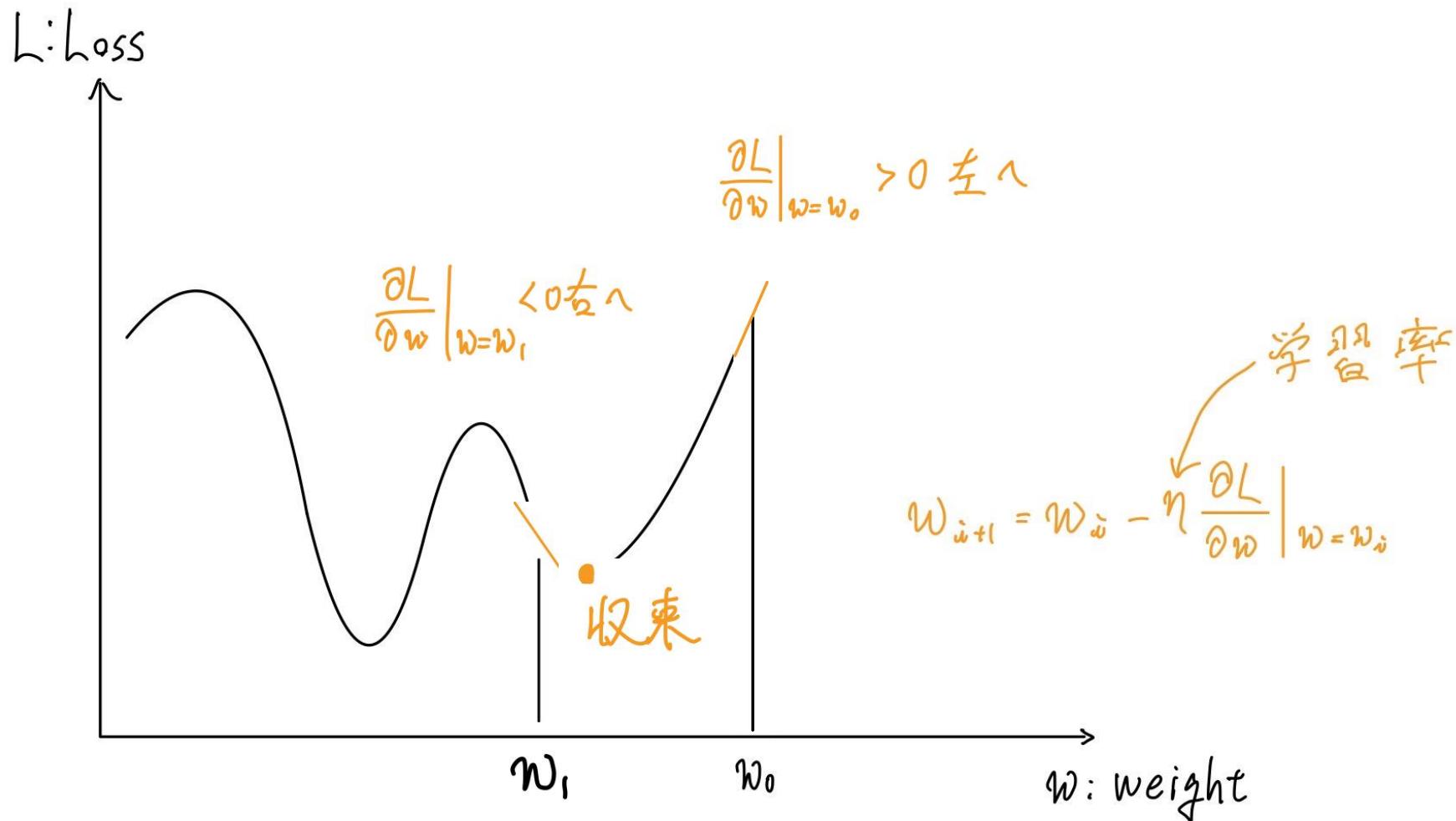
ニューラルネットワーク-勾配法



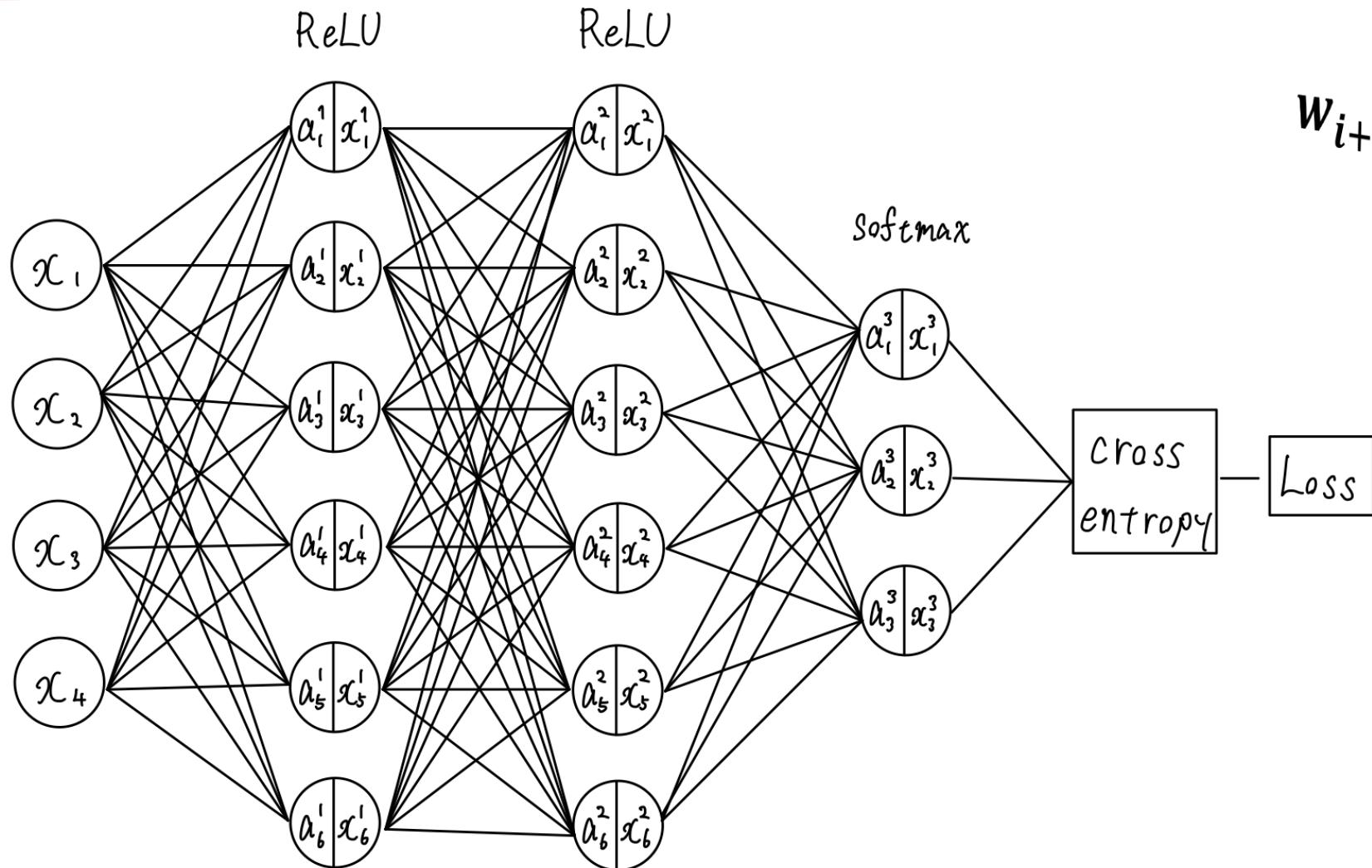
ニューラルネットワーク-勾配法



ニューラルネットワーク-勾配法

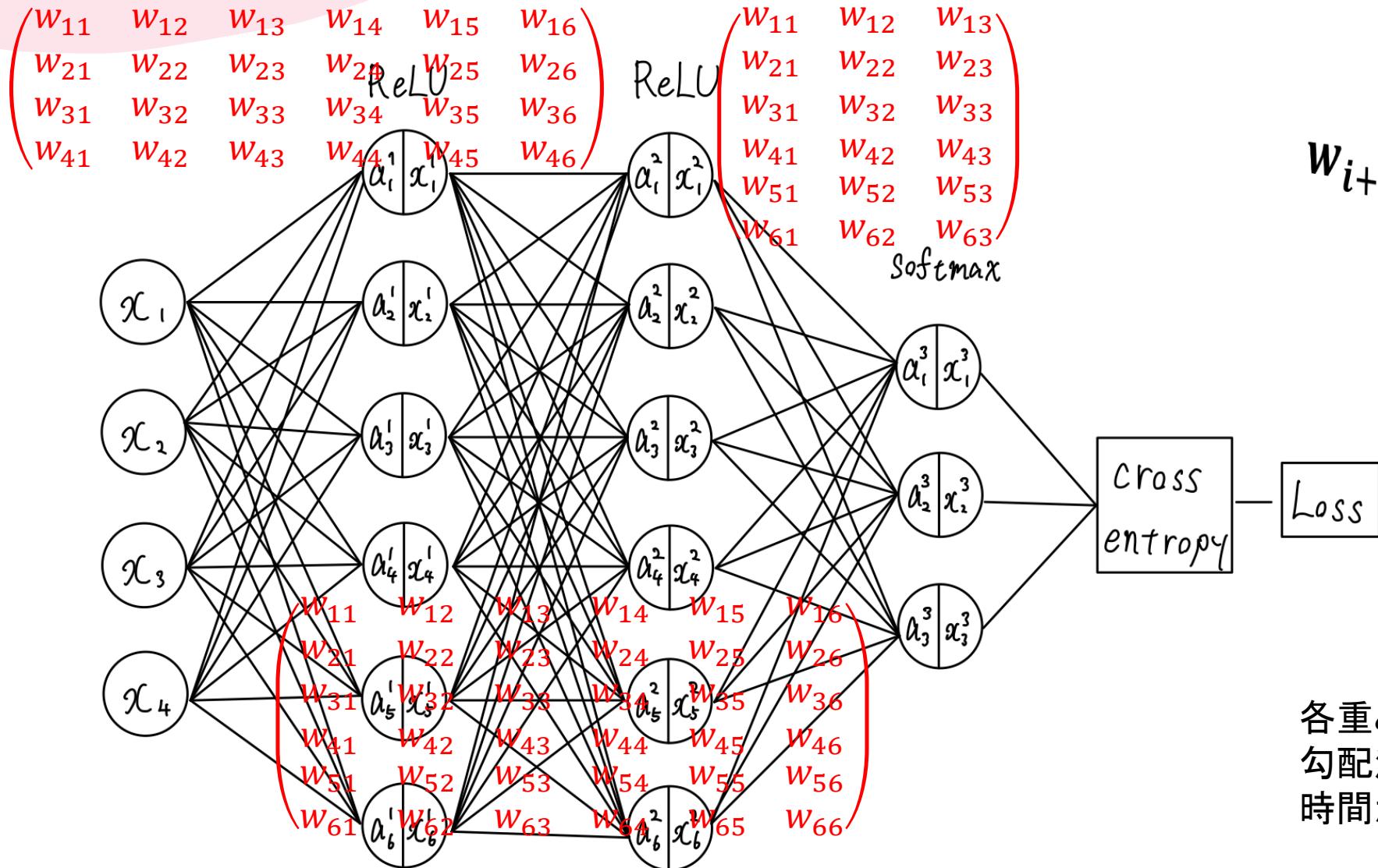


ニューラルネットワーク-勾配法



$$w_{i+1} = w_i - \eta \frac{\partial L}{\partial w} \Big|_{w=w_i}$$

ニューラルネットワーク-勾配法



$$w_{i+1} = w_i - \eta \frac{\partial L}{\partial w} \Big|_{w=w_i}$$

各重みについて
勾配法をやってると
時間がかかる



ニューラルネットワーク
学習 完



ニューラルネットワーク
次回 誤差逆伝播法