パターン認識特論
―閾値論理と線形識別機構―

担当:和田　俊和
部屋　A513
Email　twada@ieee.org
講義資料はhttp://wada1.sys.wakayama-u.ac.jp/PRA/

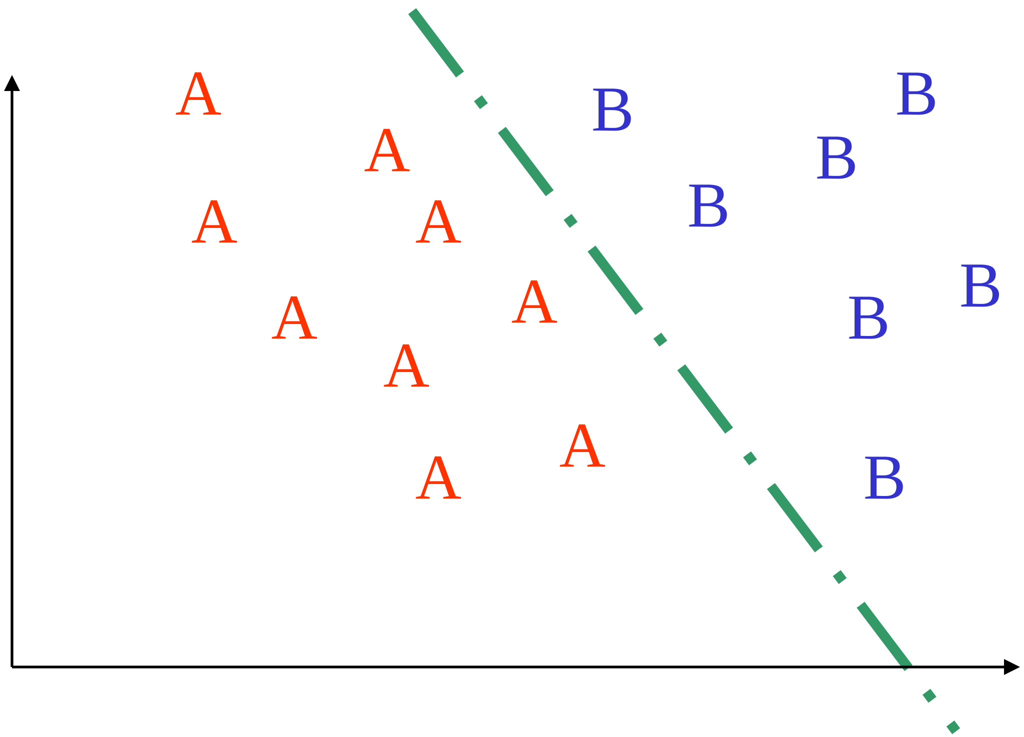
線形識別機構

閾値論理

線形識別関数とは

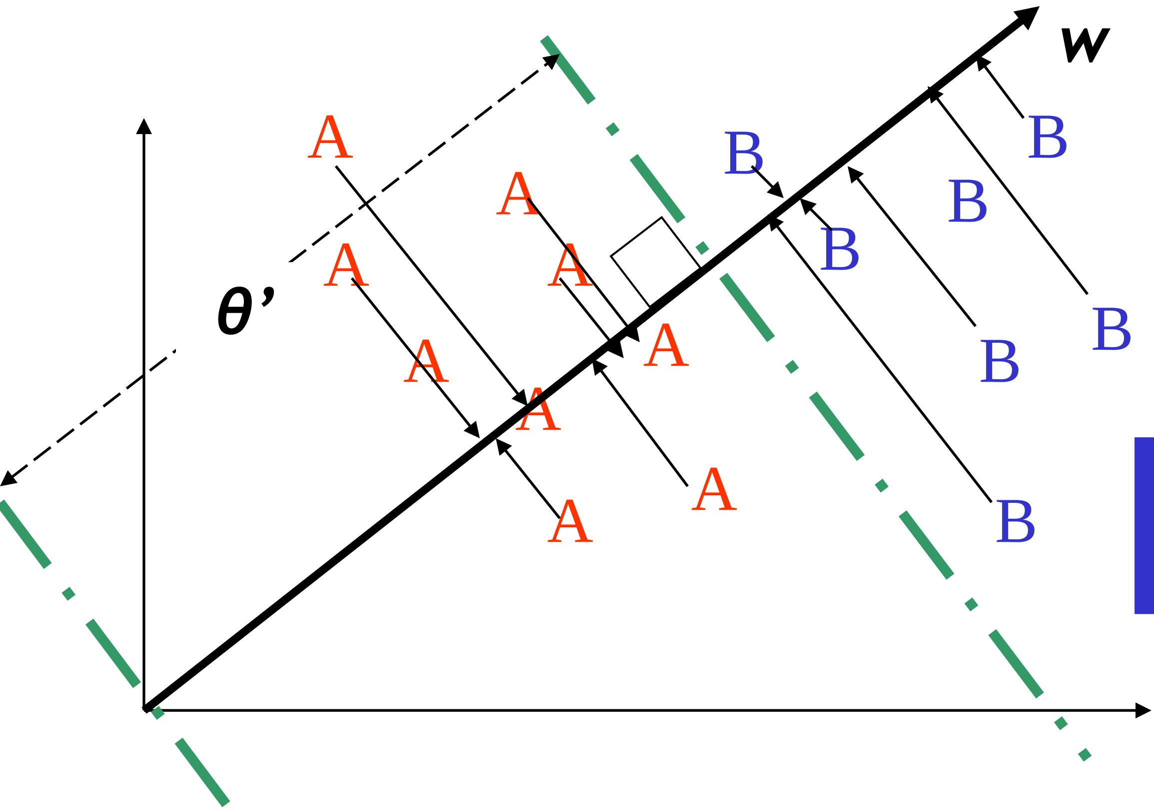
<u>**識別面**</u>：異なるカテゴリー（クラス）に属するパターンを分離する面

<u>**線形識別面**</u>：線形方程式として表現できる識別面



線形識別関数の数学的表現

<u>線形識別の方法</u>:線形識別面に直交するベクトル$\boldsymbol{w}$上にパターン$\boldsymbol{x}$を投影し、原点からその投影点までの長さVの大小関係によって帰属するクラスを決定する。



$$V = \frac{\boldsymbol{w}\cdot\boldsymbol{x}}{|\boldsymbol{w}|}$$

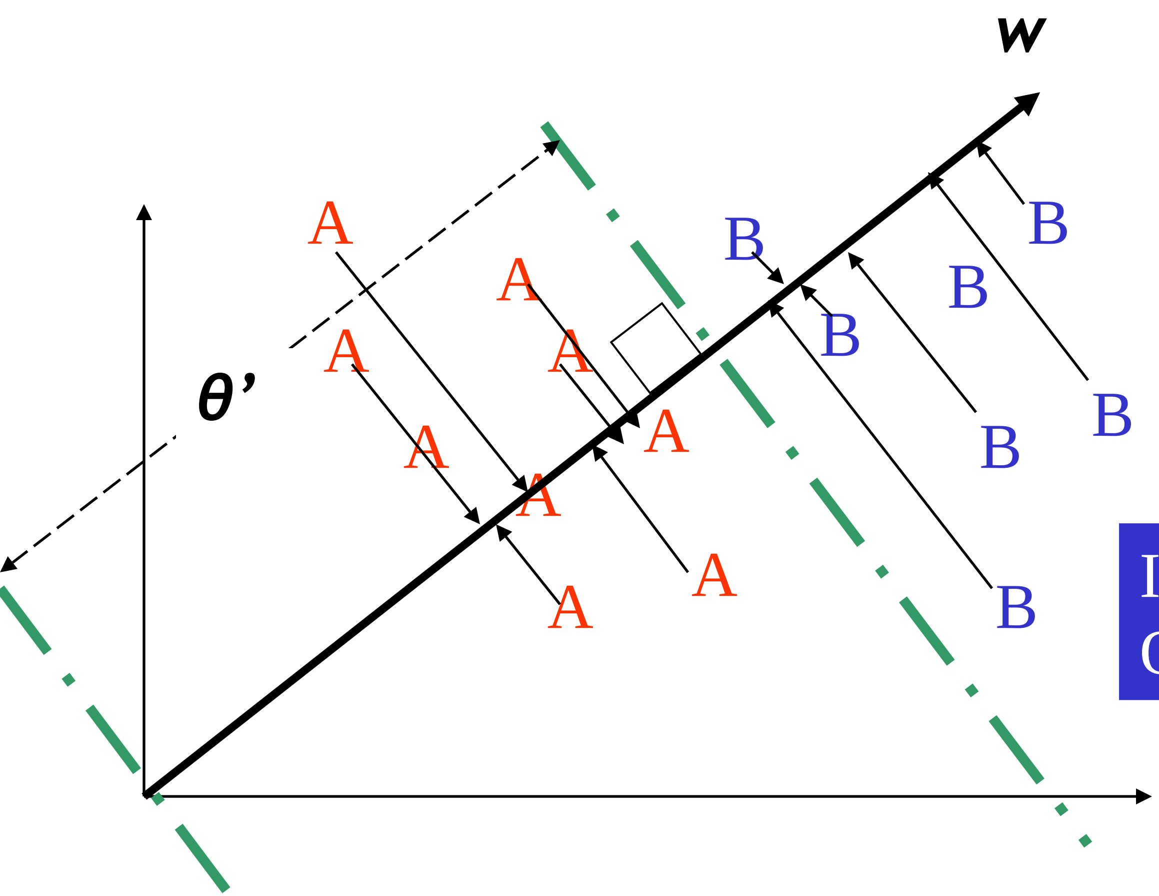
If $v<\theta'$, $\boldsymbol{x}$ belongs to A
Otherwise, $\boldsymbol{x}$ belongs to B

線形識別関数の数学的表現

真面目に長さを計らなくても良い。

$$v = \frac{w \cdot x}{|w|}$$

$$|w|v = w \cdot x$$



したがって、$w \cdot x$ と $|w|\theta'(=\theta)$ の大小関係を調べれば良い.

If $w \cdot x < \theta$, x belongs to A
Otherwise, x belongs to B

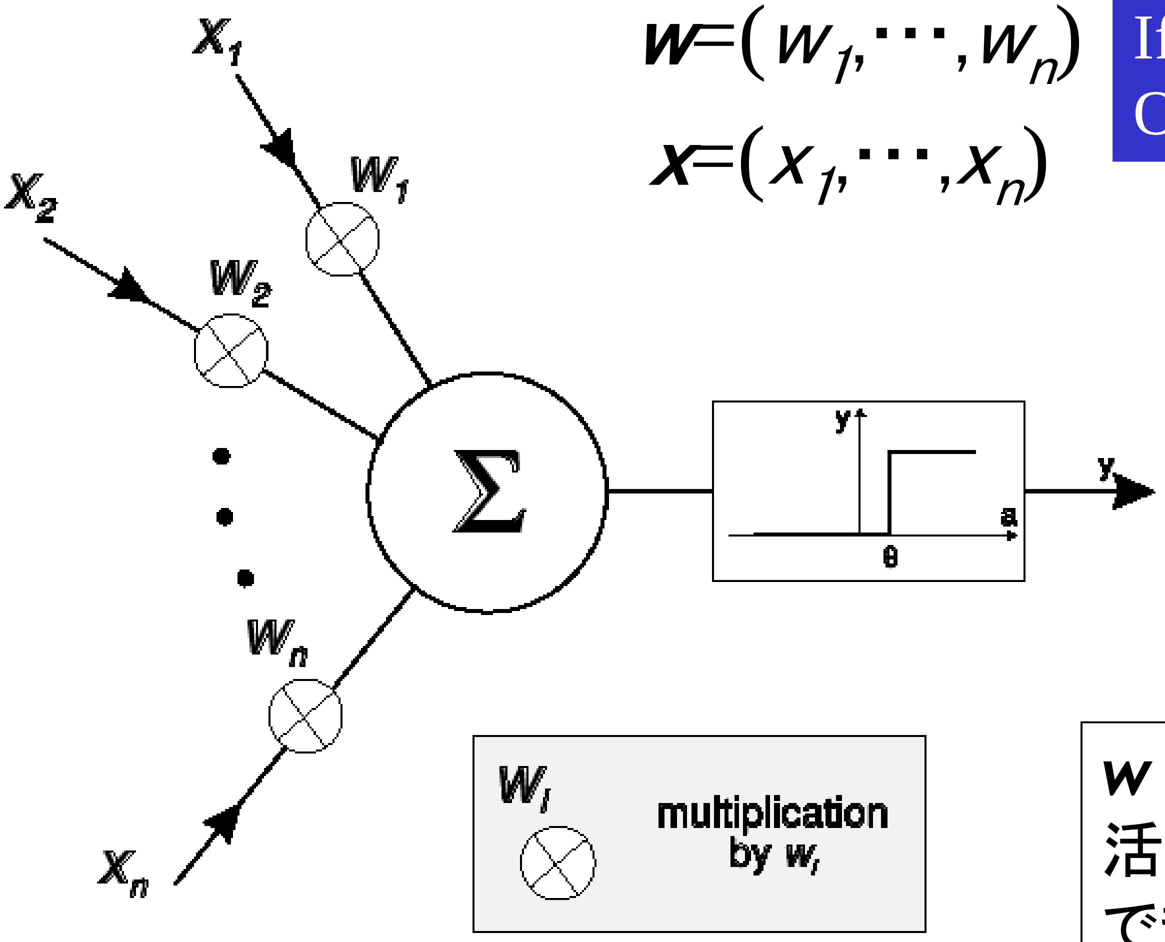
$\boldsymbol{w \cdot x}$の意味（TLUとの関係）

TLU：**T**hreshold **L**ogic **U**nit (McCulloch and Pitts, 1943)

$\boldsymbol{w}=(w_1,\cdots,w_n)$

$\boldsymbol{x}=(x_1,\cdots,x_n)$

If $\boldsymbol{w \cdot x}<\theta$, $y=0$
Otherwise, $y=1$



TLU

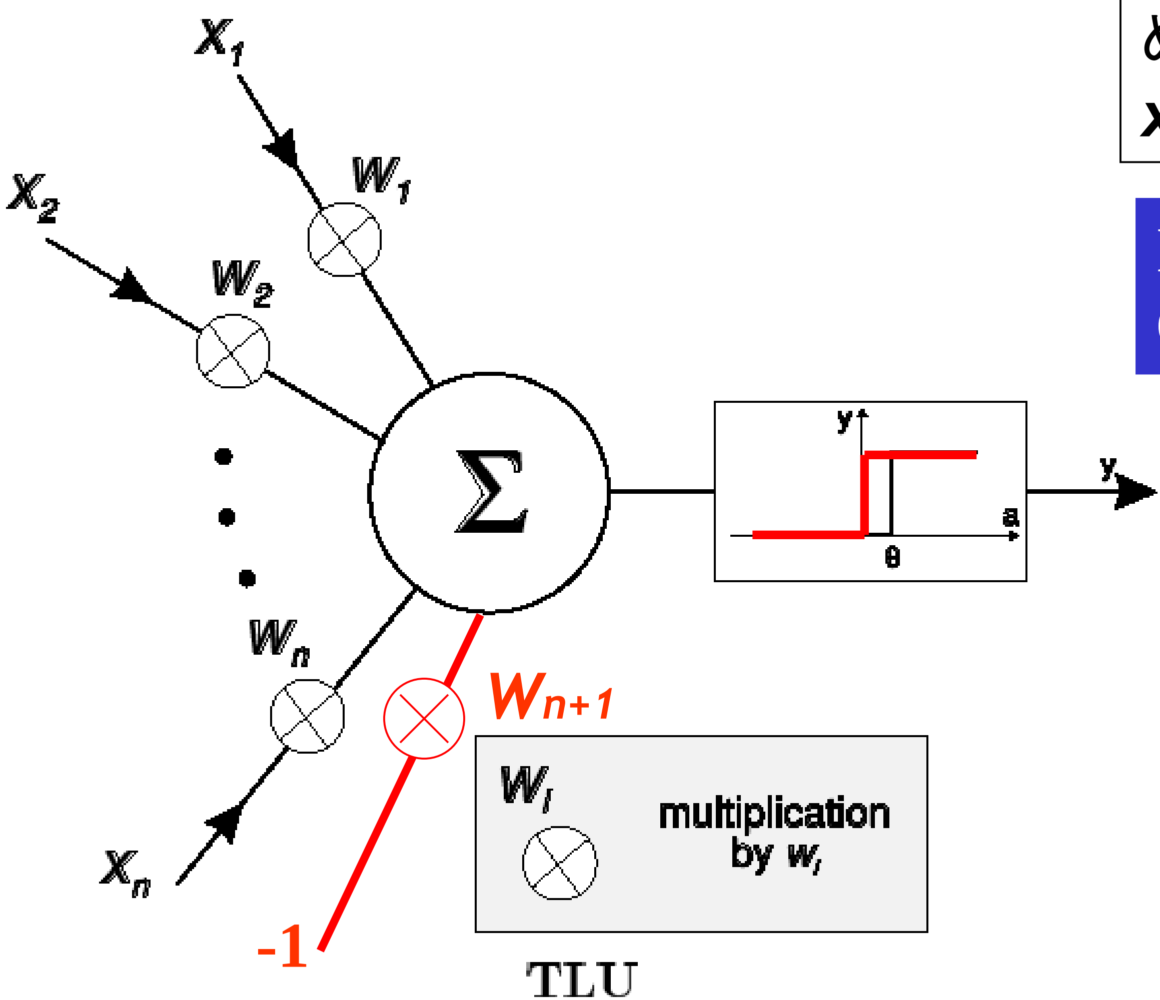
$\boldsymbol{w \cdot x}$ は、TLUの内部活性度と見なすことができる。

Wの拡張

x_{n+1}= -1につながった入力を付加する。

閾値θをベクトルwに含めてしまう。
$x_{n+1}=-1$

If $w \cdot x<0$, $y=0$
Otherwise, $y=1$

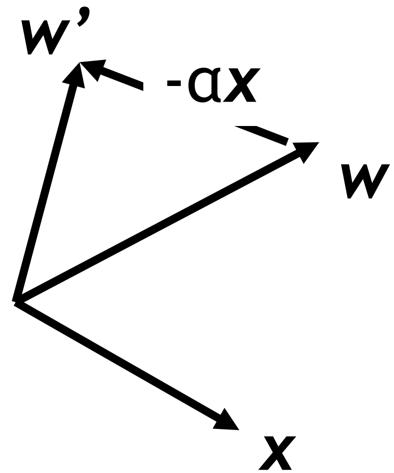


線形識別関数の学習
(誤識別した場合の*w*の調整)

教師付き学習：入力データとクラスデータのセット(x,t)を与える。(t=$\mathbf{0}$ **⇒y=0, t=**$\mathbf{1}$ **⇒y=1)**

誤識別のケース:

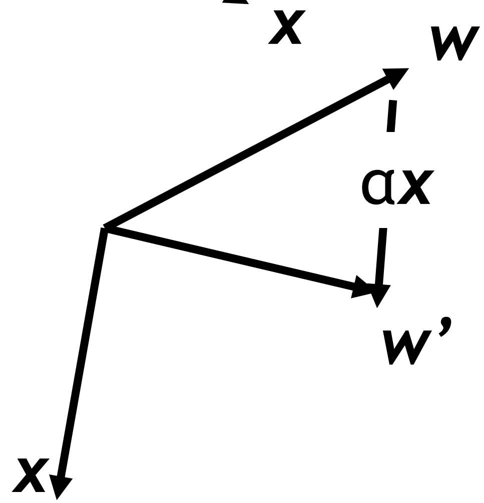
• $\boldsymbol{w} \cdot \boldsymbol{x}>0$ (y=1) **and** t =0



$$\boldsymbol{w}' = \boldsymbol{w} + \alpha(t - y)\boldsymbol{x}$$

α：学習率

• $\boldsymbol{w} \cdot \boldsymbol{x}<0$ (y=0) **and** t =1



線形識別関数の学習の例
(ANDゲート)

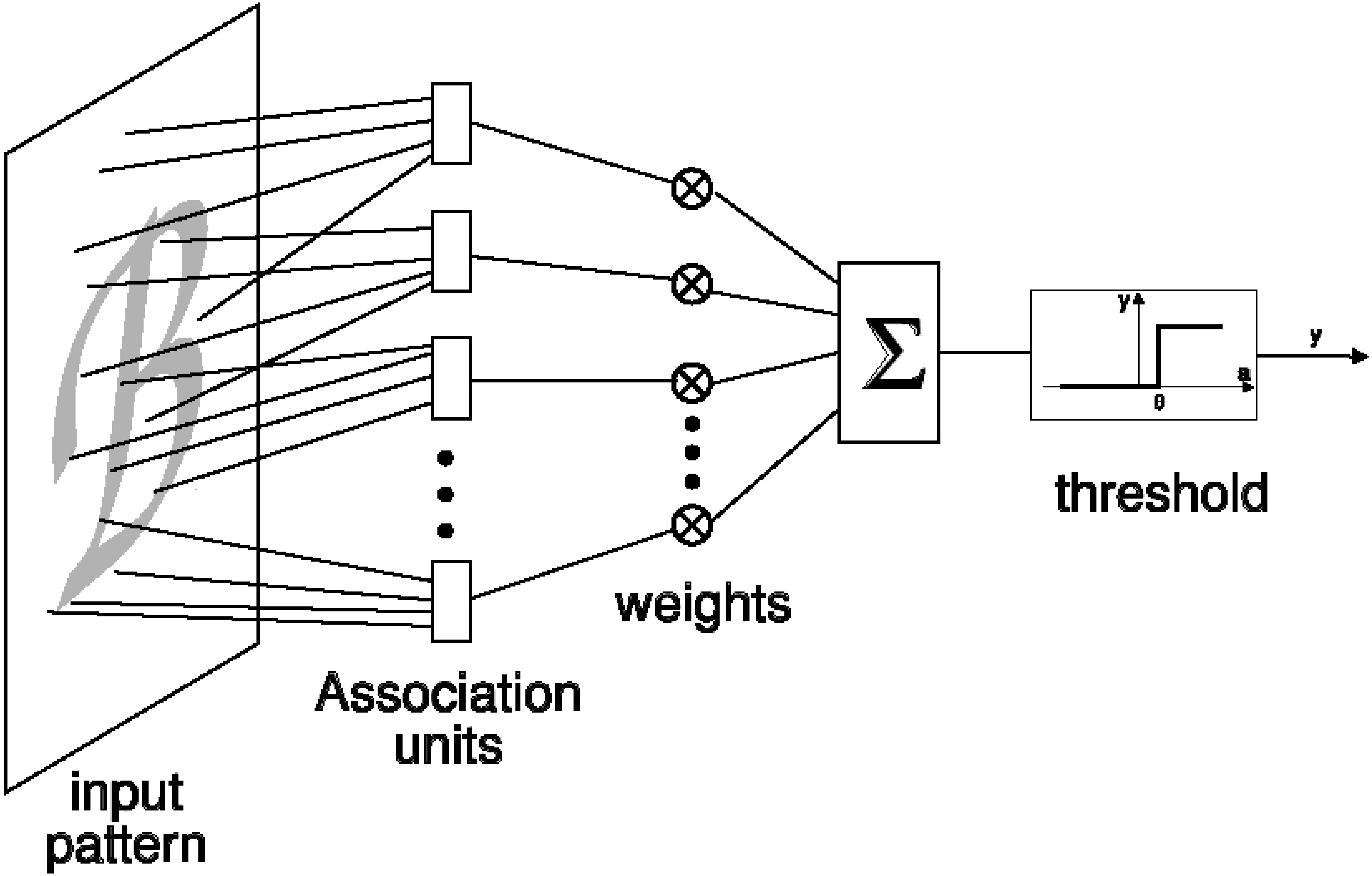
初期値：w=（0 ,0.4,0.3）, α=0.25

w_1	w_2	θ	x_1	x_2	a	y	t	$\alpha(t-y)$	δw_1	δw_2	$\delta\theta$
0.0	0.4	0.3	0	0	0	0	0	0	0	0	0
0.0	0.4	0.3	0	1	0.4	1	0	-0.25	0	-0.25	0.25
0.0	0.15	0.55	1	0	0	0	0	0	0	0	0
0.0	0.15	0.55	1	1	0.15	0	1	0.25	0.25	0.25	-0.25
0.25	0.4	0.3	0	0	0	0	0	0	0	0	0
0.25	0.4	0.3	0	1	0.4	1	0	-0.25	0	-0.25	0.25
0.25	0.15	0.55	1	0	0.25	0	0	0	0	0	0
0.25	0.15	0.55	1	1	0.4	0	1	0.25	0.25	0.25	-0.25
0.5	0.4	0.3	0	0	0	0	0	0	0	0	0
0.5	0.4	0.3	0	1	0.4	1	0	-0.25	0	-0.25	0.25
0.5	0.15	0.55	1	0	0.5	0	0	0	0	0	0

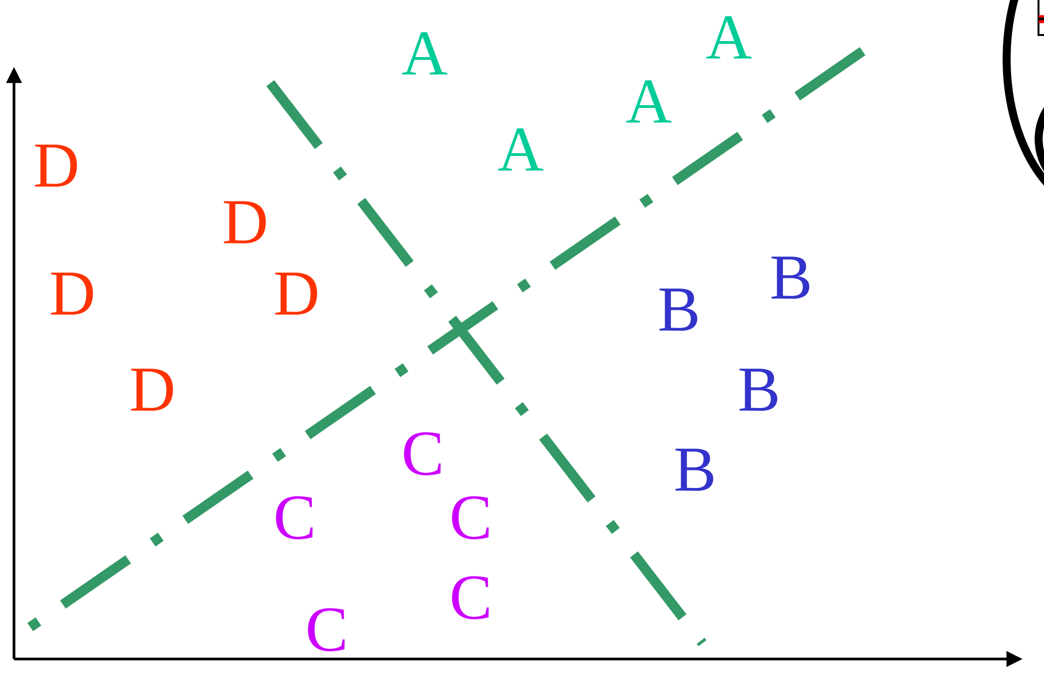
応用：Perceptron

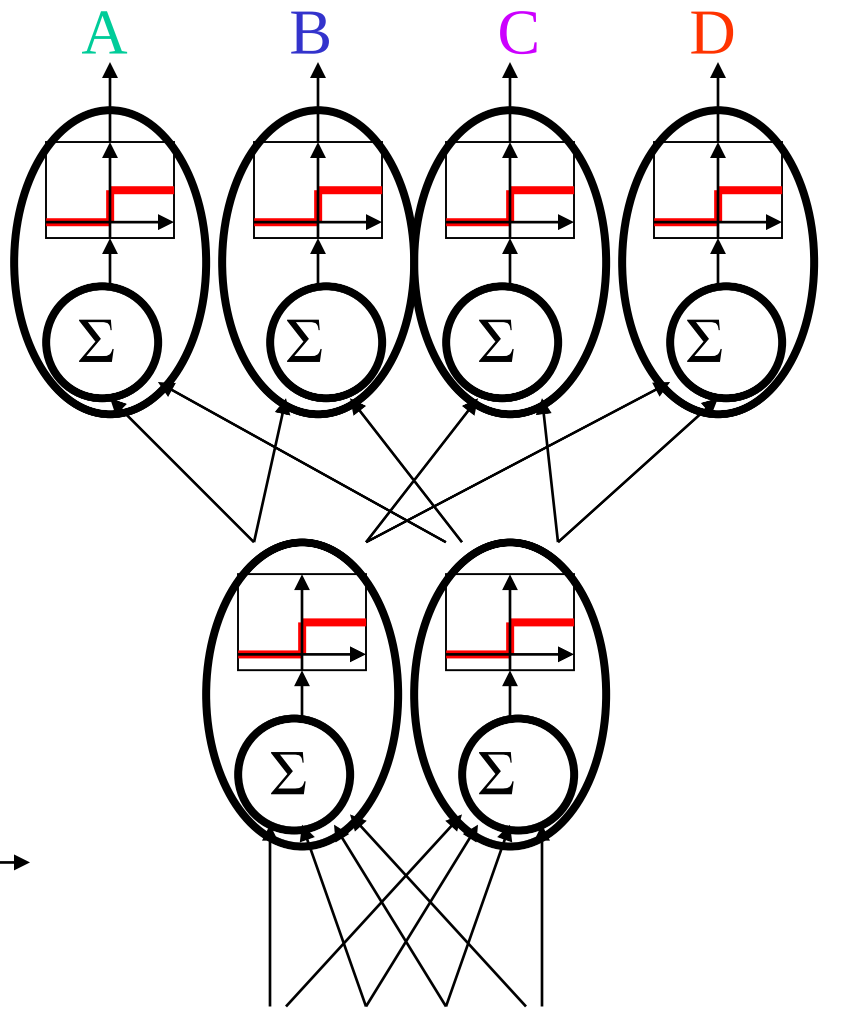
(Rosenblatt,1962) 拡張されたTLU

Association Unitは任意の二値論理関数（固定）



TLUを用いた多クラスの識別

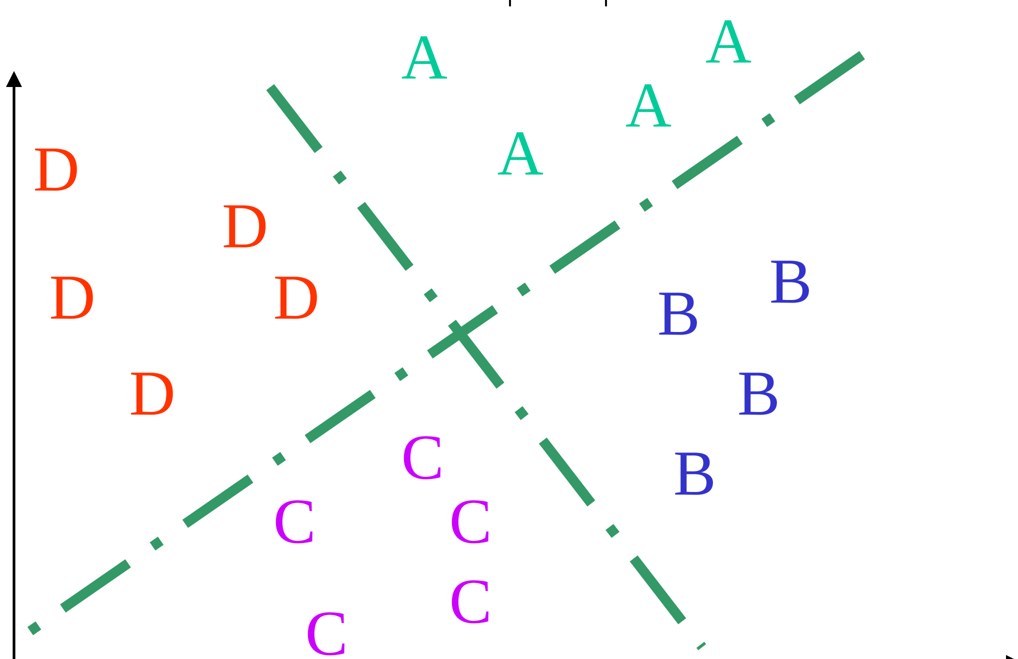


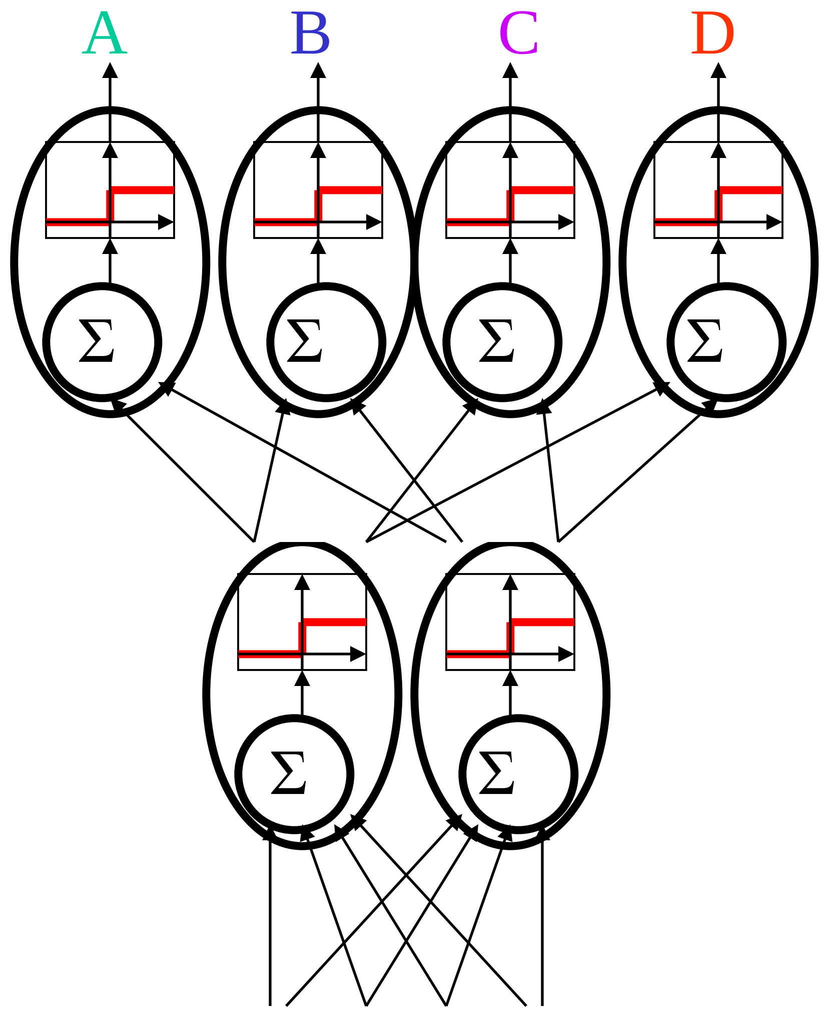


多段TLUの学習(1)

層毎に学習する方法

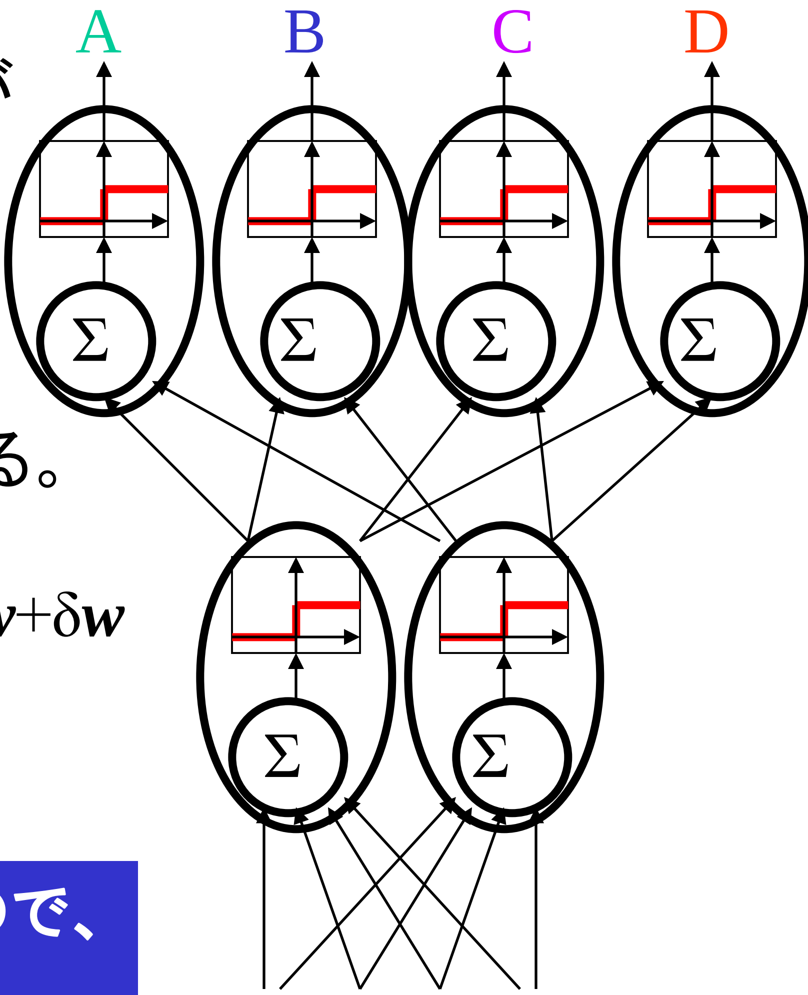
y1	y2	A	B	C	D
1	1	1	0	0	0
1	0	0	1	0	0
0	1	0	0	0	1
0	0	0	0	1	0





多段TLUの学習(2)

fは微分可能ではないが、活性度***w･x***は微分可能。***w･x*** を<u>-1,1 の2クラスに分けた教師信号を個々のTLUに与えれば</u>、学習が行えるはず。



問題： 各TLUについて $E=1/2\,(t-\boldsymbol{x}\cdot\boldsymbol{w})^2$
とし、これを最小化するようにwを変更する。

解法：

$\delta\boldsymbol{w} = -\alpha\left(\frac{\partial E}{\partial \boldsymbol{w}}\right) = \alpha(t-\boldsymbol{x}\cdot\boldsymbol{w})\boldsymbol{x}$ とし、$\boldsymbol{w}'=\boldsymbol{w}+\delta\boldsymbol{w}$
によって更新する。

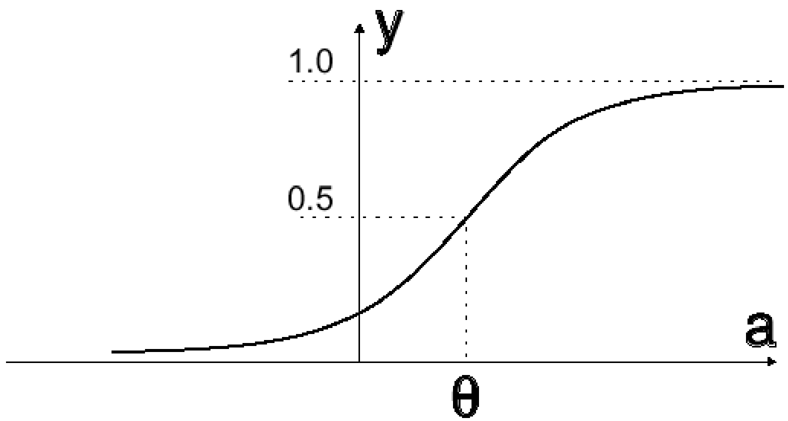
結局、全てのTLUに教師信号を与えるので、層ごとの学習と同じことになる。

問題点

- 全てのユニットに教師信号を与えなければならない。（出力層にだけ教師信号を与えたい。）

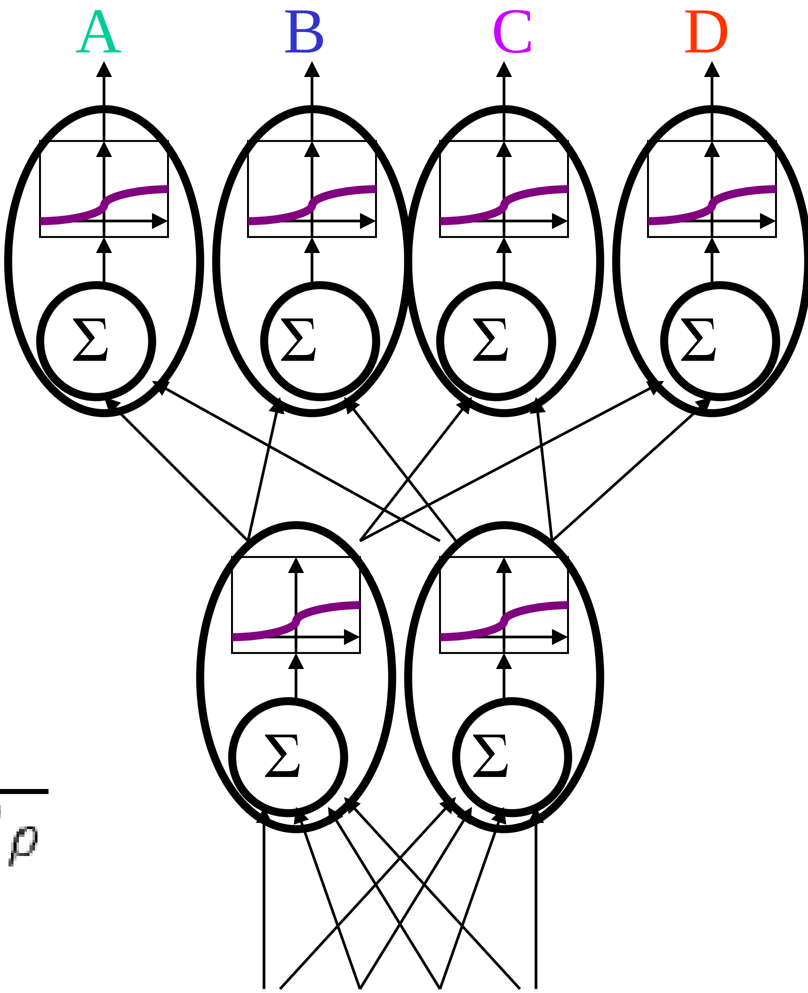
- f(x ; w)は不連続関数でありwで微分することができない。（閾値処理に起因する。）

ニューラルネット



sigmoid

$$y = \sigma(a) \equiv \frac{1}{1 + e^{-(a-\theta)/\rho}}$$



ニューラルネットの学習

$E=1/2(\boldsymbol{t}-\sigma(\underline{\boldsymbol{x}\cdot\boldsymbol{w}}))^2$を$\boldsymbol{w}$で微分すると

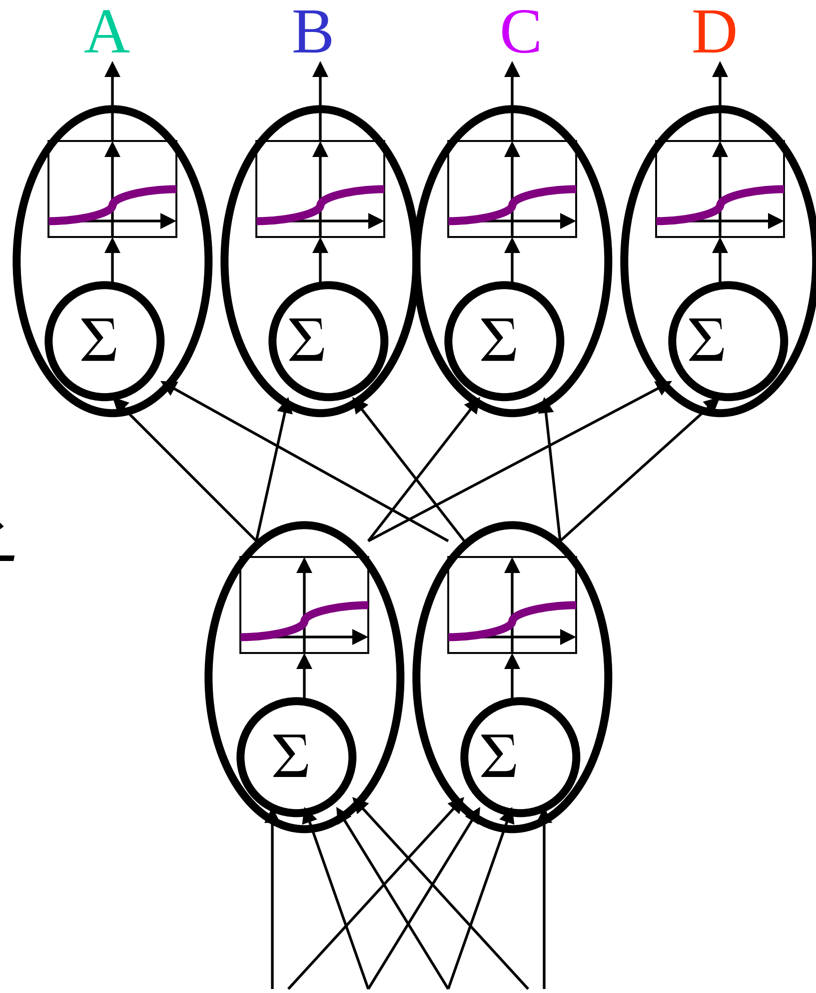
$$-\sigma'(\boldsymbol{x}\cdot\boldsymbol{w})\,(\boldsymbol{t}-\sigma(\boldsymbol{x}\cdot\boldsymbol{w}))\boldsymbol{x}$$

が得られる。このことから、

$\delta=\alpha\underline{\sigma'(\boldsymbol{x}\cdot\boldsymbol{w})}\,(\boldsymbol{t}-\sigma(\boldsymbol{x}\cdot\boldsymbol{w}))$ とし、

$\boldsymbol{w}'=\boldsymbol{w}+\delta\boldsymbol{x}$

によって更新する方法が考えられる。
(αは学習係数)



教師信号 t を全てのニューロンに与えなければならない。。。。本当か？

出力層について教師信号を与え、中間(隠れ)層のδは出力層のδから計算する。

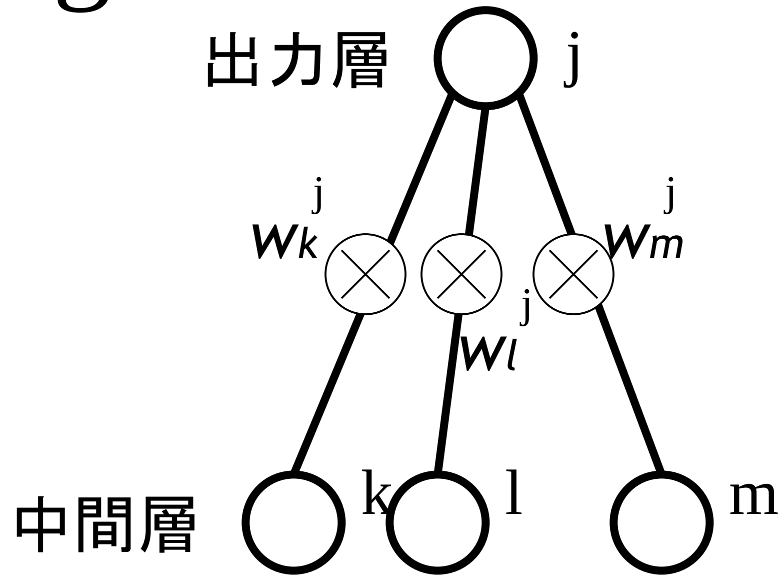
ニューラルネットの学習
back propagation

出力層（j番目）:

$\delta^j = \alpha\sigma'(\boldsymbol{x}^j \cdot \boldsymbol{w}^j)\,(t^j - \sigma(\boldsymbol{x}^j \cdot \boldsymbol{w}^j))$ とし、$\boldsymbol{w}^{j\prime} = \boldsymbol{w}^j + \delta^j \boldsymbol{x}^j$

によって更新する。

（δは$\boldsymbol{x}$に対する$\boldsymbol{w}$の修正量）



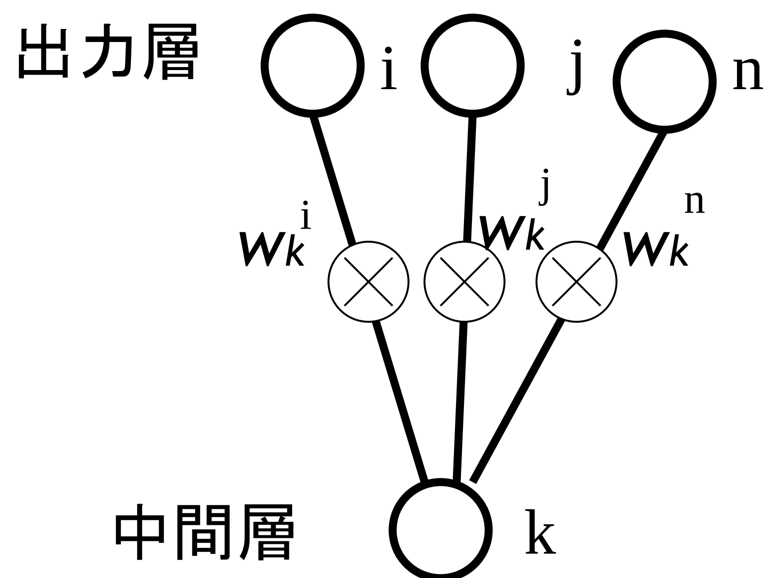
中間層(k番目):

$\delta^k = \sigma'(\boldsymbol{x}^k \cdot \boldsymbol{w}^k) \sum_{j \in Ik} \delta^j w_k^j$ とし、

$\boldsymbol{w}^{k\prime} = \boldsymbol{w}^k + \delta^k \boldsymbol{x}^k$

によって更新する。

（出力層の修正係数δを使って
　中間層の修正係数を計算する）



ニューラルネットの学習
back propagation

Forward pass:

- 入力を与え、中間層の出力を計算する。
- 中間層の出力から出力層の値を計算する。

出力層:

$\delta_j = \alpha\sigma'(\boldsymbol{x}_j \cdot \boldsymbol{w}_j)(t_j - \sigma(\boldsymbol{x}_j \cdot \boldsymbol{w}_j))$ とし、$\boldsymbol{w}_j' = \boldsymbol{w}_j + \delta_j \boldsymbol{x}_j$

によって更新する。

Backward pass:

- 教師信号と出力層の値の差からδを計算し出力層につながる重み係数を変更する。
- 出力層のδから中間層のδ を計算する。

中間層:

$\delta_k = \sigma'(\boldsymbol{x}_k \cdot \boldsymbol{w}_k) \sum_{j \in I_k} (\delta_j w_{jk})$ とし、

$\boldsymbol{w}_k' = \boldsymbol{w}_k + \delta_k \boldsymbol{x}_k$

によって更新する。

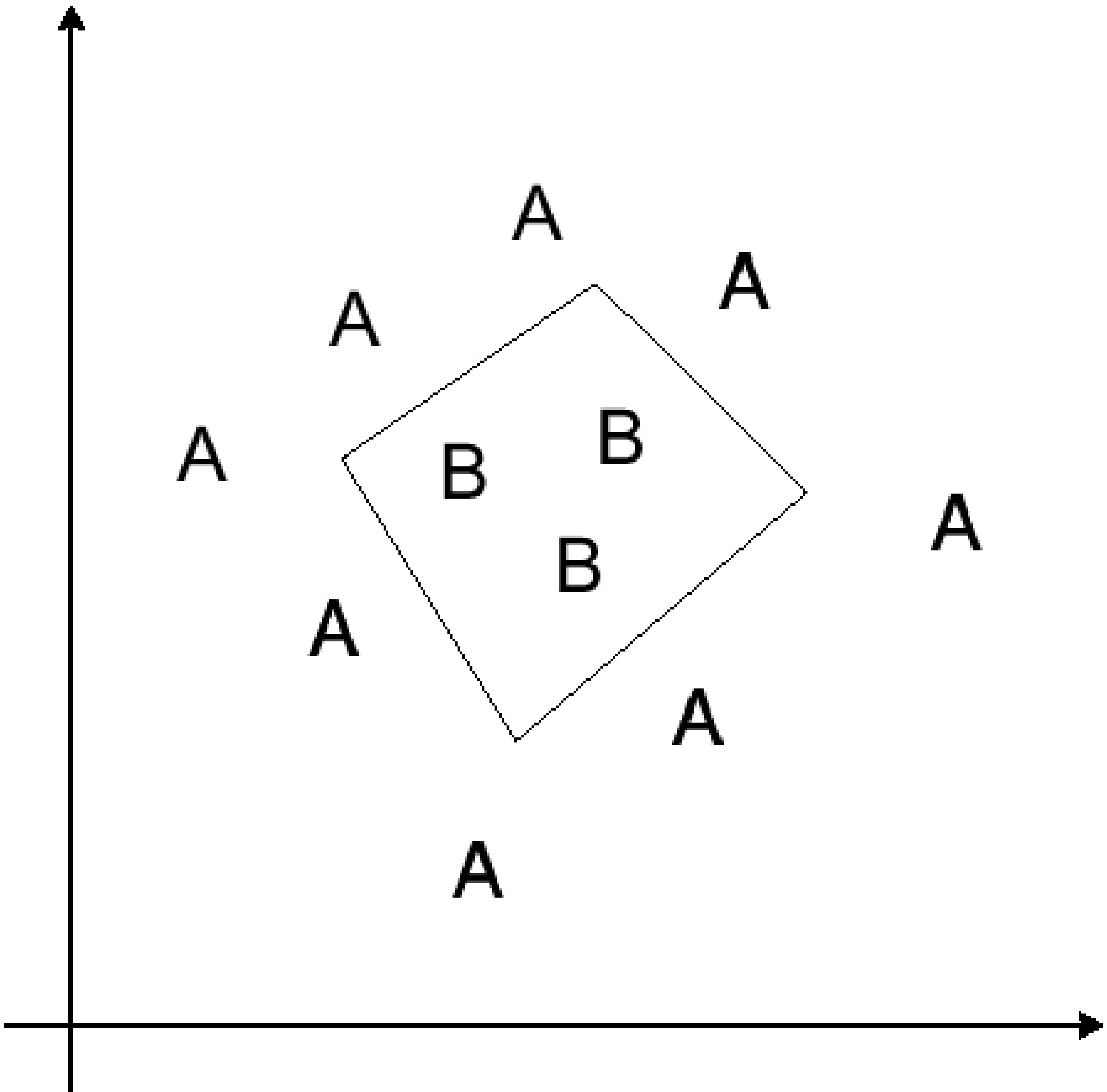
学習の効率化

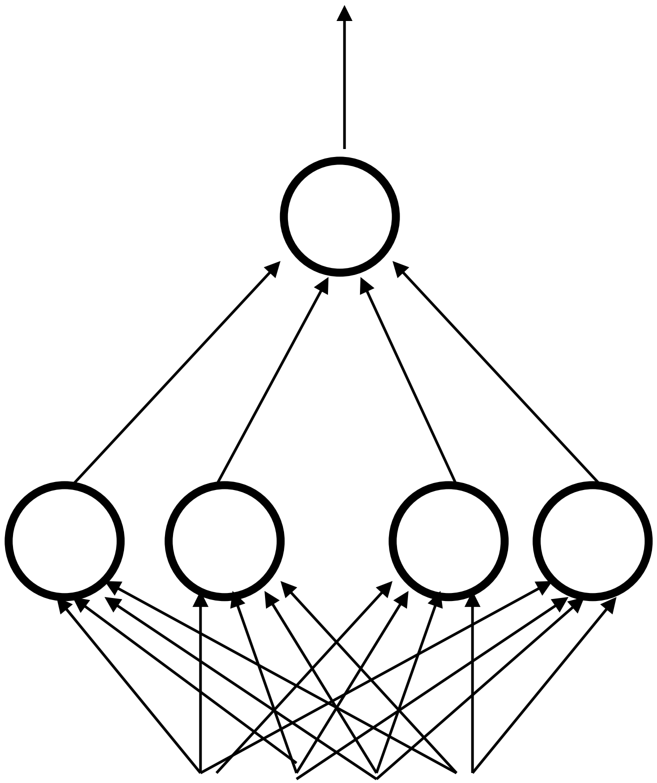
学習率α:大きくすれば、学習の効果が上がるが、学習結果が不安定になる。小さくすれば学習結果は安定になるが、学習がなかなか進まない。どうするか？

前回の係数変化の傾向を利用することにより、前回も同じ方向に変化したなら、次も同じ方向に移動しやすいバイアスをかける。

$$\Delta w_i^j(n)=\alpha\delta^j x_i^j+\lambda\Delta w_i^j(n-1)$$

非線型識別関数獲得の由来

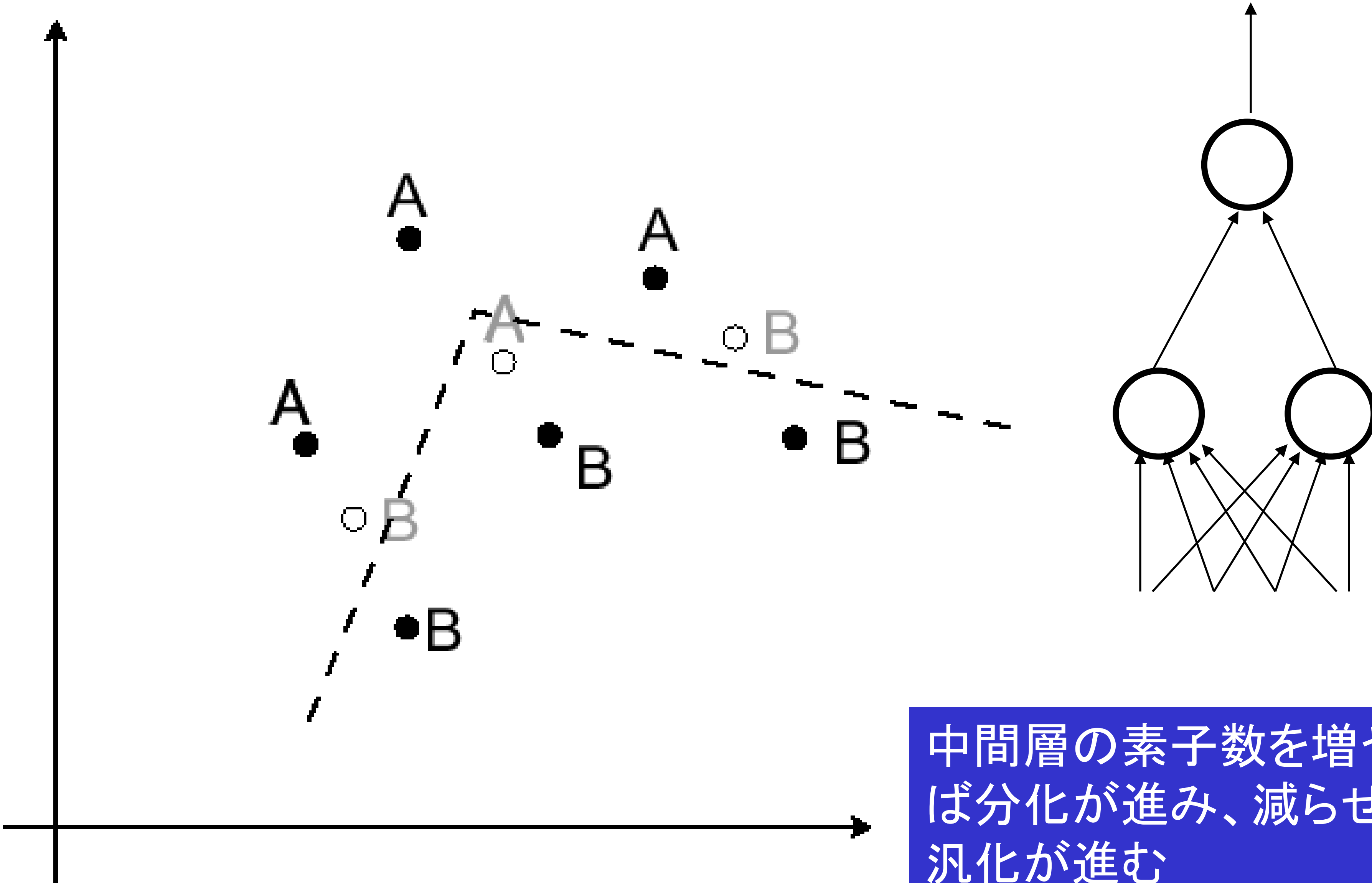




TLUでもニューロ素子でもネットワーク化すれば、非線型識別関数は獲得できる

汎化と分化の制御

中間(隠れ層)素子数の設定



中間層の素子数を増やせば分化が進み、減らせば汎化が進む

レポート課題：線形識別関数の学習
(ORゲート)

初期値：w=（0 ,0,0.7）,α=0.2として、下記の表を完成させなさい

w_1	w_2	θ	x_1	x_2	a	y	t	$\alpha(t-y)$	δw_1	δw_2	$\delta\theta$
			0	0	0	0	0	0	0	0	0
			0	1			1				
			1	0			1				
			1	1			1				
			0	0			0				
			0	1			1				
			1	0			1				
			1	1			1				
			0	0			0				
			0	1			1				
			1	0			1				