令和6年度酒田市地域デジタル人材等育成委託事業

機械学習入門

―手書き数字を見分けるシステムを作ろう―

カロル　ノヴァコフスキ
Karol Nowakowski

東北公益文科大学
karol@koeki-u.ac.jp

東北公益文科大学
Tohoku University of Community Service and Science

I
機械学習の基礎

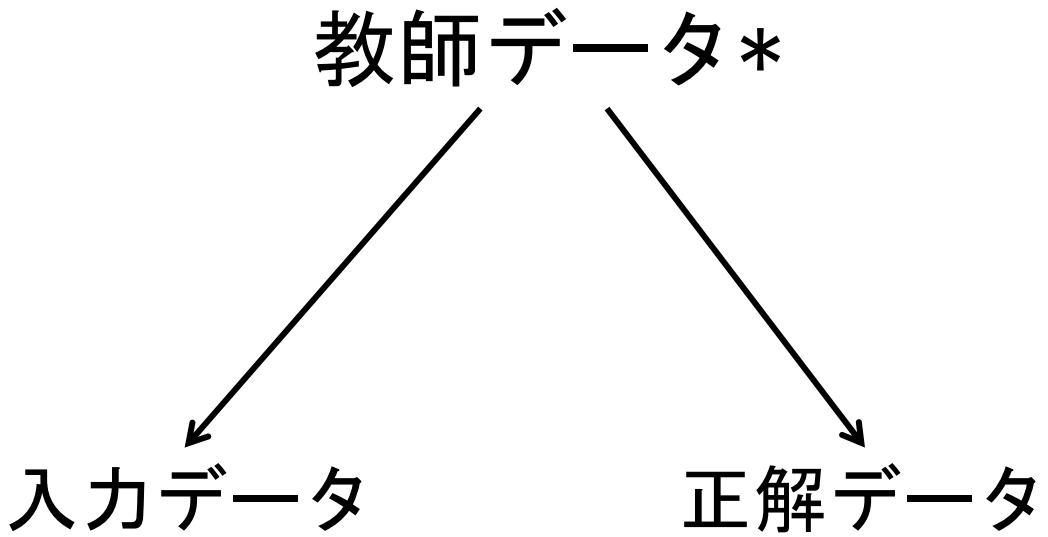
機械学習の基礎

東北公益文科大学
Tohoku University of Community Service and Science

機械学習

- データから（規則性やパターンを）学習し、それを基に未知のデータに関して予測できるプログラムを開発する技術
- 人工知能の下位分野

機械学習の基礎

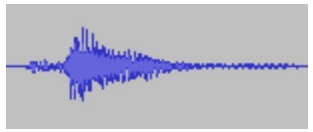
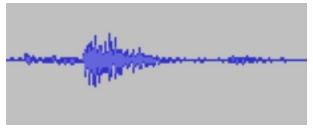
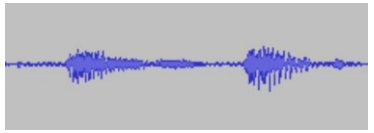


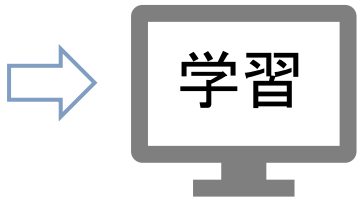
＊「教師あり学習」の場合（「教師なし学習」や「強化学習」は「正解データ」を使わない）。

機械学習の基礎

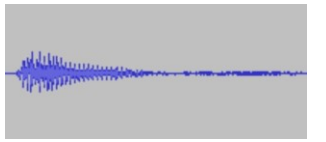
東北公益文科大学

例（音声認識）：

教師データ		
入力データ（音声）		正解データ（テキスト）
	→	dog
	→	cat
	→	hamster



（新しい）入力：



→ horse

出力：

機械学習の基礎

東北公益文科大学

例（機械翻訳）：

教師データ		
入力データ （原言語の文）		正解データ （目的言語の文）
あれが私の車です	→	That’s my car
あの白いシャツが好き	→	I like that white shirt
この馬は速い	→	This horse is fast

（新しい）入力：

あの白い車は速い →

出力：

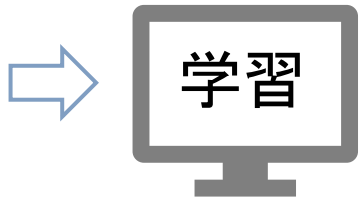
That white car is fast

機械学習の基礎

東北公益文科大学

例（言語モデル(ChatGPTなど)）：

教師データ		
入力データ（文）		正解データ（次の単語）
昔々、あるところに、お爺さんと…	→	お婆さん
昔々、あるところに、お爺さんと婆さん…	→	が
昔々、あるところに、お爺さんとお婆さんが…	→	住んで



（新しい）入力：

MNISTデータベースを用いて、手書き…

→

出力：

数字

機械学習の基礎

東北公益文科大学

例（文字認識）：

教師データ		
入力データ（画像）		正解データ（文字）
	→	0
	→	1
	→	2

（新しい）入力：

→ 出力：1

機械学習の基礎

東北公益文科大学

MNISTデータベース：

7万枚の手書き数字の画像（+正解ラベル）を含むデータベース。機械学習による画像処理システムの学習に広く使われている。

“A few samples from the MNIST test dataset” by Josef Steppan (CC BY-SA 4.0)

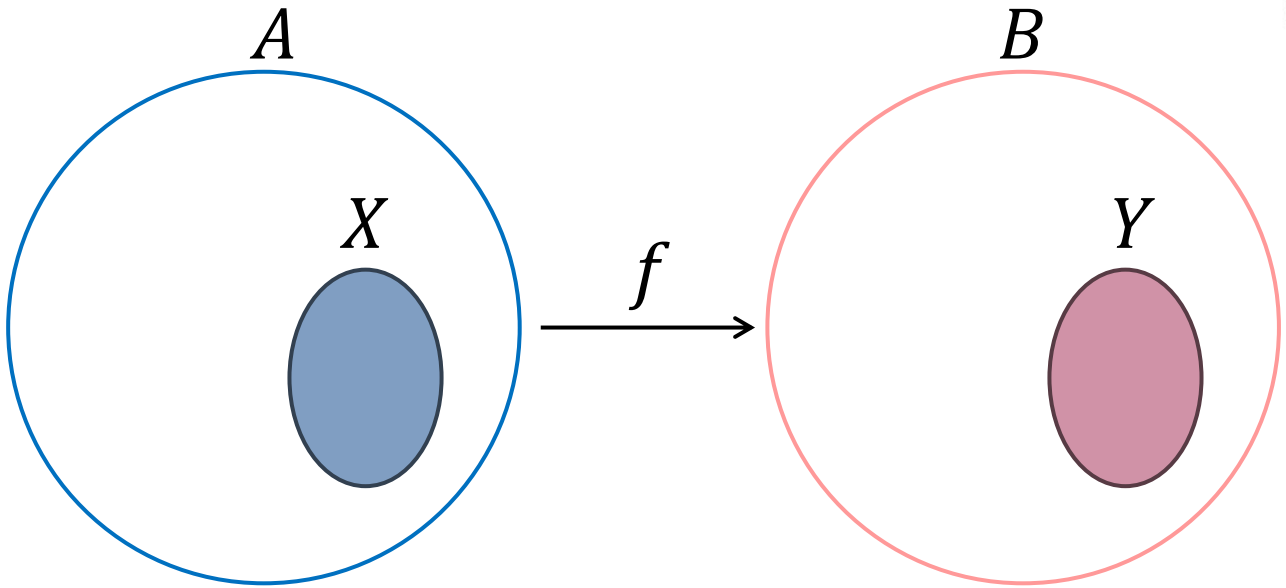
機械学習の基礎

A – 全ての入力データ

B – 全ての正解データ

X – 獲得済みの入力データ

Y – 獲得済みの正解データ

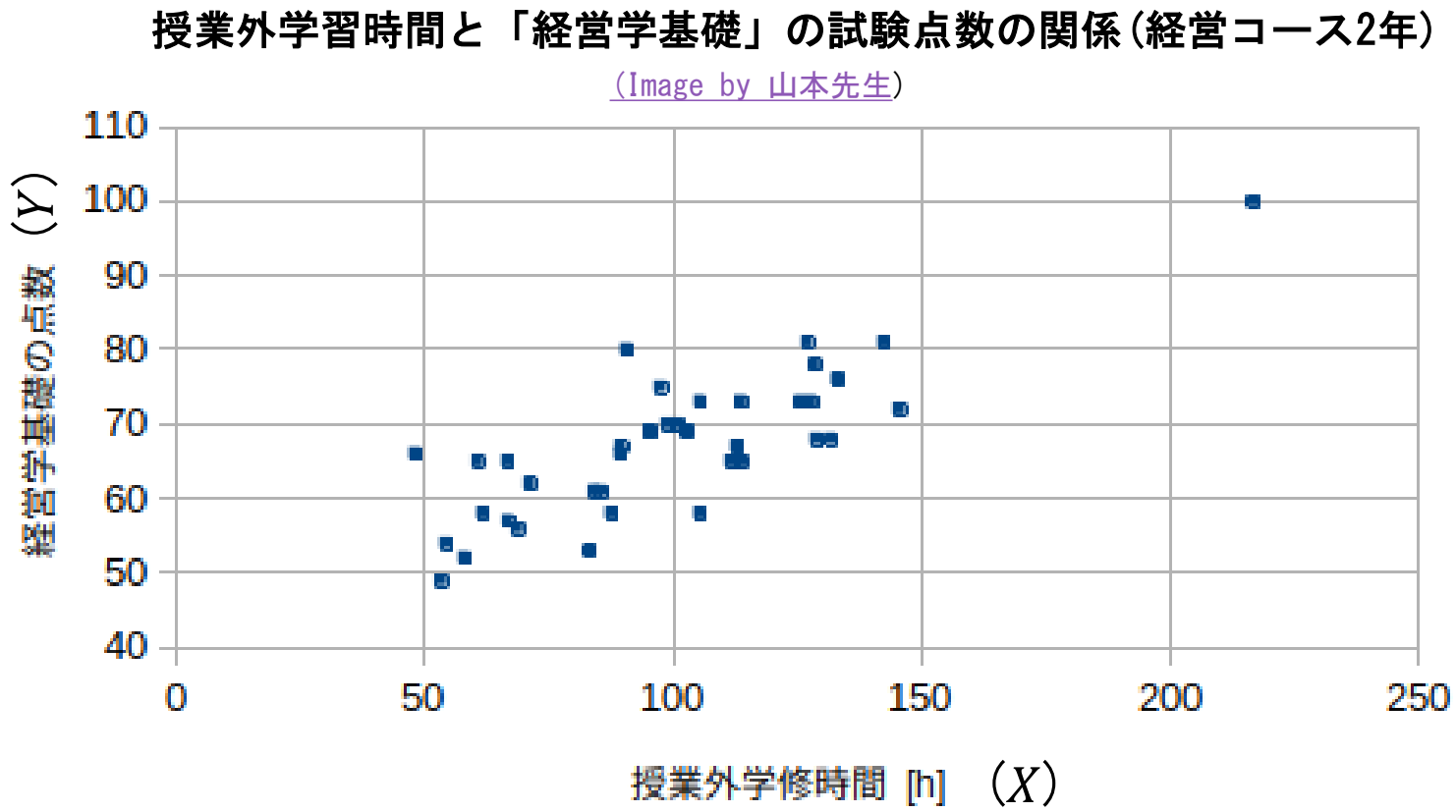


- $f: A \rightarrow B$ という関数が存在すると仮定する（非常に複雑な関数になる場合もある）。
- **機械学習の目的**： X と Y に基づいてf関数を（近似的に）求める。結果はh(hypothesis)関数である（「モデル」とも呼ばれる）。

機械学習の基礎

東北公益文科大学

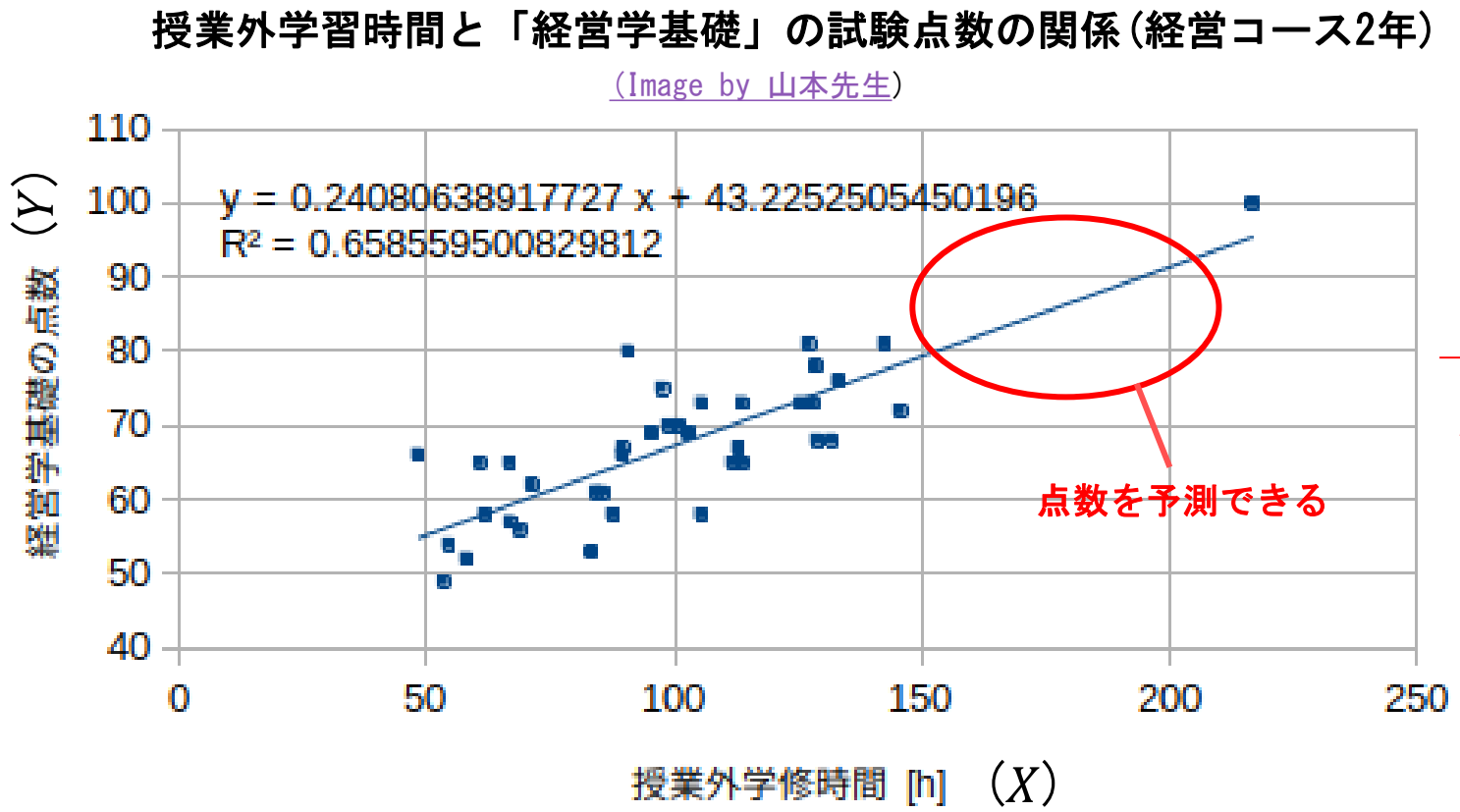
例：



機械学習の基礎

東北公益文科大学

例：



（線形回帰による）モデル：

$h(x) = 0.24x + 43.23$

モデルの「パラメーター」

機械学習の基礎

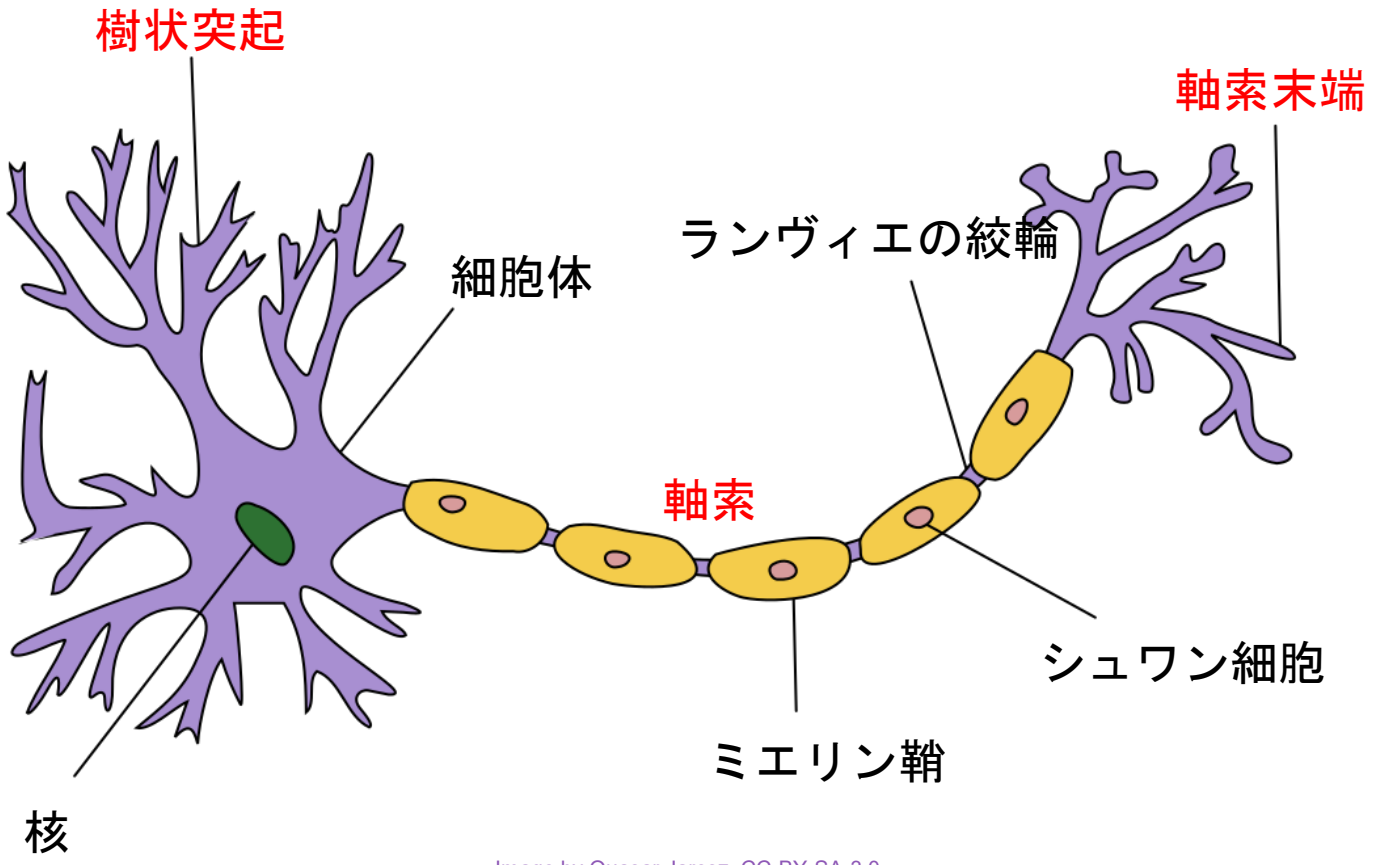
- h関数の妥当性を評価するためには学習に使用されない「**評価データ**」が必要
- XとYの一部（1~2割程度）を学習に使わず評価のために置いておくのが一般的である

“A few samples from the MNIST test dataset” by Josef Steppan (CC BY-SA 4.0)

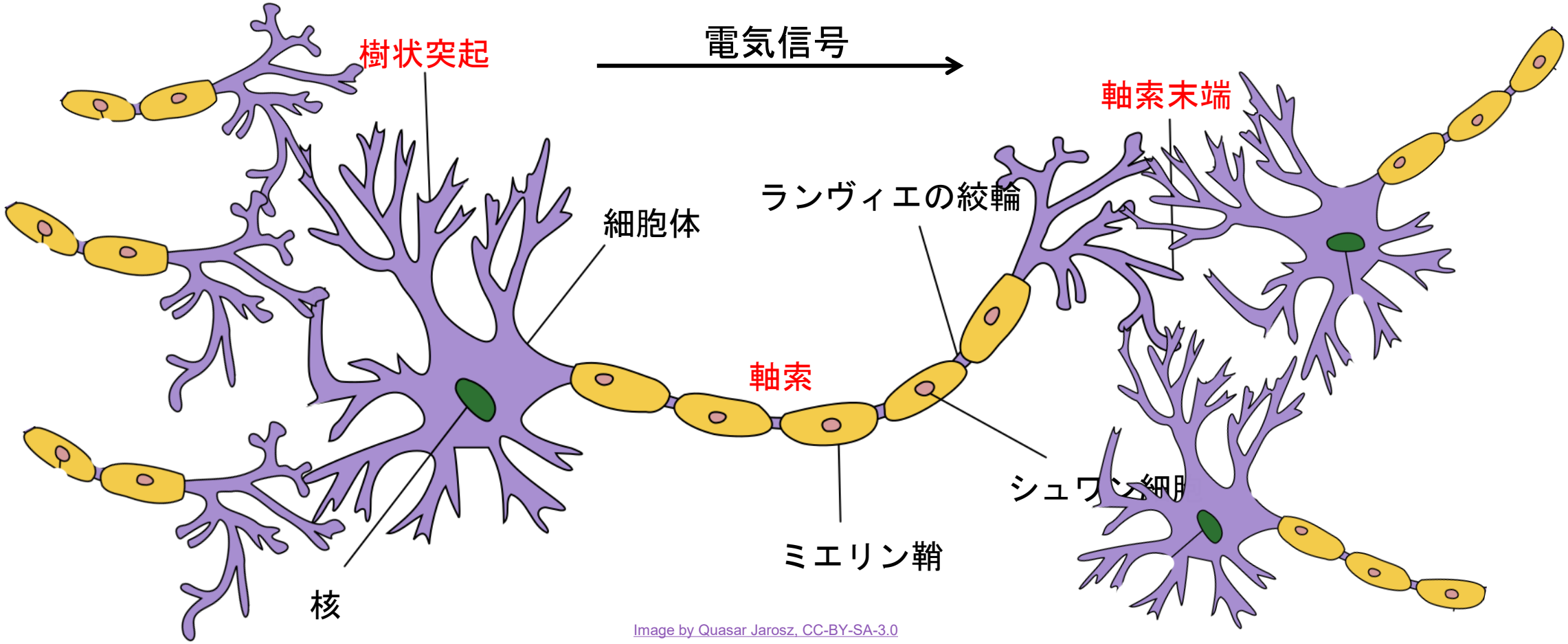
東北公益文科大学
Tohoku University of Community Service and Science

II
人工ニューロン
（パーセプトロン）

ニューロン（神経細胞）



ニューロン（神経細胞）



人工ニューロン（パーセプトロン）

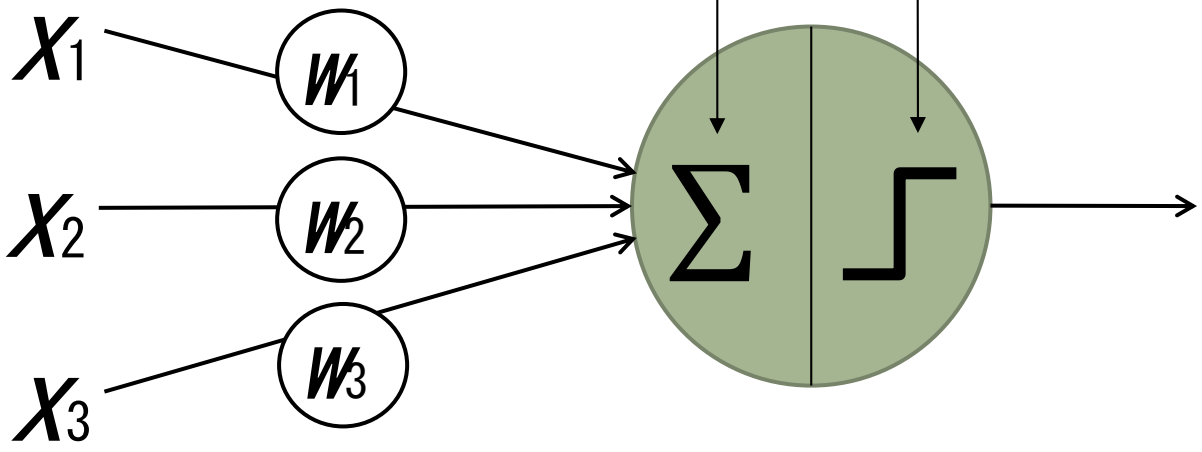
東北公益文科大学

加重和： $z = \vec{x} \cdot \vec{w} = x_1 * w_1 + x_2 * w_2 + x_3 * w_3$

入力信号：

重み（パラメーター）：

活性化関数（ここでは「ステップ関数」）



予測値（0または1）

ステップ関数の定義：

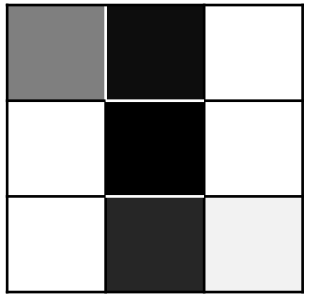
$$step(z) = \begin{cases} 0 \; if \; z < 0 \\ 1 \; if \; z \geq 0 \end{cases}$$

Frank Rosenblatt (1958). “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”. *Psychological Review*.

人工ニューロン（パーセプトロン）

東北公益文科大学

画像（３ｘ３画素）：



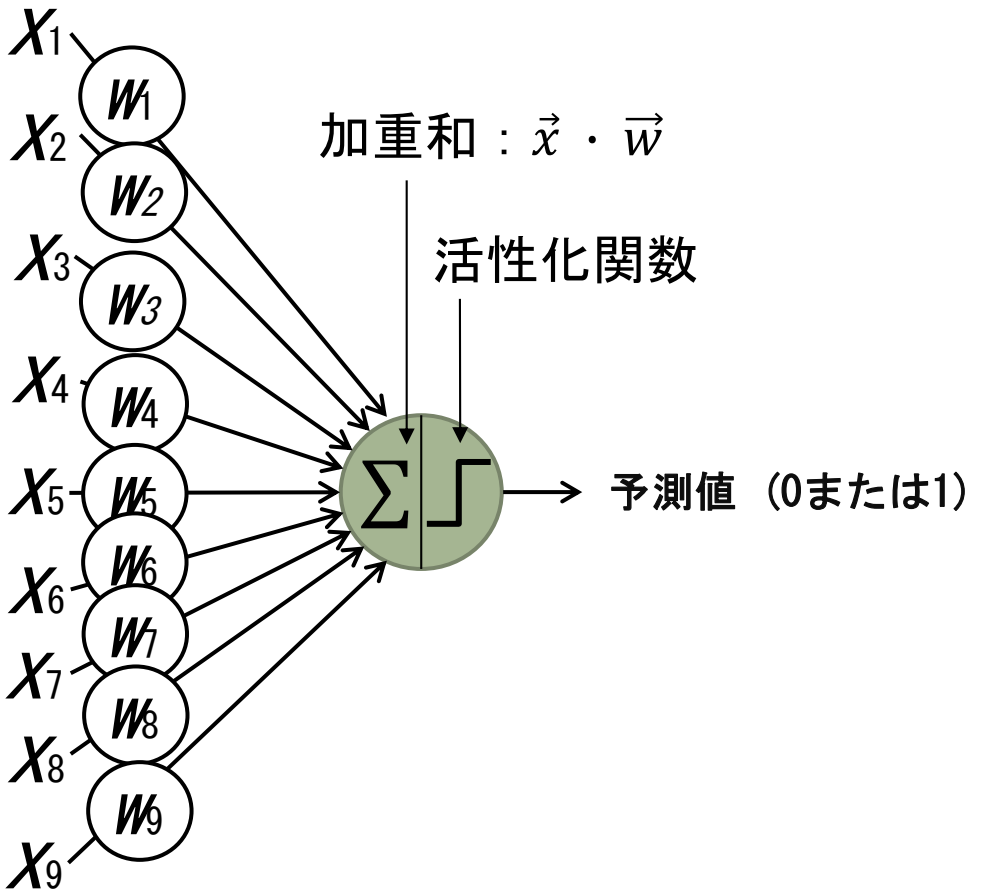
パーセプトロンの
入力値（画素値）：

$x_1 = 125$	$x_2 = 247$	$x_3 = 0$
$x_4 = 0$	$x_5 = 255$	$x_6 = 0$
$x_7 = 0$	$x_8 = 221$	$x_9 = 28$

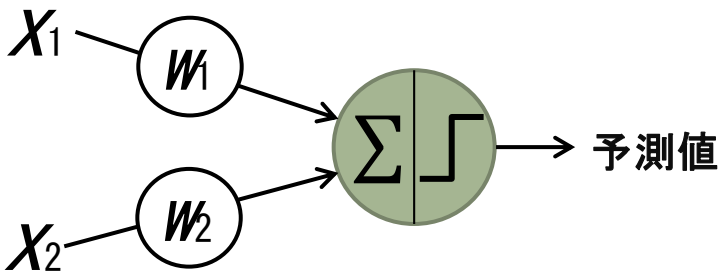
真っ白　→　0

真っ黒　→　255

灰色　→　1~254

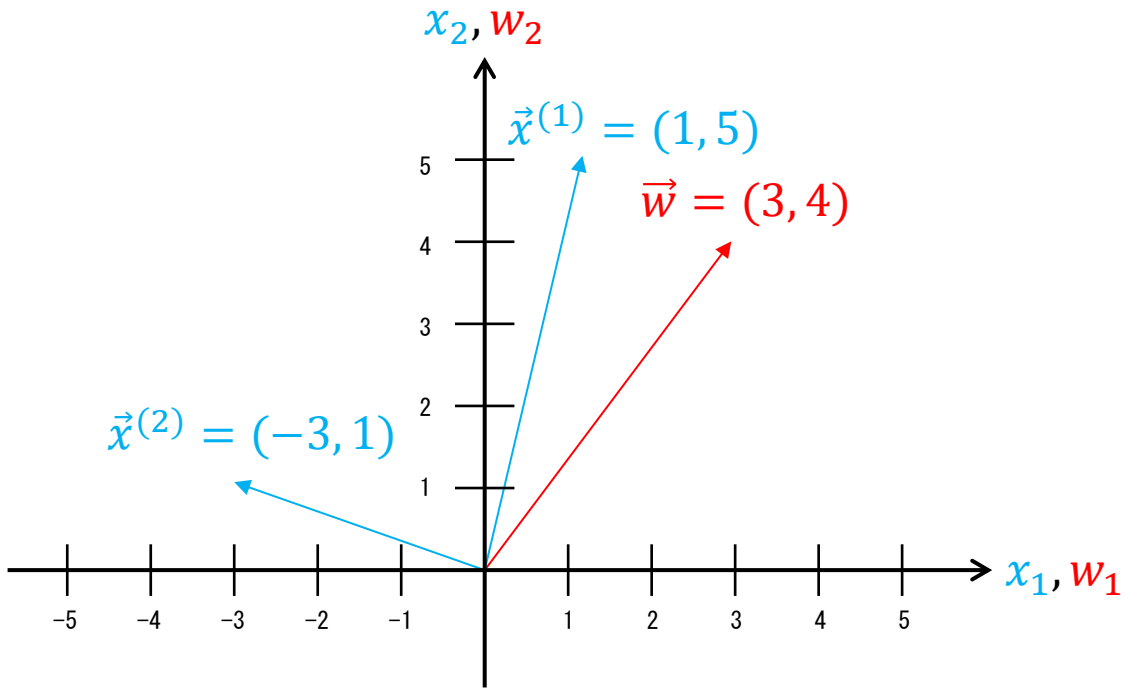


人工ニューロン（パーセプトロン）



$$z = \vec{x} \cdot \vec{w}$$

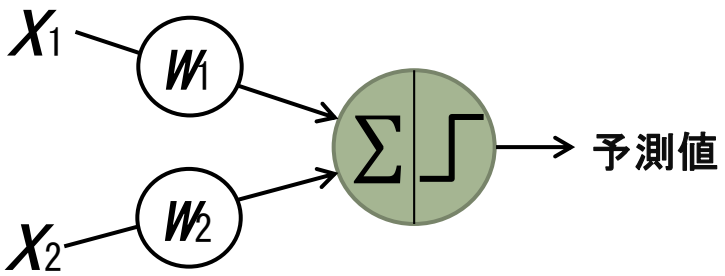
$$予測値 = step(z) = \begin{cases} 0 \; if \; z < 0 \\ 1 \; if \; z \geq 0 \end{cases}$$



$\vec{x}^{(1)} \cdot \vec{w} = 23$ → 予測値：1

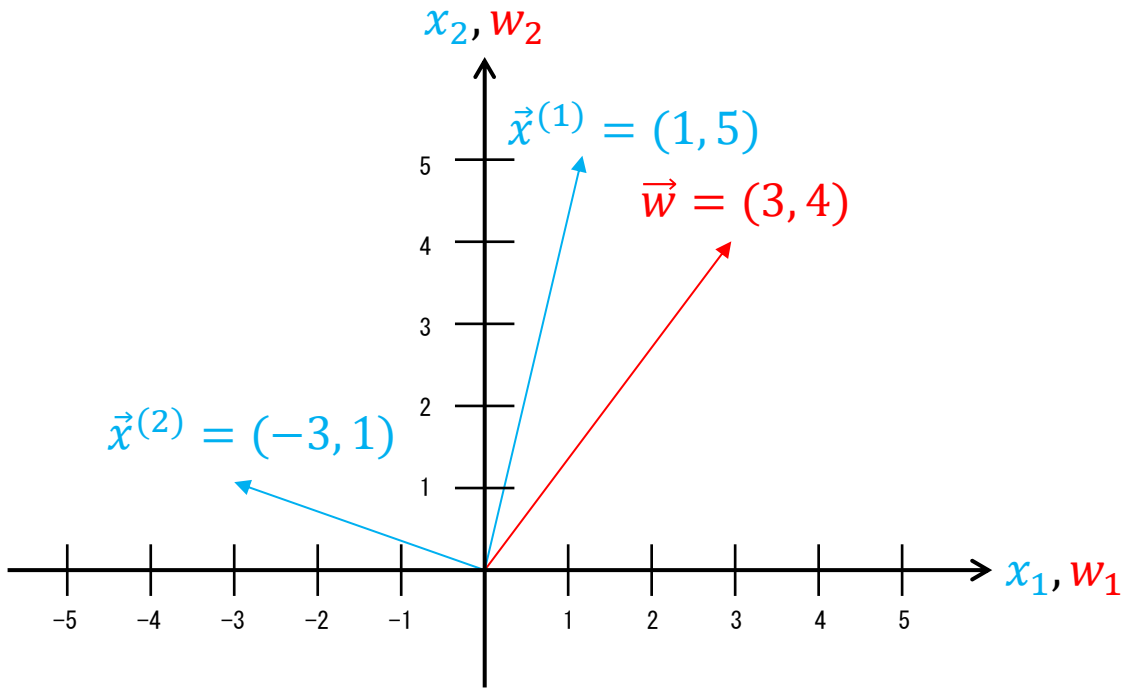
$\vec{x}^{(2)} \cdot \vec{w} = -5$ → 予測値：0

人工ニューロン（パーセプトロン）



$$z = \vec{x} \cdot \vec{w}$$

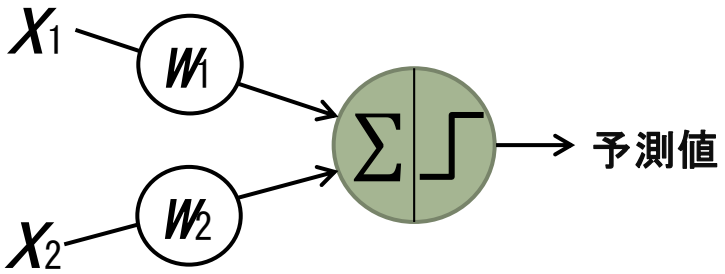
$$予測値 = step(z) = \begin{cases} 0 \; if \; z < 0 \\ 1 \; if \; z \geq 0 \end{cases}$$



$\vec{x}^{(1)} \cdot \vec{w} = 23$ → 予測値：1

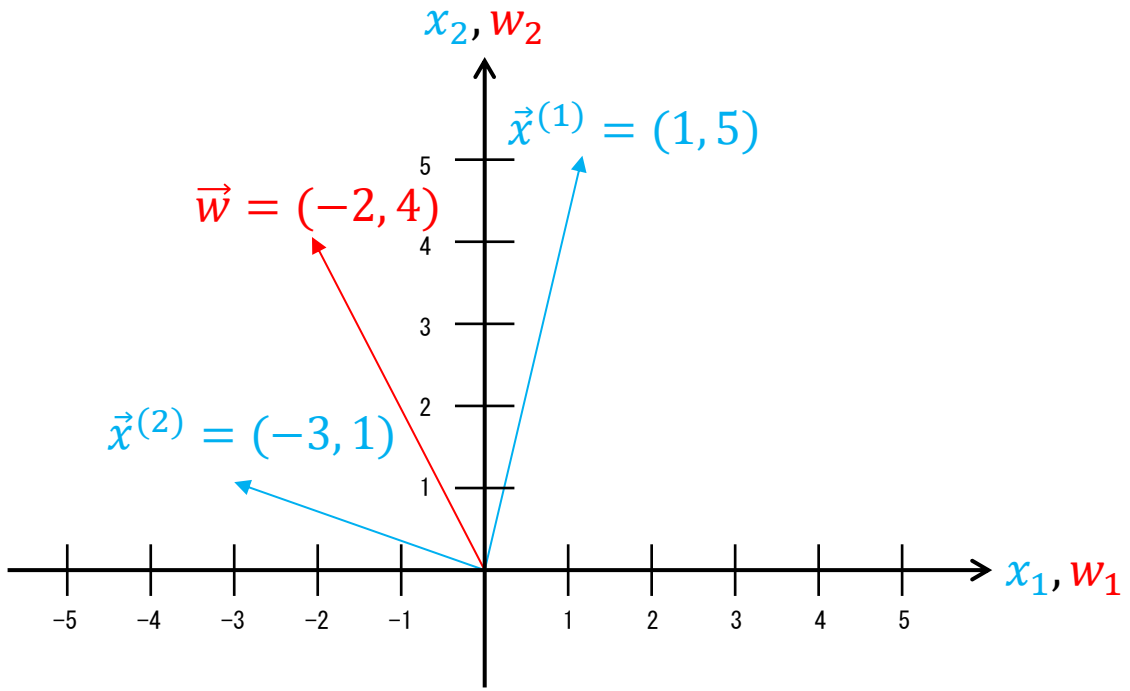
$\vec{x}^{(2)} \cdot \vec{w} = -5$ → 予測値：0 （正解：1）

人工ニューロン（パーセプトロン）



$$z = \vec{x} \cdot \vec{w}$$

$$予測値 = step(z) = \begin{cases} 0 \; if \; z < 0 \\ 1 \; if \; z \geq 0 \end{cases}$$



$\vec{x}^{(1)} \cdot \vec{w} = 18$ → 予測値：1

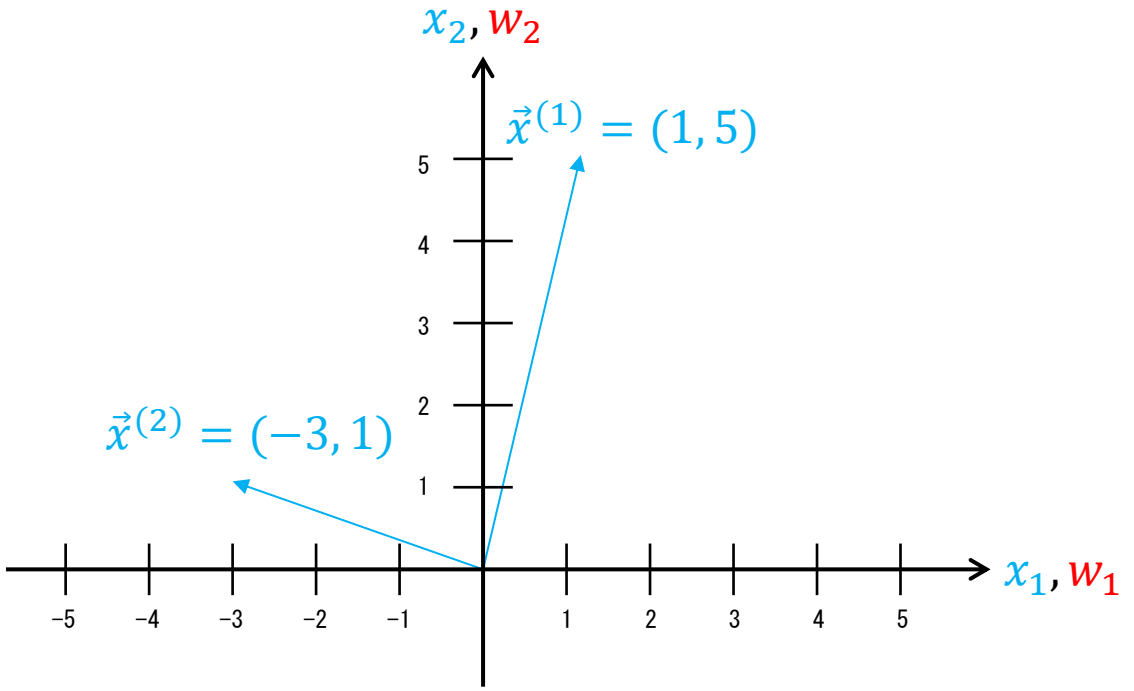
$\vec{x}^{(2)} \cdot \vec{w} = 10$ → 予測値：1 （正解：1）

人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す



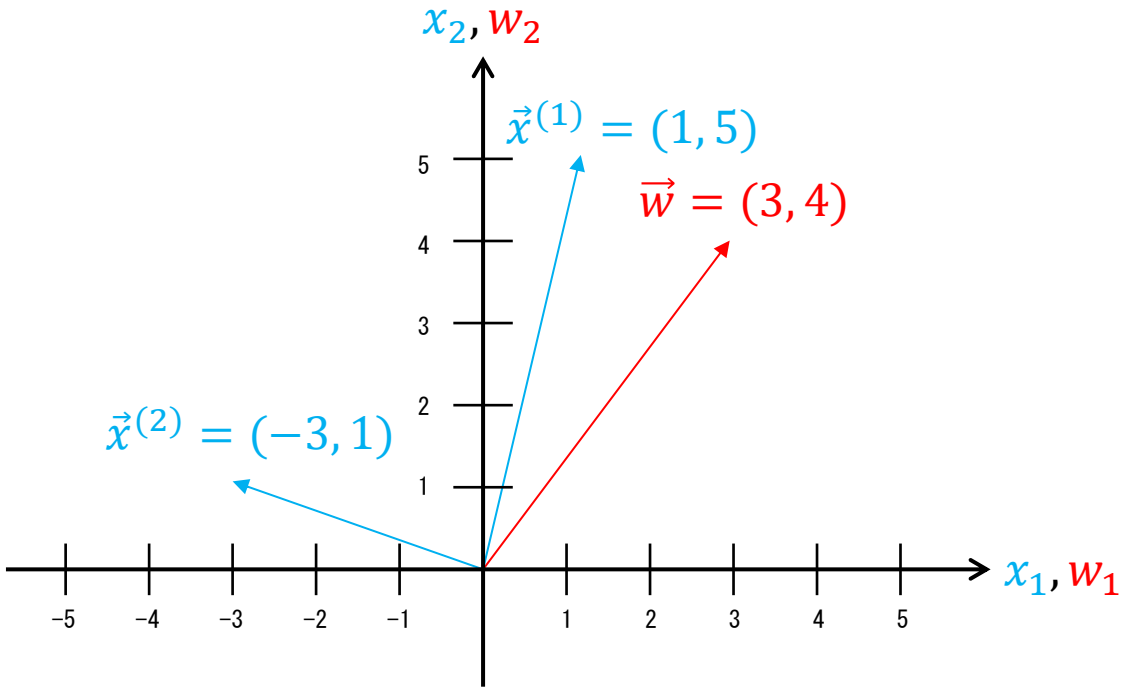
※0.1などの小さな固定値

人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す



初期化：$\vec{w} \leftarrow (3, 4)$

※0.1などの小さな固定値

人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

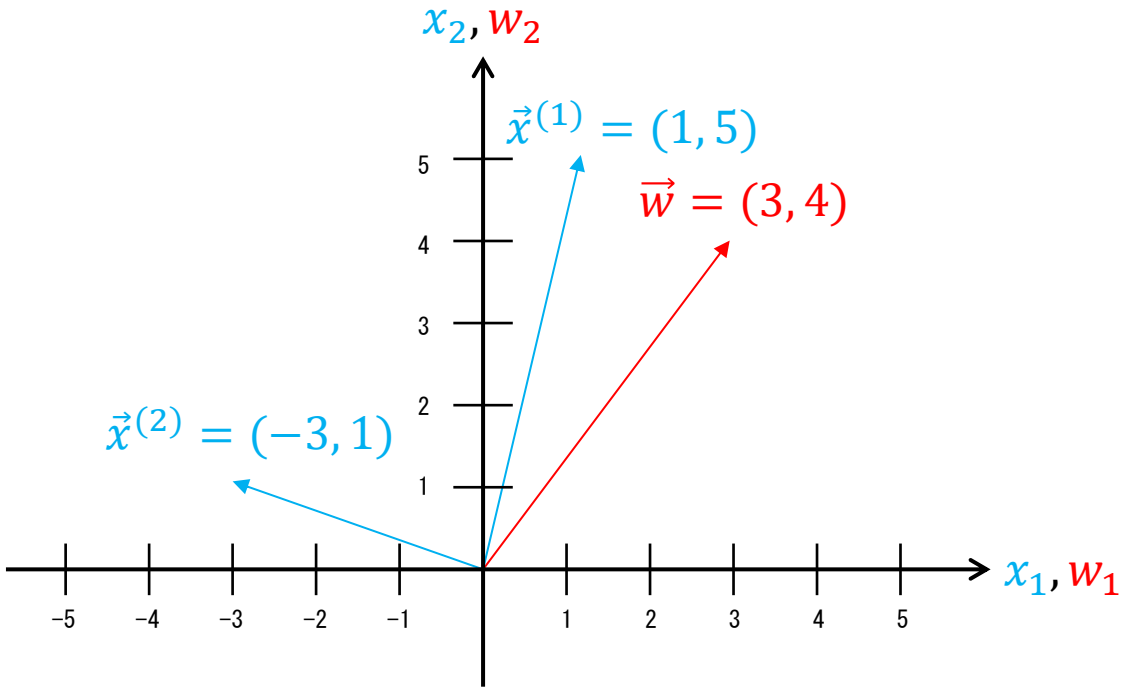
1．初期化：重み（$\vec{w}$）を乱数のベクトルにする

2．最適化：
各学習サンプルを使って次の処理を行う：
for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y

予測を行う：
予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
正解と予測値の差を求める：
誤差 $\leftarrow y^{(i)} -$ ***予測値***
重みを一つ一つ更新する：
for w_j in $\vec{w}$

$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※

3．必要に応じて最適化を繰り返す



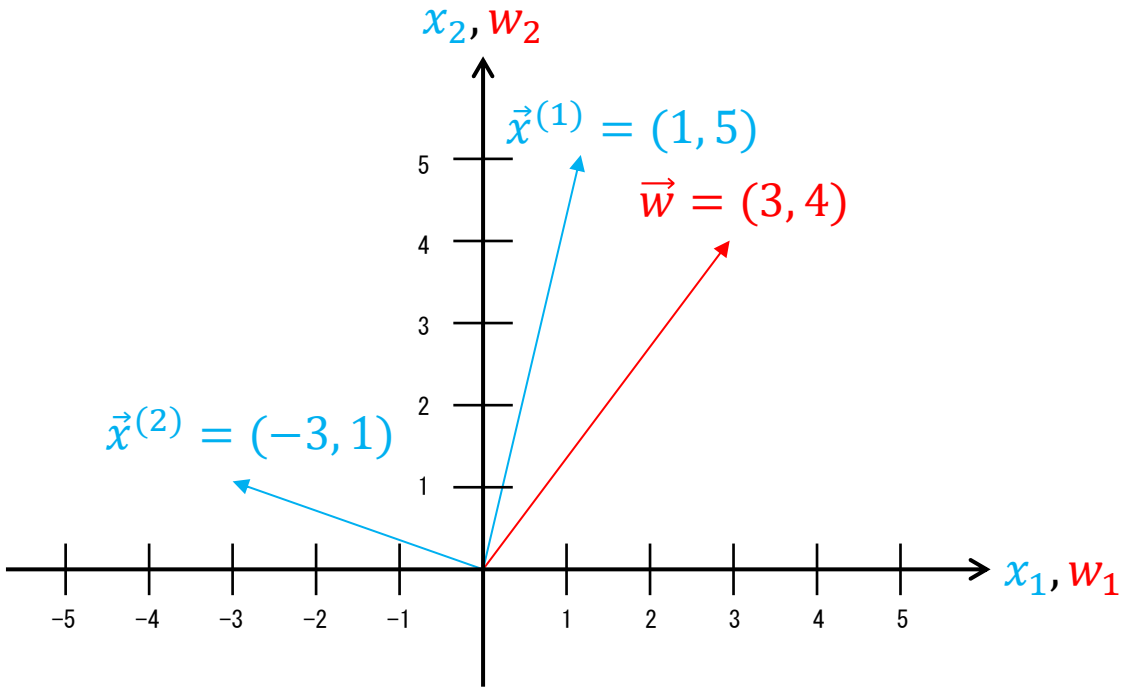
※0.1などの小さな固定値

人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す



予測：$step\left(\vec{x}^{(1)} \cdot \vec{w}\right) = step(23) = 1$

※0.1などの小さな固定値

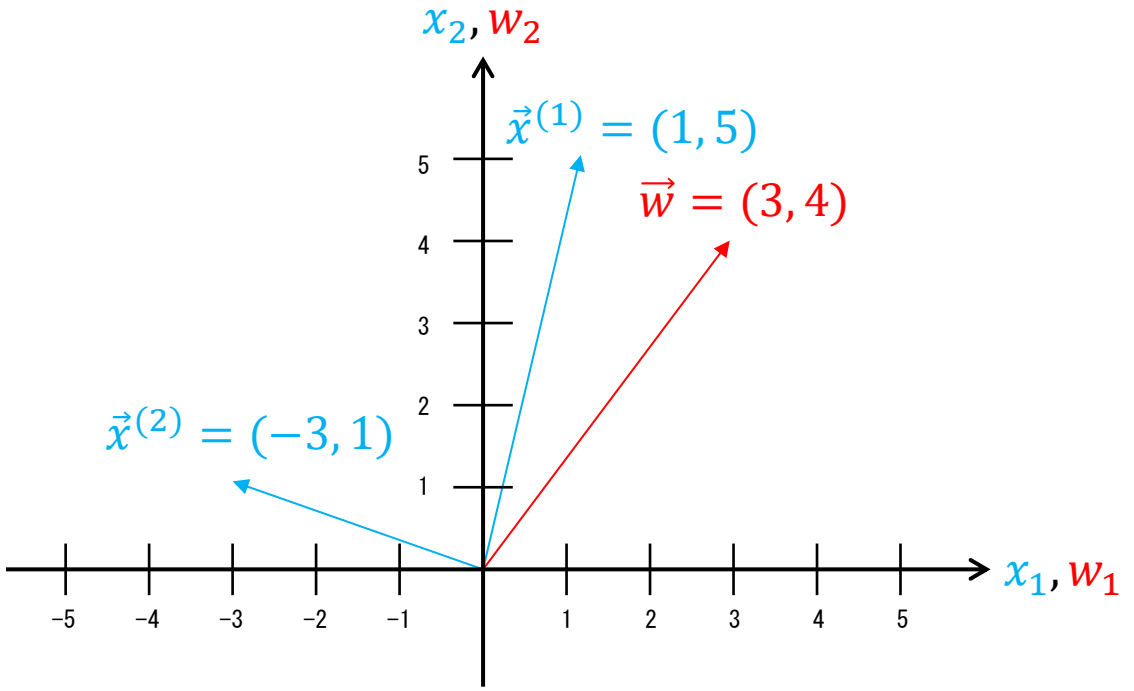
人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す

※0.1などの小さな固定値



予測：$step\left(\vec{x}^{(1)} \cdot \vec{w}\right) = step(23) = 1$

正解：$y^{(1)} = 1$

誤差：$1 - 1 = 0$

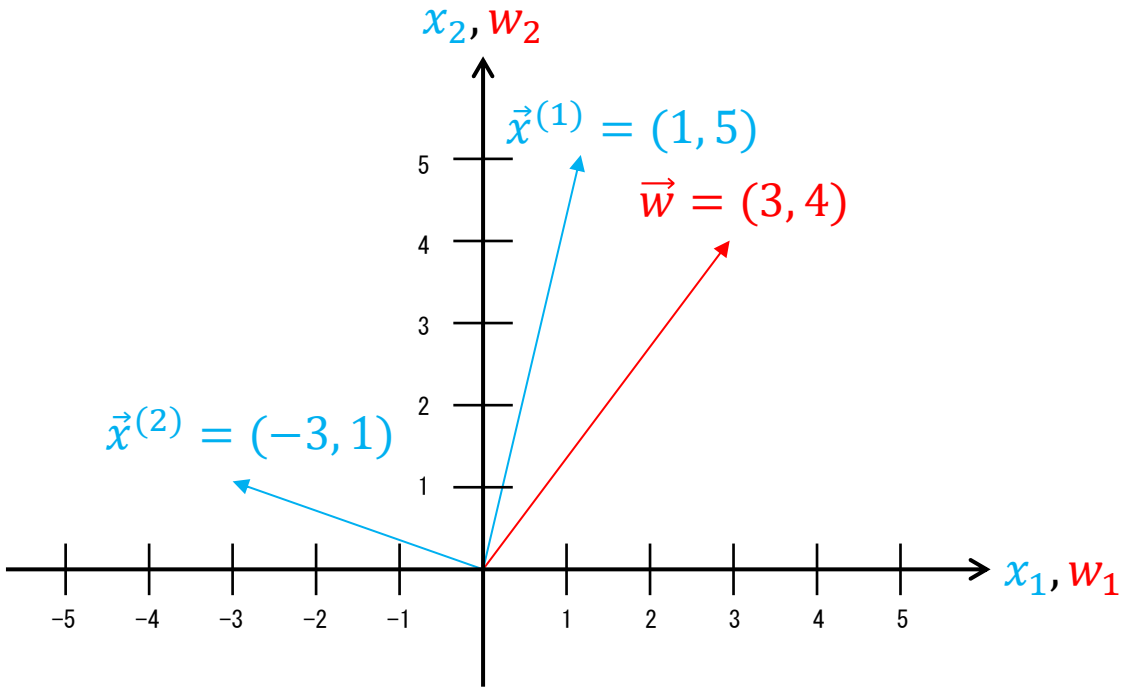
人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す

※0.1などの小さな固定値



重みの更新：

$w_1 \leftarrow w_1 + 0 \times 1 \times$ *学習率*（更新なし）

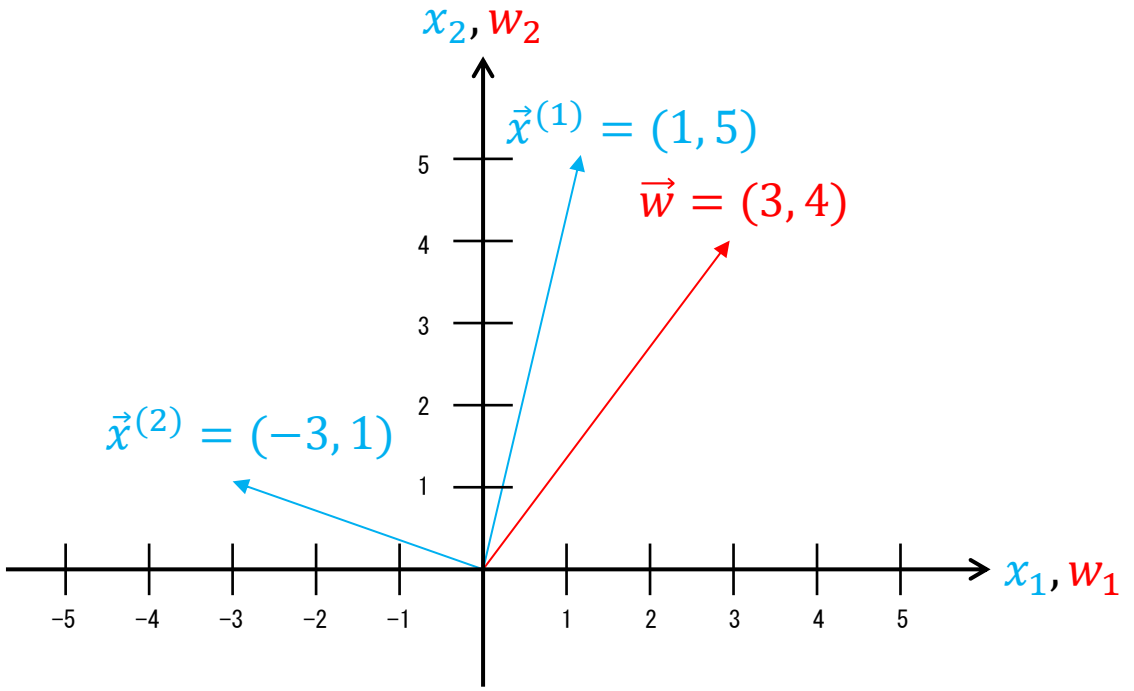
$w_2 \leftarrow w_2 + 0 \times 5 \times$ *学習率*（更新なし）

人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す



予測：$step(\vec{x}^{(2)} \cdot \vec{w}) = step(-5) = 0$

※0.1などの小さな固定値

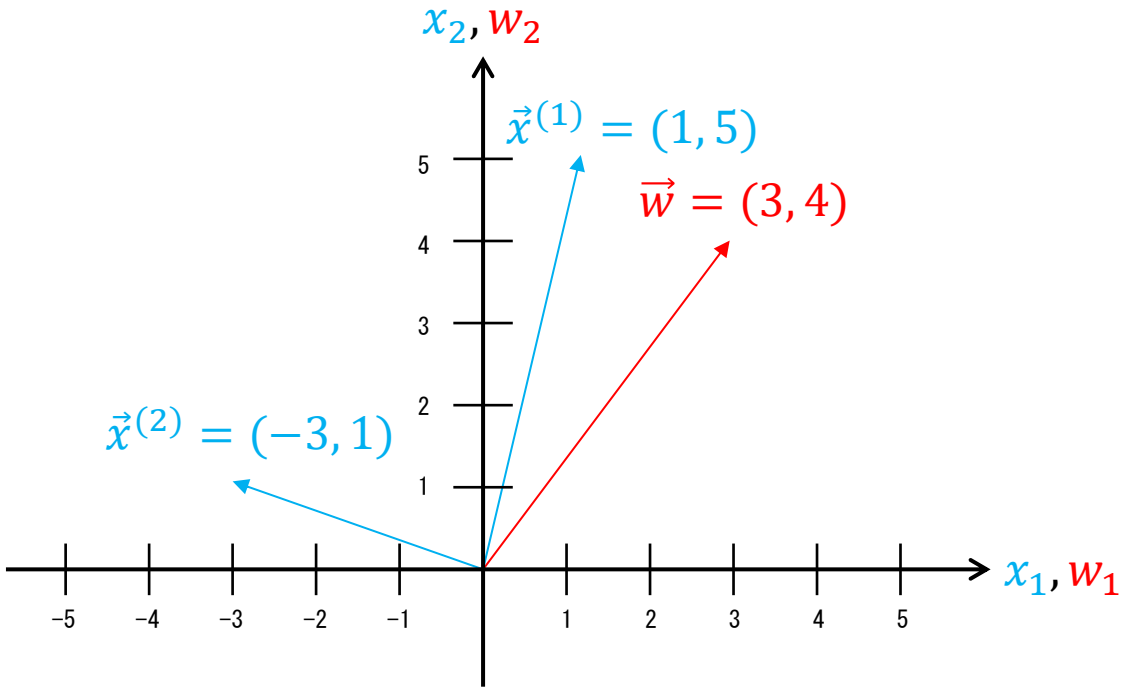
人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す

※0.1などの小さな固定値



予測：$step\left(\vec{x}^{(2)} \cdot \vec{w}\right) = step(-5) = 0$

正解：$y^{(2)} = 1$

誤差：$1 - 0 = 1$

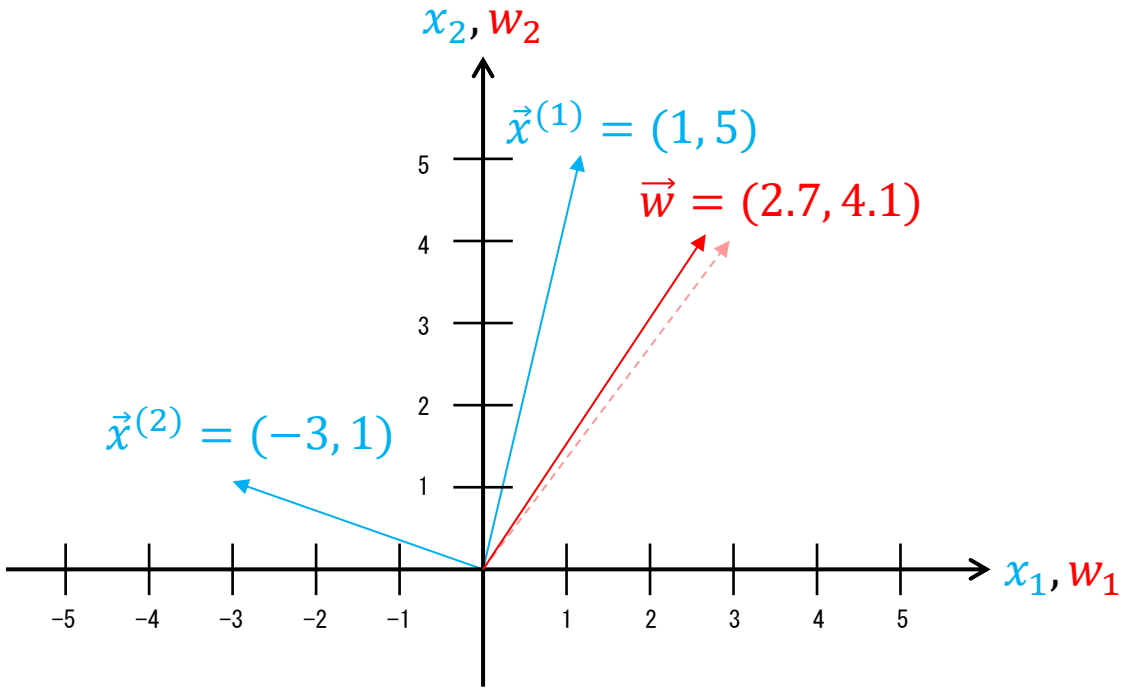
人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す

※0.1などの小さな固定値



重みの更新：

$w_1 \leftarrow w_1 + 1 \times -3 \times$ *学習率*$(0.1) = 3 + (-0.3)$

$w_2 \leftarrow w_2 + 1 \times 1 \times$ *学習率*$(0.1) = 4 + 0.1$

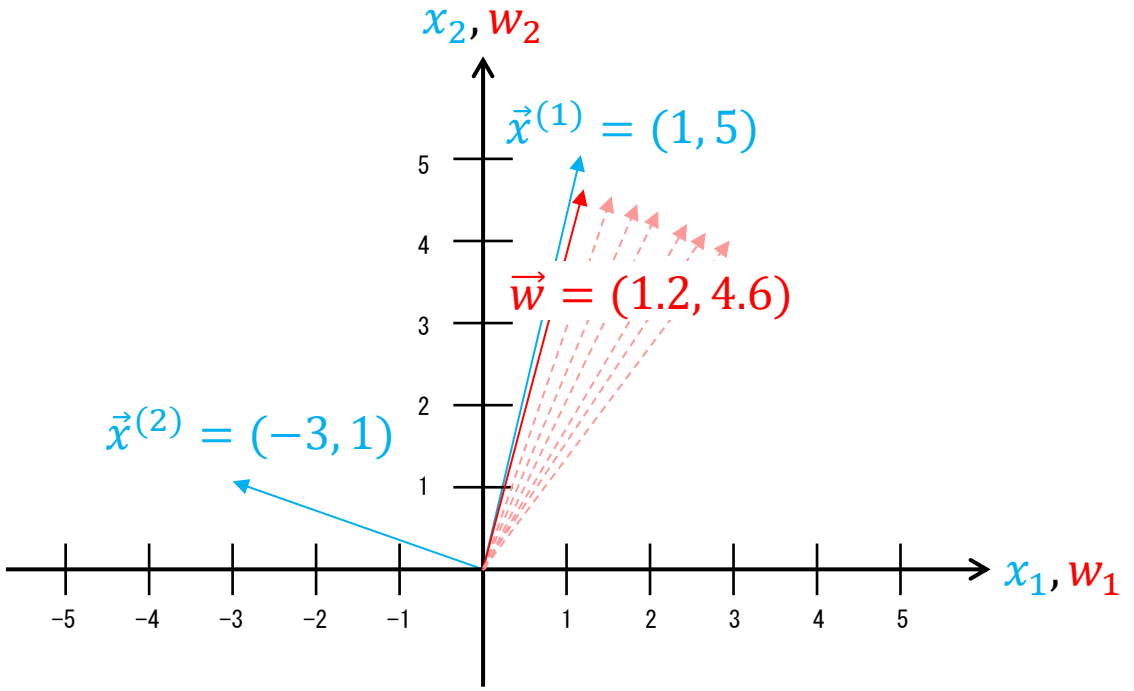
人工ニューロン（パーセプトロン）

東北公益文科大学

学習の手順：

1．初期化：重み（$\vec{w}$）を乱数のベクトルにする
2．最適化：
　各学習サンプルを使って次の処理を行う：
　for $\vec{x}^{(i)}$, $y^{(i)}$ in X, Y
　　予測を行う：
　　予測値 $\leftarrow step(\vec{x}^{(i)} \cdot \vec{w})$
　　正解と予測値の差を求める：
　　誤差 $\leftarrow y^{(i)} -$ ***予測値***
　　重みを一つ一つ更新する：
　　for w_j in $\vec{w}$
　　$w_j \leftarrow w_j +$ ***誤差*** $\times x_j^{(i)} \times$ ***学習率***※
3．必要に応じて最適化を繰り返す

※0.1などの小さな固定値



$\vec{x}^{(1)} \cdot \vec{w} = 24.2$ → 予測値：1（正解：1）

$\vec{x}^{(2)} \cdot \vec{w} = 1$ → 予測値：1（正解：1）

人工ニューロン（パーセプトロン）

東北公益文科大学

パーセプトロンを用いて手書き数字認識システムを実装したPythonのソースコード:

Colab Notebook