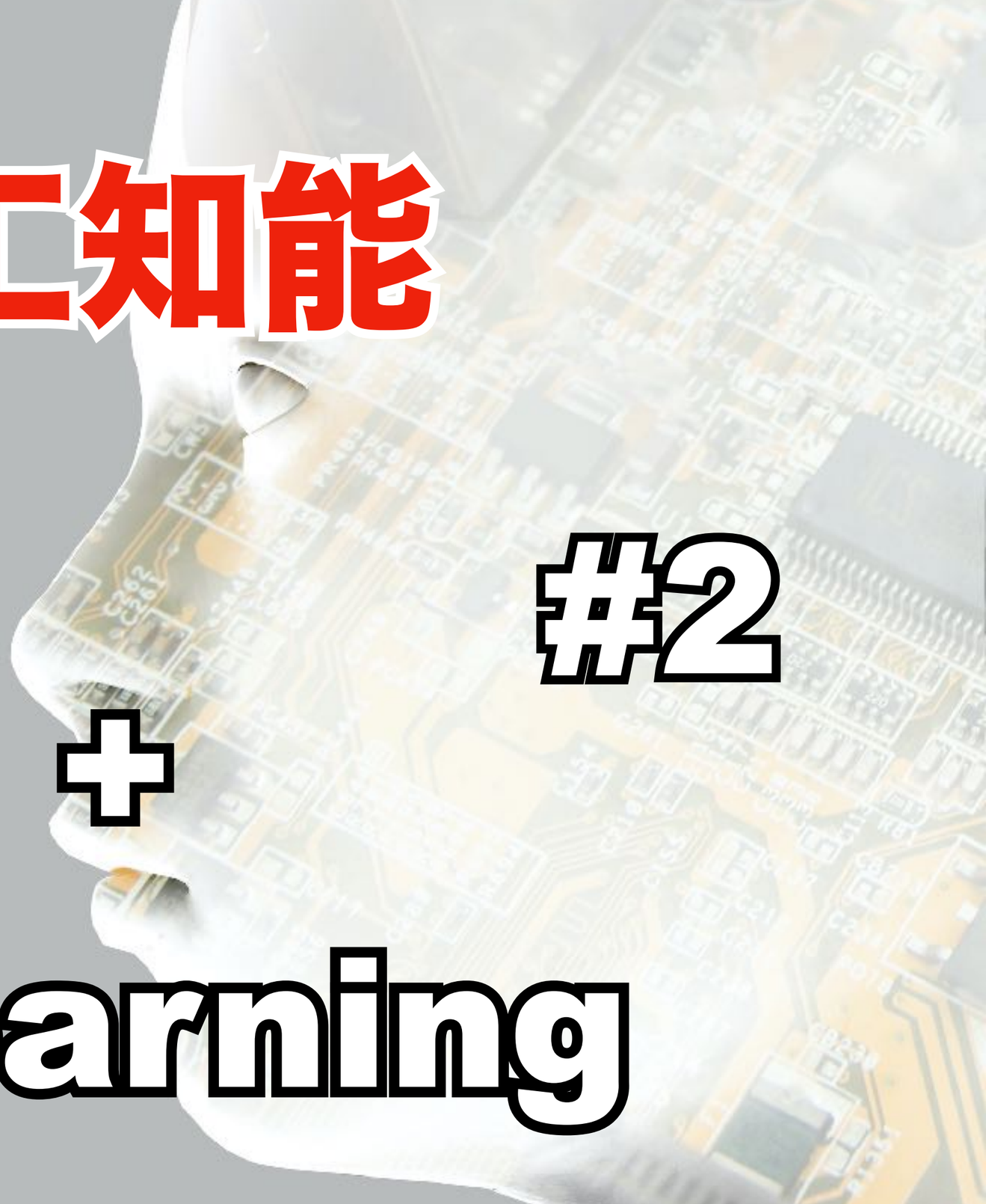
Live! 人工知能

ARTIFICIAL INTELLIGENCE



#2

PyTorch + Deep Learning

ARTIFICIAL INTELLIGENCE

第2講の概要

講座の内容

第1講. イントロダクション

第2講. PyTorchで実装する簡単なディープラーニング

第3講. PyTorchの様々な機能

第4講. 畳み込みニューラルネットワーク（CNN）

第5講. 再帰型ニューラルネットワーク（RNN）

第6講. AIアプリのデプロイ

今回の内容

1. 第2講の概要
2. 勾配降下法
3. 活性化関数と損失関数
4. 最適化アルゴリズム
5. 簡単なディープラーニングの実装
6. 演習
7. 質疑応答

教材の紹介

- Pythonの基礎
- 第2講の教材: **simple_dl.ipynb**
- 第2講の演習: **exercise.ipynb**

ハッシュタグ

#Live人工知能

演習の解答 -第1講-

https://github.com/yukinaga/lecture_pytorch/blob/master/lecture1/exercise.ipynb

ARTIFICIAL INTELLIGENCE

勾配降下法

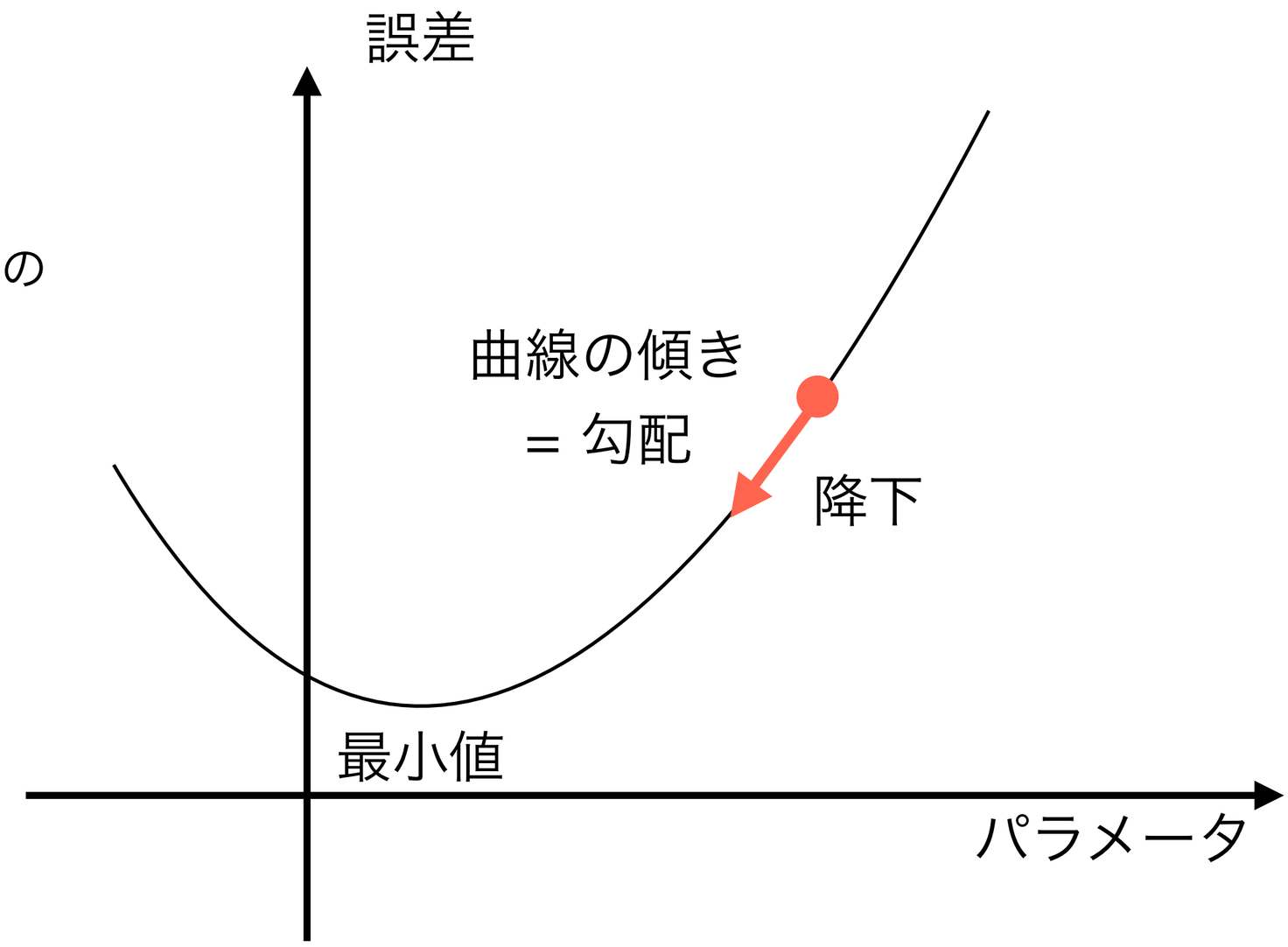
勾配降下法とは？

- 勾配降下法は、ニューラルネットワークの

  学習に使われる

- 出力と正解の間の誤差を、

  曲線を滑り落ちるようにして最小化する



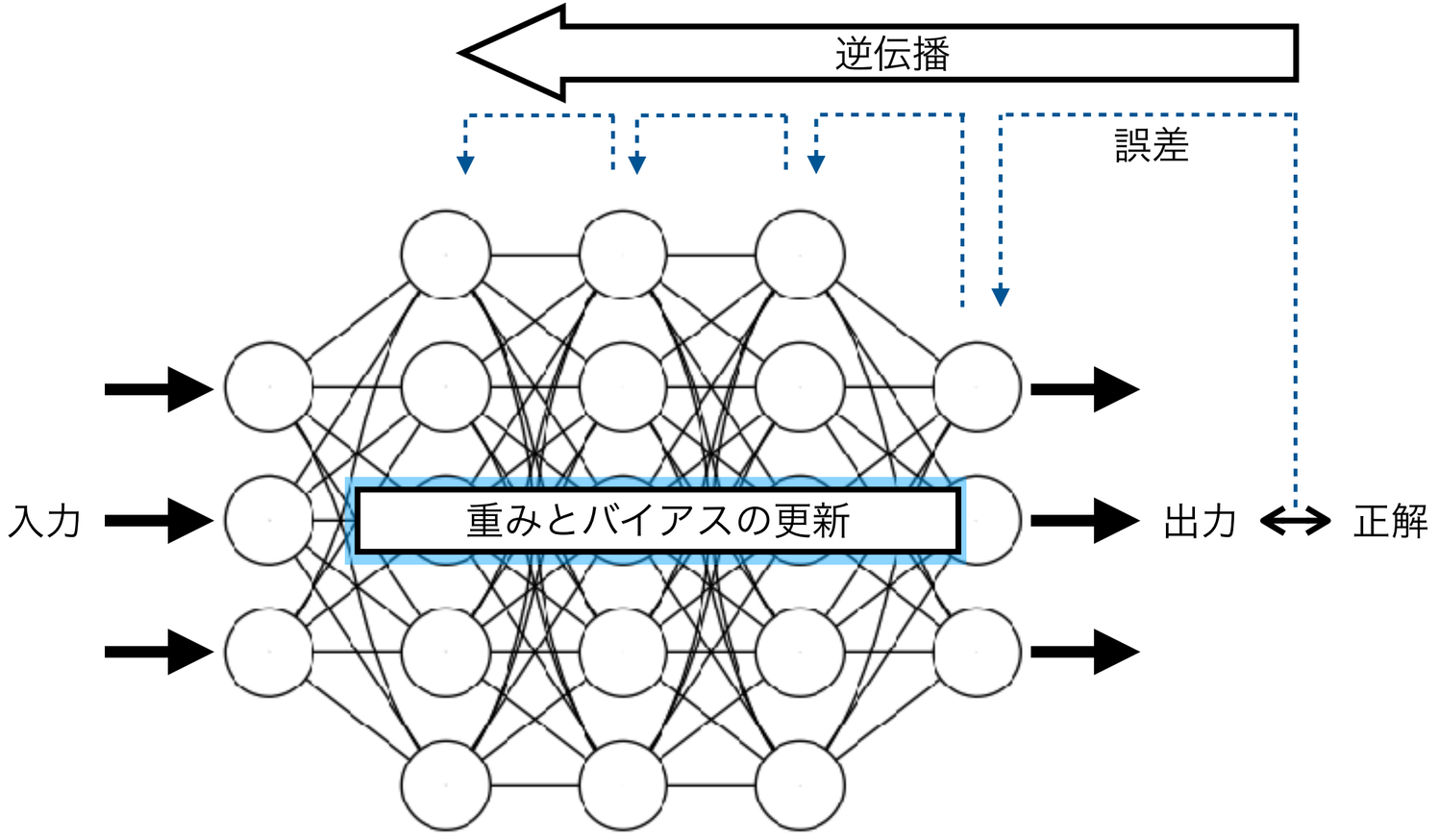
バックプロパゲーションによる学習

- ニューラルネットワークは、

  出力と正解の誤差が小さくなるよう

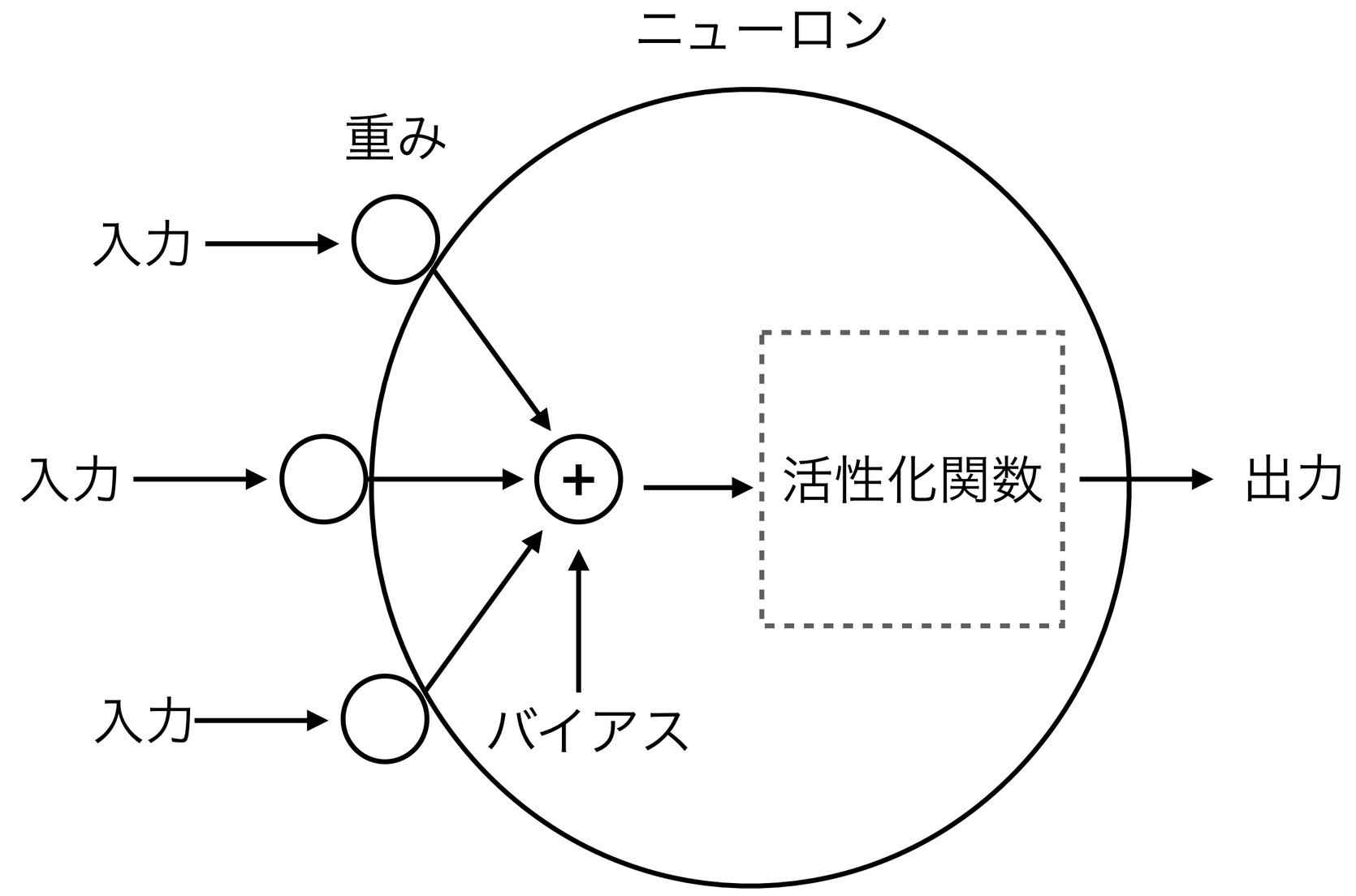
  に重みとバイアスを調整することで

  学習することができる



単一ニューロン

- ニューロンへの入力に**重み**をかけた値を統べて足し合わせ、**バイアス**を加える
- 上記の値を**活性化関数**で処理し出力とする
- 重みとバイアスがパラメータ



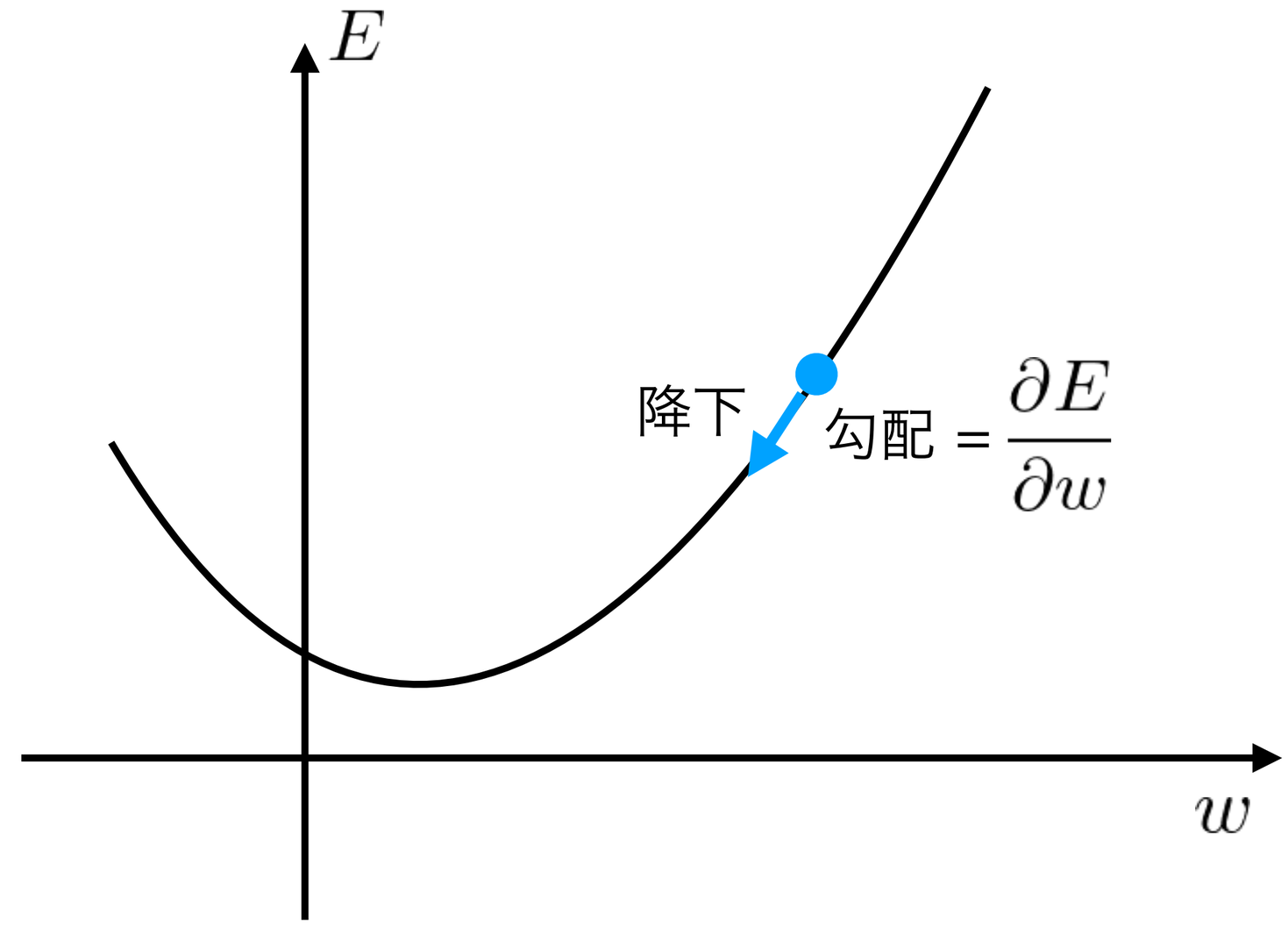
勾配降下法による誤差の最小化

$$w \leftarrow w - \eta \frac{\partial E}{\partial w}$$

w：重み

η：学習係数

E：誤差



ARTIFICIAL INTELLIGENCE

活性化関数と損失関数

活性化関数と損失関数

- **活性化関数**

  → ニューロンの興奮/抑制状態を決める関数

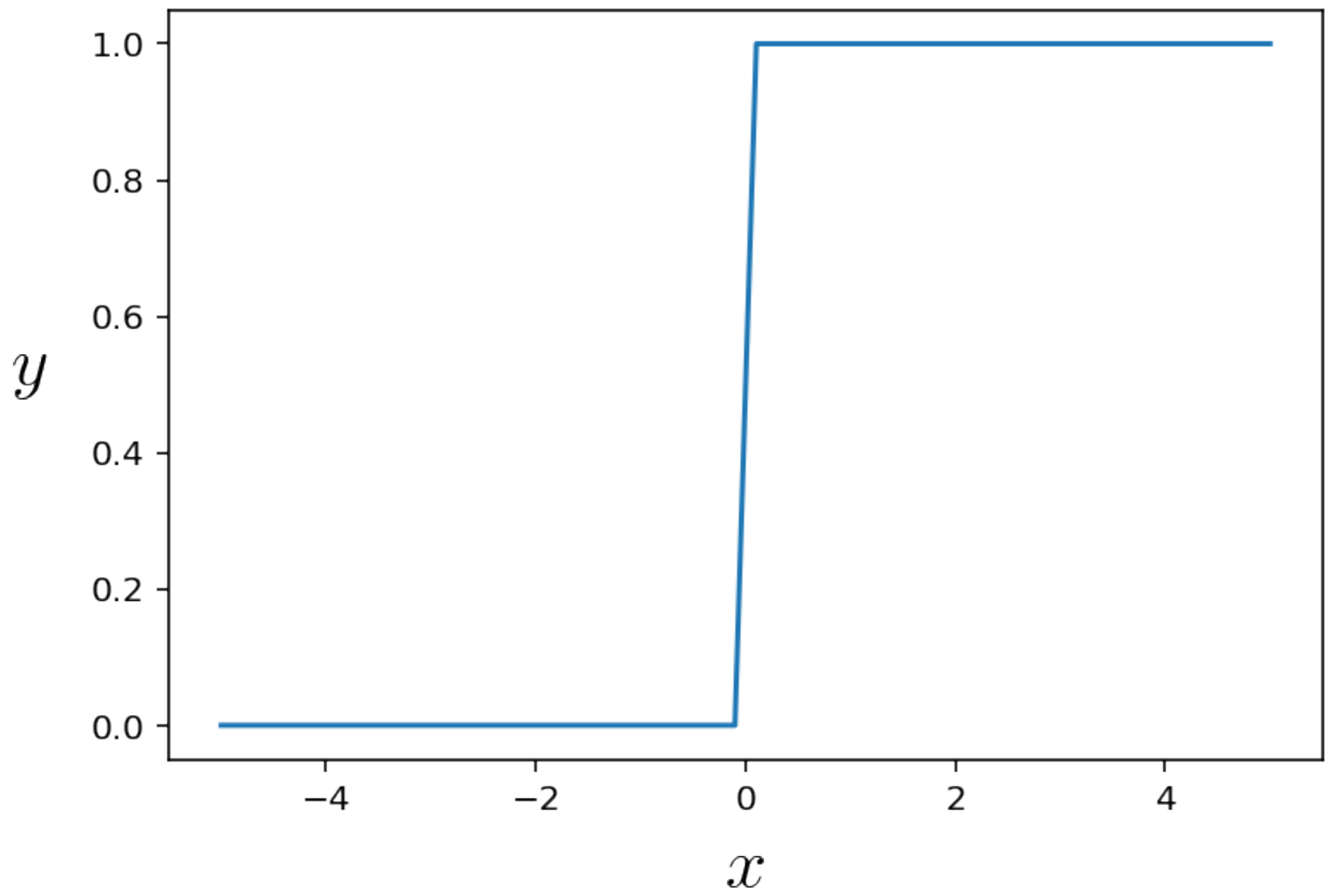
  → 関数への入力を、興奮/抑制状態を表す値に変換する

- **損失関数**（誤差関数）

  → 出力と正解の間の「誤差」を定義する関数

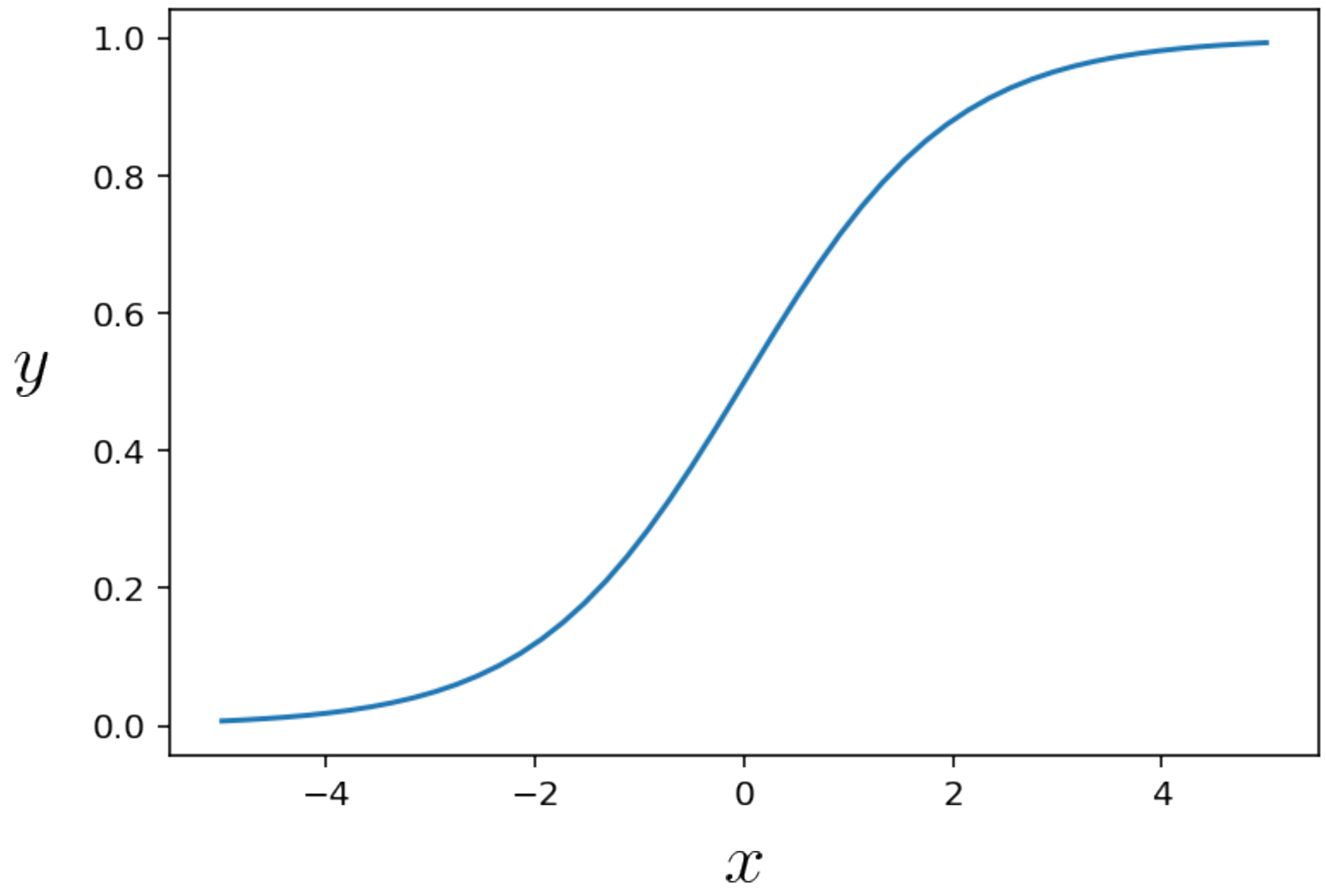
活性化関数: ステップ関数

$$y = \begin{cases} 0 & (x \leqq 0) \\ 1 & (x > 0) \end{cases}$$



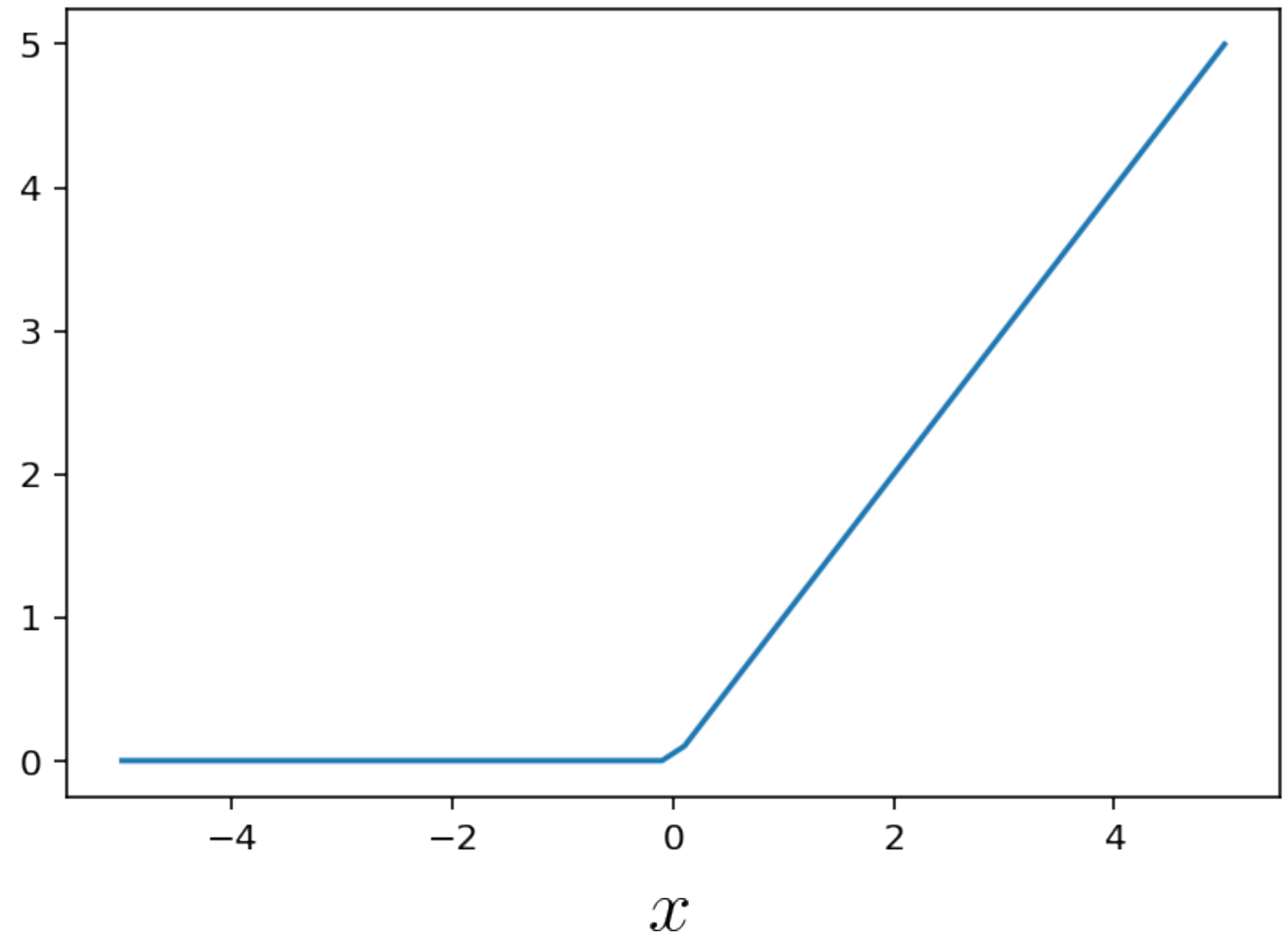
活性化関数: シグモイド関数

$$y = \frac{1}{1 + \exp(-x)}$$



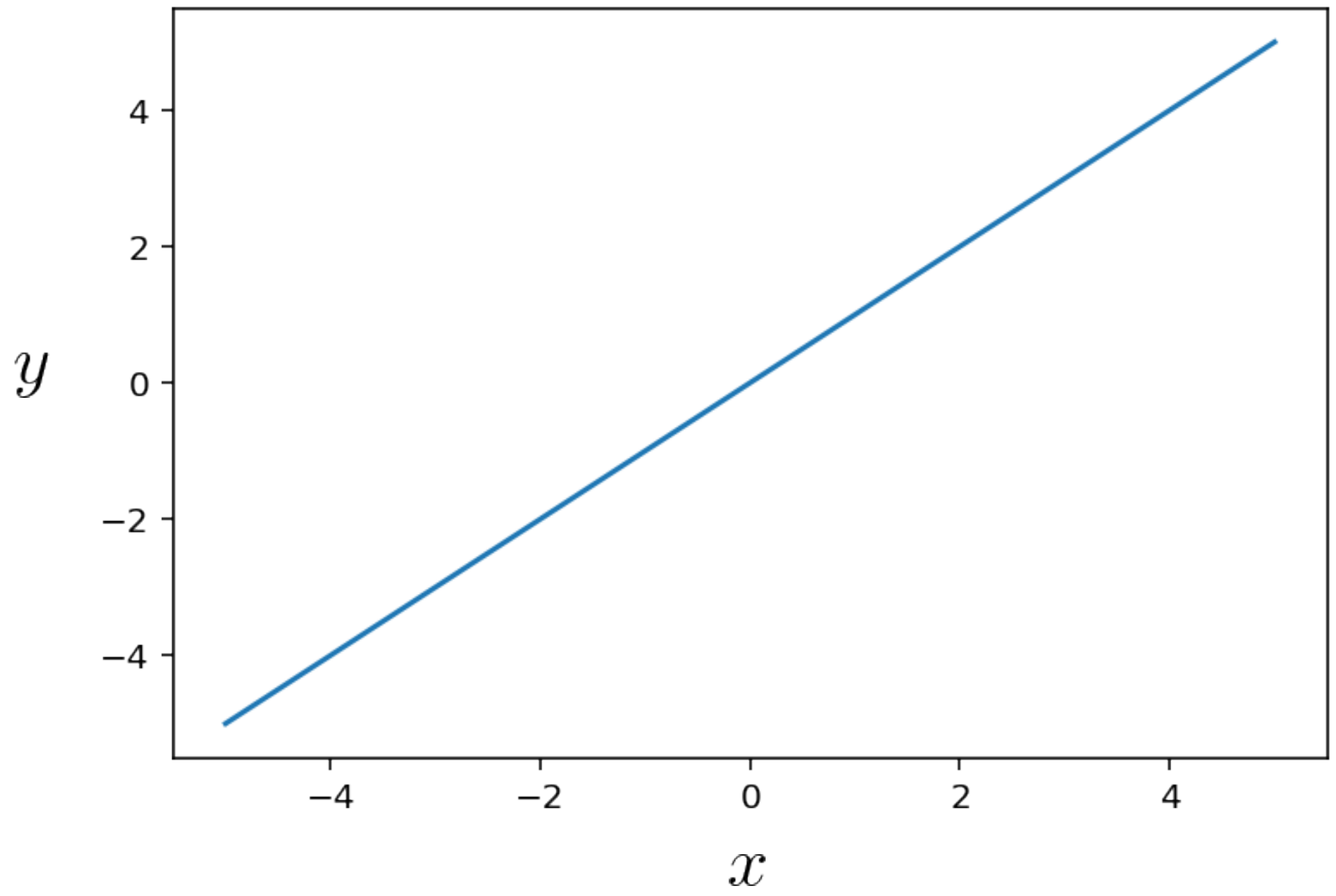
活性化関数: ReLU

$$y = \begin{cases} 0 & (x \leqq 0) \\ x & (x > 0) \end{cases}$$

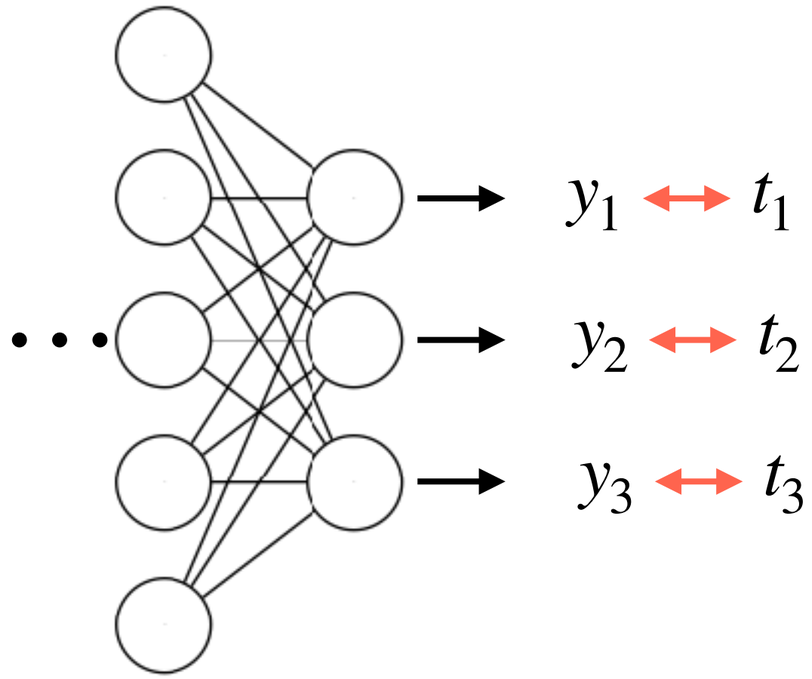


活性化関数: 恒等関数

$$y = x$$



損失関数: 二乗和誤差

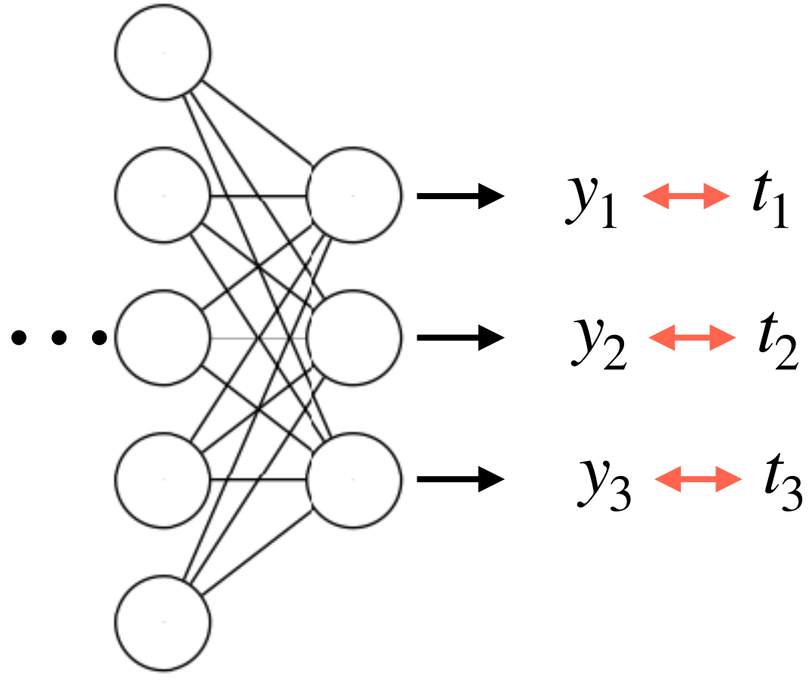


$$E = \frac{1}{2}\sum_{k=1}^{n}(y_k - t_k)^2$$

n：出力層のニューロン数

損失関数: 交差エントロピー誤差

- 多クラス分類問題でよく使われる損失関数
- 多クラス分類問題では、正解は1つだけ1で残りは0



$$E = -\log(y) = -\log\left(\frac{\exp(x)}{\sum_{k=1}^{n}\exp(x_k)}\right)$$

x：活性化関数への入力

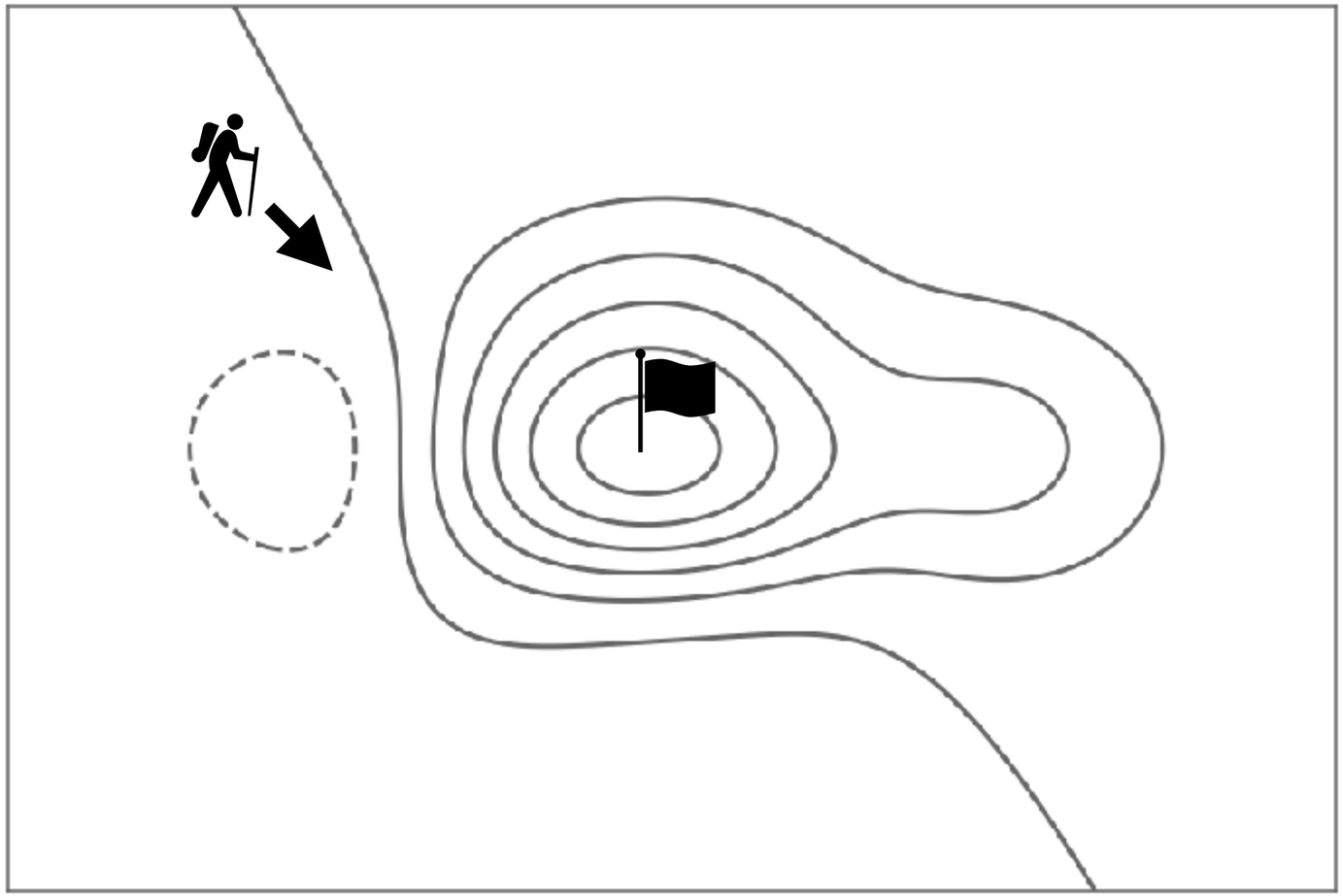
n：出力層のニューロン数

ARTIFICIAL INTELLIGENCE

最適化アルゴリズム

最適化アルゴリズムとは？

- パラメータを調整し誤差を最小化するためのアルゴリズム
- 例えるなら夜の山岳地帯で谷底を目指すための



SGD（確率的勾配降下法）

$$w \leftarrow w - \eta \frac{\partial E}{\partial w}$$

w：重み

η：学習係数

E：誤差

AdaGrad

- 学習が進むとともにパラメータの更新量が低下
- 最初は広い領域で探索し、次第に探索範囲を絞る

$$h \leftarrow h + (\frac{\partial E}{\partial w})^2$$

$$w \leftarrow w - \eta \frac{1}{\sqrt{h}} \frac{\partial E}{\partial w}$$

w：重み

η：学習係数

E：誤差

Adam

- 学習の進行度合いや過去の履歴に基づき、更新量が調整される

$$m_0 = v_0 = 0$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\frac{\partial E}{\partial w}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\frac{\partial E}{\partial w})^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$w \leftarrow w - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

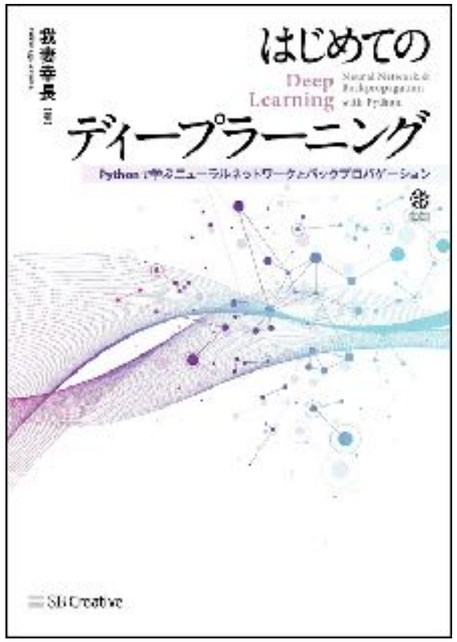
$\beta_1, \beta_2, \eta, \epsilon$：定数

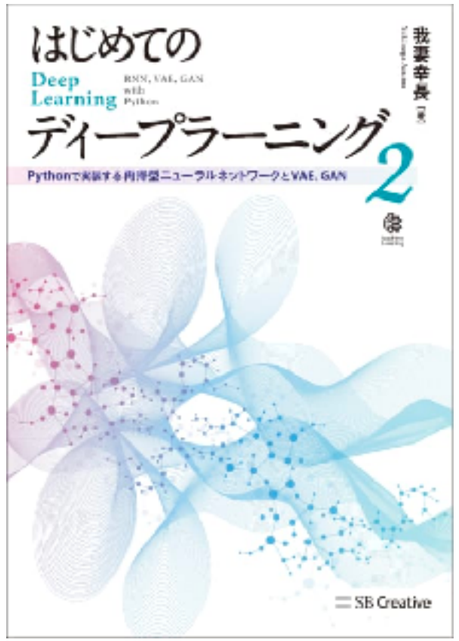
w：重み

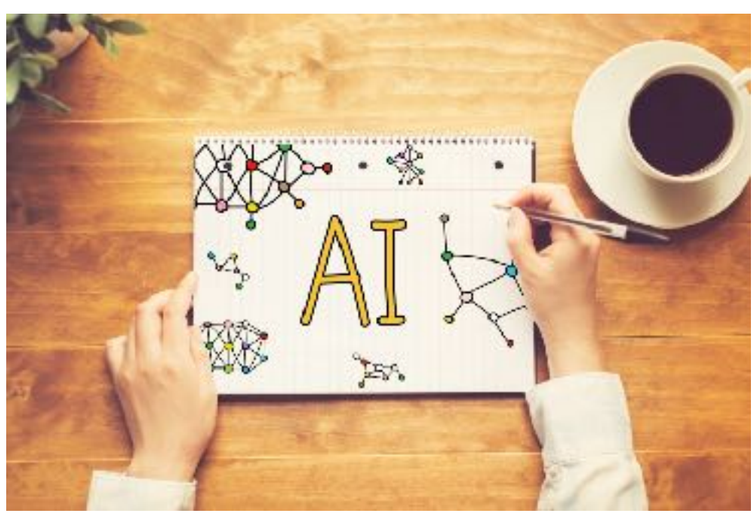
η：学習係数

E：誤差

より詳しく学びたい方へ...







Udemy講座

ディープラーニング：Pythonで
ゼロから構築し学ぶ人工知能
（AI）と深層学習の原理

ARTIFICIAL INTELLIGENCE

簡単なディープラーニングの実装

ARTIFICIAL INTELLIGENCE

演習

次回

第1講. イントロダクション

第2講. PyTorchで実装する簡単なディープラーニング

第3講. PyTorchの様々な機能

第4講. 畳み込みニューラルネットワーク（CNN）

第5講. 再帰型ニューラルネットワーク（RNN）

第6講. AIアプリのデプロイ

ARTIFICIAL INTELLIGENCE

質疑応答