# 機械学習エンジニアコース
# Sprint

－ DNN_深層学習 －

DIVE INTO CODE

# 今回のモチベーション

## 目的はなにか

1. **理解する**
   **スクラッチを通してニューラルネットワークの発展的内容を理解する。**

# このスライドは?

ここでは、深層学習の基本的な知識を学びましょう

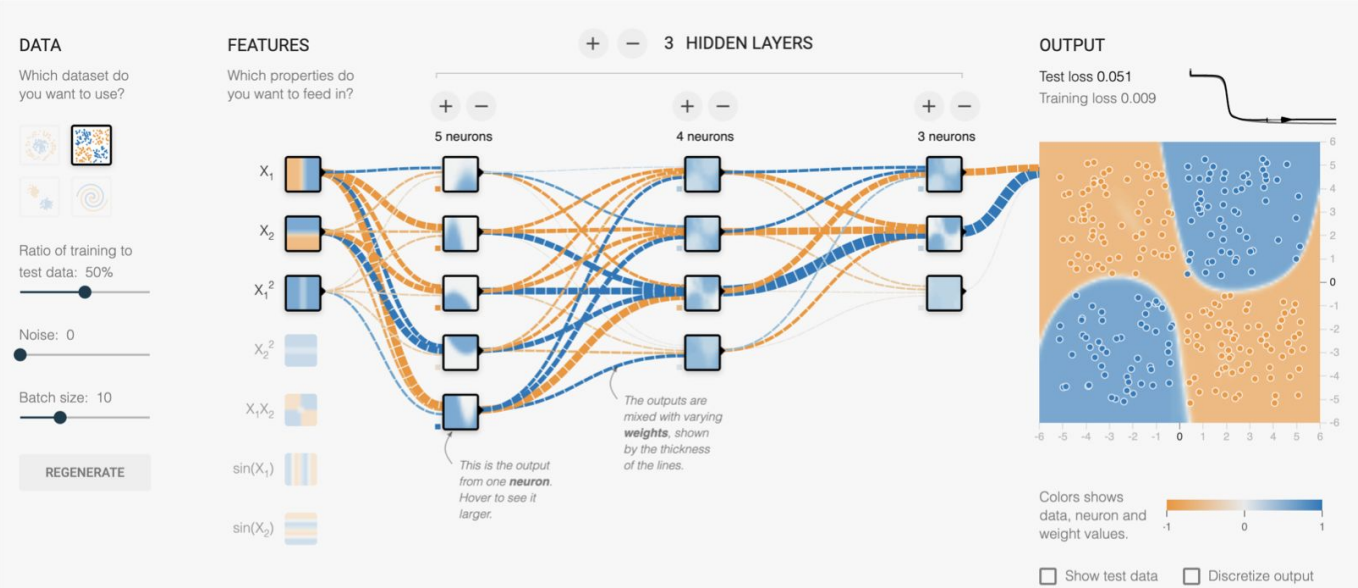
# Deep Neural Network

## Deep Neural Network

ネットワークの全貌を眺めてみよう。

こちらのサイトで、
隠れ層（HiddenLayers）あるいはノード数（neurons）を増減させてみよう。
さらに活性化関数（Activation）も変えてみよう。

http://playground.tensorflow.org/#activation=relu&batchSize=10&dataset=spiral&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=0&networkShape=5,4,3&seed=0.48429&showTestData=false&discretize=false&percTrainData=50&x=true&y=true&xTimesY=false&xSquared=true&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=classification&initZero=false&hideText=false



# 深層学習の論点を知る

① 表現能力　←←← 今回はこれ

ネットワークの構造に依存

② 汎化能力

ネットワークの構造&学習アルゴリズムに依存

③ 最適化能力

ネットワークの構造&学習アルゴリズムに依存

# 深層学習の論点を知る

ニューラルネットワークはどれだけ複雑な関数を学習できるか、いわば、その表現能力についての活発な議論は80年代後半が盛んであった。

これは、モデルはどこまで真の連続関数(1)を近似できるか（**関数近似能力**）を問うことに他ならない。

(1) 関数f(x)が定義域のすべてのxの値で連続であるとき、f(x)は連続関数である。

# 浅いニューラルネットの場合

「3層のニューラルネット（中間層1層）のような浅いモデルでも、中間層のノード数を無限に増やせば任意の関数を任意の精度で近似できる。」

——万能近似定理より [Cybenko (1989)]

https://pdfs.semanticscholar.org/05ce/b32839c26c8d2cb38d5529cf7720a68c3fab.pdf

3層のニューラルネットの関数近似能力の定式化：
真の連続関数　$f : \mathbb{R}^m \to \mathbb{C}$　を以下のような $g(x)$ で近似する

出力層のパラメータ
シグモイド的関数

$$g(\mathrm{x}) = \sum_{j=1}^{J} c_j \eta\left(\mathrm{a}_j \cdot \mathrm{x} - b_j\right)$$

中間層のノード

$\left(\mathrm{a}_j, b_j, c_j\right) \in \mathbb{R}^m \times \mathbb{R} \times \mathbb{C}$

$\left(\mathrm{a}_j, b_j\right) \in \mathbb{R}^m \times \mathbb{R}$　は中間層のパラメータ

シグモイド的：　$h(x) \to \begin{cases} 1 & (x \to \infty) \\ 0 & (x \to -\infty) \end{cases}$

表 2.1: $g(x) := \sum_{j=1}^{J} c_j \eta(a_j \cdot x - b_j)$ による $f(x)$ の近似可能性

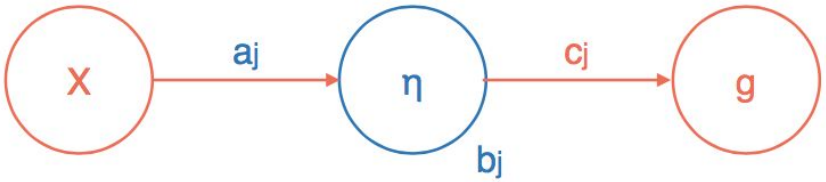
| | $f$ | $\eta$ | 位相 | 証明方針 |
|---|---|---|---|---|
| Irie and Miyake 1988 | $L^1$ | $L^1$ | 各点 | Fourier 反転公式 |
| Cybenko 1989 | $C$ | シグモイド | 広義一様 | Hahn-Banach |
| Hornik+ 1989 | $C$ | シグモイド | 広義一様 | Stone-Weierstrass |
| Funahashi 1989 | $C$ | 有界連続かつ単調増加 | 広義一様 | Fourier 反転公式 |
| Carroll and Dickinson 1989 | $\mathcal{D}$ | tanh | $L^2$ | Radon 反転公式 |
| Mhaskar and Micchelli 1992 | $L^p$ | $\mathcal{S}_0'$ | $L^p$ | $B$-spline |
| Leshno+ 1993 | $C$ | $\mathcal{S}_0'$ | 広義一様 | 近似単位元 |

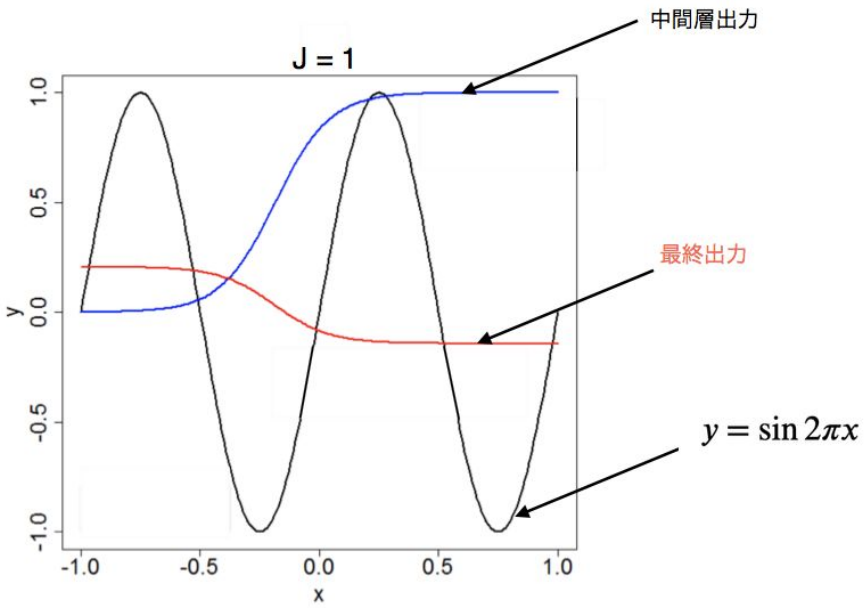
# 浅いニューラルネットの場合

近似したい関数： $y = \sin 2\pi x$

ノード数 1 のNN： $g(x) = \sum_{j=1}^{1} c_j \eta\left(a_j \cdot x - b_j\right)$

中間層のノード



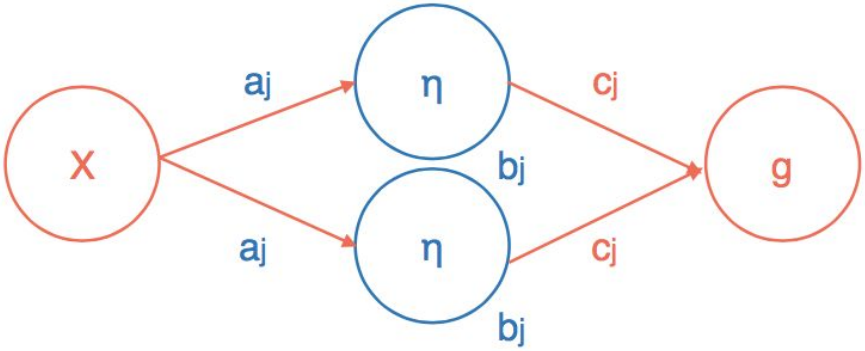



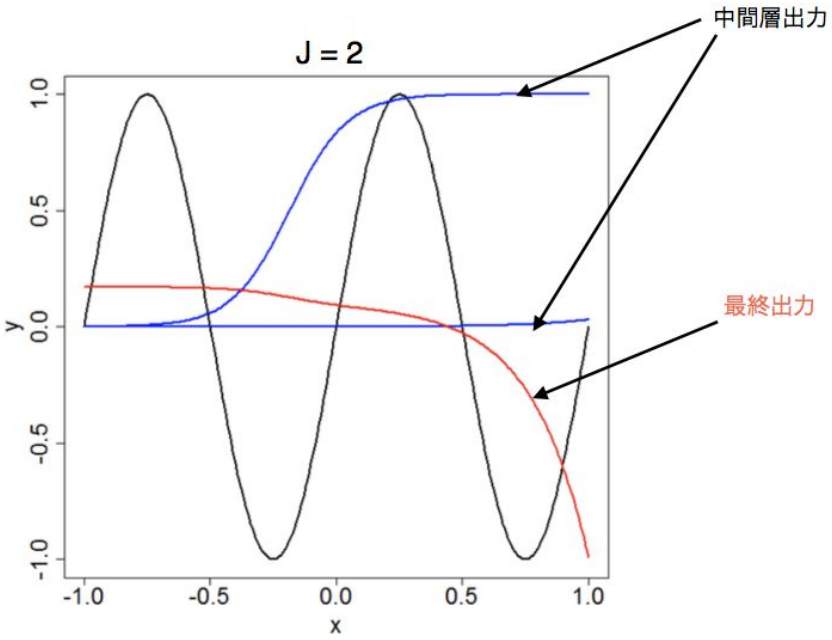
# 浅いニューラルネットの場合

近似したい関数： $y = \sin 2\pi x$

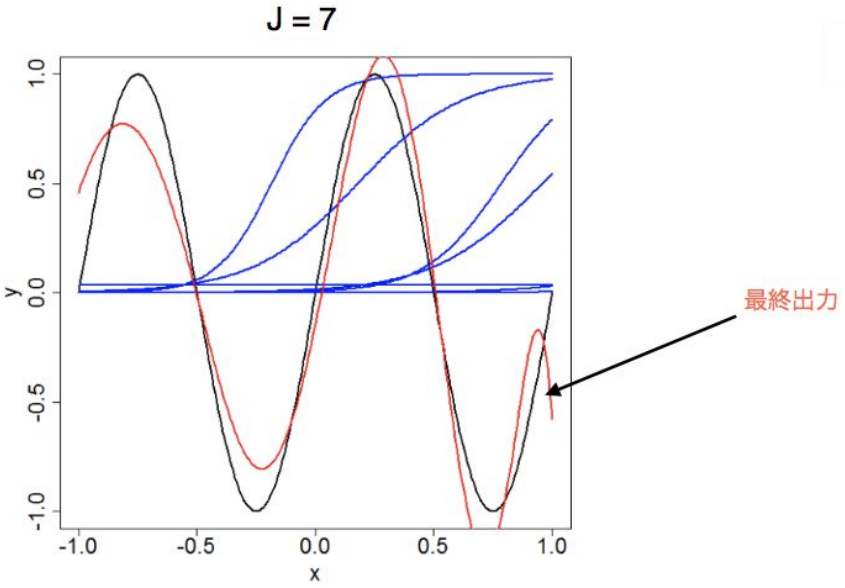
ノード数２のNN： $g(x) = \sum_{j=1}^{2} c_j \eta\left(a_j \cdot x - b_j\right)$

中間層のノード

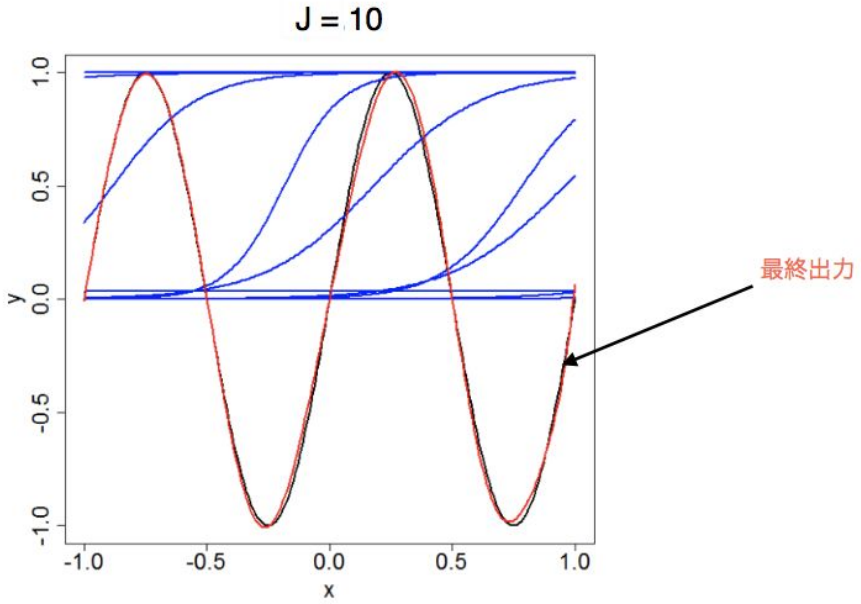


# 浅いニューラルネットの場合




$$g(\mathrm{x}) = \sum_{j=1}^{7} c_j \eta\left(\mathrm{a}_j \cdot \mathrm{x} - b_j\right)$$




$$g(\mathrm{x}) = \sum_{j=1}^{10} c_j \eta\left(\mathrm{a}_j \cdot \mathrm{x} - b_j\right)$$

# 深いニューラルネットの場合

## 深いニューラルネットの場合

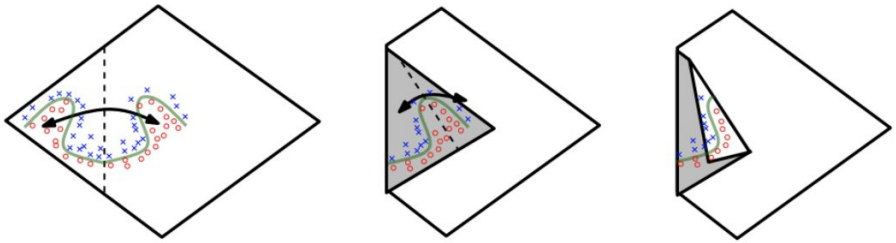
浅いニューラルネットにおいて、中間層のノードが無限にあれば、
任意の関数を近似できることは証明されている。

$$g(\mathrm{x}) = \sum_{j=1}^{\infty} c_j \eta\left(\mathrm{a}_j \cdot \mathrm{x} - b_j\right)$$

それでも層を増やす必要はあるか？

任意の関数を近似するには一つの中間層で十分であるが、その一つの層が非現実的に巨大なサイズになる可能性がある。

ノード数を増やし層を横に広げることで、表現能力は多項式的に上がる一方、**層の数を増やすことによって**表現力は**指数関数的に上がる**。



表現能力：領域をいくつの多面体に分けられるか

中間層のユニット数

$$\left(\prod_{i=1}^{L-1} \left\lfloor \frac{n_i}{n_0} \right\rfloor^{n_0}\right) \sum_{j=0}^{n_0} \binom{n_L}{j}.$$

$L$：層の数
$n$：中間層の横幅
$n_0$：入力の次元

$$n_i \geq n_0 \text{ for all } i \in [L]$$

https://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of-deep-neural-networks.pdf

# 深層学習の論点を知る

① 表現能力

ネットワークの構造に依存

② 汎化能力 ←←← ちょっと紹介

ネットワークの構造＆学習アルゴリズムに依存

③ 最適化能力

ネットワークの構造＆学習アルゴリズムに依存

# 汎化能力

## 汎化性の高いモデルを手に入れたい

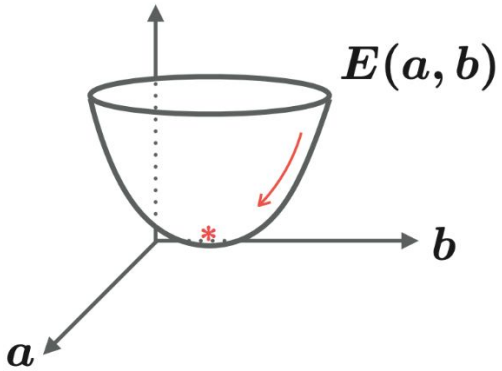
有限の訓練データからルールや知識を獲得し、（同じ分布からサンプリングされる）訓練データには含まれないが、訓練データと同様の性質をもつ未知のデータに対してもうまく推論できるようなモデルを獲得すること。

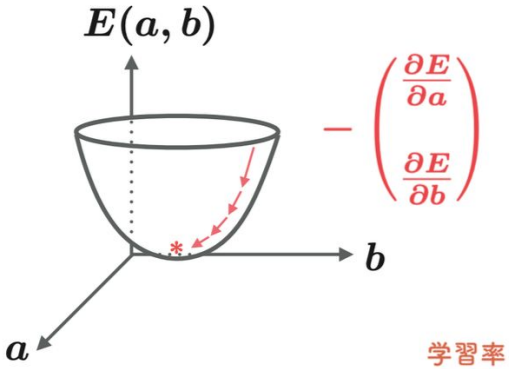
**勾配降下法**：

最適化問題（データの学習）を数値的に解く

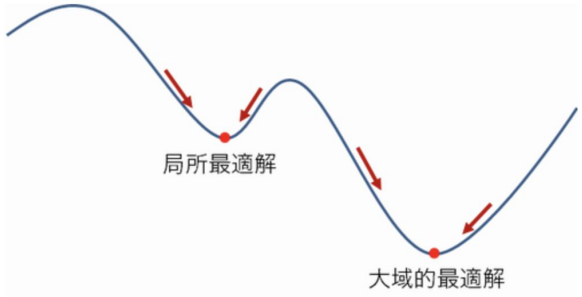
→ **極小値を求める**

$$a^*, b^* = \mathrm{argmin}\, E(a, b)$$




$$-\begin{pmatrix} \frac{\partial E}{\partial a} \\ \frac{\partial E}{\partial b} \end{pmatrix}$$




学習率

$$a^{t+1} = a^t - 0.01\frac{\partial E(a^t, b^t)}{\partial a}$$

$$b^{t+1} = b^t - 0.01\frac{\partial E(a^t, b^t)}{\partial b}$$

# 汎化能力

## 極小値には、汎化性にとって良いものと悪いものがある

良いもの：平坦性(flatness)

悪いもの：鋭度(sharpness)

**確率的勾配降下法**（SGD）

バッチサイズが**小さい**：

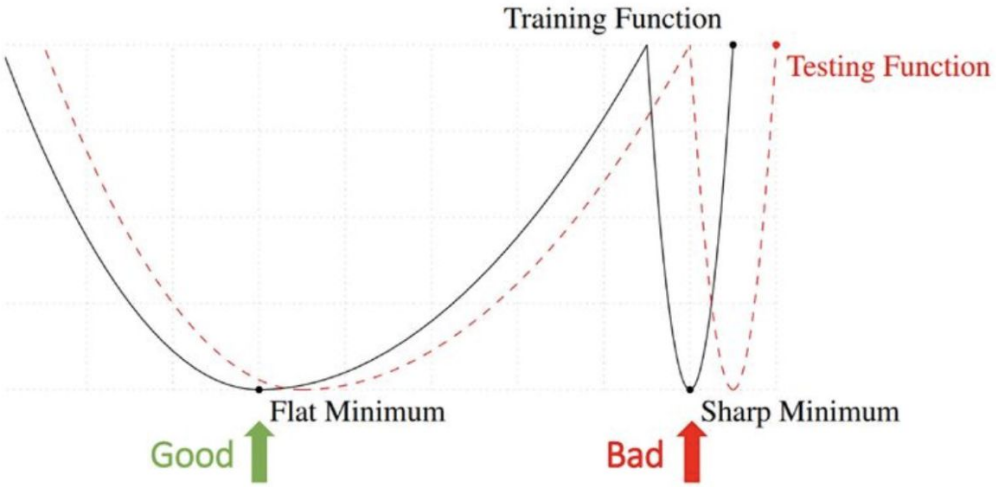
勾配のノイズが増加し、平坦な大域最適解に収束し、かつ汎化性能がよい。

バッチサイズが**大きい**：

シャープな大域最適解に収束し、汎化性能が低い。

暗黙的に正則化を行っている？？？

https://arxiv.org/pdf/1710.06451.pdf




学習率を自動調整するAdaGrad学習は高速だが汎化能力の低い最適解に収束するという議論がある

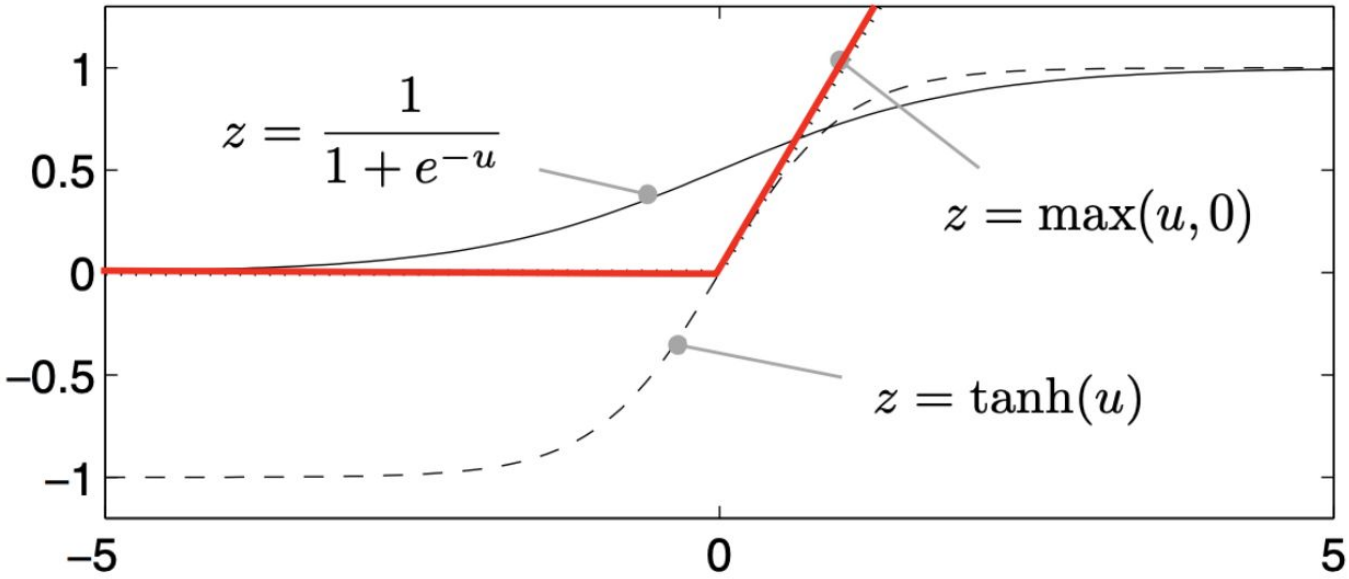
# おまけ

## 活性化関数について

# 活性化関数と初期値のよい組み合わせ

```
Xavier ――Sigmoid/Tanh
He ――ReLU
```

Xavier：ザビエル、
He：フー
ReLU：レルー
Tanh：ハイパボリックタンジェント（タンボリック）

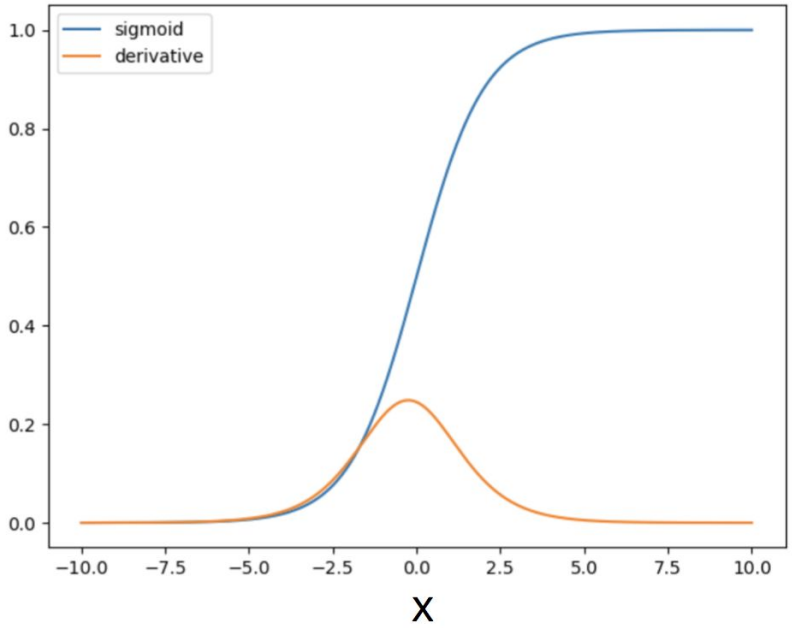


# 勾配消失

## 活性化関数について

ネットワークのアーキテクチャを様々工夫してみたけれど、なぜか学習が進まないときがある。

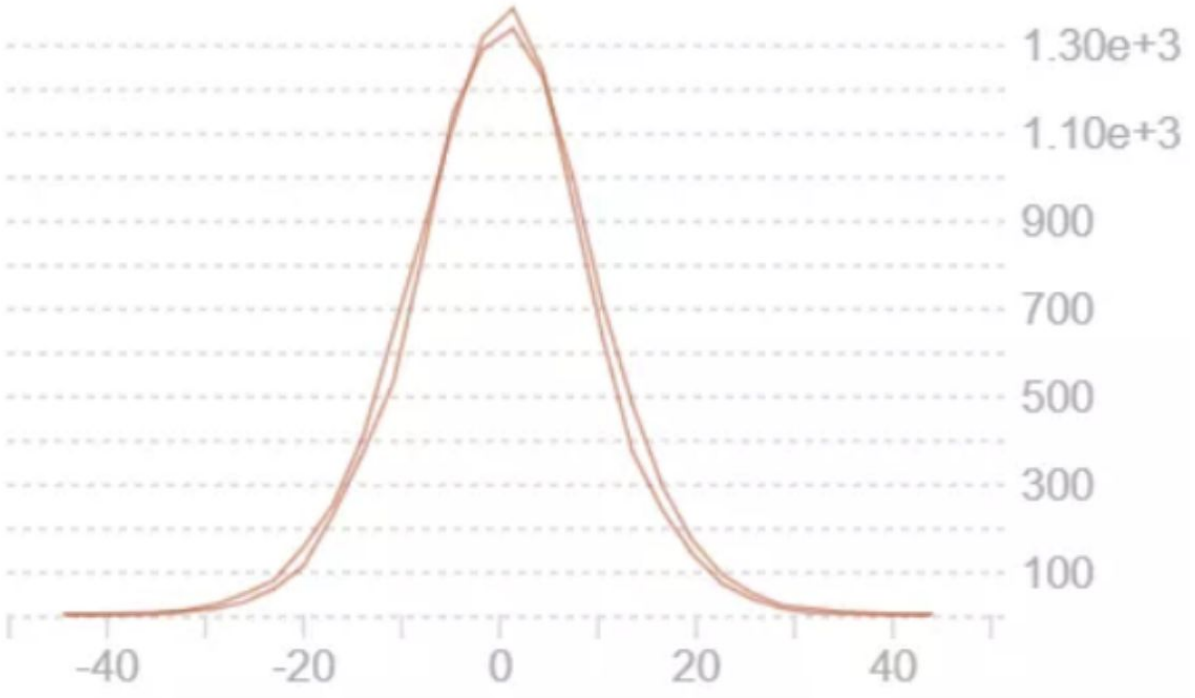
逆伝搬がうまくいかない理由の一つに、隠れ層の飽和が挙げられる。
下の青い関数はSigmoid関数で、オレンジの関数はその一次導関数である。
abs(x)＞6のとき、導関数は0に近づく（飽和する）ことがわかる。重みを更新する逆伝播法は活性化関数の導関数に依存するため、ノードの出力が飽和領域に至ると、学習が遅くなるか、まったく行われなくなる。



# 勾配消失

重みの初期値の分散を1.0として、Sigmoid関数に通してみよう。

入力を以下のような分布としよう。



# 勾配消失

## 出力の分布

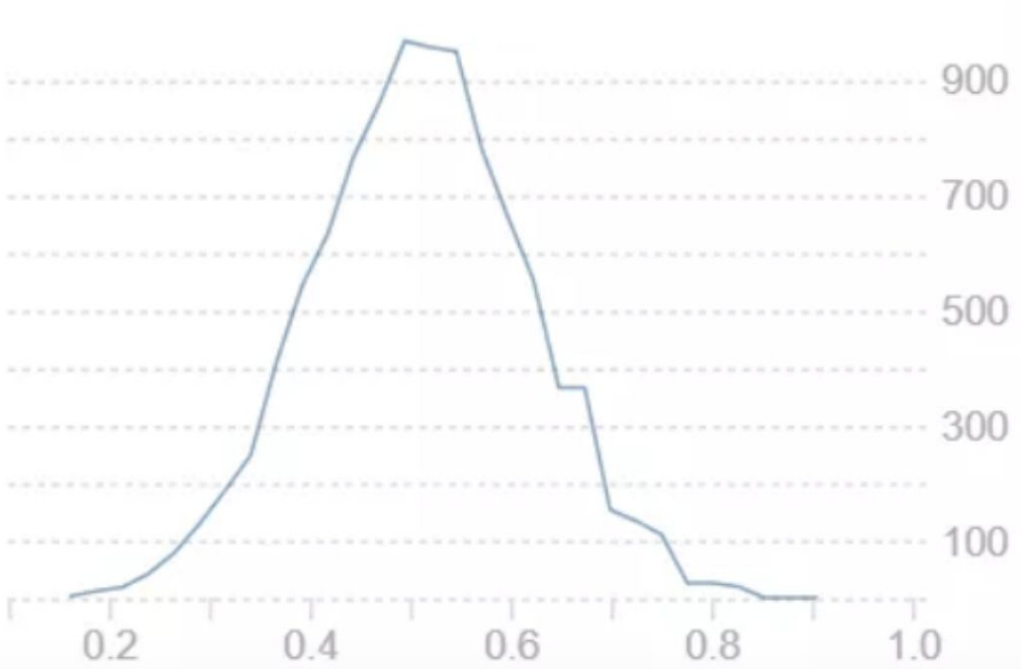
Sigmoidを通過後、Sigmoid曲線の飽和領域、すなわち0と1に近い出力に絞り込まれた。

逆伝搬では、流れてきたデルタにSigmoid関数の導関数をアダマール積するので、導関数がほぼ0になると勾配が消失する可能性がある。

# 勾配消失(回避)

## 重みの初期値がXavierの場合

出力はSigmoid関数の線形領域（0.5）を中心とした分布となり、飽和は生じなかった。



# DNN_深層学習
# 完