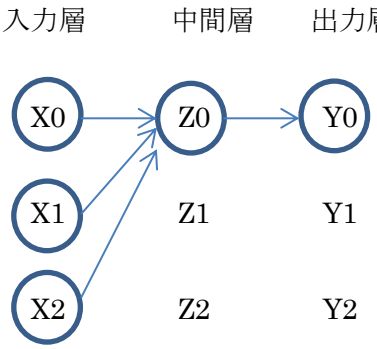
1　Section1_入力層~中間層

ニューラルネットワークの構造（２層）は以下のようになる。



入力層 X0,X1,X2 として、中間層へは W00・X0、W01・X1、W02・X2 を渡す。

入力総和 Z0＝W00・X0＋W01・X1＋W02・X2

入力総和 Z1＝W10・X0＋W11・X1＋W12・X2

入力総和 Z2＝W20・X0＋W21・X1＋W22・X2

行列式より

$$\begin{bmatrix} W00 & W01 & W02 \\ W10 & W11 & W12 \\ W20 & W21 & W22 \end{bmatrix} \begin{bmatrix} X0 \\ X1 \\ X2 \end{bmatrix} = \begin{bmatrix} Z0 \\ Z1 \\ Z2 \end{bmatrix}$$

２　Section2_活性化関数

ニューラルネットワークにおける活性化関数とは、あるニューロンから次のニューロンへと出力する際に、あらゆる入力値を別の数値に変換して出力する関数である。

１）ReLU

関数への入力値が０以下の場合は、出力値が常に０、入力値が０より大きい場合は、出力値が入力値と同じになる。

２）Mish

関数への入力値が０以下の場合は、出力値がほぼ０、入力値が０より大きい場合は、出力値が入力値と同じになる。

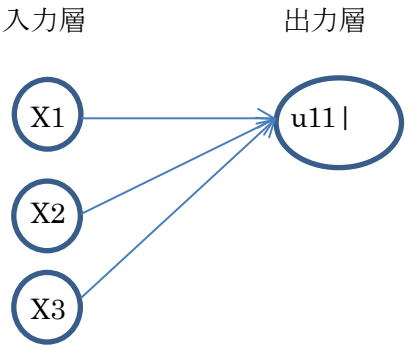
３）活性化関数一覧

No	活性化関数	概要
1	ステップ関数	Y= 1(x>0) 0(x≦0)
2	恒等関数	Y=x（順伝播） δ y / δ x ＝１（逆伝播）
3	Bent Idenntity	y ＝1/2(√x2+1-1)+x
4	HardShirink	Y=x(x＜-λ or λ＜x) 0
5	SoftShrink	Y=x+λ(x＜-λ)（順伝播） x－λ(λ＞x) 0 δ y / δ x ＝１（逆伝播） 0
6	Threshold	Y=x（順伝播） 0 Y=1（逆伝播） 0
7	シグモイド関数	Y=1/(1+e-x)（順伝播） δ y / δ x ＝y(1-y)（逆伝播）
8	HardSigmoid	Y=1 0.2x+0.5 0 δ y / δ x =0.2 0
9	logSigmoid	Y=log(1/(1+e-x) δ y / δ x =1/(1+ex)
10	Tanh	Y=tanh(x)=(ex-e-x)/(ex+e-x) δ y / δ x =sech2(x)=1/cosh2(x)=4/(ex+e-x)2
11	tanhSirink	Y=x-tanh(x) δ y / δ x =tanh2(x)

12	Hardtanh	Y=1 -1 X δ y / δ x =0 1
13	ReLU 関数	Y=x(x＞0) 0(x≦0) δ y / δ x =1(x＞0) 0(x≦0)
14	ReLU6	Y=0(x≦0) 6(x≧6) X δ y / δ x =0((x≦0) or (x≧6)) 1
15	Leak-ReLU	Y=x(x＞0) 0.01x(x≦0) δ y / δ x =1(x＞0) 0.01(x≦0)
16	ELU	Y=x (x≧0) α(ex-1) (x＜0) δ y / δ x =1 α ex
17	SELU	Y= λ x λ α(ex-1) δ y / δ x = λ λ α ex
18	CELU	Y=x α(e(x/ α)-1) δ y / δ x =1 e(x/ α)
19	ソフトマックス関数	Yi=exi/ Σ exk δ y / δ x = yi(1-yi) Σ yiyj= Σ yi(δ ij-yj)
20	Softmin	Yi=e-xi/ Σ e-xk δ y / δ x =yi(1-yi)+ Σ yiyj= Σ yi(δ ij-yj)
21	Logsoftmax	Yi=log(exi/ Σ exk) δ y / δ x = Σ (δ ij-eyj)
22	Softplus	Y=log(1+ex)=ln(1+ex) δ y / δ x =ex/(1+ex)=1/(1+e-x)
23	Softsign	Y=x/(1+|x|) δ y / δ x =1/(1+|x|)2

24	Swish	Y=x/(1+e- β x) δ y / δ x = β y+(1- β y)/(1+e- β x)
25	hardSwish	Y=0 X(x+3)/6 X δ y / δ x =0 (2x+3)/6 1
26	ACON	
27	Mish	
28	tanhExp	Y=xtanh(ex) δ y / δ x =tanh(ex)-xex(tanh2(ex)-1)

3　Section3_出力層



出力層では、活性化関数を使用して非線形変換を行う。

出力層の設計

1）ニューラルネットワークは分類、回帰の両方に用いられる。

2）どちらの問題を解決するかで、活性化関数を変更する必要がある。

3）回帰問題では恒等関数を、分類問題ではソフトマックス関数を用いる。

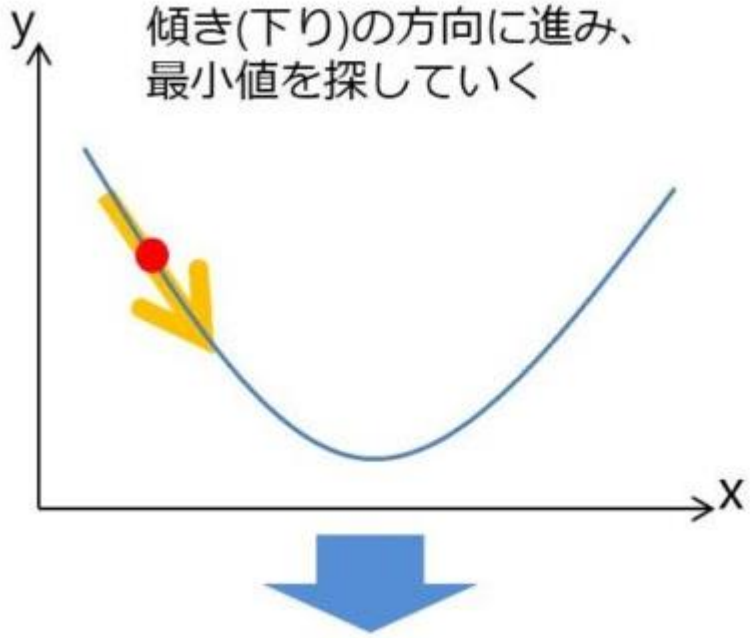
4）クラス分類では、出力層のニューロンの数はクラス数

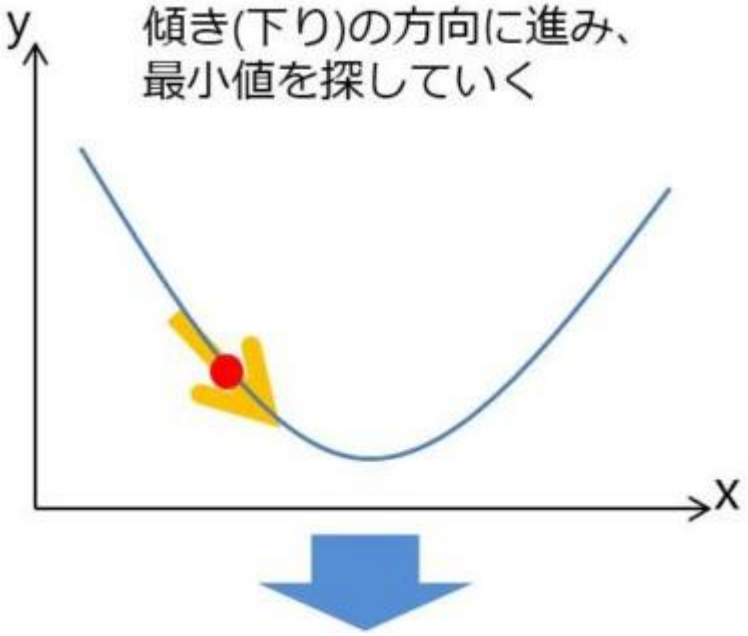
４　Section4_勾配降下法

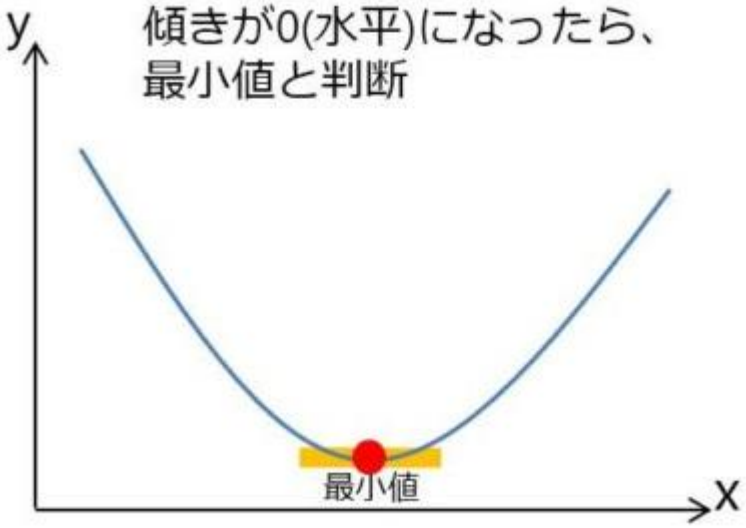
モデルに対してコストが最小になるようにパラメータを少しづつ変化させて、トレーニングに適合したパラメータを算出するアルゴリズムである。

$x = x - \eta \left(df(x) / dx \right)$

１）連鎖律







手法	利用データ	計算時間	メリット	デメリット
バッチ勾配降下法	全てのデータ	大	・解への到達が早い ・結果が安定	・メモリの使用量が多い ・局所解にはまりやすい
SGD	１つのデータ	小	・メモリの使用量が少ない ・オンライン学習が可能 ・局所解を回避する可能性がある	・解への到達が遅いことがある ・はづれ値の影響を大きく受ける
ﾐﾆﾊﾞｯﾁ勾配降下法	一部のデータ	中	・バッチ勾配降下法と SGD のそれぞれのメリットがある	それぞれのデメリットがある

5　Section5_誤差逆伝播法

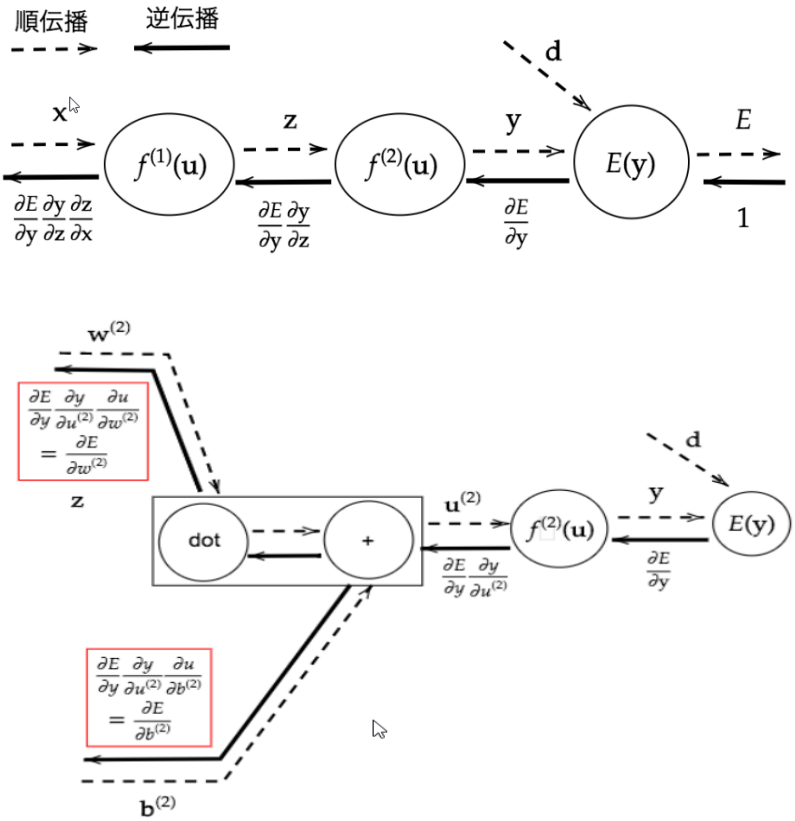
（1）誤差逆伝播法の概要

$$\nabla E = \frac{\partial E}{\partial \mathbf{w}} = \left[\frac{\partial E}{\partial w_1} \cdots \frac{\partial E}{\partial w_M} \right]$$

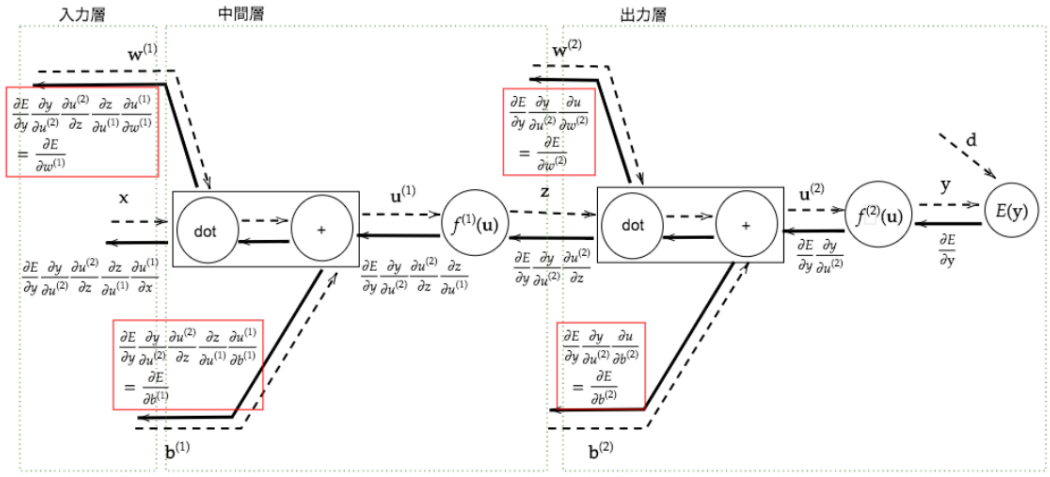
【誤差逆伝播法】

算出された誤差を、出力層側から順に微分し、前の層前の層へと伝播させる。

最小限の計算で各パラメータでの微分値を解析的に計算する手法である。



計算結果（=誤差）から微分を逆算することで、不要な再帰的計算を避けて微分を算出する。



計算結果（=誤差）から微分を逆算することで、不要な再帰的計算を避けて微分を算出できる。

（1）誤差勾配の計算について

$$E(\mathbf{y}) = \frac{1}{2}\sum_{j=1}^{J}(y_j - d_j)^2 = \frac{1}{2}\|\mathbf{y}-\mathbf{d}\|^2 \quad : \text{誤差関数} = \text{二乗誤差関数}$$

$$\mathbf{y} = \mathbf{u}^{(L)} \quad : \text{出力層の活性化関数} = \text{恒等写像}$$

$$\mathbf{u}^{(l)} = \mathbf{w}^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)} \quad : \text{総入力の計算}$$

$$\frac{\partial E}{\partial w_{ji}^{(2)}} = \frac{\partial E}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial w_{ji}^{(2)}}$$

$$\frac{\partial E(\mathbf{y})}{\partial \mathbf{y}} = \frac{\partial}{\partial \mathbf{y}}\frac{1}{2}\|\mathbf{y}-\mathbf{d}\|^2 = \mathbf{y}-\mathbf{d}$$

$$\frac{\partial \mathbf{y}(\mathbf{u})}{\partial \mathbf{u}} = \frac{\partial \mathbf{u}}{\partial \mathbf{u}} = 1$$

$$\frac{\partial \mathbf{u}(\mathbf{w})}{\partial w_{ji}} = \frac{\partial}{\partial w_{ji}}\left(\mathbf{w}^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)}\right) = \frac{\partial}{\partial w_{ji}}\left(\begin{bmatrix} w_{11}z_1 + \ldots + w_{1i}z_i + \ldots w_{1I}z_I \\ \vdots \\ w_{j1}z_1 + \ldots + w_{ji}z_i + \ldots w_{jI}z_I \\ \vdots \\ w_{J1}z_1 + \ldots + w_{Ji}z_i + \ldots w_{JI}z_I \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_j \\ \vdots \\ b_J \end{bmatrix}\right) = \begin{bmatrix} 0 \\ \vdots \\ z_i \\ \vdots \\ 0 \end{bmatrix}$$

$$\frac{\partial E}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial w_{ji}^{(2)}} = (\mathbf{y}-\mathbf{d})\cdot\begin{bmatrix} 0 \\ \vdots \\ z_i \\ \vdots \\ 0 \end{bmatrix} = (y_j - d_j)z_i$$