ラビットチャレンジ課題　作成者:内山 貴雄

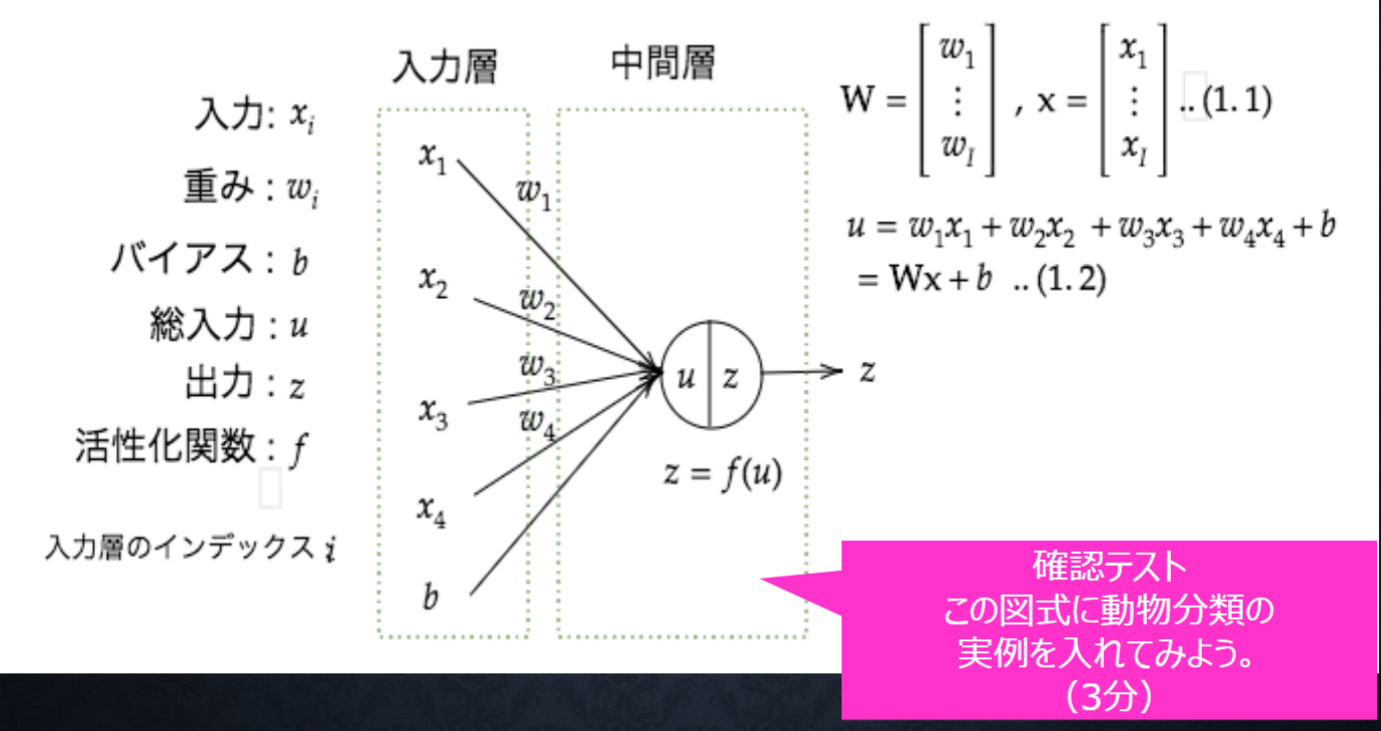
深層学習の各章の要約及び確認テスト等の考察

●入力~中間層

本章では、動物の分類問題を例にニューラルネットワーク(NN)による分類問題の学習におけるその入力~中間層の役割や入出力の計算方法を学んだ。また、確認テストでは、前述の動物分類問題をNNにて学習した際を例とし、入力層・中間層の役割概念を学んだ後に、それらをPythonにて実装する手法を学んだ。

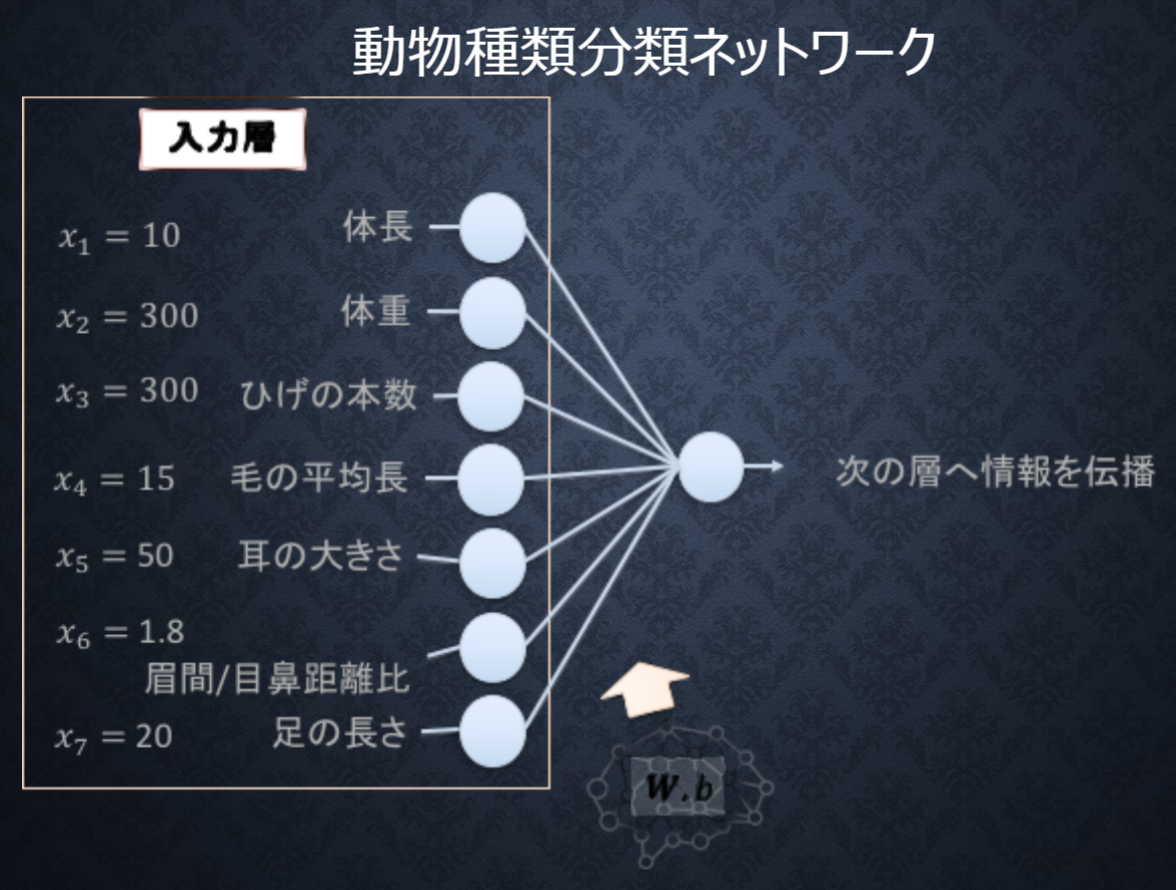
〇確認テスト1

下記図式に動物分類の実例を示せ。



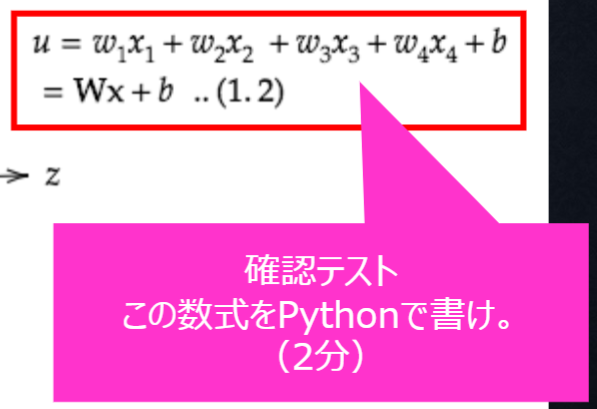
答案：

分類する動物の各情報(体重、体長、ひげの本数)を入力(x)とし、中間層に入力する値(総入力:u)は入力とそれに対応する重み(w)の積の総和となる。



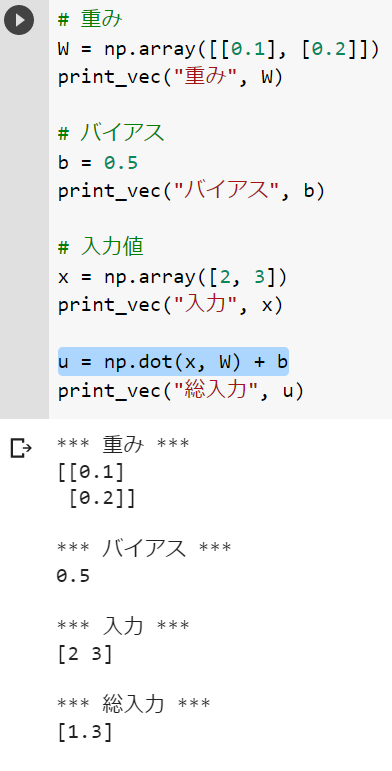
〇確認テスト2

下記式をPythonにて記述せよ。



答案：

下記コードにて重み、バイアス、入力を定義し中間層の総入力を青のマーカーに示す様に実装した。



〇確認テスト3

「1\_1\_forward\_propagation.ipynb」内にて中間層の出力を定義しているコードを抜き出せ。

答案：

下記コード内では、3層のNNを実装しており、中間層への総入力をu2、総出力をz2とし青のマーカー内にて中間層の出力を定義している。また、中間層の活性化関数として、入力値が0以下の場合には出力値が0、入力値が0より上の場合には出力値が入力値と同じ値となる関数ReLU（Rectified Linear Unit）を採用している。



●活性化関数  
本章では、NNにおける活性化関数の性質及び種類、その定義等を学習した。

まず、活性化関数は非線形な関数であるため、確認テストにて非線形関数の定義を確認した。次に活性化関数として代表的な3つの下記関数の定義・性質を学習した。

1. ステップ関数

出力は常に１か０であり、しきい値を超えたら発火する関数。パーセプトロンにて使用されていた。

・課題：

0 -1間の間を表現できず、線形分離可能なものしか学習できなかった

1. シグモイド関数

入力値により出力地が0 ~ 1の間を緩やかに変化する関数。ステップ関数は出力値が0,1であるのに対し、出力値が0から1の間であり、その強弱を伝えられるため予想ニューラルネットワーク普及の足掛けとなった関数。

課題；

値が大きな入力値において出力の変化が微小なため、勾配消失問題を引き起こすリスクがある。

1. ReLU関数

入力値が0以下の場合には出力値が0、入力値が0より上の場合には出力値が入力値と同じ値となる関数

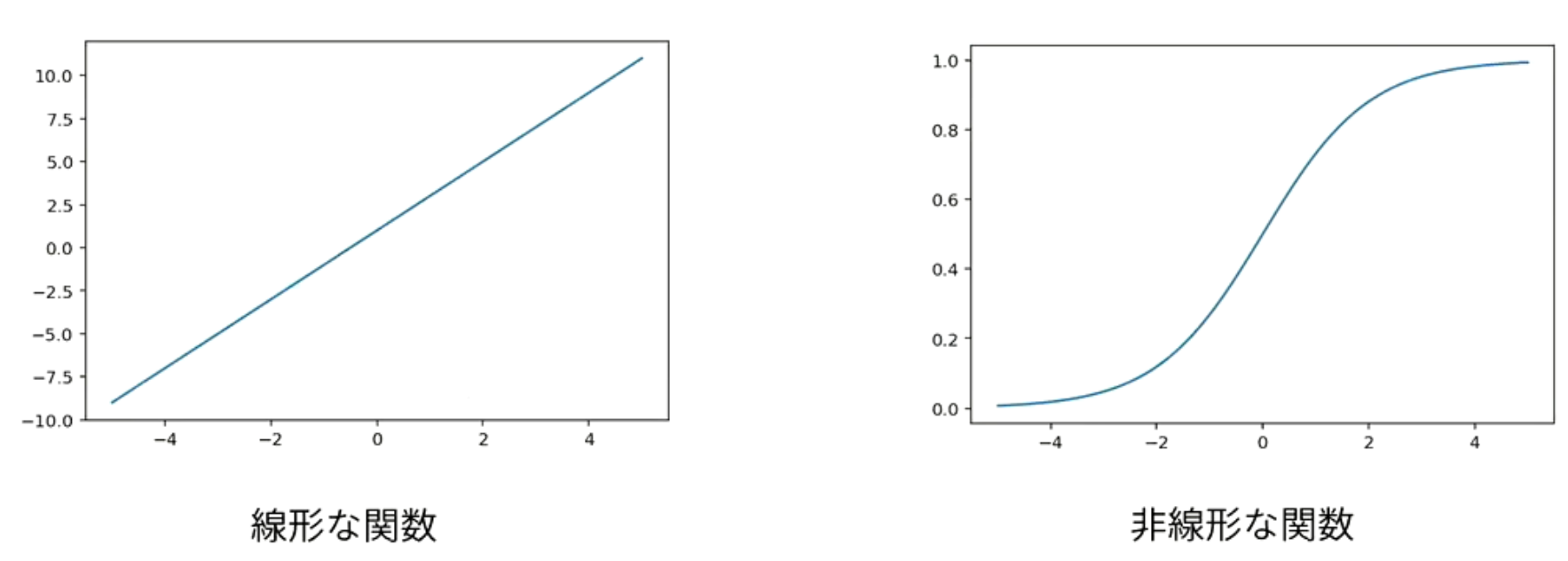
〇確認テスト：

線形と非線形の違いを図にかいて簡易に説明せよ。

答案：

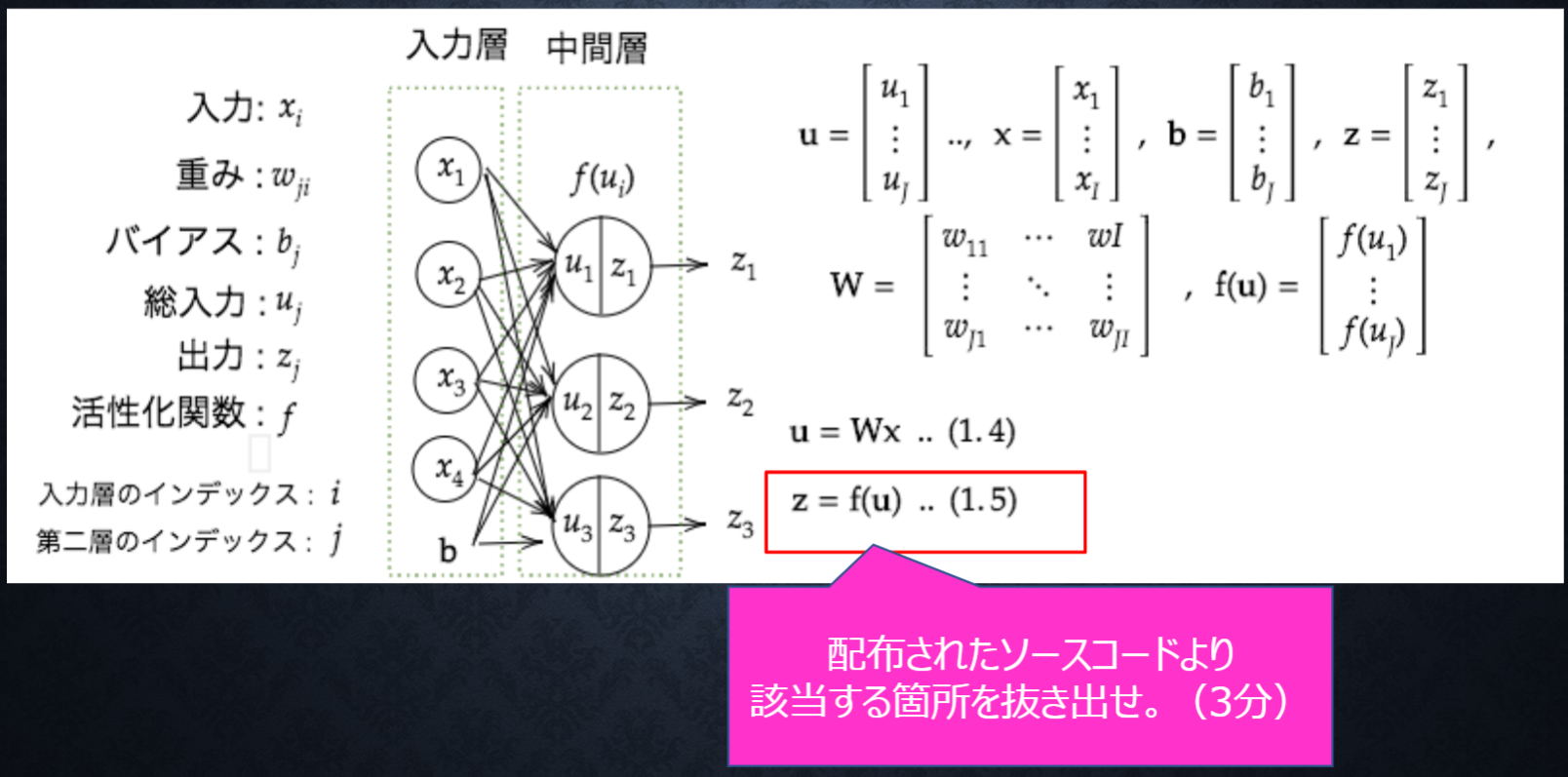
各関数を簡易に図示すると下記の様になる。

また、線形関数は加法性・斉次性を満たし、非線形関数はそれらの性質を持たない。



〇確認テスト

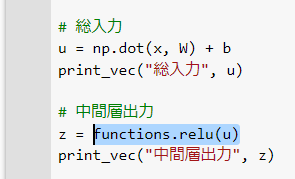
配布されたコードより下記図のマークされた式に該当する箇所を抜き出せ。



答案：

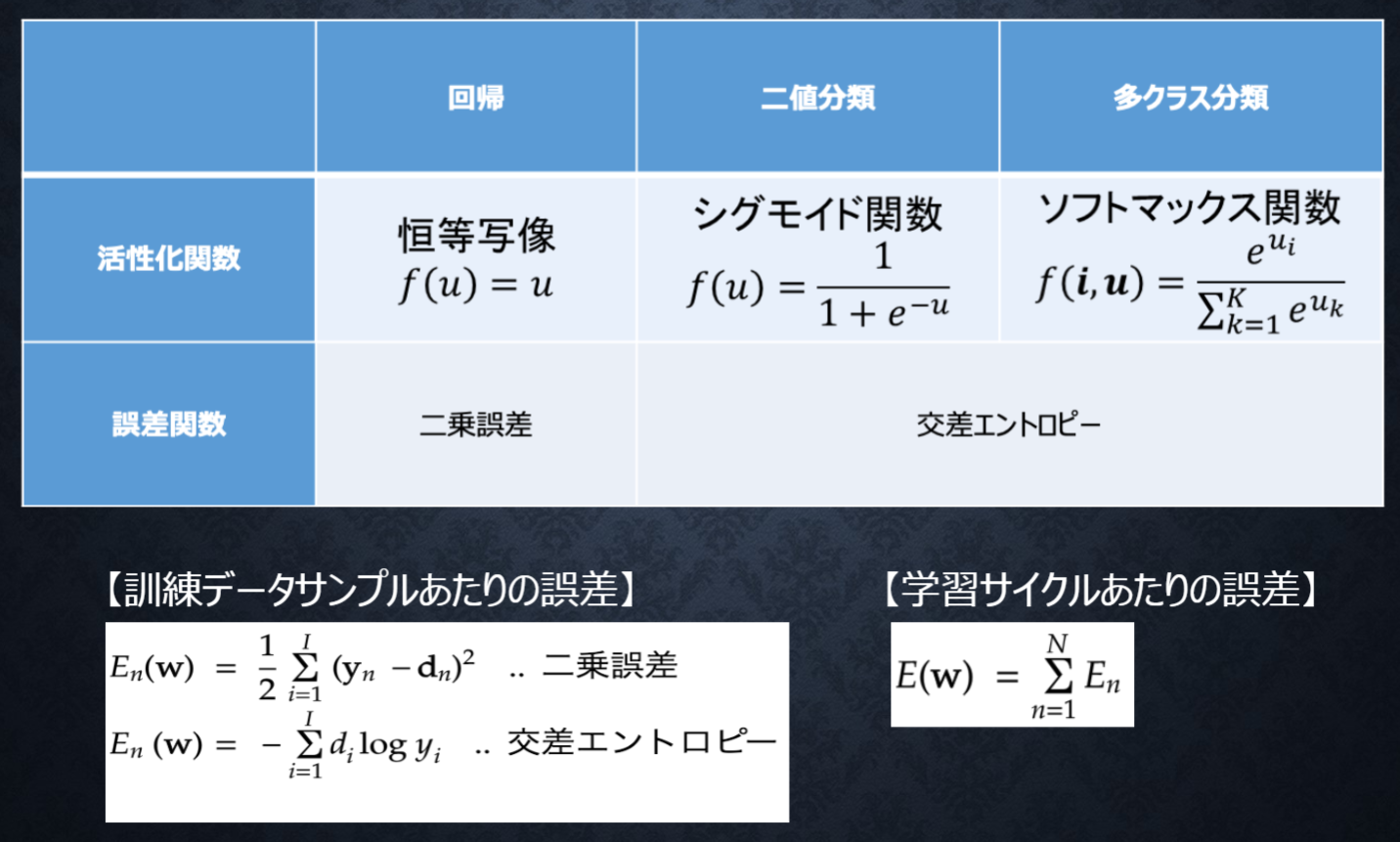
該当箇所を下記図の青マーカーにて示す。

また該当箇所においてReLU関数を採用しており、関数の実装は配布コードのCommonフォルダ内functions.pyにて実装している。



●出力層

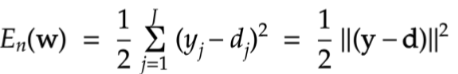
本章では、まず出力層の役割、各学習手法における使用する出力層の活性化関数及び誤差関数の定義を下記表にて学んだ。そのうえで、テキスト及び確認テストでは、各活性化関数の詳細説明とPythonでの実装を配布コードにて確認・学習した。



〇確認テスト

誤差関数に関しなぜ引き算ではなく2乗するか答えよ。

また、下記式の1/2はどういう意味を持つか答えよ。



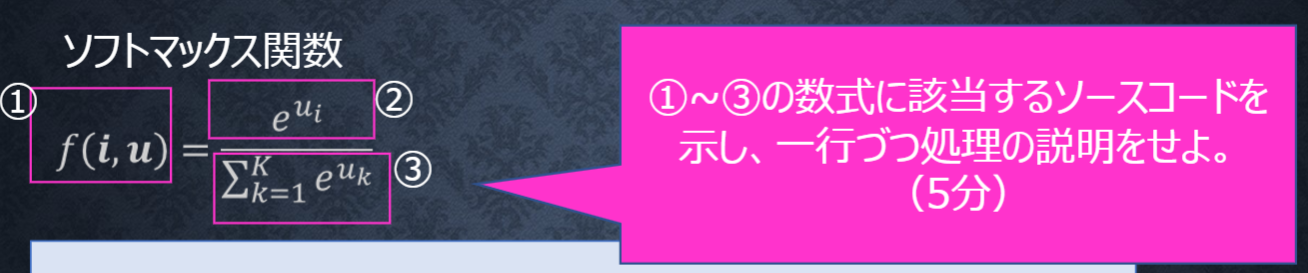
答案：

誤差dが正解値yよりも大きい場合、上記関数は誤差の大きさを示しているのに関わらず、負の値となってしまうため、2乗している。

また、1/2を掛けている理由は、誤差逆伝搬の計算の中で、誤差関数の微分を行う際の計算を簡易にするためである。

〇確認テスト

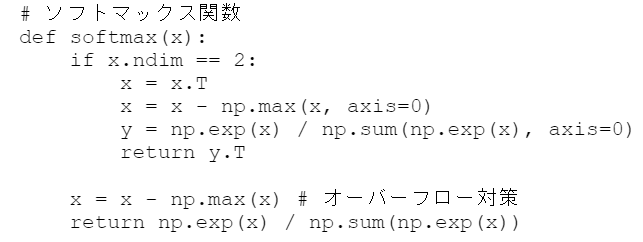
1~3の数式に該当するソースコードを示し、一行づつ処理の説明をせよ。



答案：

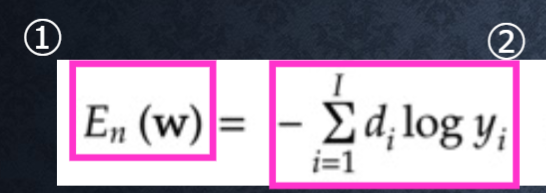
上記式のKは分類するクラス数を示しており、③式は各クラスの②式を総和したものであるため、①式は全クラスの②式の総和に占めるiクラスの②式の割合と言える。

配布コードcommonフォルダ内functions.pyにおいて上記ソフトマックス関数は下記の様に実装されており、コメント文を抜いた下記コードの1行目にて①式の引数を含めた表現をしており、8行目にて②、③式の表現を実装している。



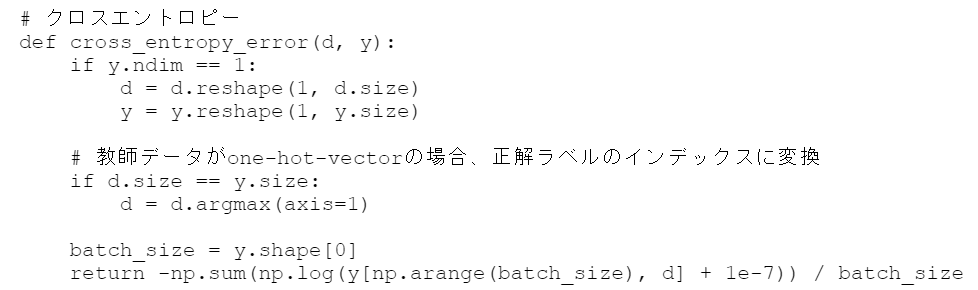
〇確認テスト

1~2の数式に該当するソースコードを示し、一行づつ処理の説明をせよ。



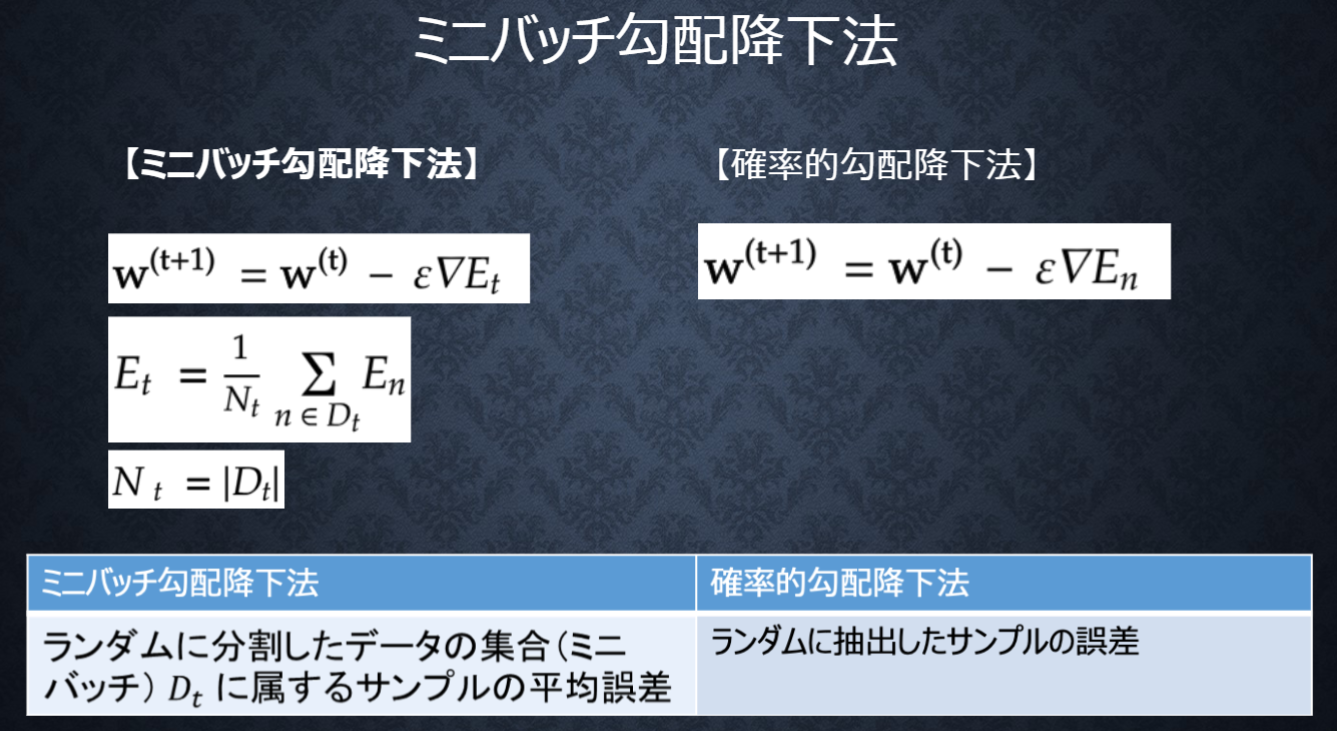
答案：

配布コードcommonフォルダ内functions.pyにおいて上記交差エントロピーは下記の様に実装されており、コメント文を抜いた下記コードの1行目にて①式の引数を含めた表現をしており、8行目にて②式の表現の実装をしている。8行目について、batch\_sizeはミニバッチ学習の際のバッチ数を示しており、1e-7については、log関数の出力値が-∞とならないために微小な値を敢えて入力している工夫である。



●勾配降下法

　本章では、誤差関数を最小或いは最大化する重みWを解析的ではなく、誤差関数の重みWによる偏微分関数に各変数値を代入し出力した値と学習率の積を更新前の重みWから引きそれを更新する操作を繰り返すことで、前述の条件を満たす重みWを算出する勾配降下法を学んだ。



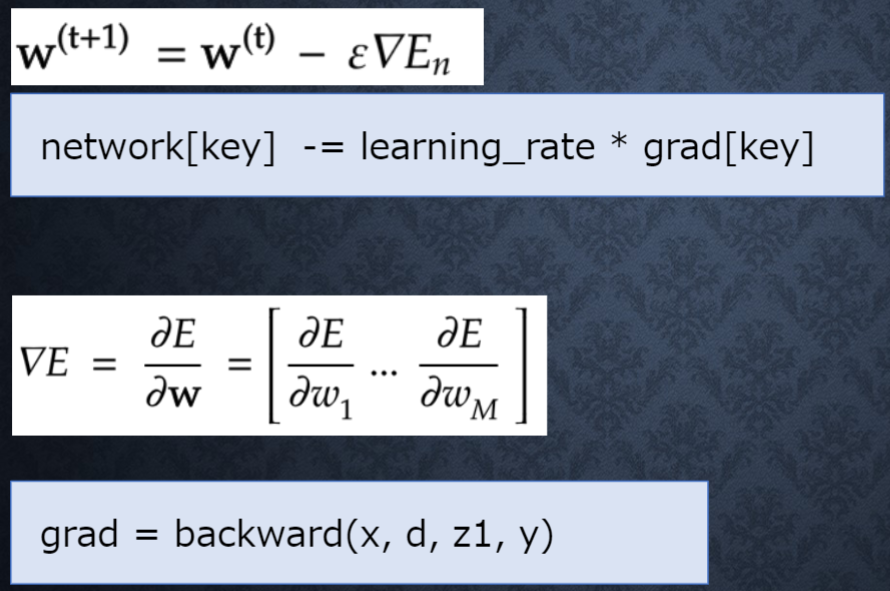
〇確認テスト

下記式を実装しているコードを示せ。

答案：

「1\_3\_stochastic\_gradient\_descent.ipynb」における「確率的勾配降下法」内にて、上記式は下記コードにて示されている。

　下記コード内network[key]は更新後の重みWを示し、leaning\_rateは学習率、grad[key]は誤差関数を重みWにて微分した式に既知の各変数を代入し算出された値を示す。またその値はbackwrad()関数にて求められる。



〇確認テスト

オンライン学習とは何か2行でまとめよ。

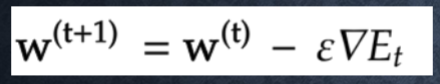
答案：

学習データが入ってくるたびに都度パラメータを更新し、学習を進めていく方法。

その一方で、バッチ学習は一度に全てのデータを用いパラメータを更新する方法である。

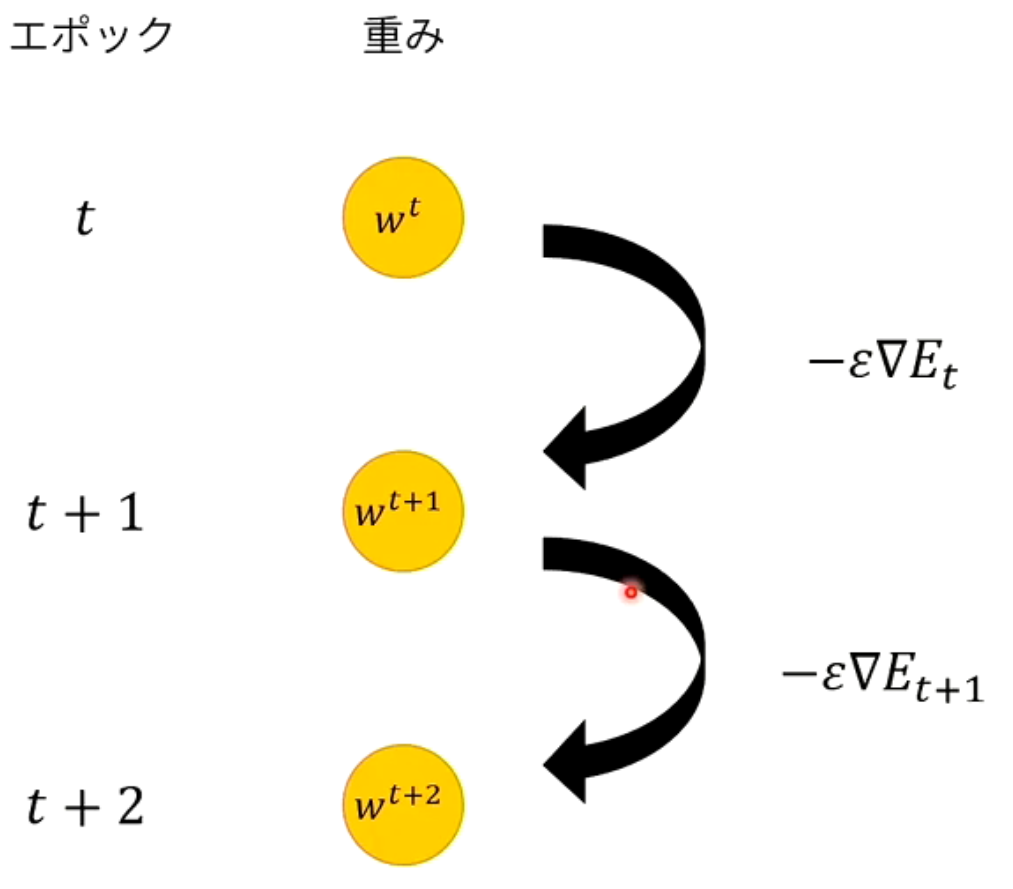
〇確認テスト

下記数式の意味を図に書いて説明せよ。



答案：

エポックのインデックスをt、各エポックにおける重みWをWt、各エポックのWの更新量を-εΔEttとして下記図に示している。



●誤差逆伝搬法

本章では、まずNNの勾配降下法による重み、バイアス値の更新を行う方法を学んだ。

まず、その方法としてプログラムで微小な数値hを生成し擬似的に微分を計算する一般的な手法である数値微分法を学習した。また、その方法の欠点について、各重みの更新を行う度に、誤差関数Eに(w+h),(w-h)を代入し順伝搬の計算から行い出力値を算出する必要があるため、計算量が大きくなる懸念があることを学んだ。

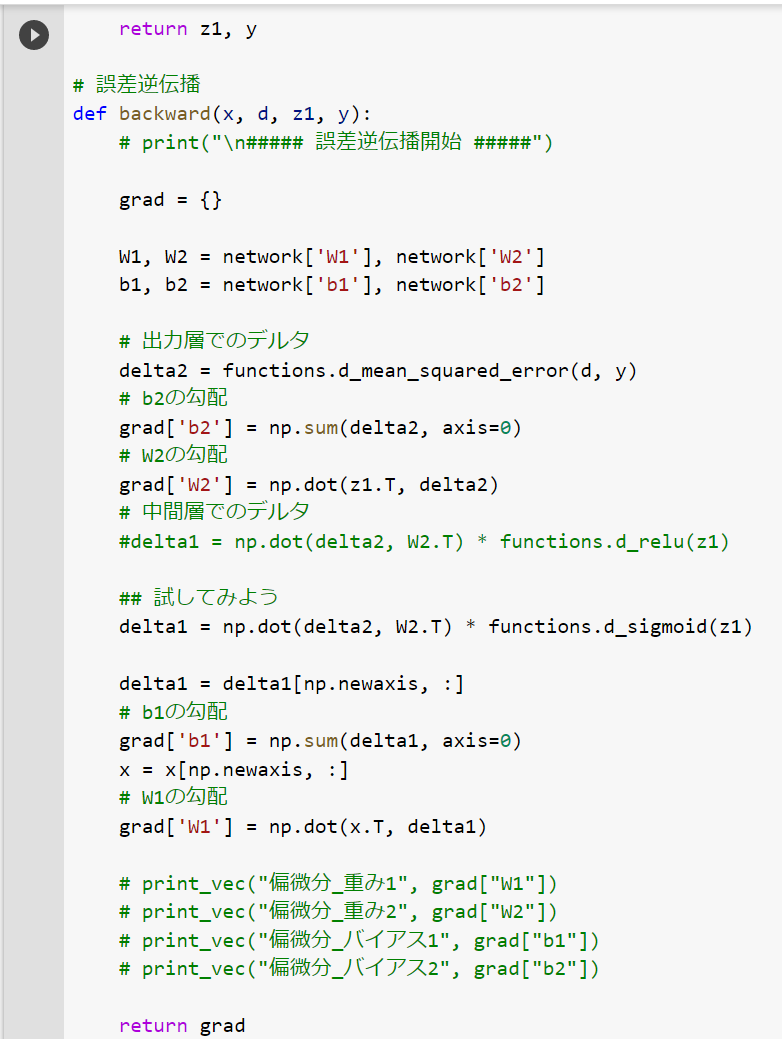
次に、前述のパラメータ更新の方法であり、数値微分法の欠点を補う、算出された誤差を、出力層側から順に微分し、前の層前の層へと伝播。最小限の計算で各パラメータでの微分値を解析的に計算する誤差逆伝搬法を学んだ。

〇確認テスト

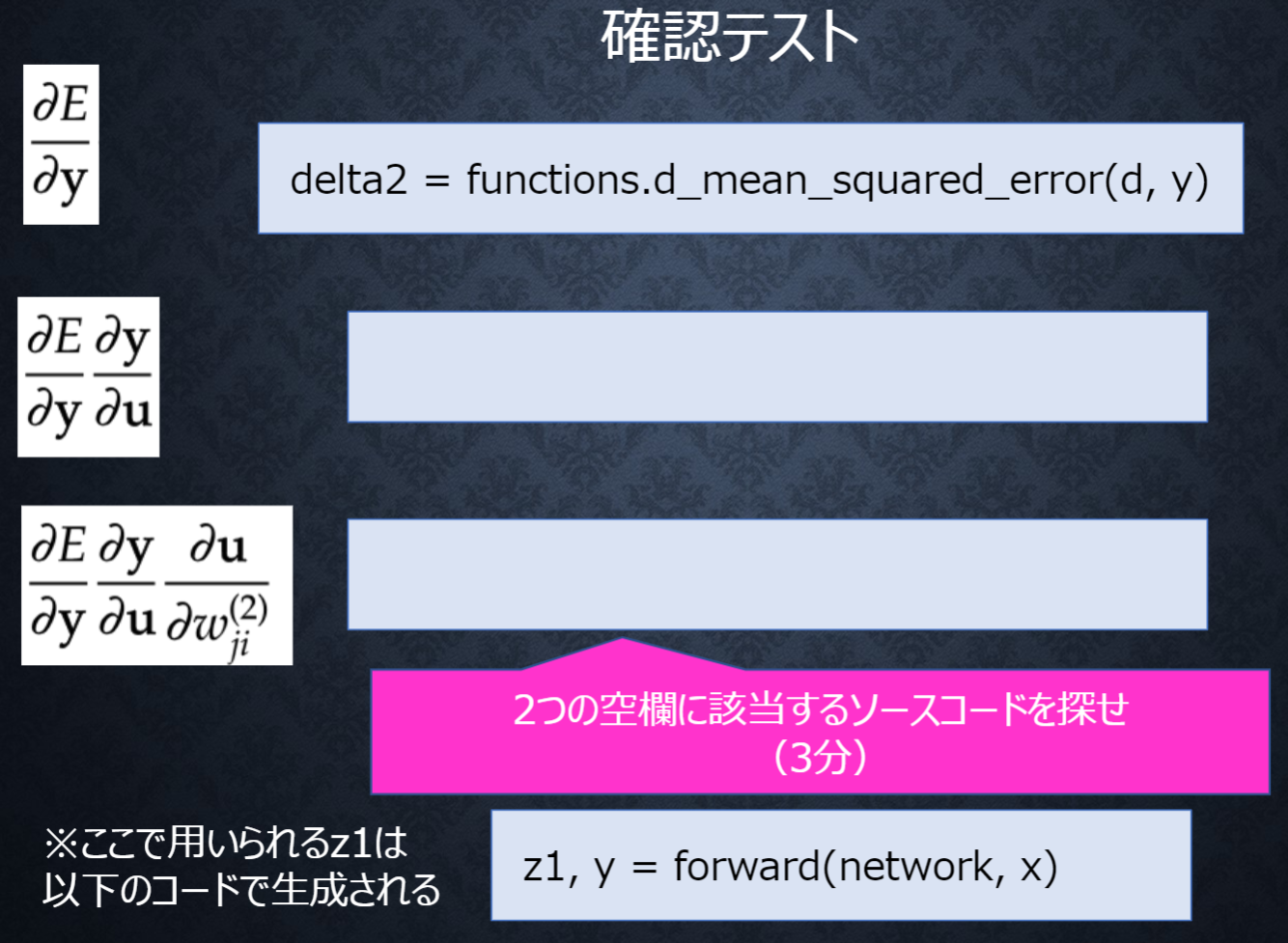
誤差逆伝播法では不要な再帰的処理を避ける事が出来る。既に行った計算結果を保持しているソースコードを抽出せよ。

答案：

下記の赤枠に示す。



〇確認テスト



答案：

下記の赤枠に示す。

