Deep Learning Seminar
~2~

渡邊研究室　M1 二見悠樹

今日の目標

パーセプトロンを拡張して，多層化にしよう

そんで多クラス分類を行おう

目次

1. 前回のおさらい・課題解説
2. Pythonとは何ぞや？
3. kerasとは？
4. ソフトマックス関数
5. NNの学習(パラメータの最適化)

前回のおさらい

単純なパーセプトロンを実装しました

課題の答え

出来たパーセプトロンのパラメータ

w: [2.14037745, 1.2763927]　b: -9

w[0]*x1 + w[1] *x2 +b = 0 という関数がニューロンが発火するかどうかの境界線になるのでこれを図示すればいい

Pythonとは？

インタプリタ言語の一種

コードを上から一行づつ読み込んでいく言語

逐次的に機械語に変換しながら同時に実行するため、メモリや速度を犠牲に・・・

ただし、Pythonは裏で高速なCやC++が動いている

とにかく分かり易い！！

「読みやすく、効率もよいコードをなるべく簡単に書けるようにする」を思想に

型の宣言をする必要がない（例外あり）

listを使うことで、データの処置・管理がやり易い

二つの系統が存在

2.x系：昔から開発がされており、今でも多くのライブラリがこれに準拠

3.x系：新しく開発が進んでいる系統、今から始めるならこちらがオススメ

インストール

PythonをWindowsにインストール

http://qiita.com/taipon_rock/items/f574dd2cddf8851fb02c

アナコンダという便利な物があるらしい w（あらゆるパッケージを一括ダウンロード）

PythonをLinuxにインストール

sudo yum -y install python (RedHat/CentOS)

sudo apt-get install python2.7 (Ubuntu/Debian系)

http://www.tohoho-web.com/python/start.html

https://hombre-nuevo.com/python/python0009/

PythonをMacにインストール

http://qiita.com/ms-rock/items/6e4498a5963f3d9c4a67

kerasとは？

- python用のライブラリ
- Tensorflow/theano のラッパーライブラリにあたる
- Googleの人が開発
- 日本語ドキュメントも充実してる（優しい）

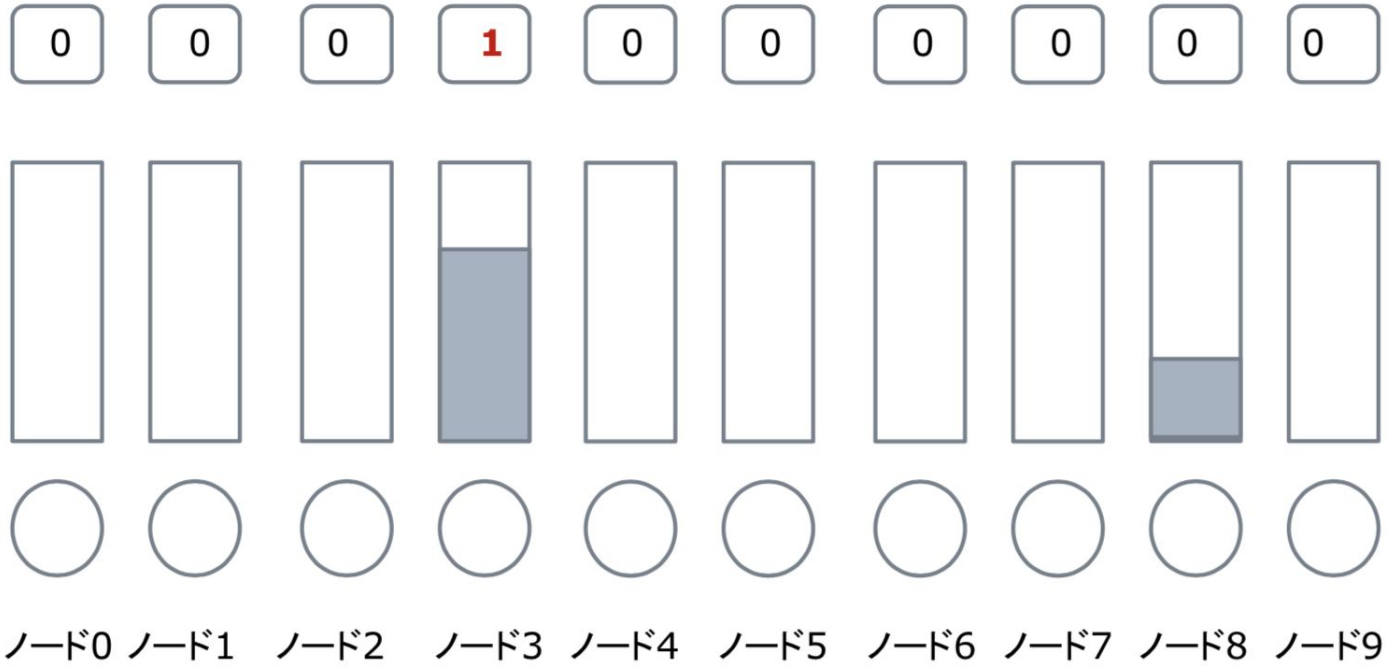
ソフトマックス関数

- クラス分けの場合に出力に利用される活性化関数(activator)
- MNIST(手書き数字画像)だと入力された画像が0～9までの10種類(10クラス)の数字のどれかだと判断しなければならない

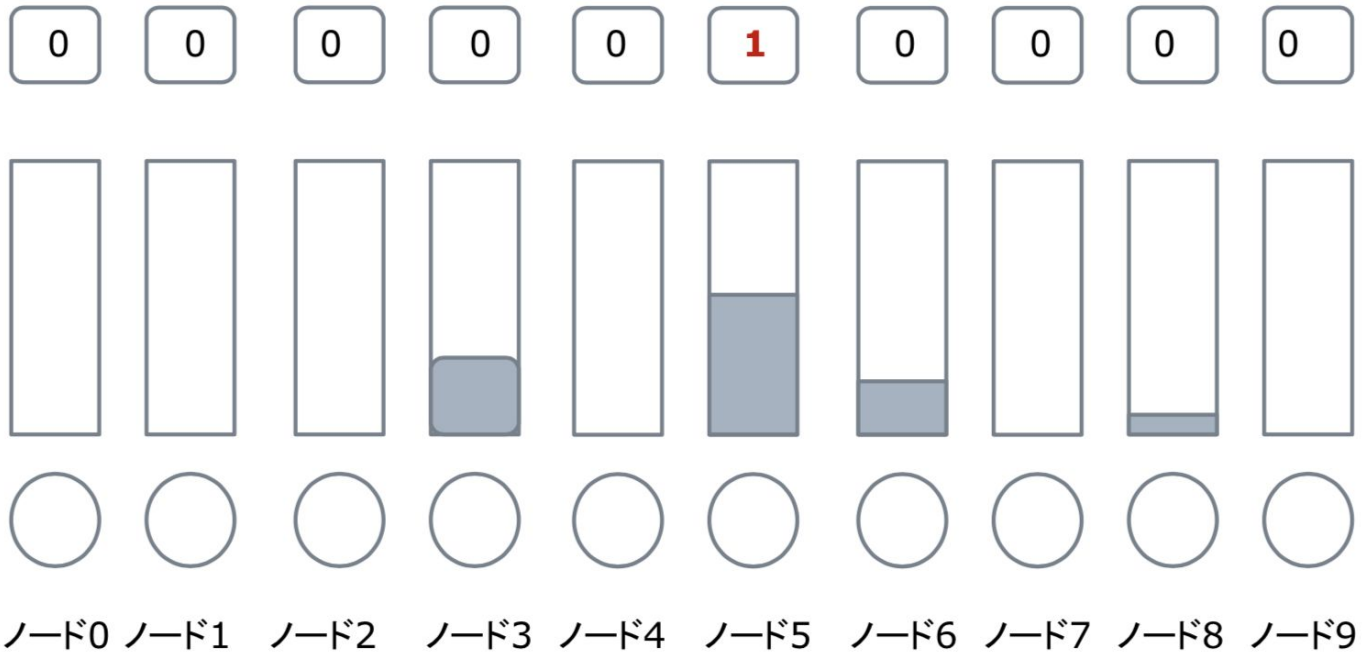
→ 出力層に0～9までの10クラスに対応したニューロンを並べそれぞれのノードの値が**「提示された画像がその数字である確率」**であるようにする

ニューロンごとに確率を計算するのがsoftmax関数で，　最終的に最も確率が最大であるものが選ばれる

softmax 関数の出力例

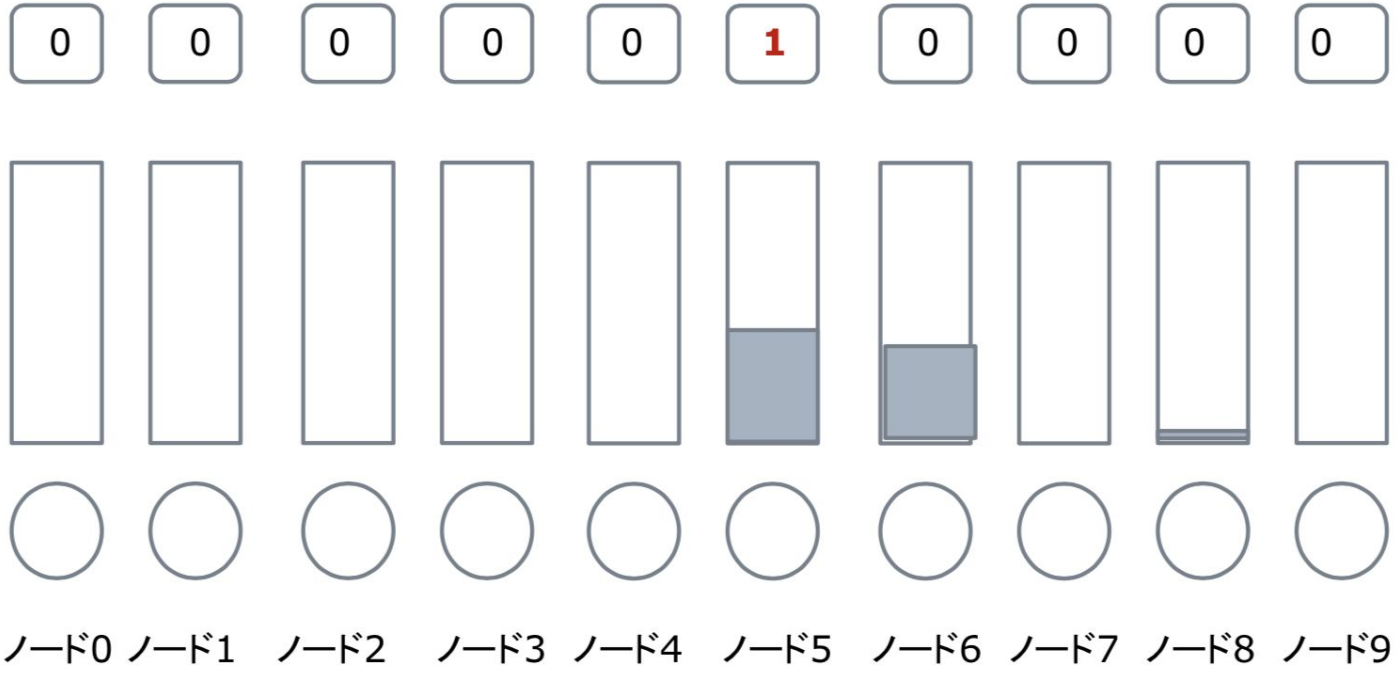


softmax 関数の出力例



出力の値を足したら1になる！

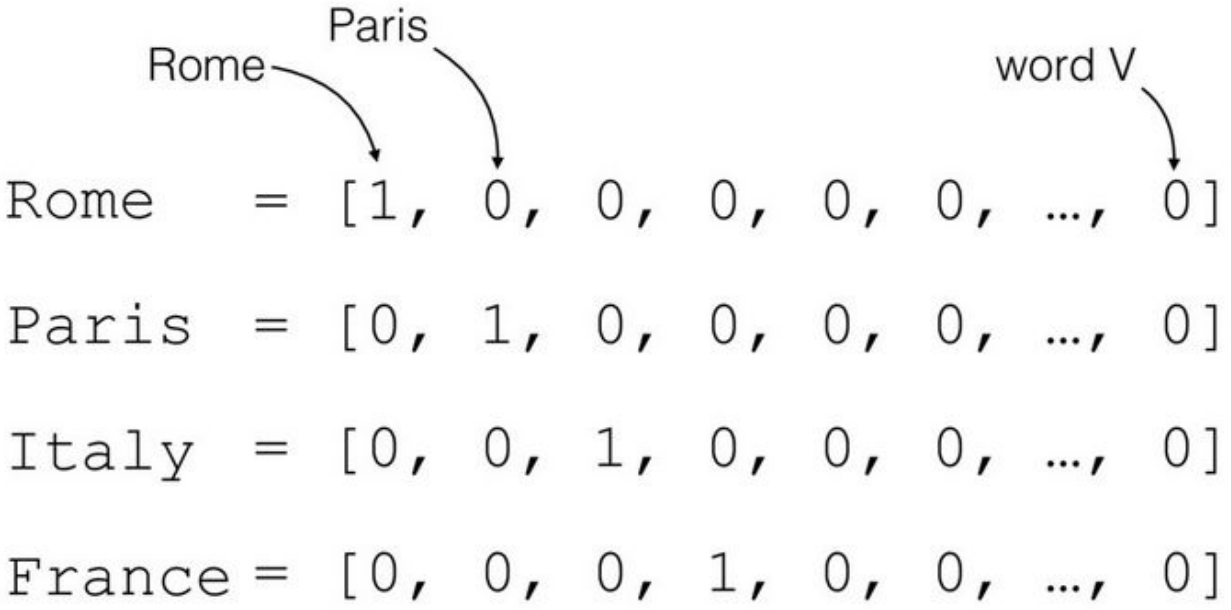
softmax 関数の出力例



1-of-K 表現 (one-hot dector)

- このようなクラス分類問題を解く際の教師ラベルには　正解クラスに**1**を残りは全て**0**になっている

→ 自前のデータを使うにはこのようなデータ整形が必要となる...



NNのパラメータ最適化(学習)

- 損失関数 (Loss Function)
- 勾配降下法 (Gradient Descent)
- クロスエントロピー

損失関数（Loss Function）

出力結果と正解(そうなって欲しい出力)とのずれ

- 出力結果と予期していた「正解とのズレ」を何らかの方法で定式化する必要がある
- そのズレを関数として表したものが**Loss Function(損失関数)**または**Cost Function(コスト関数)** と呼ぶ
- 出力が「正解」と一致した場合Lossの値が最小値となる

→ つまり，ニューラルネットの学習とはこのLoss値を最小にすることを目標としているだけ！？

Gradient Descent（勾配降下法）

- パラメータ更新で，最もよく使われる方法が，Gradient Descent(勾配降下法)
- Loss FunctionをJとする
- ニューロンのパラメータ(W:重み, b:バイアス)がそれぞれ少し変化した際に，このJがどう変化するかを調べる
- このJの変化が「勾配」であるJの値が下がるように，Wとbを変化させる.
- ∂J/∂W と ∂J/∂b をチェックすることににになる．Jはニューラルネットワークの出力と正解の「ズレ」なので　　φ(WX + b)の項を含む．活性化関数に微分可能な関数が好まれるのはこのため

Gradient Descent（勾配降下法）

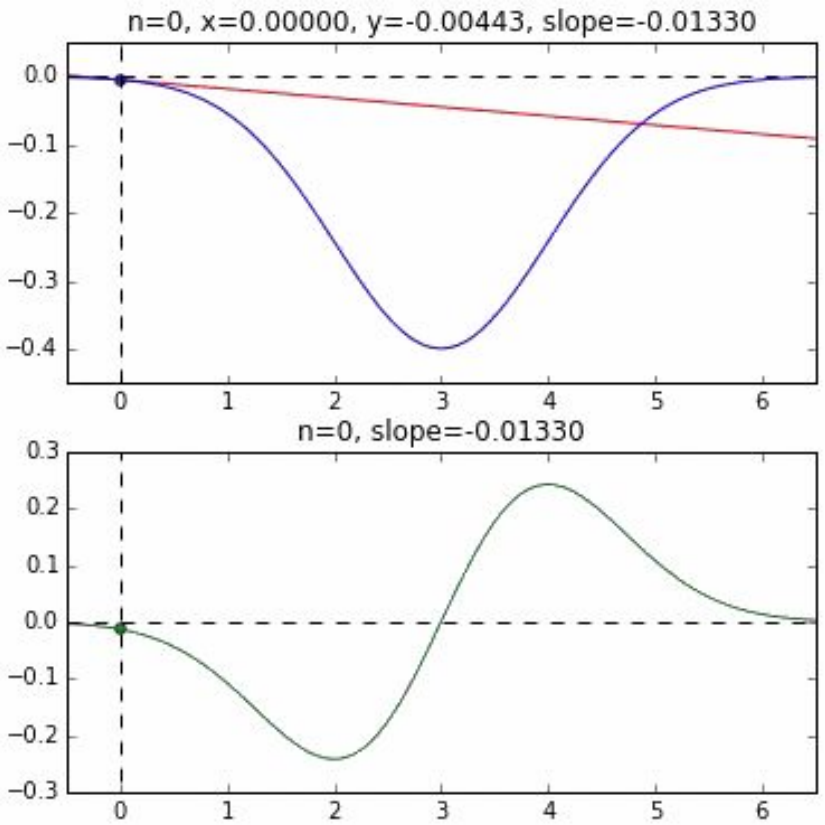
ある関数があって　$J(\theta_0, \theta_1)$

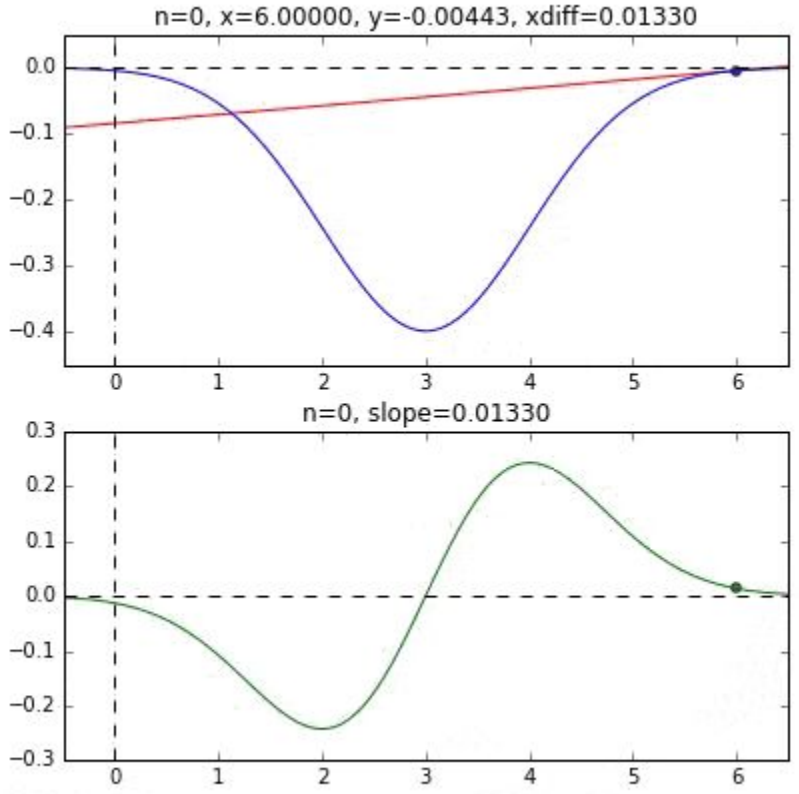
その値を最小にしたい　$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

基本的な考え方:

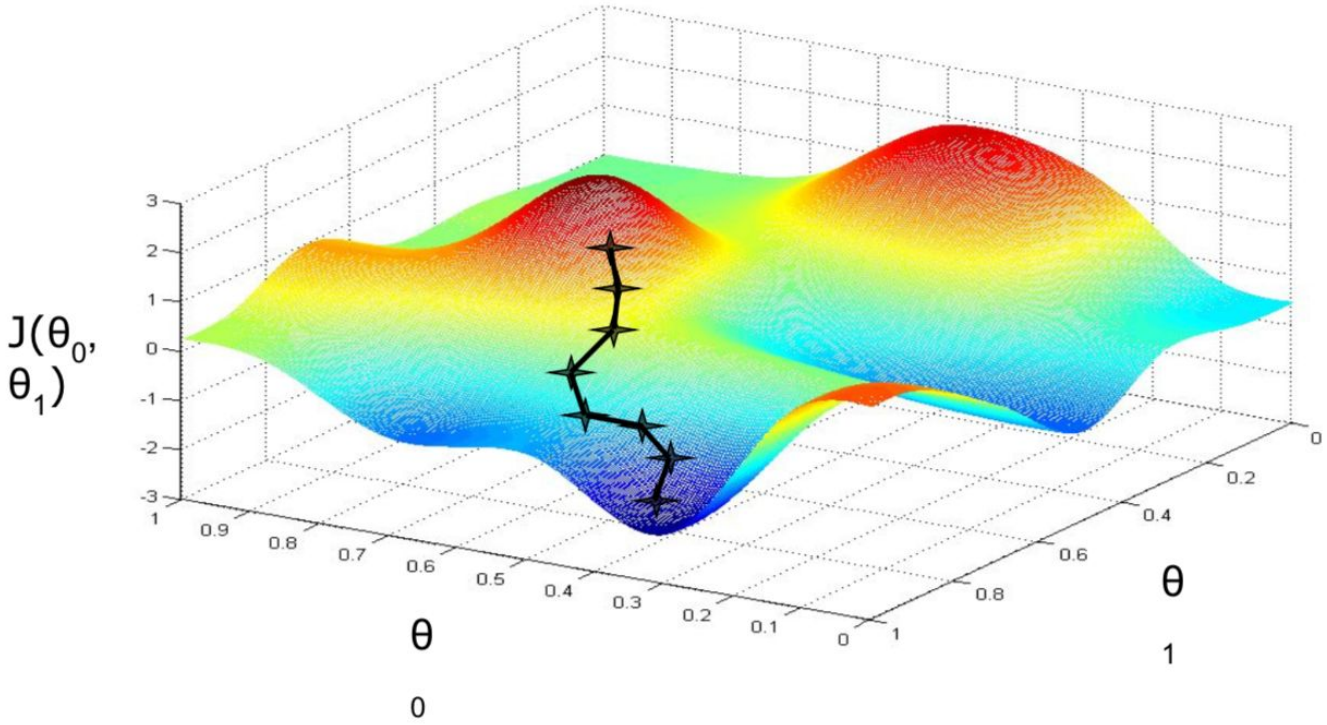
- ある　θ_0, θ_1　からはじめる。
- θ_0, θ_1　の値を、$J(\theta_0, \theta_1)$　が減少するように変化させる。
- 最小値になるまで、繰り返す。

すごくわかりやすいGif



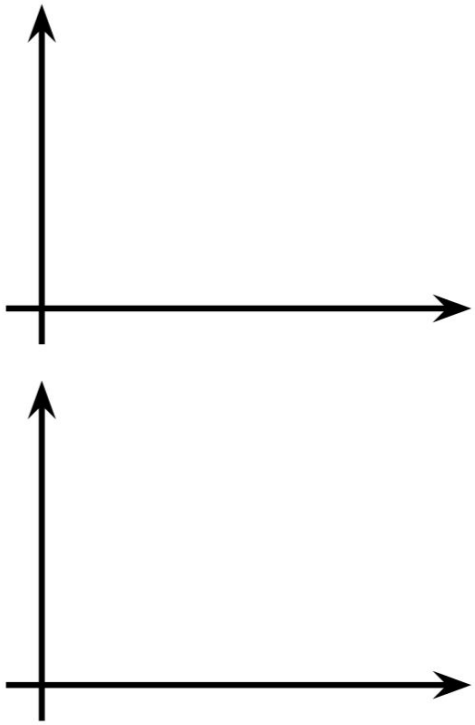


2次元に拡張すると...



kwsk

直感的な説明



簡単にするために、二次元のグラフで考えよう。ある点から始めて、その点から極小点に近ずくためには、どうすればいいか？

二つの場合がある。
その点が極小点の右側にある場合には、その点のx座標を少し減らせばいい。
その点が極小点の左側にある場合には、その点のx座標を少し増やせばいい。

その点が、極小点の右にあるか左にあるかは、その点での接戦の傾きが、正であるか負であるかでわかる。

よって、α を正の数値とすれば、次の式が、より近づいた点の位置を与える。

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

損失関数を、パラメーターで微分している

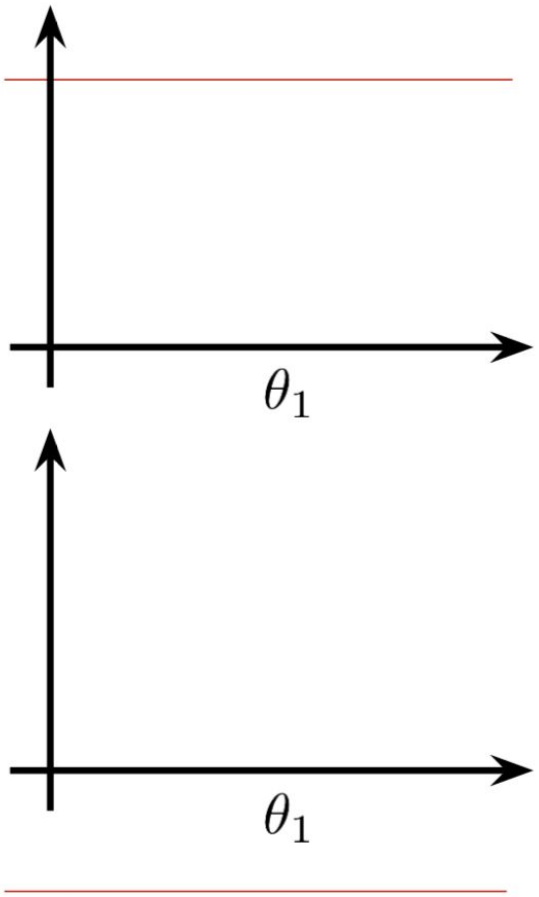
kwsk

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

このアルファを学習率
（Learning Rate）という。

アルファが、あまりに小さいと、Gradient descentは、ゆっくりにしか進まない。

アルファが、あまりに大きいと、Gradient descentは、最小値を通り越してしまう。場合によっては、振動したり、収束しない場合もある。



確率的勾配降下法

- ただの勾配降下法は学習データセット全体に対するパラメータについて，コスト関数の勾配を計算する

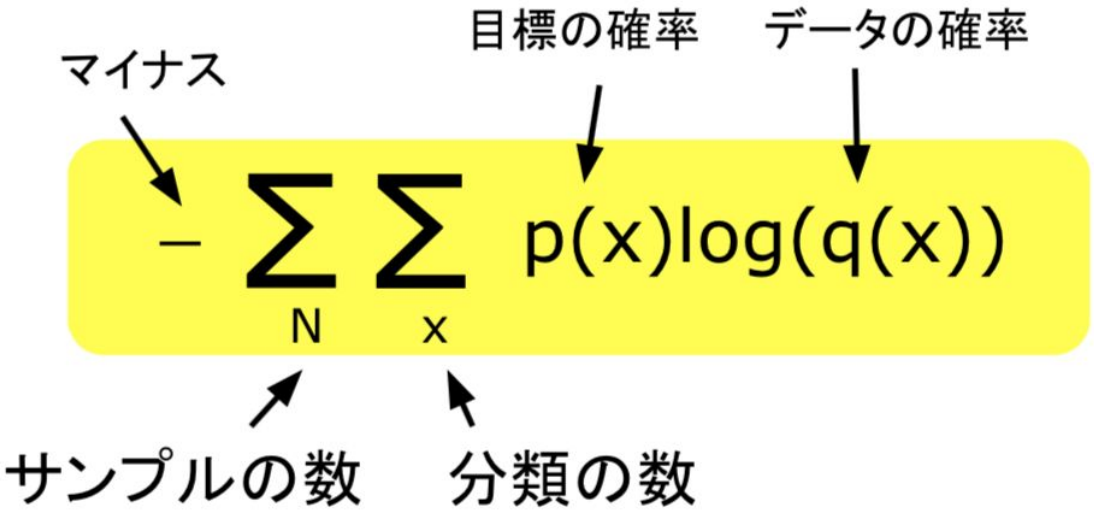
→ **つまり，たった1回の更新を行うのに全データセットに対して勾配を計算する必要がある．効率が悪い．アホ**

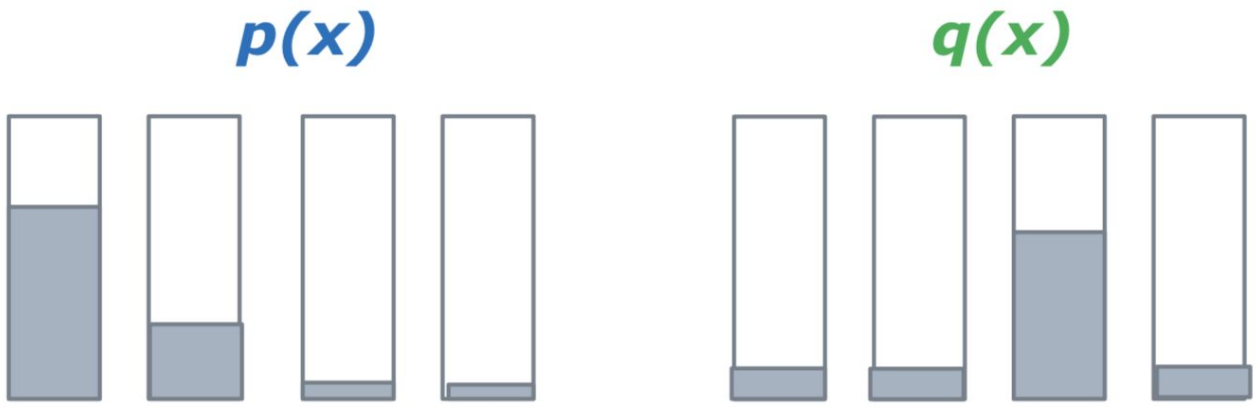
- 確率的勾配降下（Stochastic Gradient Descent；SGD）
  - データをランダムに1つ選んでパラメータを更新
- ミニバッチ勾配降下法
  - N個のデータをM(≦N)個ずつのかたまり(ミニバッチ)に分けて学習を行うもの．
  - 確率的勾配降下は**M=1のときと同等**なので，便宜上どちらの手法もひとまとめに確率的勾配降下と呼ぶことが多い

https://postd.cc/optimizing-gradient-descent/#gradientdescentvariants

クロスエントロピー

- さきほどのクラス分類によく使われるSoftmaxの出力の場合だと...
- 数字の場合は0~9の10クラスのノードに，ノードに対応した数字である確率が入ってくる.
- こうした場合の損失関数に利用されるのが，クロスエントロピーである





P(1)=2/3,p(2)=1/4,p(3)=5/24,p(4)=5/24　q(1)=1/8,q(2)=1/8, q(3)=5/8,q(4)=1/8

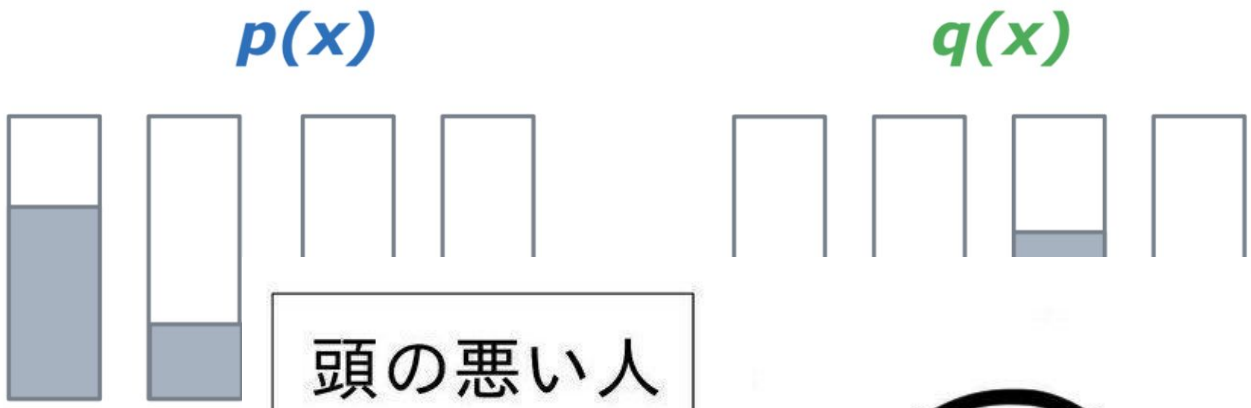
x=1の時　2/3log(1/8) = -2/3log8 = -2log2
x=2の時　1/4log(1/8) = -1/4log(8) = -3/4log2
x=3の時　5/24log(5/8)= 5/24log(5) - 15/24log2
x=4の時　5/24log(1/8)= -5/24log(8) = -15/24log2
全部足し合わせると
-(48log2+18log2-5log5+15log2+15log2)/24=-(96log2-5log5)/24

H(p,q) = (96log2-5log5)/24

$$\sum_{x} p(x)\log(q(x))$$

の計算

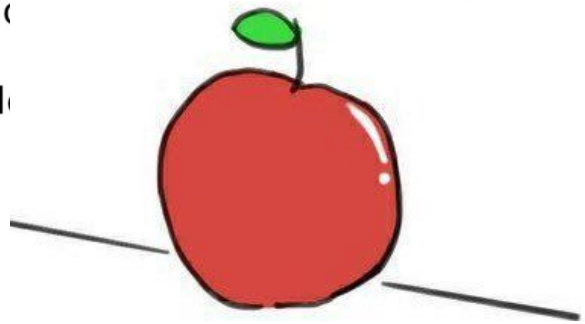
省略します



P(1)=2/3,p(2)=1/

x=1の時　2/3lo
x=2の時　1/4lo
x=3の時　5/24l
x=4の時　5/24l
全部足し合わせると
-(48log2+18l

H(p,q) = (96l



いろいろ言ったけど

kerasだとこんだけで実装できます

```
from keras.optimizers import SGD,
 optimizer=SGD(),
loss='categorical_crossentropy',
```

なので．．．

中の数式を理解するというよりはどういうアルゴリズムで，どういうときに使うものなのか？

という方が大事かも…(理解するに越したことはないが)

まとめ

モデルの学習を行うには...

1. モデルの出力を式で表す
2. 誤差関数を定義する
3. 誤差関数を最小化するべく，パラメータに対する勾配を求める
4. 勾配降下法により最適なパラメータを探索

MNIST（手書き文字）

機械学習関連のアルゴリズムのベンチマークデータ

何かモデルを構築した際は、このデータで動作確認すると良い

その前に...

MNISTのデータ構造について知ろう！

Tensorflowのチュートリアルがわかりやすい

https://www.tensorflow.org/versions/r1.1/get_started/mnist/beginners

次回予定

- Caltech101を使って100クラス分類をしよう！

- 今後つまづくであろうデータ整形に関するお話
- 学習経過を可視化出来るTensorboardの使い方

課題

自分で書いた数字画像を学習モデルに入力し，

モデルがどの数字と認識してるか出力させよう

ヒント:MNISTのデータ形式と同じように画像をデータ整形し，モデルに入力させる必要がある

model.predict()