1　ロジスティック回帰とは

・名前が紛らわしいが、回帰でなく分類手法になる

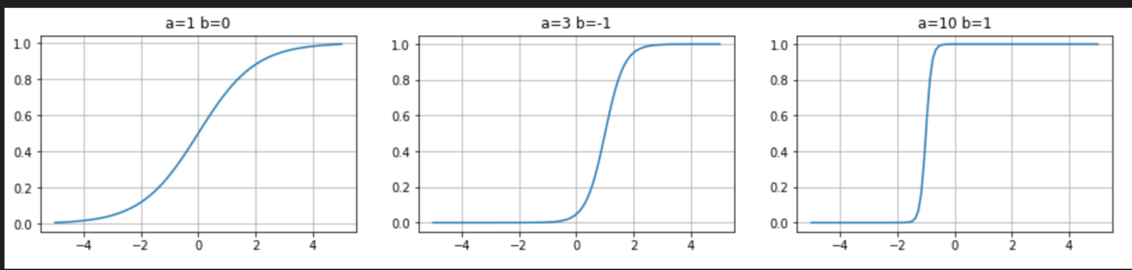
・二値分類モデル

・ラベルを 0 か 1 としてそのまま確率 (ラベル 1 である確率) として表現する

・モデルはシグモイド関数 (実際の $a(x+b)$ はパラメータ $\mathbf{w}$ を含む関数に置き換わる、と考える)

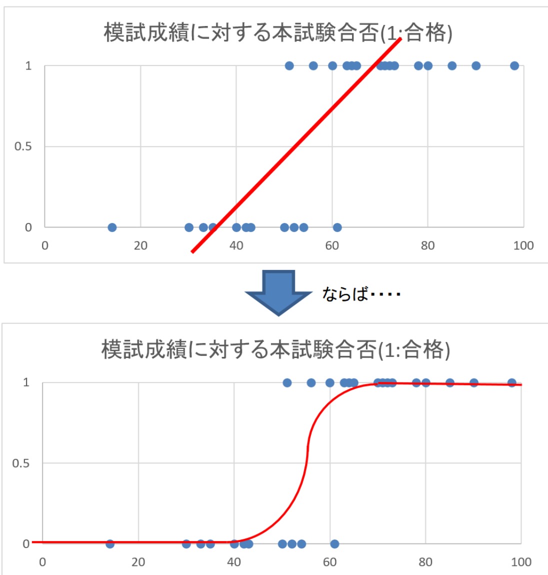
```python
def sigmoidFunc(x,a,b):
    return 1/(1 + np.exp(-a*(x + b)))

x = np.linspace(-5,5,100)
```



・以下の様な理解で良いはず (ex:模試成績に対する本試験合否分類)

模試成績(100点満点)	本試験合否
14	0
30	0
33	0
35	0
40	0
42	0
43	0
50	0
51	1
52	0
54	0
56	1
60	1
61	0
63	1
64	1
65	1
70	1
71	1
72	1
73	1
78	1
80	1
85	1
90	1
98	1



線形では説明できない
合否分岐を如何表現する？
↓
境界を確率で表現できれば都合よい

シグモイド関数で表現すれば如何か？
ということ

2 ロジスティック回帰モデル

・シグモイド関数を採用したもう一つの理由：微分式が簡潔になるから

・e^x の微分はそのまま e^x となる (オイラーが定義)

・この性質に対して微分の連鎖律を適用して計算を簡素にする

シグモイド関数式 $\sigma(x) = \frac{1}{1+exp(-ax)}$ の時

$\frac{d\sigma(x)}{dx} = \frac{d}{dx}(\frac{1}{1+exp(-ax)}) = (-1)\{1+exp(-ax)\}^{-2} \cdot exp(-ax) \cdot (-a)$

$= \frac{a \cdot exp(-ax)}{\{1+exp(-ax)\}^2} = \frac{a}{1+exp(-ax)} \cdot \frac{1+exp(-ax)-1}{1+exp(-ax)}$

$= a\sigma(x)(1-\sigma(x))$

・実際の x はパラメータとデータの線形結合となる (Y=1 となる確率に対応させる)

$P(Y = 1 \mid \mathbf{x}) = \sigma(w_0 + w_1x_1 + \cdots + w_mx_m)$

(切片はバイアスとして作用し 0 → 1 分岐の値を決める作用をしそう)

3 最尤推定法

・確率に対応させる:最尤推定を利用する

・最尤推定:目の前に見えた事象が発生する様な分布を推定する

・確率分布はベルヌーイ分布を採用 (離散確率分布で確率 p で 1、確率 1-p で 0 をとる)

$P(y) = p^y(1-p)^{1-y}$

・同時確率

あるデータが得られた時、それが同時に得られる確率

確率変数は独立である事を仮定すると、それぞれの確率の掛け算となる

(例１：サイコロを２つ用意し、同時に転がす (独立した事象)、片方が４以上、片方が偶数、

となる確率がそれぞれ 1/2 とする、同時確率は 1/2*1/2 で 1/4 となる)

(例２：サイコロを１つ用意し、転がす、４以上かつ偶数となる確率は

単純に 1/3 という事 (これは独立した事象でないので同時確率ではない))

・尤度関数

データは固定し、パラメータを変化させる

誘導関数を最大化する様なパラメータ p を選ぶ方法を最尤推定という

$P(y) = p^y(1-p)^{1-y}$ → 一回の試行で $y = y_1$ となる確率

$P(y_1, y_2, \ldots, y_n; p) = \prod_{i=1}^n p^{y_i}(1-p)^{1-y_i}$ →n 回試行で y_1〜y_n が同時に起こる確率 (p 固定)

$P(y_1, y_2, \ldots, y_n; p) = \prod_{i=1}^n p^{y_i}(1-p)^{1-y_i}$ →y_1〜y_n のデータが得られた際の尤度関数

p はシグモイド関数に置き換える $p = \sigma(\mathbf{w}^T\mathbf{x})$

この置き換えにより未知パラメータ $\mathbf{w}$ の推定問題となる

$P(y_1, y_2, \ldots, y_n; \mathbf{x}) = \prod_{i=1}^n \sigma(\mathbf{w}^T\mathbf{x}_i)^{y_i}(1-\sigma(\mathbf{w}^T\mathbf{x}_i))^{1-y_i}$

4 対数尤度関数

・尤度関数の積を微分するのは困難 $\rightarrow$ 対数尤度関数に置き換えて微分しやすくする

同時確率の積 $\rightarrow$ 対数を取ると和に変換可能なので指数演算は和算で計算可能になる

・尤度関数の最大点と対数尤度関数の最大点は同じ

$$L(\mathbf{w};\mathbf{x}) = \prod_{i=1}^{n} \sigma(\mathbf{w}^T\mathbf{x}_i)^{y_i}(1-\sigma(\mathbf{w}^T\mathbf{x}_i))^{1-y_i} \rightarrow \text{尤度関数}$$

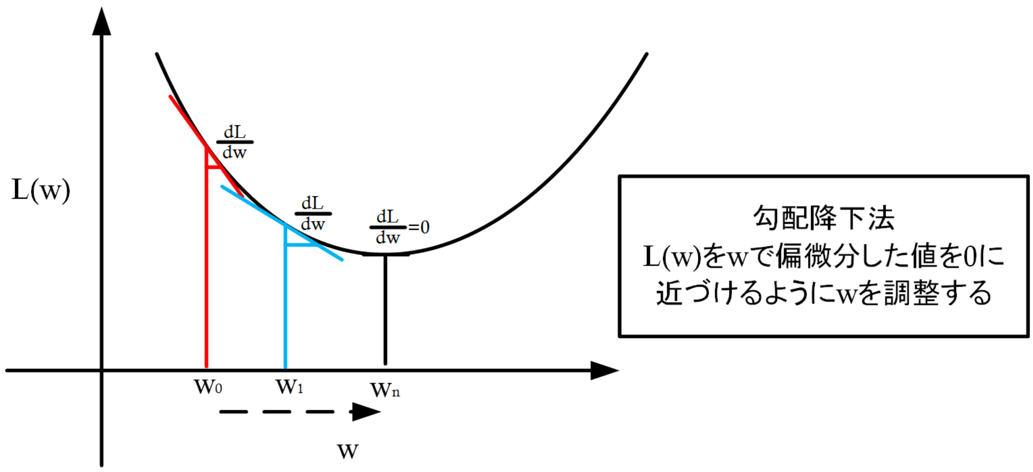
$$logL(\mathbf{w};\mathbf{x}) = \sum_{i=1}^{n}\{(y_i)log(\sigma(\mathbf{w}^T\mathbf{x}_i)) + (1-y_i)log(1-\sigma(\mathbf{w}^T\mathbf{x}_i))\} \rightarrow \text{対数尤度関数}$$

5 勾配降下法

・対数尤度関数の微分値が 0 をとるようなパラメータ $\mathbf{w}$ を学習する際に勾配降下法を用いる

・単純に「微分値つまり傾きが 0 に近づく様に $\mathbf{w}$ を調整していく」様なアルゴリズムとなる

・ロジスティック回帰の場合パラメータ $\mathbf{w}$ の二次式となり凸関数である事が保証される



・確率的勾配降下法 (SGD)

データを１つづつランダムに「確率的に」選んでパラメータを更新する方法

勾配降下法と比較して計算量が圧縮できる

DNN 等モデルの形が単純な凸関数で表現されない様な場合 (谷底が複数あり最良解をそこから探す) にこの手法をとる場合がある

6 性能指標

・混同行列 (Confusion Matrix)

検証用データをモデルで予測した結果をまとめた表

後述するがこの指標だけでは「用途に適したモデル」か如何か判断しにくい

単純な成績 (数値) としては評価しやすくなる

confusion matrix(混同行列)
(モデルの予測結果を個数で評価)

		検証用データの結果	
		positive	negative
モデルの予測結果	positive	真陽性 (True Positive)	偽陰性 (False Positive)
	negative	偽陽性 (False Negative)	真陰性 (True Negative)

正しい予測の個数

間違った予測の個数

・正解率

モデルが正しく予測した割合を評価 (単純に以下の通り、正解数を全数で割る)

$\frac{TP+TN}{TP+FN+FP+TN}$

これも単純な数字としては指標になるが大体のケースで不十分

・再現率 (Recall)

本当に Positive なデータからモデルが Positive と判定できた確率

$\frac{TP}{TP+FN}$

・適合率 (Precision)

モデルが Positive と予想したものの中で本当に Positive である確率

$\frac{TP}{TP+FP}$

・おさえるべきポイント

1. 例えば病気診断モデルとした場合、疾患を見落とすのが致命的欠陥とみなされる
   - → 病気じゃない人を病気と誤判定した方が安全 (再検査すれば良い)
   - → この場合再現率 (Recall) 重視で評価すべき
2. 例えばスパムメール判別器モデルとした場合、本当にスパムメールであるものを検出したい
   - → この場合検出漏れは許容できるので (逆に普通のメールをスパム判定したくない)
   - → 適合率 (Precision) 重視で評価すべき

等、用途に応じてそれを評価する様にしましょう、という理解をすること

7 実習

ハンズオンコードに対して課題を追加実施 (提出コードに追記した)

コードは本ドキュメントと同じレポジトリに提出する

`https://github.com/toruuno/report_ml/blob/master/skl_logistic_regression.ipynb`