人間と情報システム II 第 13 回　1116202017 賈書瑞

(画像は最後にある)

2 次元ベクトルにおける 4 クラス識別器の設計

１．はじめに

機械学習は本質的に標本xからラベル y への変換$f: y = f(x, w)$（ここでwはモデルパラメータ）を設定し、最適化する作業と思われる。変換$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{w})$が最も単純な線形関数を使用する場合は、線形識別関数が得られる。

２．約束

訓練データセット：$\{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^D, 1 \le i \le N, N = 1000, D = 2\}$

標本$\boldsymbol{x}_i$に対応するクラスのラベル：$y_i (y_i \in \mathbb{Z}, 1 \le y_i \le K, K = 4)$

訓練データセットの標本総数は N、標本の次元は 2、クラス総数は 4 である。

３．線形識別関数

標本$\boldsymbol{x}_i$とパラメータwの線形識別関数の形式は

$$s = f(\boldsymbol{x}_i, \boldsymbol{w}, b) = x_{i1} w_1 + x_{i2} w_2 + \cdots + x_{iD} w_D + b$$

ここでsはスコアで、$w = (w_1, w_2, \ldots, w_D)^T \in \mathbb{R}^D$、$b$は定数項である。$s$はスカラーなので 1 クラスのスコアしか表せない。K個のクラス全てのスコアを得るためにK個の線形関数が必要であり

$$s_j = f(\boldsymbol{x}_i, \boldsymbol{w}, b) = w_{j1} x_{i1} + w_{j2} x_{i2} + \cdots + w_{jD} x_{iD} + b_1$$

j番目の方程式はクラスjのスコアを表す。行列でまとめて書き換えると

$$\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix} = \boldsymbol{s}_i = \begin{bmatrix} \boldsymbol{w}_1^T \boldsymbol{x}_i + b_1 \\ \boldsymbol{w}_2^T \boldsymbol{x}_i + b_2 \\ \vdots \\ \boldsymbol{w}_K^T \boldsymbol{x}_i + b_K \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}_1^T, b_1 \\ \boldsymbol{w}_2^T, b_2 \\ \vdots \\ \boldsymbol{w}_K^T, b_K \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix} \ (1 \le i \le N).$$

さらに

$$\begin{bmatrix} s_{11} & s_{21} & \cdots & s_{N1} \\ s_{12} & s_{22} & \cdots & s_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ s_{K1} & s_{K2} & \cdots & S_{KN} \end{bmatrix} = [\boldsymbol{s}_1 \boldsymbol{s}_2 \cdots \boldsymbol{s}_N] = \begin{bmatrix} \boldsymbol{w}_1^T, b_1 \\ \boldsymbol{w}_2^T, b_2 \\ \vdots \\ \boldsymbol{w}_K^T, b_K \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \boldsymbol{x}_2 \cdots \boldsymbol{x}_N \\ 1 \ 1 \cdots \ 1 \end{bmatrix}$$

すなわち

$$S = WX.$$

と書ける。ここでSはスコア行列。

最適化アルゴリズムはこの重み係数行列Wに対して行われる。

４．損失関数（loss function）

損失関数は S の質を評価する。S と希望されるラベルの差異が小さいほど損失関数の値が小さくなる。ベクトル$\boldsymbol{x}$に対して

$$\text{softmax}(\boldsymbol{x}) = \frac{e^{\boldsymbol{x}}}{\boldsymbol{1}^T e^{\boldsymbol{x}}}$$

とする。**関数値の成分は全て(0,1)上の実数なので「確率」の目安として捉えて良いだろう**。ただし、$\boldsymbol{1}$は全ての成分が 1 の列ベクトルとする。

K次元の列ベクトル$\hat{\boldsymbol{y}}_i$をy_iの one-hot ベクトル表現として

$$L_i = -\hat{\boldsymbol{y}}_i^T \log \text{softmax}(\boldsymbol{s}_i) \ (1 \le i \le N)$$

が標本$\boldsymbol{x}_i$における損失である。全体の損失はL_iの平均値である。

5． 重み係数の最適化

最急降下法（gradient descent method）を使う。まずは L の勾配を求める。

$$
\begin{aligned}
L_i &= -\hat{\boldsymbol{y}}_i^T \log \operatorname{softmax}(\boldsymbol{s}_i) \\
&= -\hat{\boldsymbol{y}}_i^T (\log e^{\boldsymbol{s}_i} - \mathbf{1} \log (\mathbf{1}^T e^{\boldsymbol{s}_i}) \\
&= -\hat{\boldsymbol{y}}_i^T \log(e^{\boldsymbol{s}_i}) + \log(\mathbf{1}^T e^{\boldsymbol{s}_i}) \ (\because \hat{\boldsymbol{y}}_i^T \mathbf{1} = 1) \\
&= -\hat{\boldsymbol{y}}_i^T \boldsymbol{s}_i + \log(\mathbf{1}^T e^{\boldsymbol{s}_i}),
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{d}L_i &= -\hat{\boldsymbol{y}}_i^T \mathrm{d}\boldsymbol{s}_i + \frac{\mathbf{1}^T (e^{\boldsymbol{s}_i \odot \mathrm{d}\boldsymbol{s}_i})}{\mathbf{1}^T e^{\boldsymbol{s}_i}} \\
&= -\hat{\boldsymbol{y}}_i^T \mathrm{d}\boldsymbol{s}_i + \frac{(e^{\boldsymbol{s}_i})^T \mathrm{d}s_i}{\mathbf{1}^T e^{\boldsymbol{s}_i}} \\
&= tr\left(-\hat{\boldsymbol{y}}_i^T \mathrm{d}\boldsymbol{s}_i + \operatorname{softmax}(\boldsymbol{s}_i)^T \mathrm{d}\boldsymbol{s}_i\right) \\
&= tr\left(\left(\operatorname{softmax}(\boldsymbol{s}_i)^T - \hat{\boldsymbol{y}}_i^T\right)\mathrm{d}\boldsymbol{s}_i\right) \\
&= tr\left((\operatorname{softmax}(\boldsymbol{s}_i) - \hat{\boldsymbol{y}}_i)^T \mathrm{d}W \begin{bmatrix}\boldsymbol{x}_i \\ 1\end{bmatrix}\right) \\
&= tr\left(\begin{bmatrix}\boldsymbol{x}_i \\ 1\end{bmatrix} (\operatorname{softmax}(\boldsymbol{s}_i) - \hat{\boldsymbol{y}}_i)^T \mathrm{d}W\right)
\end{aligned}
$$

よって

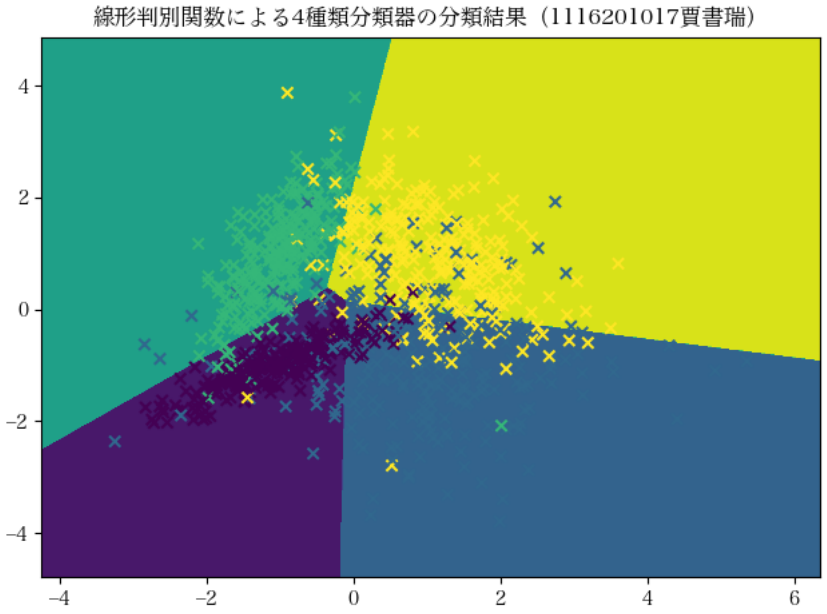
$$
\frac{\partial L_i}{\partial W} = (\operatorname{softmax}(\boldsymbol{s}_i) - \hat{\boldsymbol{y}}_i) \begin{bmatrix}\boldsymbol{x}_i \\ 1\end{bmatrix}^T
$$

$$
\operatorname{grad} L = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial L_i}{\partial W}
$$

$$
= \frac{1}{N} [\operatorname{softmax}(\boldsymbol{s}_1) - \hat{\boldsymbol{y}}_1 \quad \dots \quad \operatorname{softmax}(\boldsymbol{s}_N) - \hat{\boldsymbol{y}}_N] \begin{bmatrix} \boldsymbol{x}_1^T & 1 \\ \boldsymbol{x}_2^T & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_N^T & 1 \end{bmatrix}
$$

$$
= \frac{1}{N} \left(\operatorname{softmax}(S) - \hat{Y}\right) X^T.
$$

が得られる。これにより訓練すると、線形識別可能なデータに対してほぼ 100%の識別率が達成できる。実際の訓練では、線形識別不可なデータでも 81.3%が出た。



参考文献

[1] https://ja.wikipedia.org/wiki/最急降下法

[2] https://www.ituring.com.cn/book/tupubarticle/25626

[3] https://zhuanlan.zhihu.com/p/24709748（行列に対する微分）

[4] https://numpy.org/doc/stable/

[5] https://blog.csdn.net/mqq9931/article/details/83829849