先端データ解析論　第 7 回レポート

電子情報学専攻　48-176403 石毛真修

2017 年 5 月 30 日

大問 1.

ガウスカーネルモデルを用いた、最小二乗確率的分類の実装

```python
import numpy as np
from numpy.random import randn
from matplotlib import pyplot


class DataGenerator:
    def __init__(self, n=90, c=3):
        assert(n % c == 0)
        length = n // c
        self._classes = np.arange(1, c + 1)
        self._y = np.array([np.ones(length) * i for i in range(1, c + 1)])
        self._x = randn(c, length) + np.array([np.linspace(-3, 3, c)] * length).T

    @property
    def x_1d(self):
        return self._x.reshape(-1)

    @property
    def y_1d(self):
        return self._y.reshape(-1)

    @property
    def classes(self) -> list:
        return list(self._classes)


class GauseKernelRegression:
    H = .2
    LAMBDA = .1

    def __init__(self, data: DataGenerator):
        self._data = data
        self._K = self._build_K(data.x_1d)                                          c
        self._thetas = dict.fromkeys(data.classes, None)

    def _kernel(self, x: np.array, c: np.array) -> np.array:
        diff = np.abs(x - c)
        return np.exp(- diff**2 / (2 * self.H**2))

    def _build_K(self, X: np.array) -> np.array:
        return np.array([self._line_kernel(v) for v in X])

    def _line_kernel(self, v: float) -> np.array:
        vec = np.ones(self._data.x_1d.shape) * v
        return self._kernel(vec, self._data.x_1d)

    def _theta(self, label: int) -> np.array:
        if self._thetas[label] is not None:
            return self._thetas[label]
        else:
            Q = self._K.T @ self._K + self.LAMBDA * np.identity(len(self._data.x_1d))
            inv_Q = np.linalg.inv(Q)
            th = inv_Q @ self._K.T @ (self._data.y_1d == label)
            self._thetas[label] = th
            return th

    def _psudo_prob(self, x: float, y: int) -> float:
        return self._line_kernel(x) @ self._theta(y)

    def probs(self, x: float) -> np.array:
        ps = np.array([max(0, self._psudo_prob(x, y)) for y in self._data.classes])
        return ps / np.sum(ps)

        data = DataGenerator(n=90, c=3)
        gkr = GauseKernelRegression(data)
```

```
# Create Data and plot
N = 100
X = np.linspace(-5, 5, N)
probs = np.array([gkr.probs(x) for x in X])

axes = plt.gca()
axes.set_xlim([-5, 5])
axes.set_ylim([-.5, 1.5])

plt.plot(X, probs[:, 0], 'b')
plt.plot(X, probs[:, 1], 'r--')
plt.plot(X, probs[:, 2], 'g|')

plt.plot(data._x[0], -.1 * np.ones(data._x.shape[1]), 'bo')
plt.plot(data._x[1], -.2 * np.ones(data._x.shape[1]), 'rx')
plt.plot(data._x[2], -.1 * np.ones(data._x.shape[1]), 'gv')
```

c

実行すると次のような結果を得る.

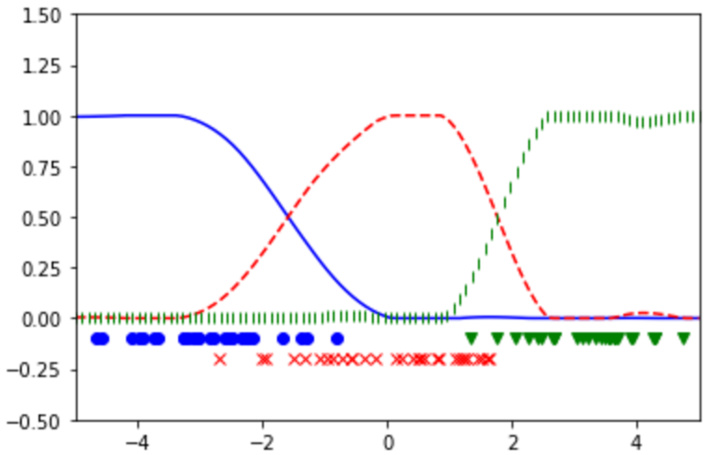

図 1: ガウスカーネルモデルによる最小自乗確率的分類の結果

大問 2.

$$B_\tau(y) = \sum_{y^{(\tau+1)},\ldots,y^{(m_i)}=1}^{c} exp\left(\sum_{k=\tau+2}^{m_i} \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) + \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right)$$

を次のような再帰的な表現に変形する。

$$B_\tau(y^{(\tau)}) = \sum_{y^{(\tau+1)}=1}^{c} B_{\tau+1}(y^{(\tau+1)}) exp\left(\zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right)$$

証明

$$\begin{aligned}
B_\tau(y^{(\tau)}) &= \sum_{y^{(\tau+1)},\ldots,y^{(m_i)}=1}^{c} exp\left(\sum_{k=\tau+2}^{m_i} \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) + \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right) \\
&= \sum_{y^{(\tau+1)}=1}^{c} \sum_{y^{(\tau+2)},\ldots,y^{(m_i)}=1}^{c} exp\left(\sum_{k=\tau+3}^{m_i} \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) + \zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right) \\
&\qquad exp\left(\zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right) \\
&= \sum_{y^{(\tau+1)}=1}^{c} B_{\tau+1}(y^{(\tau+1)}) exp\left(\zeta^{\mathrm{T}}\varphi(\mathbf{x}_i^{(\tau+1)}, y^{(\tau+1)}, y^{(\tau)})\right)
\end{aligned}$$

大問 3.

$$P_\tau(y) = \max_{y^{(1)},\ldots,y^{(\tau-1)}\in\{1,\ldots,c\}} \left[\sum_{k=1}^{\tau-1} \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) + \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau)}, y^{(\tau)}, y^{(\tau-1)}) \right]$$

を次のような再帰的な表現に変形する。

$$P_\tau(y^{(\tau)}) = \max_{y^{(\tau-1)}\in\{1,\ldots,c\}} \left[P_{\tau-1}(y^{(\tau-1)}) + \zeta^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau)}, y^{(\tau)}, y^{(\tau-1)}) \right]$$

証明

$$\begin{aligned}
P_\tau(y) &= \max_{y^{(1)},\ldots,y^{(\tau-1)}\in\{1,\ldots,c\}} \left[\sum_{k=1}^{\tau-1} \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) + \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau)}, y^{(\tau)}, y^{(\tau-1)}) \right] \\
&= \max_{y^{(\tau-1)}\in\{1,\ldots,c\}} \Bigg[\hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau)}, y^{(\tau)}, y^{(\tau-1)}) \\
&\quad + \max_{y^{(1)},\ldots,y^{(\tau-2)}\in\{1,\ldots,c\}} \left[\sum_{k=1}^{\tau-1} \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(k)}, y^{(k)}, y^{(k-1)}) \hat{\zeta}^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau-1)}, y^{(\tau-1)}, y^{(\tau-2)}) \right] \Bigg] \\
&= \max_{y^{(\tau-1)}\in\{1,\ldots,c\}} \left[P_{\tau-1}(y^{(\tau-1)}) + \zeta^{\mathrm{T}} \varphi(\mathbf{x}_i^{(\tau)}, y^{(\tau)}, y^{(\tau-1)}) \right]
\end{aligned}$$