先端データ解析論
第十回　レポート

37-196360　森田涼介

2019 年 6 月 26 日

宿題 1

$$\boldsymbol{x}_i \in \mathbb{R}^d,\ \boldsymbol{T} \in \mathbb{R}^{m\times d},\ \boldsymbol{W} \in \mathbb{R}^{n\times n},\ \boldsymbol{W}^{\mathrm{T}} = \boldsymbol{W} \tag{1}$$

$$\boldsymbol{X} = (\boldsymbol{x}_1,\ \cdots,\ \boldsymbol{x}_n) \tag{2}$$

$$\boldsymbol{D} = \mathrm{diag}\left(\sum_{i'=1}^{n} W_{i,i'}\right) \tag{3}$$

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{4}$$

に対し，

$$\sum_{i,i'=1}^{n} W_{i,i'}||\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{x}_{i'}||^2 = 2\mathrm{tr}\left(\boldsymbol{T}\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{T}^{\mathrm{T}}\right) \tag{5}$$

が成立することを示す。

いま，

$$\boldsymbol{z}_i = \boldsymbol{T}\boldsymbol{x}_i \tag{6}$$

とおくと，

$$\begin{aligned}
\sum_{i,i'=1}^{n} W_{i,i'}||\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{x}_{i'}||^2 &= \sum_{i,i'=1}^{n} W_{i,i'}||\boldsymbol{z}_i - \boldsymbol{z}_{i'}||^2 \\
&= \sum_{i,i'=1}^{n} W_{i,i'}(\boldsymbol{z}_i - \boldsymbol{z}_{i'})^{\mathrm{T}}(\boldsymbol{z}_i - \boldsymbol{z}_{i'}) \\
&= \sum_{i,i'=1}^{n} W_{i,i'}\left\{||\boldsymbol{z}_i||^2 + ||\boldsymbol{z}_{i'}||^2 - 2\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_{i'}\right\}
\end{aligned}$$

ここで，$W_{i,i'} = W_{i',i}$ なので，

$$\begin{aligned}
\sum_{i,i'=1}^{n} W_{i,i'}||\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{x}_{i'}||^2 &= 2\sum_{i,i'=1}^{n} W_{i,i'}\left(||\boldsymbol{z}_i||^2 - \boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_{i'}\right) \\
&= 2\left\{\sum_{i=1}^{n}\left(\sum_{i'=1}^{n} W_{i,i'}||\boldsymbol{z}_i||^2\right) - \sum_{i,i'=1}^{n} W_{i,i'}\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_{i'}\right\}
\end{aligned} \tag{7}$$

いま，

$$\boldsymbol{Z} = (\boldsymbol{z}_1,\ \cdots,\ \boldsymbol{z}_n) \tag{8}$$

を考えると，

$$\mathrm{tr}\left(\boldsymbol{Z}\boldsymbol{D}\boldsymbol{Z}^{\mathrm{T}}\right) = \sum_{i=1}^{n}\left(\sum_{i'=1}^{n} W_{i,i'}||\boldsymbol{z}_i||^2\right) \tag{9}$$

となり，また，

$$\mathrm{tr}\left(\boldsymbol{Z}\boldsymbol{W}\boldsymbol{Z}^{\mathrm{T}}\right) = \sum_{i,i'=1}^{n} W_{i,i'}\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_{i'} \tag{10}$$

となる。

式 (7)，(9)，(10) と $\boldsymbol{Z} = \boldsymbol{TX}$ より，

$$\sum_{i,i'=1}^{n} W_{i,i'} ||\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{x}_{i'}||^2 = 2\mathrm{tr}\left(\boldsymbol{Z}\boldsymbol{D}\boldsymbol{Z}^{\mathrm{T}}\right) - 2\mathrm{tr}\left(\boldsymbol{Z}\boldsymbol{W}\boldsymbol{Z}^{\mathrm{T}}\right) \tag{11}$$

$$= 2\mathrm{tr}\left(\boldsymbol{Z}\boldsymbol{L}\boldsymbol{Z}^{\mathrm{T}}\right) \tag{12}$$

$$= 2\mathrm{tr}\left(\boldsymbol{T}\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{T}^{\mathrm{T}}\right) \tag{13}$$

となって，式 (5) が示された。

宿題 2

適当な類似度行列に対して局所性保存射影を実装する。

目的関数は，

$$J = \frac{1}{2}\sum_{i,i'=1}^{n} W_{i,i'}||\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{x}_{i'}||^2 = \mathrm{tr}\left(\boldsymbol{T}\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{T}^{\mathrm{T}}\right) \tag{14}$$

であり，拘束条件は

$$\boldsymbol{T}\boldsymbol{X}\boldsymbol{D}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{T}^{\mathrm{T}} = \boldsymbol{I} \tag{15}$$

である。J を最小にするような $\boldsymbol{T}$ を求めることを考える。これはまず，次の一般化固有値問題を解けばよい。

$$\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{\xi} = \lambda\boldsymbol{X}\boldsymbol{D}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{\xi} \tag{16}$$

$$\lambda_1 \geq \cdots \geq \lambda_d \tag{17}$$

$$\boldsymbol{\xi}_j^{\mathrm{T}}\boldsymbol{X}\boldsymbol{D}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{\xi}_j = 1 \tag{18}$$

ここから，下位 m 個の一般化固有ベクトルを並べ，

$$\boldsymbol{T} = \boldsymbol{T}_{\mathrm{LPP}} = \left(\boldsymbol{\xi}_d, \cdots, \boldsymbol{\xi}_{d-m+1}\right)^{\mathrm{T}} \tag{19}$$

とすればよい。

また，類似度行列にはガウスカーネル

$$K(\boldsymbol{x}, \boldsymbol{c}) = \exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{c}||^2}{2h^2}\right) \tag{20}$$

$$\boldsymbol{W} = \boldsymbol{K} = \begin{bmatrix} K(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & K(\boldsymbol{x}_n, \boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & K(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix} \tag{21}$$

を用いることとする。

$h = 1.0$ としてこれを実装したものが，5 ページの Listing 1 である。結果は図 1，2 に示した通りである。これより，元のデータのクラスタ構造がうまく保存されたまま次元が削減されていることがわかる。

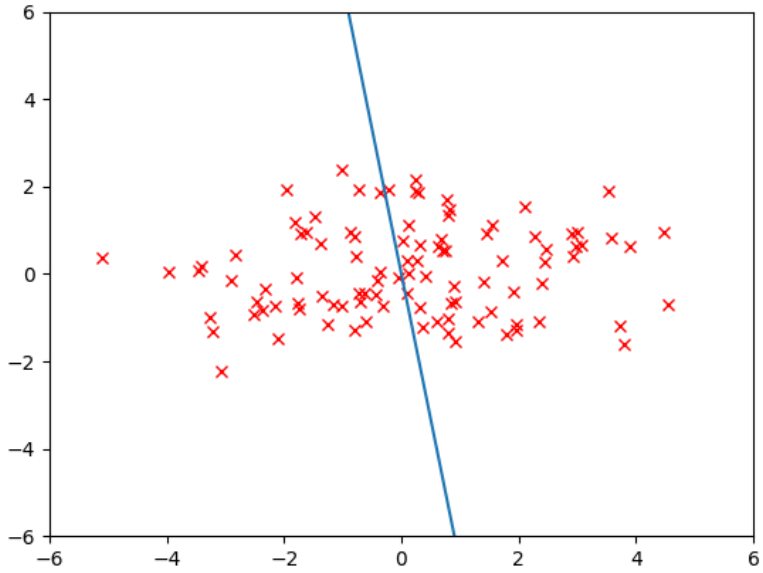

図 1: データ 1 に対する結果

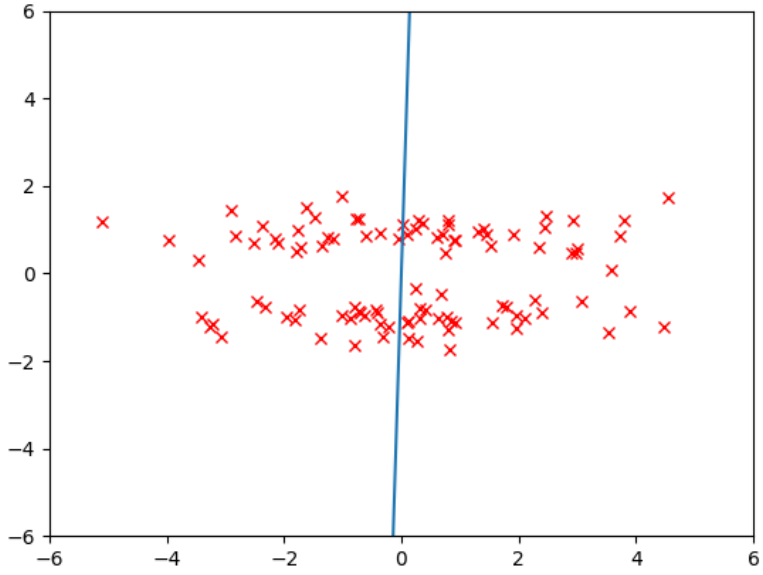

図 2: データ 2 に対する結果

プログラム

実行環境と用いた言語・ライブラリを以下の表 1 に示す。

表 1: プログラムの実行環境

OS	:	Microsoft Windows 10 Pro (64bit)
CPU	:	Intel(R) Core(TM) i5-4300U
RAM	:	4.00 GB
使用言語	:	Python3.6
可視化	:	matplotlib ライブラリ

Listings 1: `assignment2.py`

```
# -*- coding: utf-8 -*-


import numpy as np
import scipy.linalg
import matplotlib.pyplot as plt


def generate_data_1(n=100):
    x = np.concatenate([
            np.random.randn(n, 1) * 2,
            np.random.randn(n, 1)
        ], axis=1)
    return x


def generate_data_2(n=100):
    x = np.concatenate([
            np.random.randn(n, 1) * 2,
            2 * np.round(np.random.rand(n, 1)) - 1 + np.random.randn(n, 1) / 3.
        ], axis=1)
    return x


def calc_norm(x, c, save_memory=False):
    if save_memory:
        n_x = x.shape[1]
        n_c = c.shape[0]
        d = x.shape[-1]
        norm = np.zeros((n_c, n_x))
        x = np.reshape(x, (n_x, d))
```

```
        c = np.reshape(c, (n_c, d))
        for i in range(len(x)):
            x_i = x[i]
            norm[i, :] = np.sum((x_i - c)**2, axis=-1)
    else:
        norm = np.sum((x - c) ** 2, axis=-1)
    return norm


def gauss_kernel(x, c, h, save_memory=False):
    norm = calc_norm(x, c, save_memory)
    ker = np.exp(- norm / (2*h**2))
    return ker


def calc_design_matrix(x, c, h, kernel):
    return kernel(x[None], c[:, None], h)


def train(x, n_components=1, h=1., kernel=gauss_kernel):
    K = calc_design_matrix(x, x, h, kernel)

    W = K
    D = np.diag(W.sum(axis=1))
    L = D - W

    X = x.T
    A = X.dot(L).dot(X.T)
    B = X.dot(D).dot(X.T)

    eigen_values, eigen_vectors = scipy.linalg.eig(A, B)

    # normalize
    for i in range(len(eigen_vectors)):
        eigen_vectors[i] = eigen_vectors[i]/np.linalg.norm(eigen_vectors[i])

    T = eigen_vectors[::-1][:n_components]
    return T


def visualize(x, T, h=1., grid_size=100, path=None):
    plt.xlim(-6., 6.)
    plt.ylim(-6., 6.)
    plt.plot(x[:, 0], x[:, 1], 'rx')
    plt.plot(np.array([-T[:, 0], T[:, 0]]) * 9, np.array([-T[:, 1], T[:,
    1]]) * 9)

```

```
78      if path:
79          plt.savefig(path)
80      plt.show()
81
82
83  def main():
84      # settings
85      n = 100
86      n_components = 1
87      h = 1.0
88      fig_path = '../figures/assignment2_result_data1.png'
89      np.random.seed(0)
90
91
92      # generate data
93      x = generate_data_1(n)
94      #print(x.shape)
95
96
97      # train
98      T = train(x, h=h)
99
100
101     # result
102     print(f'#data: {n}  #Component: {n_components}  h = {h}')
103     #print('T = \n', T)
104
105     visualize(x, T, h=h, path=fig_path)
106
107
108 if __name__ == '__main__':
109     main()
```