模式识别第四章作业

- 李佳政
- 201828013229075

1. 求S_w和S_b

由于概率相等，所以三个分类的概率都为$\frac{1}{3}$。
由多类情况下的类内散布矩阵公式得到

$$S_w = \sum_{i=1}^{c} P(\omega_i) E\{(x-m_i)(x-m_i)^T|\omega_i\} = \sum_{i=1}^{c} C_i$$

其中C_i为第i类的协方差矩阵，得到均值向量$m_1 = (\frac{4}{3}, \frac{1}{3})^T, m_2 = (-\frac{2}{3}, \frac{2}{3})^T, m_3 = (-\frac{1}{3}, -\frac{4}{3})^T$，

且$C = \frac{1}{3}(x-m)^T(x-m)$,所以

$$S_w = S_{w1} + S_{w2} + S_{w3} = C_1 + C_2 + C_3 = \begin{pmatrix} \frac{2}{3} & -\frac{1}{9} \\ -\frac{1}{9} & \frac{2}{3} \end{pmatrix}$$

根据多分类模式类间散布矩阵公式

$$S_b = \sum_{i=1}^{c} P(\omega_i) E\{(x-m_0)(x-m_0)^T\}$$

其中$m_0 = E\{x\} = \frac{1}{3} \times (\frac{1}{3}, -\frac{1}{3})^T$为总体均值向量。得到

$$S_b = \begin{pmatrix} 0.7654 & 0.1605 \\ 0.1605 & 0.7654 \end{pmatrix}$$

2. K-L变换

首先计算均值向量
$m_0 = \frac{1}{2}(\frac{1}{4}\sum_{j=1}^{4} x_{1j} + \frac{1}{4}\sum_{i=1}^{2} x_{2j}) = (0.75, 0.25, 0.125)^T$，由于不为0向量，所以将这些样本减去$m_0$平移到原点，得到新的样本集。计算得到

$$R = \sum_{i=1}^{2} P(\omega_i) E\{xx^T\} = \frac{1}{2}(\frac{1}{4}\sum_{j=1}^{4} x_{1j}x_{1j}^T + \frac{1}{4}\sum_{i=1}^{2} x_{2j}x_{2j}^T) = \begin{pmatrix} 0.6875 & 0.1875 & -0.09375 \\ 0.1875 & 1.1875 & -0.53125 \\ -0.09375 & -0.53125 & 0.859375 \end{pmatrix}$$

解特征方程$|R - \lambda I| = 0$，得到特征值

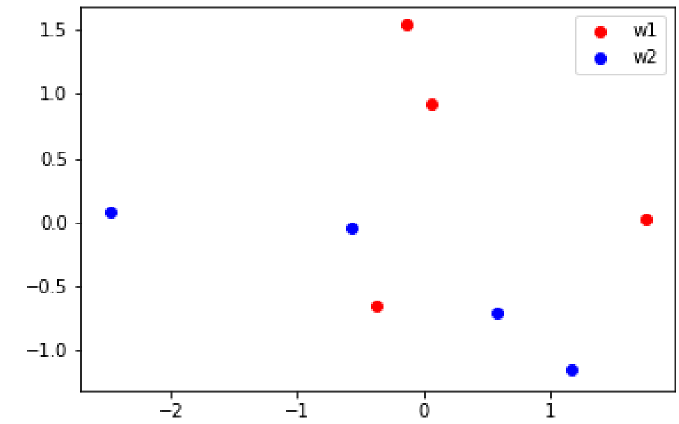
$$\lambda_1 = 1.625, \lambda_2 = 0.64876, \lambda_3 = 0.460613$$

由式子$R\phi_i = \lambda_i \phi_i$可以求得特征向量：
$\phi_1 = (0.21539, 0.95853, -0.18661)^T$
$\phi_2 = (0.78975, -0.05859, 0.61062)^T$
$\phi_3 = (-0.57437, 0.27889, 0.76962)^T$

2.1 降维到二维

选λ_1和λ_2对应的变换向量作为变换矩阵，经过$y = \phi^T x$变换得到二维模式特征为

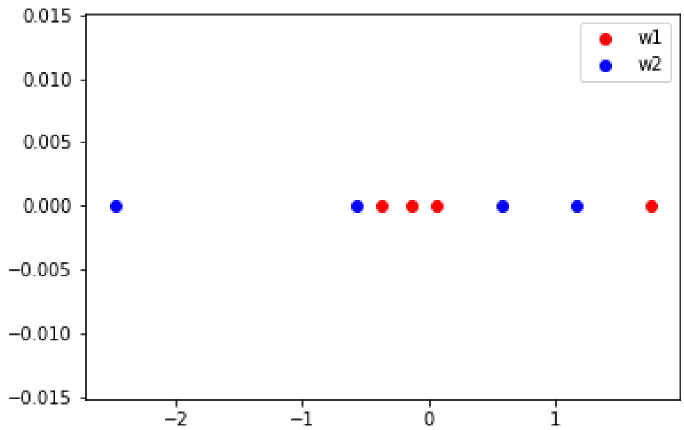
$$\omega_1 = \{(-0.37785, -0.6540)^T, (0.05293, 0.92551)^T, (-0.13368, 1.53613)^T, (1.75461, 0.01859)^T\},$$
$$\omega_2 = \{(-0.56446, -0.04338)^T, (0.58069, -0.71258)^T, (-2.48152, 0.07380)^T, (1.16929, -1.1441)^T\}.$$



2.2 降维到一维

选中λ_1的变换矩阵，得到一维模式特征为

$$\omega_1 = \{-0.37785, 0.05293, -0.13368, 1.75461\},$$
$$\omega_2 = \{-0.56446, 0.58069, -2.48152, 1.16929\}.$$



2.3 程序代码

```python
import numpy as np
import matplotlib.pyplot as plt

w1 = np.array([
    [0, 0, 0],
    [2, 0, 0],
    [2, 0, 1],
    [1, 2, 0]
])
w2 = np.array([
    [0, 0, 1],
    [0, 1, 0],
    [0, -2, 1],
    [1, 1, -2],
])

m0 = (w1.mean(axis=0) + w2.mean(axis=0)) / 2
w1_ = w1 - m0
w2_ = w2 - m0
R = (np.matmul(w1_.T, w1) / 4 + np.matmul(w2_.T, w2) / 4) / 2
lamda, vector = np.linalg.eig(R)

# 二维
w1_2 = np.matmul(w1_, vector[:2].T)
w2_2 = np.matmul(w2_, vector[:2].T)
plt.scatter(w1_2[:, 0], w1_2[:, 1], color="r")
plt.scatter(w2_2[:, 0], w2_2[:, 1], color="b")
plt.legend(["w1", "w2"]);
# 一维
w1_1 = np.matmul(w1_, vector[:1].T)
w2_1 = np.matmul(w2_, vector[:1].T)
plt.scatter(w1_1, np.zeros((4, 1)), color="r")
plt.scatter(w2_1, np.zeros((4, 1)), color="b")
plt.legend(["w1", "w2"]);
```