EN 2023

1 (a)

$C_1 \to B \quad C_2 \to D \quad C_3 \to A \quad C_4 \to C$

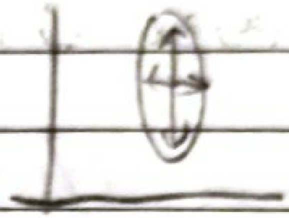
(b)

$$C_1 = \begin{pmatrix} Var(f_1) & Cov(f_1, f_2) \\ Cov(f_1, f_2) & Var(f_2) \end{pmatrix}$$

(c)

$C_2 \to$ Horizontal spread

$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ (principal) and $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$



$C_4 \to$ Vertical spread: $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ principal and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$C_3 \to$ We can see that $x = y$ so the principal eigenvector is just a $45°$ angle $\begin{pmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \end{pmatrix}$. The second one is orthogonal to the first, so $\begin{pmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{pmatrix}$

Or we could solve: $\det(C_3 - \lambda I) = 0 \Leftrightarrow (5-\lambda)^2 = 0 \Rightarrow \lambda = 5 \pm 4$
$\Rightarrow Max\ \lambda = 9$

And then solve: $(C_3 - 9I)v = 0 \Leftrightarrow x = y$ which we can normalize to $\begin{pmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \end{pmatrix}$

$C_1 \to$ Same as C_3 but the principal component is $\begin{pmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{pmatrix}$ and the second one is $\begin{pmatrix} \sqrt{2}/2 \\ \sqrt{2}/2 \end{pmatrix}$

②

ⓐ Assign new sample to the nearest class prototype according to some metric.

Training: for Euclidean MDC ⇒ Compute mean vectors of training set

for Mahalanobis MDC ⇒ Compute mean vectors + covariance matrices of the training set.

ⓑ

- $g_k(x) = \mu_k^T C^{-1} x - 0{,}5\,\mu_k^T C^{-1} \mu_k$

For class ω_1:

$$\mu_k^T C^{-1} = (2 \ \ -4)\begin{pmatrix} 0{,}2 & 0 \\ 0 & 1 \end{pmatrix} = (0{,}4 \ \ -4)$$

$$-0{,}5\,\mu_k^T C^{-1} \mu_k = -0{,}5\,(0{,}4 \ \ -4)\begin{pmatrix} 2 \\ -4 \end{pmatrix} = -0{,}5\,(0{,}8 + 16)$$
$$= -8{,}4$$

$$g_1(x) = (0{,}4 \ \ -4)\,x - 8{,}4 =$$
$$= 0{,}4\,x_1 - 4\,x_2 - 8{,}4$$

For ω_2:

$$\mu_k^T C^{-1} = (2 \ \ 4)\begin{pmatrix} 0{,}2 & 0 \\ 0 & 1 \end{pmatrix} = (0{,}4 \ \ 4)$$

$$-0{,}5\,\mu_k^T C^{-1} \mu_k = -0{,}5\left((0{,}4 \ \ 4)\begin{pmatrix} 2 \\ 4 \end{pmatrix}\right) =$$
$$= -0{,}5\,(0{,}8 + 16) = -8{,}4$$

$$g_2(x) = 0{,}4\,x_1 + 4\,x_2 - 8{,}4$$

$x = \begin{pmatrix} 1{,}1 \\ 0{,}5 \end{pmatrix}$ $\quad g_1(x) = -6 \quad g_2(x) = -2$

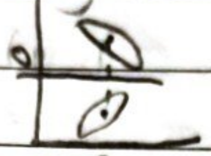
Assign to class ω_2

• $d_{12}(x) = g_1(x) - g_2(x) =$

$= (0,4x_1 - 4x_2 - 8,4) - (0,4x_1 + 4x_2 - 8,4) =$
$= 0x_1 - 8x_2 - 0 = -8x_2$

Separation plane: $-8x_2 = 0 \Leftrightarrow x_2 = 0$

• Equivalent to Euclidean, the separation is orthogonal to the segment connecting the centroid and passes through the midpoint.



③ ⓐ 2, there are two features

ⓑ Classify as ω_1 if $P(\omega_1|x) > P(\omega_2|x) \Leftrightarrow$

$$\Leftrightarrow \frac{P(x|\omega_1)P(\omega_1)}{P(x)} > \frac{P(x|\omega_2)P(\omega_2)}{P(x)}$$

$$\Leftrightarrow P(x|\omega_1)P(\omega_1) > P(x|\omega_2)P(\omega_2) \Leftrightarrow$$

$$\Leftrightarrow \underbrace{\frac{P(x|\omega_1)}{P(x|\omega_2)}}_{} > \frac{P(\omega_2)}{P(\omega_1)} = \Lambda(x)$$

LRT

in this case: $P(\omega_2) = P(\omega_1)$ so:

$$\Lambda(x) = \frac{P(x|\omega_1)}{P(x|\omega_2)} > 1$$

— If we wanted to develop the likelihoods further, we could use the Gaussian densities:

$$P(x|\omega_1) = \frac{1}{2\pi^{d/2}|C|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T \overset{I}{C^{-1}}(x-\mu_1)\right)$$

Here, $C = I$ These are given

(c) $d_{ij}(x) = g_i(x) - g_j(x)$

$d_{ij}(x) = w^T x + w_0$ where:

$\Rightarrow w = C^{-1}(\mu_i - \mu_j)$

$\Rightarrow w_0 = -\frac{1}{2}(\mu_i^T C^{-1}\mu_i - \mu_j^T C^{-1}\mu_j) + \ln\left(\frac{P(\omega_i)}{P(\omega_j)}\right)$

(d)

for $d_{12}(x)$:

$$w = I(\mu_1 - \mu_2) = \begin{pmatrix} -2 \\ -2 \end{pmatrix}$$

$P(\omega_1) = P(\omega_2)$

$$w_0 = -\frac{1}{2}\left((-1 \ \ -1)\begin{pmatrix} -1 \\ -1 \end{pmatrix} - (1 \ \ 1)\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right) + \ln(1)$$

$$= -\frac{1}{2}(2-2) + 0 = 0$$

$d_{12} = (-2 \ \ -2)\, x = -2x_1 - 2x_2$

This is equivalent to Euclidean MDC. Bayes = Mahalanobis if $P(\omega_1) = P(\omega_2)$ and Mahalanobis = Euclidean if $C = I$. Both conditions are true for this question

④

ⓐ SVMs find the optimal plane to separate the classes with the maximum margin:

→ Maximize $\gamma = \frac{2}{||w||}$ subject to the constraint that all training samples are correctly classified (for hard margin - high C) or with some penalty (for soft margin - low C).

We can formulate the problem as a primal problem (quadratic optimization):

$$\text{minimize } \Phi(w) = \frac{1}{2}||w||^2 + C\sum \xi_i$$

same as maximizing $2/||w||$ | penalty parameter | ξ_i: $\begin{cases} 0, \text{ if correct label} \\ 0 < \xi_i \leq 1, \text{ if correct but inside margin} \\ 1 \text{ if incorrect} \end{cases}$

subject to $y_i(w^T x_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$

y_i: sign of class of sample i

But in practice, this is formulated as a dual problem to be solved efficiently. Using Lagrange Multipliers (α_i), we maximize the dual function $Q(\alpha)$ subject to $\sum \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$, the solution gives us the α_i values, which are the support vectors (the difficult patterns near the boundary).

The weight vector is then given by $w = \sum_{i=1}^{N} \alpha_i y_i x_i$.
This is a convex problem with known solvers, so it's easy to find the unique minimum.

s.a. Lagrange Multipliers Methods

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i^T x_j)$$

(b) $w = \sum_i \alpha_i y_i x_i = 1(+1)\begin{pmatrix}2\\2\end{pmatrix} + 1(-1)\begin{pmatrix}3\\1\end{pmatrix} = \begin{pmatrix}-1\\1\end{pmatrix}$

Note:
$\alpha_1 = 1$
$\alpha_2 = 1$
are consistent
with the constraint
$\sum \alpha_i y_i = 0$

To compute b, we can use
the constraint: $y_i(w^T x_i + b) = 1$

using $x_1 = [2\ 2]^T$

$+1((-1\ 1)^T \begin{pmatrix}2\\2\end{pmatrix} + b) = 1 \Leftrightarrow$

$\Leftrightarrow -2 + 2 + b = 1 \Leftrightarrow b = 1$

$d(x) = w^T x + b = \begin{pmatrix}-1\\1\end{pmatrix}^T x + 1 = -x_1 + x_2 + 1$

(c) $d(\begin{pmatrix}3\\3\end{pmatrix}) = -3 + 3 + 1 = 1$ (lies on the positive margin boundary) $\Rightarrow$ its a support vector for ω_1

(d) $d(\begin{pmatrix}2\\0\end{pmatrix}) = -2 + 1 = -1 \rightarrow \omega_2$ ✗
$d(\begin{pmatrix}3\\0\end{pmatrix}) = -3 + 1 = -2 \rightarrow \omega_2$ ✓
$d(\begin{pmatrix}0\\3\end{pmatrix}) = 0 + 3 + 1 = 4 \rightarrow \omega_1$ ✓

66,67% accuracy

5

```
def knn_training (Xtr, Ttr, K):
    model={'Xtr': Xtr,
           'Ttr': Ttr,
           'K': K
          }
    return model #no training

def knn_testing (Xte, Tte, model):
    Xtr = model['Xtr']
    Ttr = model['Ttr']
    K = model['K']
    predictions = []
    Pte = Xte.shape[1]
    for i in range (Pte):
        test_sample= Xte[:, i].reshape(-1, 1)

        #distance to all training samples
        dists= np.linalg.norm (Xtr - test_sample, axis=0)
        #indices of first K
        nearest_idxs = np.argsort (distances) [:K]
        # labels of these neighbours
        nearest_labels = Ttr.flatten() [nearest_idxs]
        # find most frequent
        ct=np.bincount(nearest_labels.astype(int))
        pred= np.argmax (counts)
        predictions.append(pred)
    predictions=np.array(predictions)
    true = Tte.flatten()
```

```
TP = np.sum((predictions =1) & (true = 1))
TN =  "            " 2  "   " 2
FP =  "            " 1  "   " 2
FN =  "            " 2  "   " 1

se = TP/(TP + FN)
sp = TN/(TN + FP)
return se, sp
```