MLE Exercise sheet 2

Sunday, 18 May 2025 8:21 PM

Exercise 1 : Decision Boundaries of Generative Classifiers

- Task 1 : Gaussian Discriminant Analysis

(a) QDA:

We assume A & B with Gaussian conditionals

$$p(x|y=k) = \frac{1}{\sqrt{(2\pi)^N \det \Sigma_k}} \exp\left[-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\right]$$

& Also $\pi_k = p(y=k)$. The MAP-discriminant is,

$$g_k(x) = \log p(x|y=k) + \log \pi_k$$

$$= -\frac{1}{2}\log\det\Sigma_k - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log \pi_k$$

Setting $g_A(x) = g_B(x)$

$$-\frac{1}{2}\log\det\Sigma_A - \frac{1}{2}(x-\mu_A)^T\Sigma_A^{-1}(x-\mu_A) + \log\pi_A = -\frac{1}{2}\log\det\Sigma_B - \frac{1}{2}(x-\mu_B)^T\Sigma_B^{-1} + \log\pi_B$$

Taking them to one side & expanding the quadratic forms:

$$-\frac{1}{2}x^T(\Sigma_A^{-1}-\Sigma_B^{-1})x + (\Sigma_A^{-1}\mu_A - \Sigma_B^{-1}\mu_B)^T x + \left[-\frac{1}{2}(\mu_A^T\Sigma_A^{-1}\mu_A - \mu_B\Sigma_B^{-1}\mu_B) - \frac{1}{2}\ln\frac{\det\Sigma_A}{\det\Sigma_B} + \ln\frac{\pi_A}{\pi_B}\right] = 0$$

Let, $\Lambda = -\frac{1}{2}(\Sigma_A^{-1}-\Sigma_B^{-1})$, $\omega = \Sigma_A^{-1}\mu_A - \Sigma_B^{-1}\mu_B$, $b = -\frac{1}{2}(\mu_A^T\Sigma_A\mu_A - \mu_B\Sigma_B^{-1}\mu_B) - \frac{1}{2}\ln\frac{\det\Sigma_A}{\det\Sigma_B} + \ln\frac{\pi_A}{\pi_B}$

So the Decision Boundary is,

$$x^T\Lambda x + \omega^T x + b = 0$$

$\Rightarrow$ Here $x^T\Lambda x = \sum_{ij}\Lambda_{ij}x_i x_j$ is a 2^{nd} degree func. of the components of x. Upon solving the eq^n, we'll get set of points satisfying a degree-2 polynomial equation – i.e. A conic in 2D. Hence quadratic term indicates non-linear decision boundary.

(b) LDA:

Let $\Sigma_A = \Sigma_B = \Sigma$

$$g_k(x) = \log p(x|y=k) + \log\pi_k$$

$$= -\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k) + \log\pi_k$$

& setting $g_A(x) = g_B(x)$,

$$-\frac{1}{2}\left[(x-\mu_A)^T\Sigma^{-1}(x-\mu_A) - (x-\mu_B)^T\Sigma^{-1}(x-\mu_B)\right] + \log\frac{\pi_A}{\pi_B} = 0$$

Expanding & cancelling the common $x^T\Sigma^{-1}x$ terms,

$$x\left[\Sigma^{-1}(\mu_A-\mu_B)\right]^T - \frac{1}{2}\left(\mu_A^T\Sigma^{-1}\mu_A - \mu_B^T\Sigma^{-1}\mu_B\right) + \log\frac{\pi_A}{\pi_B} = 0$$

$\therefore$ Decision boundary can be written as,

$$w^T x + b = 0$$

$$w = \Sigma^{-1}(\mu_A-\mu_B)$$

$$b = -\frac{1}{2}\left[\mu_A^T\Sigma^{-1}\mu_A - \mu_B^T\Sigma^{-1}\mu_B\right] + \log\frac{\pi_A}{\pi_B}$$

- Task 2 :

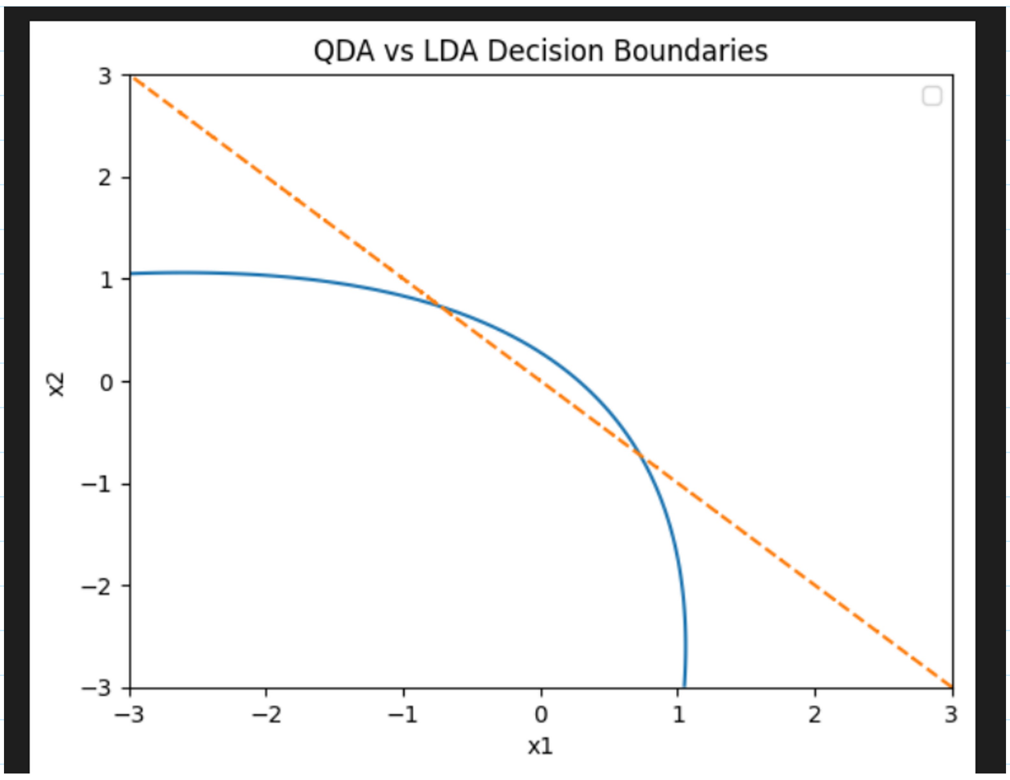
a)

```python
muA = np.array([-1.0, -1.0])
muB = np.array([1.0, 1.0])
SigmaA = np.array([[1.0, 0.3],
                   [0.3, 1.0]])
SigmaB = np.array([[1.5, -0.2],
                   [-0.2, 1.5]])

piA, piB = 0.5, 0.5

# Inverses
invA = np.linalg.inv(SigmaA)
invB = np.linalg.inv(SigmaB)

# QDA parameters
Lambda = -0.5 * (invA - invB)
w_qda = invA.dot(muA) - invB.dot(muB)
b_qda = (
    -0.5 * (muA.T.dot(invA).dot(muA) - muB.T.dot(invB).dot(muB))
    -0.5 * np.log(np.linalg.det(SigmaA) / np.linalg.det(SigmaB))
    + np.log(piA / piB)
)

# LDA parameters
Sigma_pooled = 0.5 * (SigmaA + SigmaB)
inv_pooled = np.linalg.inv(Sigma_pooled)
w_lda = inv_pooled.dot(muA - muB)
b_lda = (
    -0.5 * muA.T.dot(inv_pooled).dot(muA) + 0.5 * muB.T.dot(inv_pooled).dot(muB)
    + np.log(piA / piB)
)

print("QDA parameters:")
print("Lambda:\n", Lambda)
print("w_qda:\n", w_qda)
print("b_qda:\n", b_qda)
print("\nLDA parameters:")
print("w_lda:\n", w_lda)
print("b_lda:\n", b_lda)
```

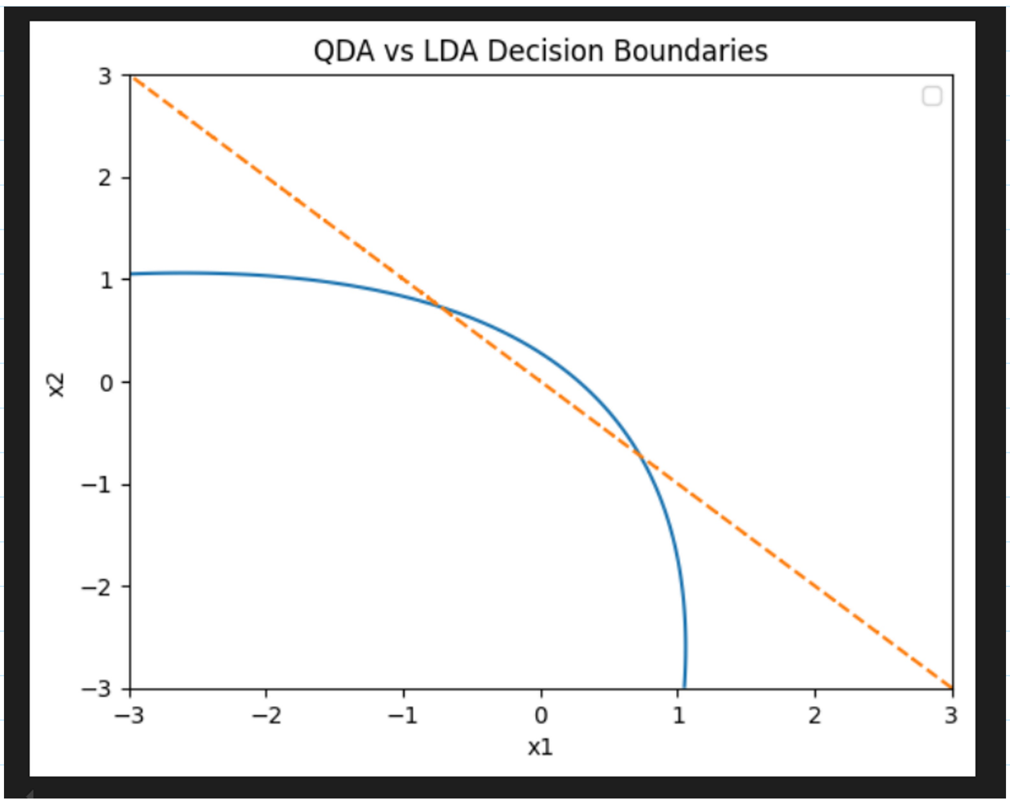
✓ 0.1s

```
QDA parameters:
Lambda:
 [[-0.21008403  0.21008403]
 [ 0.21008403 -0.21008403]]
w_qda:
 [-1.53846154 -1.53846154]
b_qda:
 0.44365159750045136

LDA parameters:
w_lda:
 [-1.53846154 -1.53846154]
b_lda:
 0.0
```

b)



b)



c) when $\Sigma_A \neq \Sigma_B$, each class has its own "shape" & orientation in feature space. QDA plugs both Σ_A^{-1} & Σ_B^{-1} into discriminant, so the boundary retains those 2^{nd} order $(x_i x_j)$ terms. Geometrically, this means the decision curve can bend, twist even form ellipses or hyperbola, following exactly how each Gaussian contours differ.

On other hand, LDA forces $\Sigma_A = \Sigma_B = \Sigma$, we get

$$w^T x + b = 0$$

a single straight line that ignores any per-class anisotropy.

- QDA can be preferred when we have ample training samples & strong evidence that Σ_A & Σ_B realy differ.
- LDA can be preferred when data is limited or maybe classes have same covariance struct. or we need linearrule