Introducción al problema

Comparativa de métodos para mejorar la eficiencia de predicción:

- `QDA`: Métodos estándar con ciclos `for`.
- `TensorizedQDA`: Uso de tensores para mayor rapidez.
- `FasterQDA`: Implementación optimizada sin ciclos.

Propósito: Acelerar la predicción eliminando ciclos innecesarios.

QDA: Implementación básica

Utiliza el método `_predict_one(x)`:

- Recorre cada clase (`for` interno).
- Método iterativo para calcular la verosimilitud.

Limitación: Ineficiente para grandes volúmenes de datos.

Tiempo: Lento debido a múltiples ciclos.

TensorizedQDA: Uso de tensores

Expande el set de predicción $X \in \mathbb{R}^{p \times n}$:

$$X_{\text{expanded}} \in \mathbb{R}^{\text{clases} \times p \times n}$$

Los promedios se almacenan en:

$$\text{tensor\_means} \in \mathbb{R}^{\text{clases} \times p \times 1}$$

Permite operaciones matriciales sin ciclos adicionales.

Mejora significativa en eficiencia.

FasterQDA: Optimización total

Cálculo del producto interno necesario para la verosimilitud:

inner_prod_mat = unbiased_X_transposed @ self.tensor_inv_cov @ unbiased

Diagonalización para reducir dimensiones.

Uso del determinante de la matriz de covarianza:

$$\frac{1}{2} \log |\Sigma_j^{-1}|$$

Todo en una sola pasada sin ciclos.

Comparativa de Tiempos

Resultados experimentales:

`QDA`:

Tiempo: 0.00242 s (std: 0.00055)

`TensorizedQDA`:

Tiempo: 0.00062 s (std: 0.00051)

`FasterQDA`:

Tiempo: 0.00010 s (std: 0.00030)

Aceleración significativa al eliminar ciclos.