Linear Dimensionality Reduction

Principal Component Analysis

} eigen decomposition of cov. matrix

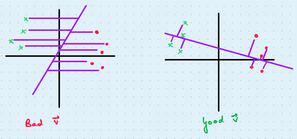
- Unsupervised, may mix classes together
- Focuses on maximising variance
- Linear
- Components always orthogonal
- Global components, may miss local patterns

When to use it	When to stay away
• Classification tasks with labeled data • When classes are roughly Gaussian • When linear separation is reasonable • Preprocessing for linear classifiers	• Data has complex, nonlinear structure • Classes have very different covariances • Very high dimensional data with few samples • Want to preserve overall data structure (not just class separation)

Linear Discriminant Analysis (LDA)

minimise within class variance — maximise between class variance

- Find some vector v to project onto that ensures both of the above.



Assumptions

- Each feature → Gaussian distribution
- Each class → identical cov matrix → shape, size, orientation same

NOTE: Still performs well if above not met.

Math

Say we have classes C_1, C_2 with n_1, n_2 samples respectively. Samples are d-dimensional and $\in \{x_1, x_2, \dots, x_n\}$

We are projecting onto a unit vector $\vec{v}$. So for any x_i:

$v^T x_i$ = projection length along $\vec{v}$

μ_1 = mean (C_1)
μ_2 = mean (C_2)

→ $|\mu_1 - \mu_2|$ = measure of separation between classes before projection

Find separation after projection

$$\bar{\mu}_1 = \frac{1}{n_1}\sum_{x_i \in C_1} v^T x_i = v^T\left[\sum_{x_i \in C_1}^{n_1} x_i\right] = v^T \mu_1$$

Illy $\bar{\mu}_2 = v^T \mu_2$

$$|\bar{\mu}_1 - \bar{\mu}_2| = v^T |\mu_1 - \mu_2|$$

Account for variance of the classes ... on a factor ∝ variance

Scatter → raw spread = sample variance × n
→ $\Sigma(x_i - \mu_x)^2$
→ scatter ≡ variance but diff scale

Normalising $|\bar{\mu}_1 - \bar{\mu}_2|$ by scatter → Fisher LDA

let $y_i = v^T x_i$

Scatter for C_1: $\tilde{S}_1^2 = \Sigma(y_i - \bar{\mu}_1)^2$
Scatter for C_2: $\tilde{S}_2^2 = \Sigma(y_i - \bar{\mu}_2)^2$

$\boxed{\text{Project on } \vec{v} \text{ that maximises } J(\vec{v})}$

$$\boxed{J(\vec{v}) = \frac{(\bar{\mu}_1 - \bar{\mu}_2)^2}{\tilde{S}_1^2 + \tilde{S}_2^2}} \quad \begin{matrix} \} S_B \\ \} S_W \end{matrix}$$

Express $J(\vec{v})$ in terms of v alone and maximise

Fisher LDA

① Class matrices before projection

$S_1 \to C_1$, $S_2 \to C_2$

$$S_1^2 = \sum_{x_i \in C_1} (x_i - \mu_1)(x_i - \mu_1)^T$$

$$S_2^2 = \sum_{x_i \in C_2} (x_i - \mu_2)(x_i - \mu_2)^T$$

⇒ Total scatter within = $\boxed{S_W = S_1^2 + S_2^2}$

② Class matrices after projection

$$\tilde{S}_1^2 = \sum_{y \in C_1} (y_i - \tilde{\mu}_1)^2$$

$$\tilde{S}_2^2 = \sum_{y_i \in C_2} (y_i - \tilde{\mu}_2)^2$$

where $y_i = v^T x_i$, $\tilde{\mu}_1 = v^T \mu_1$, $\tilde{\mu}_2 = v^T \mu_2$

Substituting,

$$\tilde{S}_1^2 = \sum_{y \in C_1} (v^T x_i - v^T \mu_1)^2 = \Sigma(v^T(x_i - \mu_1))^2$$
$$= \Sigma(v^T(x_i - \mu_1))^T(v^T(x_i - \mu_1))$$
$$= \Sigma((x_i - \mu_1)^T v)(v^T(x_i - \mu_1))$$

$\boxed{\text{Identity: } (a^T b)(b^T a) = v^T(ba)^T v}$

$$\Rightarrow \tilde{S}_1^2 = \Sigma(v^T)(x_i - \mu_1)(x_i - \mu_1)^T(v) = v^T\underbrace{\left[\Sigma(x_i - \mu_1)(x_i - \mu_1)^T\right]}_{S_1^2} v$$

$$\boxed{\tilde{S}_1^2 = v^T S_1^2 v}$$

Illy, $\boxed{\tilde{S}_2^2 = v^T S_2^2 v}$

Total scatter within after projection: $\tilde{S}_W = \tilde{S}_1^2 + \tilde{S}_2^2$
$= v^T(S_1^2 + S_2^2)v$

$$\boxed{\tilde{S}_W = v^T S_W v}$$

③ Similar process for between-classes

$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$ } Before projection

$$\tilde{S}_B = (\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (v^T\mu_1 - v^T\mu_2)^2$$
$$= (v^T(\mu_1 - \mu_2))^T(v^T(\mu_1 - \mu_2))$$
$$= v^T\underbrace{(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T}_{S_B} v$$

} After projection

$$\boxed{\tilde{S}_B = v^T S_B v}$$

④ Substitute in $J(\vec{v})$

$$\boxed{J(\vec{v}) = \frac{\tilde{S}_B}{\tilde{S}_W} = \frac{v^T S_B v}{v^T S_W v}}$$

} quantitative measure that captures variation between classes as well as within them

Maximising $J(\vec{v})$ → increase $\tilde{S}_B$, decrease $\tilde{S}_W$ → goal of LDA

Limitation: Small Sample Size (SSS) Problem

SSS : n (samples) < n (features)

① Singular S_W matrix

- If $n < d$, S_W becomes singular → non-invertible
- $S_B v = \lambda S_W v$ becomes unsolvable
  → Rayleigh quotient maximises: eigenvector associated with largest eigenvalue here gives ideal solution for $J(\vec{v})$

② High dimensionality issues

Overfit on limited training data

Solutions:

- Regularised LDA
- PCA + LDA
- Feature selection first

② Mathematical limitations

Max rank of S_W = $n - c$

When rank < d, cannot estimate all parameters

	PCA	LDA
Supervision	Unsupervised	Supervised
Objective	Maximize variance	Maximize class separation
Max Components	min(features, samples-1)	min(features, classes-1)
Use Case	Visualization, compression	Classification preprocessing

Independent Component Analysis (ICA)

Say you have some signal $X = AS$ → unknown source signals: each individual talking → we want to find this

X: observed mixed signals: bunch of people talking
A: unknown mixing matrix

Goal: Find some unmixing matrix W s.t. $WX \approx S$

Assumptions

- Source signals are independent
- Source signals are non-gaussian
- Sources combine linearly
- Sources are stationary

Why Non-Gaussian?

From CLT: When we mix two independent random vars, their sum tends to be more Gaussian than either of them

⇒ X more Gaussian than sources in S

⇒ To find original sources in S, maximize non-Gaussianity

Methods to determine non-Gaussianity

① Kurtosis

- 4 moments: mean (location), variance (spread), skewness (symmetry), kurtosis (tailedness: how heavy are the tails?)
- Kurtosis = 0 ⇒ Gaussian distribution. ∴ W → maximise $|k|$

Non-Gaussianity with kurtosis:

(a) Super-Gaussian ($k > 0$)
Heavy tails, sharp peak

(b) Sub-Gaussian ($k < 0$)
Light tails, flat peak

② Negentropy

- Used since kurtosis can be sensitive to outliers
- Entropy: measure of randomness of a variable.
  NOTE: ∀ distributions w/ identical variance, Gaussian has max value of entropy (most random)

Negentropy = difference b/w Gaussian variable entropy and entropy of variable to be analysed

$$J(y) = H(y_{Gaussian}) - H(y)$$

Maximise negentropy = maximise non-Gaussianity

Algorithm

Centre and whiten → Calculate weight vector → Deflate → Convergence

① Preprocessing: centre and whiten

- Subtract mean to centre mean to 0.
- Whitening / Sphering: Transform data to make components uncorrelated, unit variance
  Cov matrix = I
  why → dimensionality reduction
  → faster convergence

How whitening?

Calculate covariance matrix → Eigen-decompose $C = EDE^T$ → Whitening transform $W_{whiten} = D^{-1/2}E^T$ → Apply to centred data

$D^{-1/2}$: scale by root of eigenvalue

② Update weight

- Start with a random guess for weight vector w (row of unmixing matrix W)
- Update using some algo: FastICA, gradient ascent
- Maximise non-Gaussianity for $y = wZ$

③ Deflation

- Once one source component has been found, remove it from input mixture before searching further
- Usually done by orthogonalisation: ensure every weight vector found is orthogonal to the ones found so far → prevents algo from finding the same component again

④ Convergence

- Stop once weight vector stops changing significantly.
- Checked by taking dot product of old and new and seeing if absolute value ≈ 1 → direction is about the same

$|w_{old} \cdot w_{new}| \approx 1$

⇓

direction stopped changing ⇒ this direction maximises non-Gaussianity ⇒ one source component found. STOP

	PCA	ICA
Objective	Maximize variance	Maximize independence
Constraints	Orthogonal components	Independent components
Assumptions	None specific	Non-Gaussian sources
Applications	Compression, visualization	Signal separation
Uniqueness	Up to rotation	Up to scaling & permutation
Interpretability	Variance directions	Independent factors

Choose ICA when:

- you suspect data is mixture of independent sources
- sources have non-Gaussian distributions
- you need to separate mixed signals
- independence is more important than variance