Lecture 12

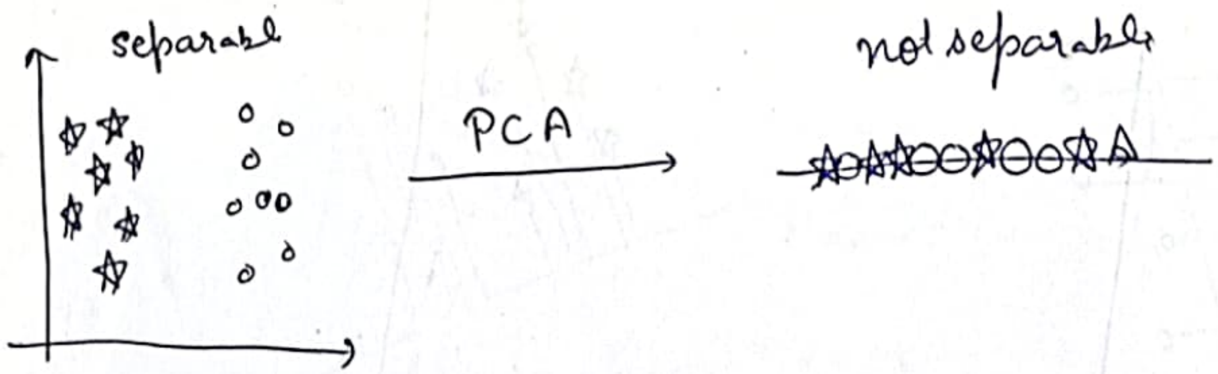
Principal Component Analysis

mt - matrix

① Calculate the covariance mt of data point.
② Calculate eigenvector + correspond eigenvalue.
③ Sort eigenvectors according to their given value in decreasing order.
④ Choose first k eigenvectors + that will be the new k dimension.
⑤ Transform original n-dim to k-dimension.

PCA is an unsupervised learning technique that aims to maximize the variance of the data along the principal components.
→ Aim is to identify the directions (components) that captures as much variance as possible.

However the directions of maximum variance may be useless for classification



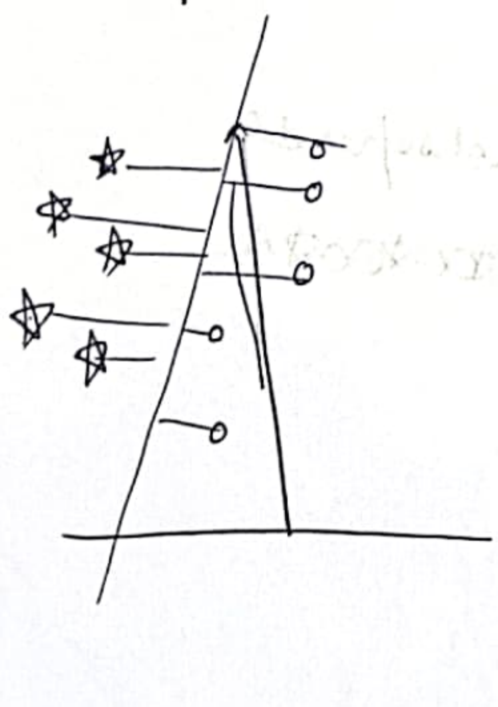
If the data is linearly separable in higher dimensional space then the linearly separable should be preserved in the lower dimensionality space.

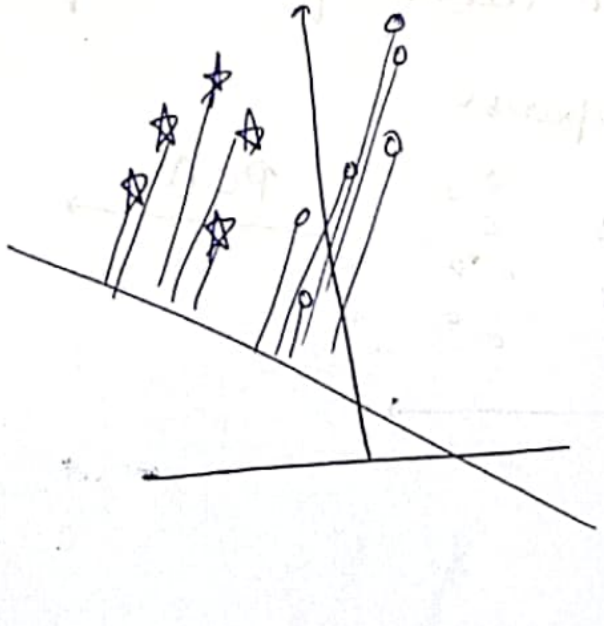
Here LDA (Linear Discriminant Analysis) comes to the rescue.

→ Unlike ~~LDA~~ PCA, LDA is a supervised learning method, which means it takes class labels into account when finding directions of maximum Variance.

LDA aims to maximize the separation b/w different classes in the data.

→ goal is to identify the directions that capture the most separation b/w the classes



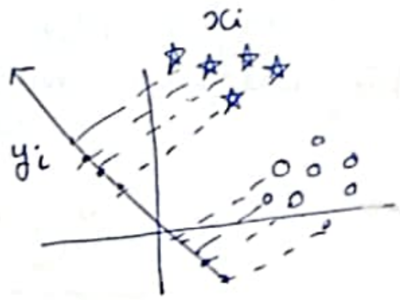


Given a data set $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ where

n_1 sample are coming from class C_1

n_2 ——— do ——————— C_2

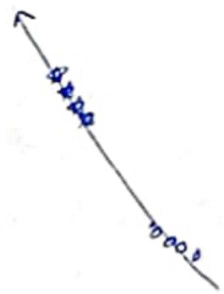
Our Aim is to find a unit vector direction that "best discriminate" b/w classes



Consider any unit vector $v \in \mathbb{R}^d$

1D projection of points are

$$y_i = v^T x_i \qquad i = 1, 2, \ldots, n$$



Let μ_1 & μ_2 be the mean of centroid of class C_1 and C_2, respectively before projection

Let $\tilde{\mu}_1$ denote mean of sample of class C_1 after the projection then

$$\tilde{\mu}_1 = \frac{1}{n_1} \sum_{x_i \in C_1}^{n_1} v^T x_i = v^T \left(\frac{1}{n_1} \sum_{x_i \in C_1}^{n_1} x_i \right)$$

$$= v^T (\mu_1)$$

$$\tilde{\mu}_2 = \frac{1}{n_2} \sum_{x_i \in C_2}^{n_2} v^T x_i = v^T (\mu_2)$$

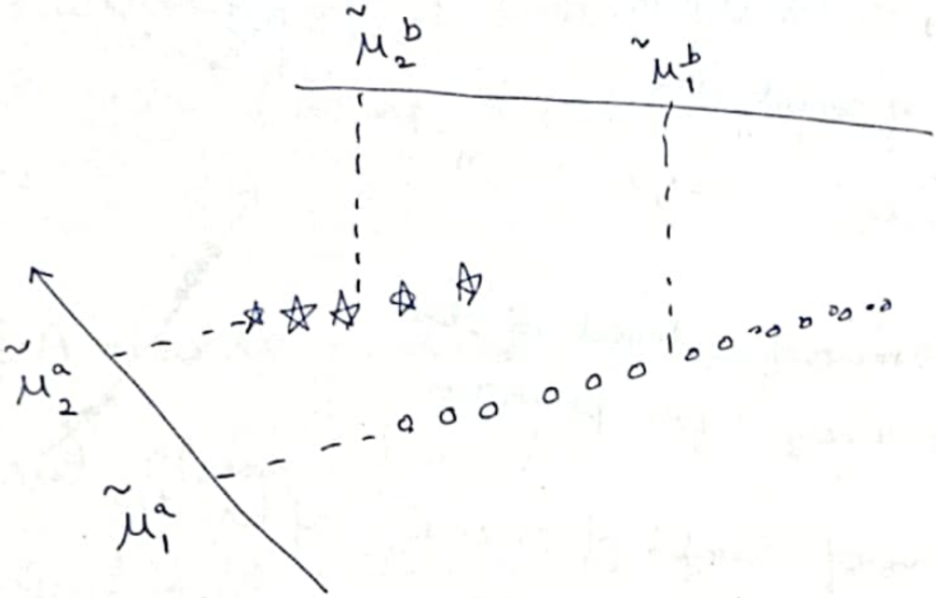
What one solve in LDA ?

$$\max_{v : \|v\| = 1} |\tilde{\mu}_1 - \tilde{\mu}_2| \quad \text{where } \tilde{\mu}_j = v^T \mu_j, \quad j = 1, 2$$

Mean

maximize $|\tilde{\mu}_1 - \tilde{\mu}_2|$ w.r.t v, while ensuring v remain unit vector.

But this criteria might not always work



So we also need to pay attention to the variance of the projected class.

Let $y_i = v^T x_i$ be the projected samples.

Then scatter for sample of class C_1 is

$$\tilde{s}_1^2 = \sum_{x_i \in C_1} (y_i - \tilde{\mu}_1)^2 \qquad \text{lets ignore } \frac{1}{n} \text{ for now}$$

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

We use them to normalize $(\tilde{\mu}_1 - \tilde{\mu}_2)$ by variance.

Ideally, projected classes have both

→ far away mean

→ small variances.

Thus, one need to project the data onto a line having direction v which maximizes

$$J(v) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

class 1 and class 2 scatter after project should be small

$$\max_{v : \|v\| = 1} \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

optimal v should.

→ $(\tilde{\mu}_1 - \tilde{\mu}_2)$ large

→ $(\tilde{s}_1^2 + \tilde{s}_2)$ small

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (v^T \mu_1 - v^T \mu_2)^2 = (v^T (\mu_1 - \mu_2))^2$$
$$= v^T (\mu_1 - \mu_2) \cdot (\mu_1 - \mu_2)^T v$$
$$= v^T S_B v$$

where

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T \in \mathbb{R}^{d \times d}$$ ~~is class~~.

called between-class scatter mtx

Remark. S_B is square, symmetric and positive semidefinite mtx.
Moreover $\operatorname{rank}(S_B) = 1$, ⇒ only 1 +ve eigenvalue.

1

Define separate class scatter mt S_1 & S_2 of class C_1 & C_2

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

$$S_2 = \sum_{x_i \in C_1} (x_i - \mu_2)(x_i - \mu_2)^T$$

Again $\frac{1}{n}$ is not taken.

Now define within the class scatter mt as

$$S_W = S_1 + S_2$$

For each class $j = 1, 2$, variance of projection (onto v):

$$\tilde{s}_j^2 = \sum_{x_i \in C_j} (y_i - \tilde{\mu}_j)^2 = \sum_{x_i \in C_j} (v^T x_i - v^T \mu_j)^2$$

$$= \sum_{x_i \in C_j} v^T (x_i - \mu_j)(x_i - \mu_j)^T v$$

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

$$\tilde{s}_2^2 = \sum_{x_i \in C_1} v^T (x_i - \mu_2)(x_i - \mu_2)^T v$$

$$\tilde{s}_j^2 = v^T \left[\sum_{x_i \in C_j} (x_i - \mu_j)(x_i - \mu_j)^T \right] v$$

$$= v^T S_j v$$

$$\tilde{s}_1^2 = v^T S_1 v \qquad \tilde{s}_2^2 = v^T S_2 v$$

$$\tilde{S}_1^2 + \tilde{S}_2^2 = v^T S_1 v + v^T S_2 v$$
$$= v^T (S_1 + S_2) v$$
$$= v^T S_W v$$

$S_W = S_1 + S_2 =$

$$\sum_{x_i \in C_1} (x_i - \mu_1)(x_i - \mu_1)^T + \sum_{x_i \in C_2} (x_i - \mu_2)(x_i - \mu_2)^T$$

S_W is called total within-class scatter mtx of original data

$S_W \in \mathbb{R}^{d \times d}$ is also square, symmetric & Positive semidefinite

Putting everything together

$$\max_{v : \|v\| = 1} \frac{v^T S_B v}{v^T S_W v} = J(v)$$

$$\frac{d}{d(v)} J(v) = 0 \qquad \text{after some step}$$

$$\Rightarrow \quad S_B v - \underbrace{\left(\frac{v^T S_B v}{v^T S_W v} \right)}_{\text{scalar value. assume it to } \lambda} (S_W v) = 0$$

$$S_B v - \lambda S_W v = 0$$

$$S_B v = \lambda S_W (\lambda v)$$

$$\underbrace{(S_W^{-1} S_B)}_{\text{Matrix, let say } M} v = \lambda v$$

$$\boxed{M v = \lambda v} \rightarrow \text{eigenvector \& eigenvalue}$$

largest eigenvalue

v is the eigenvector of $S_W^{-1} S_B$ corresponding to largest eigenvalue.

However it is not computational efficient method as.

① First one has to invert $S_W \rightarrow S_W^{-1}$
② Multiply it to $S_B \rightarrow S_W^{-1} S_B$
③ & then solve for eigenvalue.

$$S_W^{-1} S_B v_1 = \lambda v$$

· One can do it in a smartest way

$$S_W^{-1} S_B v = \lambda v$$

$$\lambda v = S_W^{-1} S_B v$$

we know $S_B x$ points in the same direction as $\mu_1 - \mu_2$

$$S_B x = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T x$$

$$\lambda v = S_W^{-1} (\mu_1 - \mu_2) \underbrace{(\mu_1 - \mu_2)^T v}_{\text{scalar}}.$$

This implies that $v \propto S_W^{-1} (\mu_1 - \mu_2)$

and it can be computed from $S_W^{-1}(\mu_1 - \mu_2)$ through rescaling! $\boxed{\text{No need to calculate } S_B}$

Recipe for LDA

① Calculate the mean vector for each class x, y.

Mean vector class 1 → μ_1 (μ_1^x, μ_1^y)

Mean vector class 2 → μ_2 (μ_2^x, μ_2^y)

② Calculate the within-class scatter mtx

$S_W = \sum_{x_i \in C_c} (x_i - \mu_c)(x_i - \mu_c)^T$ for all data points x_i in class C $(c = 1, 2)$

③ Calculate between-class scatter mtx.

$S_b = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T$.

④ Compute eigenvalues & eigenvectors of mtx.

$S_W^{-1} \cdot S_b$

Eigenvalues represent separability of classes, and the corresponding eigenvectors are direction in which data should be projected

⑤ Sort top k eigen vectors

⑥ Create a mtx W with selected eigenvectors as column

Multiple Discriminant Analysis (MDA)

Just like LDA, MDA aims to maximize the ratio of b/w-class scatter (S_B) to within-class scatter (S_w).
However, instead of finding a single vector v, it seeks a matrix V whose columns are discriminant function weights

Soln is found by solving

$$S_w^{-1} S_B V = \Lambda V$$

columns of mat V are eigenvectors, which represent discriminant function

Λ - diagonal mat eigenvalues

	PCA	LDA
Label	unsupervised	Supervised
Criterion	variance	discrimination
classification.	features classification	data classification
dimension	reduce data lighter dry dimension	reduce data upto "number of class - 1" dimension
linear projection	Yes	Yes