EE5907: Pattern Recognition
&
EE5026: Machine Learning for Data Analytics

Dr. WANG Si

Email: si.wang@nus.edu.sg

Office: E1a-04-01

A Gentle Reminder

CA2 for EE5907 and CA1 for EE5026 are released!

- Face Recognition System (CMU PIE dataset)
  - Part 1: PCA & LDA (from scratch)
  - Part 2: KNN, GMM, SVM, CNN
- Go through Assignment with the instruction file.
- Deadline (strict): 14 Nov. 2025 1800 SGT
- Individual assignment

Outlines

- Pattern Representation Learning
  - Unsupervised Representation Learning (week 7)
  - Supervised Representation Learning (week 8)
  - Unified Framework: Graph Embedding (week 8)
- Patter Recognition Models
  - Clustering and Applications (week 9)
  - Gaussian Mixture Model and Boosting (week 10)
  - Support Vector Machines (week 11)
- Deep Learning (week 12)
- Revision and Q&A (week 13)

Pattern Representation Learning

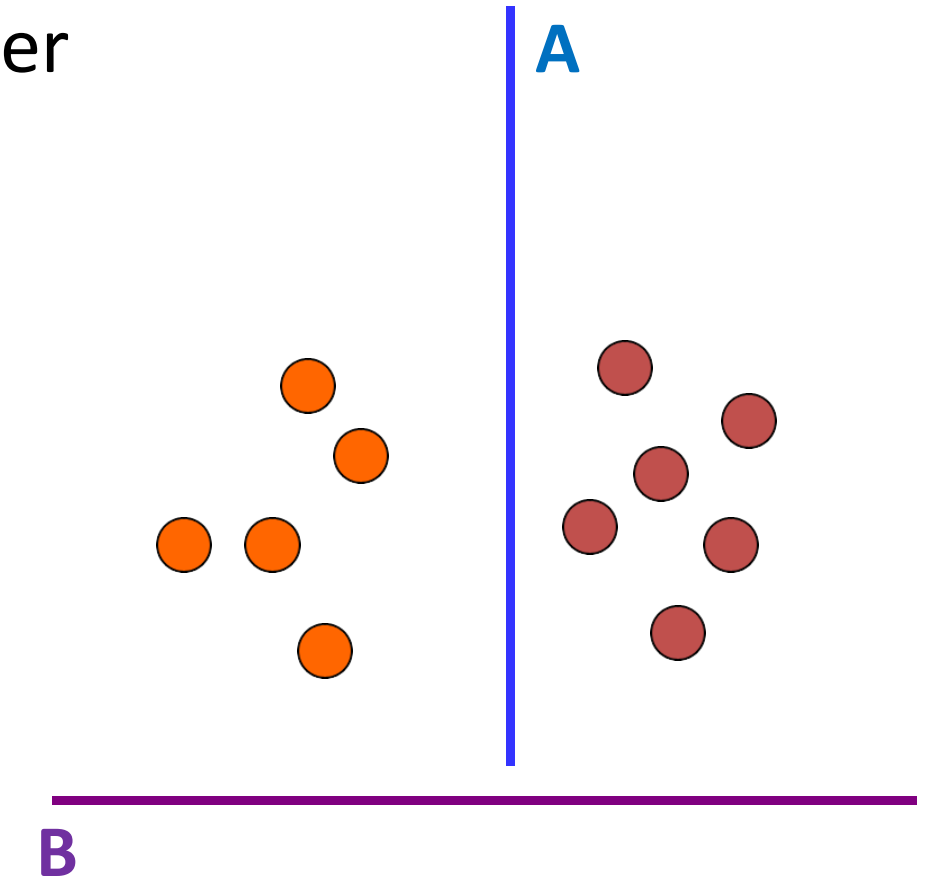
- We will discuss:
  - Linear Discriminant Analysis (LDA)
  - A Unified Framework: Graph Embedding (GE)
  - An Example of GE: Marginal Fisher Analysis (MFA)
- At the end of this lecture, you should be able to:
  - Understand rationales behind LDA and GE
  - Perform LDA

Supervised Feature Learning

Linear Discriminant Analysis

What is a Good Projection?

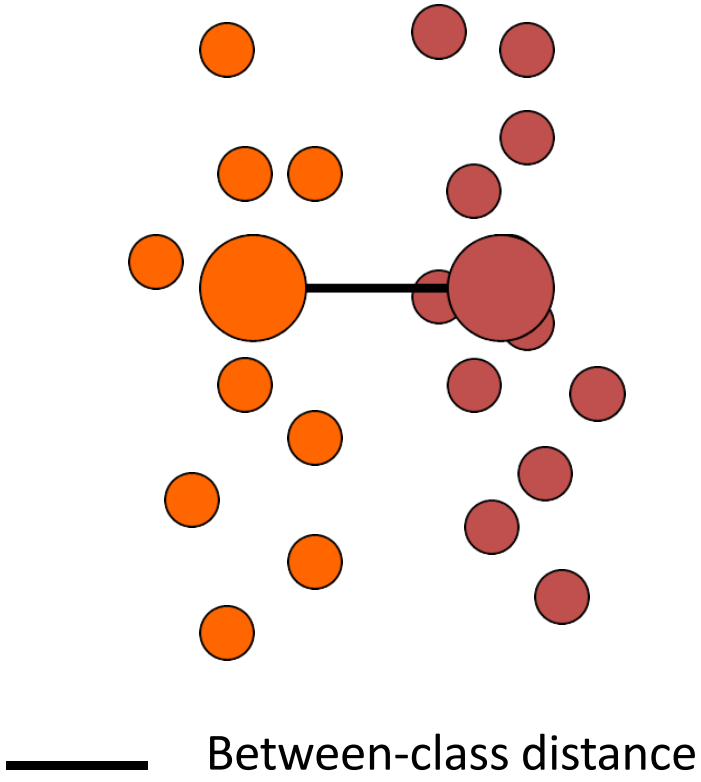
- What is a good criterion under supervised setting?
  - Separating different classes



Which projection axis is better under supervised setting?

What Class Information May be Useful?

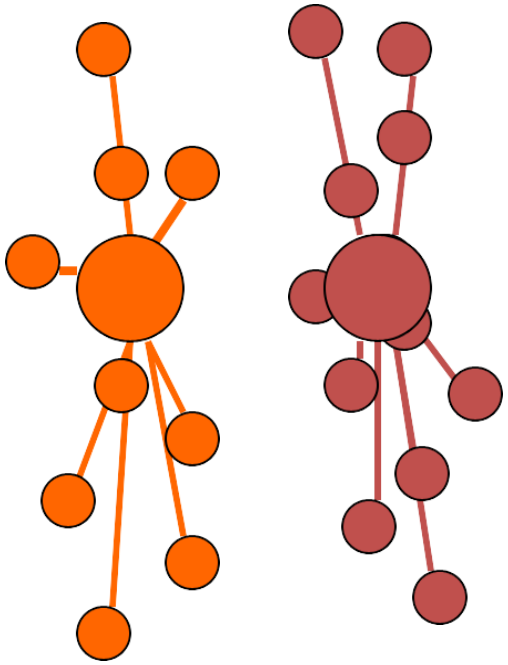
- Between-class distance
  - Distance between the centroids of different classes



What Class Information May be Useful?

- Between-class distance
  - Distance between the centroids of different classes
- Within-class distance
  - Accumulated distance of an instance to the centroid of its class

Should be large or small?

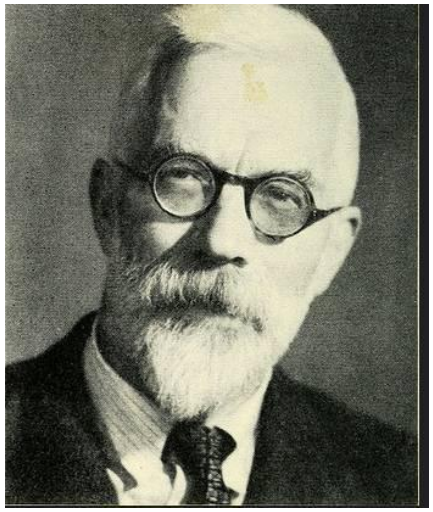


Within-class distance

Linear Discriminant Analysis!

Linear Discriminant Analysis

- A bit history
  - Fisher's Linear Discriminative Analysis (1936)
    - Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems". Annals of Eugenics.
  - Linear Discriminative Analysis (1948)
    - Rao, R. C. (1948). "The utilization of multiple measurements in problems of biological classification". Journal of the Royal Statistical Society, Series B.

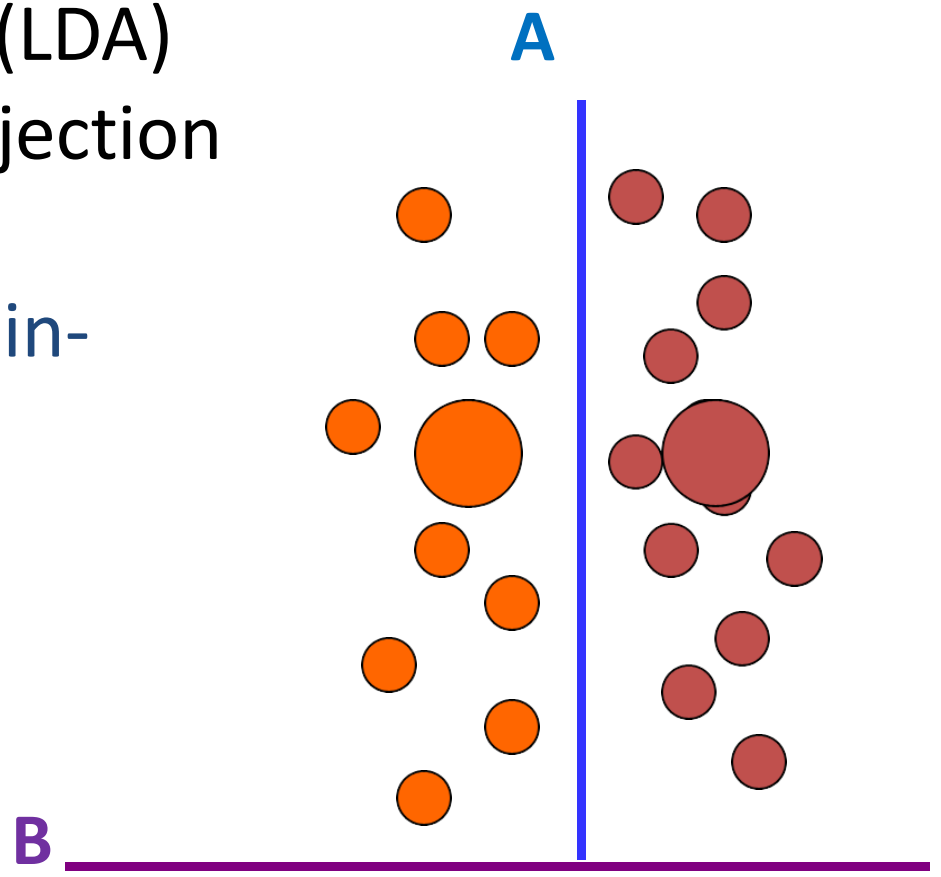


Ronald Fisher (1890-1962)

"A genius who almost single-handedly created the foundations for **modern statistical science**" – Anders Hald

Linear Discriminant Analysis

- **Linear discriminant analysis** (LDA) finds most discriminative projection by maximizing between-class distance and minimizing within-class distance



Which projection axis is possibly the 1st PC, which one is possibly the LDA projection?

Statistical Facts (1)

Quantity for Measuring Within and Between Class Distance

- Total mean vector:

$$\mu = \frac{1}{N}\sum_{\mathbf{x}} \mathbf{x},\ N \text{ is the number of all samples.}$$

- Class-specific mean vector:

$$\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{\mathbf{x}\in C_i} \mathbf{x},\ n_i \text{ is the size of class } C_i.$$

- Class-specific scatter matrix $\widehat{\boldsymbol{S}_i}$:

$$\widehat{\boldsymbol{S}_i} = \sum_{\boldsymbol{x}\in C_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T$$

Statistical Facts (2)

Quantity for Measuring Within and Between Class Distance (cont.)

- Within-class scatter matrix $\widehat{\boldsymbol{S}_w}$:

$$\widehat{\boldsymbol{S}_w} = \sum_{i=1}^{C} \widehat{\boldsymbol{S}_i}$$ C is the class number.

- Between-class scatter matrix $\widehat{\boldsymbol{S}_B}$:

$$\widehat{\boldsymbol{S}_B} = \sum_{i=1}^{C} n_i(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

- Total scatter matrix $\widehat{\boldsymbol{S}_T}$:

$$\widehat{\boldsymbol{S}_T} = \sum_{x} (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T$$

$$= \widehat{\boldsymbol{S}_W} + \widehat{\boldsymbol{S}_B}$$

see next slide

Proof

$$\widehat{S_T} = \sum_{x} (x-\mu)(x-\mu)^T \qquad \widehat{S_w} = \sum_{i=1}^{C} \sum_{x \epsilon C_i} (x-\mu_i)(x-\mu_i)^T \qquad \widehat{S_B} = \sum_{i=1}^{C} n_i(\mu_i-\mu)(\mu_i-\mu)^T$$

$$\widehat{S_w} + \widehat{S_B} = \sum_{i=1}^{C} \sum_{x \epsilon C_i} (x-\mu_i)(x-\mu_i)^T + \sum_{i=1}^{C} n_i(\mu_i-\mu)(\mu_i-\mu)^T$$

$$= \sum_{i=1}^{C} \left(\sum_{x \epsilon C_i} (x-\mu_i)(x-\mu_i)^T + n_i(\mu_i-\mu)(\mu_i-\mu)^T \right)$$

$$= \sum_{i=1}^{C} \left(\sum_{x \epsilon C_i} xx^T - 2x\mu_i^T + \mu_i\mu_i^T + n_i\mu_i\mu_i^T - 2n_i\mu_i\mu^T + n_i\mu\mu^T \right)$$

$$= \sum_{i=1}^{C} \left(n_i\mu_i\mu_i^T - 2n_i\mu_i\mu_i^T + \sum_{x \epsilon C_i} xx^T + n_i\mu_i\mu_i^T - 2n_i\mu_i\mu^T + n_i\mu\mu^T \right)$$

$$= \sum_{i=1}^{C} \left(\sum_{x \epsilon C_i} xx^T - 2 \sum_{x \epsilon C_i} x\mu^T + n_i\mu\mu^T \right)$$

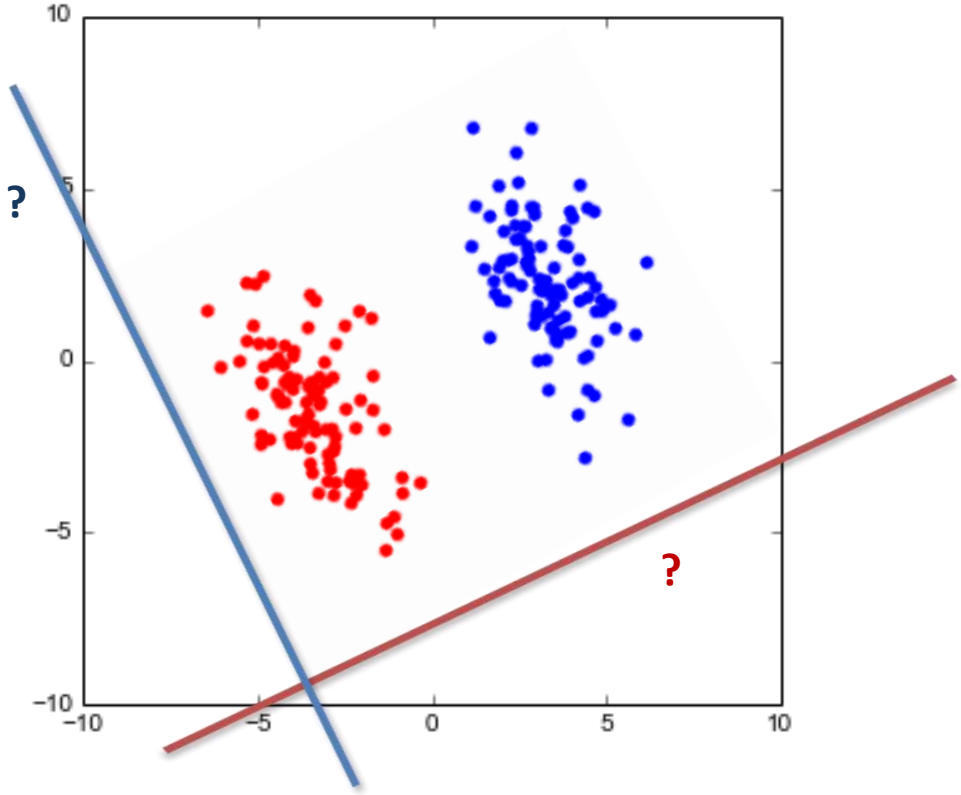
$$= \sum_{x} xx^T - 2x\mu^T + \mu\mu^T$$

$$= \sum_x (x-\mu)(x-\mu)^T = S_T$$

within-class scatter + between-class scatter
= total scatter.

Linear Discriminant Analysis - Problem

- **PROBLEM: To separate populations**
  - Suppose we have C classes from D-dimensional distributions. We want to project these classes onto a p-dimensional subspace ($p < D$) so that the variation between the classes is **as large as possible**, relative to the variation within the classes.



LDA: Two-class

A simple case

- Assume we have a set of d-dimensional samples $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, n_1 of which from class C_1, and n_2 of which from class C_2

- We seek for a linear projection vector $\boldsymbol{w}$, along which we project the samples $\boldsymbol{x}$ onto a line and get a scalar y:

$$y = \boldsymbol{w}^T \boldsymbol{x}$$

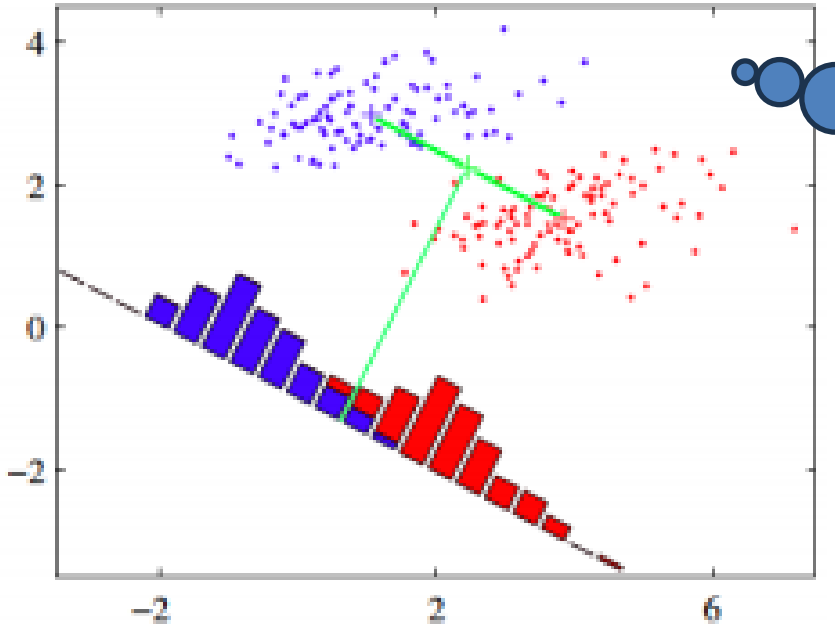
such that, the separability of the projected samples between 2 classes are maximized.

LDA: Two-class

- If the class separability measure is only maximizing the distance between projected means $\widetilde{\mu_1}$ and $\widetilde{\mu_2}$;

$$J(\boldsymbol{w}) = |\widetilde{\mu_1} - \widetilde{\mu_2}| = |\boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)|$$

where $\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\epsilon C_i} \boldsymbol{x}$ and $\widetilde{\mu}_i = \boldsymbol{w}^T\boldsymbol{\mu}_i$



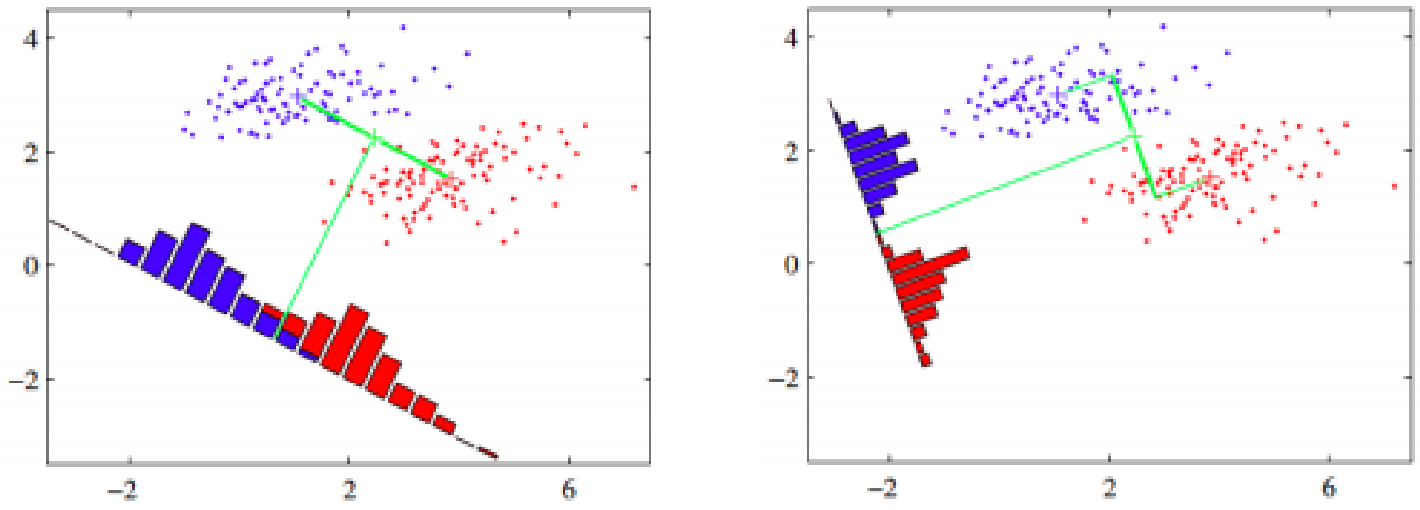
Is it the optimal solution?

LDA: Two-class

- If the class separability measure is only maximizing the distance between projected means $\widetilde{\mu_1}$ and $\widetilde{\mu_2}$:

$$J(\boldsymbol{w}) = |\widetilde{\mu_1} - \widetilde{\mu_2}| = |\boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)|$$

where $\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\epsilon C_i}\boldsymbol{x}$ and $\widetilde{\mu_i} = \boldsymbol{w}^T\boldsymbol{\mu}_i$



LDA: Two-class

- Fisher suggested maximizing the difference between the projected means, normalized by a measure of the projected within-class scatter!
- For each class, the projected scatter is:

$$\tilde{S}_i = \sum_{y \in C_i} (y - \tilde{\mu}_i)^2$$

then, $(\tilde{S}_1 + \tilde{S}_2)$ is the within-class scatter of the projected samples.

- Therefore, the criterion function: $J(\boldsymbol{w}) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{\tilde{S}_1 + \tilde{S}_2}$

LDA: Two-class

- To find the optimum $\boldsymbol{w}^*$, we need write $J(\boldsymbol{w})$ as a function of $\boldsymbol{w}$:

$$J(\boldsymbol{w}) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{\tilde{s}_1 + \tilde{s}_2} = \frac{(\boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2))^2}{\sum_{y \in C_1}(y - \tilde{\mu}_1)^2 + \sum_{y \in C_2}(y - \tilde{\mu}_2)^2}$$

$$= \frac{\boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T\boldsymbol{w}}{\sum_{i=1}^{C}\sum_{\boldsymbol{x} \in C_i}(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu}_i)^2}$$

$$= \frac{\boldsymbol{w}^T \frac{n_1 + n_2}{n_1 n_2}\widehat{\boldsymbol{S}_B}\boldsymbol{w}}{\boldsymbol{w}^T\left[\sum_{i=1}^{C}\sum_{\boldsymbol{x} \in C_i}(\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T\right]\boldsymbol{w}}$$

see next slide

$$= \frac{n_1 + n_2}{n_1 n_2}\frac{\boldsymbol{w}^T\widehat{\boldsymbol{S}_B}\boldsymbol{w}}{\boldsymbol{w}^T\widehat{\boldsymbol{S}_W}\boldsymbol{w}}$$

Note that $\widehat{\boldsymbol{S}_B}$ has a rank of at most 1.

Proof

- Prove $\widehat{\boldsymbol{S}_B} = \frac{n_1 n_2}{n_1+n_2}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T$

$$\widehat{\boldsymbol{S}_B} = \sum_{i=1}^{C} n_i(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

$$= n_1(\boldsymbol{\mu}_1 - \boldsymbol{\mu})(\boldsymbol{\mu}_1 - \boldsymbol{\mu})^T + n_2(\boldsymbol{\mu}_2 - \boldsymbol{\mu})(\boldsymbol{\mu}_2 - \boldsymbol{\mu})^T$$

$$= n_1\left(\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T - 2\boldsymbol{\mu}_1\boldsymbol{\mu}^T + \boldsymbol{\mu}\boldsymbol{\mu}^T\right) + n_2\left(\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - 2\boldsymbol{\mu}_2\boldsymbol{\mu}^T + \boldsymbol{\mu}\boldsymbol{\mu}^T\right)$$

$$(n_1+n_2)\boldsymbol{\mu} = n_1\boldsymbol{\mu}_1 + n_2\boldsymbol{\mu}_2$$

$$= n_1\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + n_2\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - (n_1+n_2)\boldsymbol{\mu}\boldsymbol{\mu}^T$$

$$= n_1\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + n_2\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - \frac{(n_1\boldsymbol{\mu}_1 + n_2\boldsymbol{\mu}_2)(n_1\boldsymbol{\mu}_1 + n_2\boldsymbol{\mu}_2)^T}{n_1+n_2}$$

$$= \frac{n_1 n_2\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + n_1 n_2\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - 2n_1 n_2\boldsymbol{\mu}_1\boldsymbol{\mu}_2^T}{n_1+n_2}$$

$$= \frac{n_1 n_2}{n_1+n_2}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T$$

LDA: Two-class

- We can finally express the Fisher criterion in terms of $\widehat{S}_W$ and $\widehat{S}_B$ as

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \widehat{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w}}$$

Ignore $\frac{n_1+n_2}{n_1 n_2}$

- To find the maximum of $J(\boldsymbol{w})$, we **take derivative** and **set to zero**

$$\frac{d}{d\boldsymbol{w}} J(\boldsymbol{w}) = \mathbf{0}$$

$$\frac{d}{dx}\left(\frac{f(x)}{g(x)}\right) = \frac{g(x)\frac{d}{dx}f(x) - f(x)\frac{d}{dx}g(x)}{(g(x))^2}$$

$$\frac{(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w}) \frac{d}{d\boldsymbol{w}}(\boldsymbol{w}^T \widehat{S}_B \boldsymbol{w}) - (\boldsymbol{w}^T \widehat{S}_B \boldsymbol{w}) \frac{d}{d\boldsymbol{w}}(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w})}{(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w})^2} = \mathbf{0}$$

$$\frac{d \boldsymbol{w}^T A \boldsymbol{w}}{d \boldsymbol{w}} = (A + A^T)\boldsymbol{w}$$

$$2(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w})\widehat{S}_B \boldsymbol{w} - 2(\boldsymbol{w}^T \widehat{S}_B \boldsymbol{w})\widehat{S}_W \boldsymbol{w} = \mathbf{0}$$

Divide by $(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w})$

$$\widehat{S}_B \boldsymbol{w} - \frac{(\boldsymbol{w}^T \widehat{S}_B \boldsymbol{w})}{(\boldsymbol{w}^T \widehat{S}_W \boldsymbol{w})} \widehat{S}_W \boldsymbol{w} = \mathbf{0}$$

$$\widehat{S}_B \boldsymbol{w} - J(\boldsymbol{w})\widehat{S}_W \boldsymbol{w} = \mathbf{0} \Rightarrow \boxed{\widehat{S}_B \boldsymbol{w} = J(\boldsymbol{w})\widehat{S}_W \boldsymbol{w}}$$

LDA: Two-class

- The generalized eigenvalue problem ($\boldsymbol{Av} = \lambda \boldsymbol{Bv}$)

$$\widehat{\boldsymbol{S}_B}\boldsymbol{w} = J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\boldsymbol{w}$$

  - If $\widehat{\boldsymbol{S}_W}$ is invertible, convert to standard eigenvalue problem ($\boldsymbol{Av} = \lambda \boldsymbol{v}$):

$$\widehat{\boldsymbol{S}_W}^{-1}\widehat{\boldsymbol{S}_B}w = J(\boldsymbol{w})\boldsymbol{w}$$

Then, $\boldsymbol{w}^*$ is the eigenvector of $\widehat{\boldsymbol{S}_W}^{-1}\widehat{\boldsymbol{S}_B}$.
Its corresponding eigenvalue, i.e., $J(\boldsymbol{w})^*$, should be the largest.

  - If $\widehat{\boldsymbol{S}_W}$ is not invertible,

$$\left(\widehat{\boldsymbol{S}_B} - J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\right)\boldsymbol{w} = \boldsymbol{0}$$

$$det\left(\widehat{\boldsymbol{S}_B} - J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\right) = 0$$

Examples: LDA: Two-class

Given the following dataset, compute the LDA projection vector **w**

Sample	Feature 1	Feature 2	Class
1	2	3	1
2	3	3	1
3	2	4	1
4	6	7	2
5	7	7	2
6	6	8	2

Examples: LDA: Two-class

N=6, n_1=3, n_2=3

Sample	Feature 1	Feature 2	Class
1	2	3	1
2	3	3	1
3	2	4	1
4	6	7	2
5	7	7	2
6	6	8	2

Step 1: Compute $\boldsymbol{\mu}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2$

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i = \frac{1}{6}\left(\begin{bmatrix}2\\3\end{bmatrix}+\begin{bmatrix}3\\3\end{bmatrix}+\begin{bmatrix}2\\4\end{bmatrix}+\begin{bmatrix}6\\7\end{bmatrix}+\begin{bmatrix}7\\7\end{bmatrix}+\begin{bmatrix}6\\8\end{bmatrix}\right) = \begin{bmatrix}4.33\\5.33\end{bmatrix}$$

$$\boldsymbol{\mu}_1 = \frac{1}{3}\sum_{i=1}^{3} \boldsymbol{x}_i = \frac{1}{3}\left(\begin{bmatrix}2\\3\end{bmatrix}+\begin{bmatrix}3\\3\end{bmatrix}+\begin{bmatrix}2\\4\end{bmatrix}\right) = \begin{bmatrix}2.33\\3.33\end{bmatrix}$$

$$\boldsymbol{\mu}_2 = \frac{1}{3}\sum_{i=4}^{6} \boldsymbol{x}_i = \frac{1}{3}\left(\begin{bmatrix}6\\7\end{bmatrix}+\begin{bmatrix}7\\7\end{bmatrix}+\begin{bmatrix}6\\8\end{bmatrix}\right) = \begin{bmatrix}6.33\\7.33\end{bmatrix}$$

Step 2: Compute $\widehat{\boldsymbol{S}_W}, \widehat{\boldsymbol{S}_B}$

$$\widehat{\boldsymbol{S}_W} = \sum_{i=1}^{C}\sum_{\boldsymbol{x}\epsilon C_i}(\boldsymbol{x}-\boldsymbol{\mu_i})(\boldsymbol{x}-\boldsymbol{\mu_i})^T = \left(\begin{bmatrix}2\\3\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)\left(\begin{bmatrix}2\\3\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)^T + \left(\begin{bmatrix}3\\3\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)\left(\begin{bmatrix}3\\3\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)^T$$
$$+\left(\begin{bmatrix}2\\4\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)\left(\begin{bmatrix}2\\4\end{bmatrix}-\begin{bmatrix}2.33\\3.33\end{bmatrix}\right)^T + \left(\begin{bmatrix}6\\7\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)\left(\begin{bmatrix}6\\7\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)^T + \left(\begin{bmatrix}7\\7\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)\left(\begin{bmatrix}7\\7\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)^T$$
$$+\left(\begin{bmatrix}6\\8\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)\left(\begin{bmatrix}6\\8\end{bmatrix}-\begin{bmatrix}6.33\\7.33\end{bmatrix}\right)^T = \begin{bmatrix}1.33 & -0.67\\-0.67 & 1.33\end{bmatrix}$$

$$\widehat{\boldsymbol{S}_B} = \sum_{i=1}^{C} n_i(\boldsymbol{\mu}_i-\boldsymbol{\mu})(\boldsymbol{\mu}_i-\boldsymbol{\mu})^T$$
$$= 3\left(\begin{bmatrix}2.33\\3.33\end{bmatrix}-\begin{bmatrix}4.33\\5.33\end{bmatrix}\right)\left(\begin{bmatrix}2.33\\3.33\end{bmatrix}-\begin{bmatrix}4.33\\5.33\end{bmatrix}\right)^T + 3\left(\begin{bmatrix}6.33\\7.33\end{bmatrix}-\begin{bmatrix}4.33\\5.33\end{bmatrix}\right)\left(\begin{bmatrix}6.33\\7.33\end{bmatrix}-\begin{bmatrix}4.33\\5.33\end{bmatrix}\right)^T = \begin{bmatrix}24 & 24\\24 & 24\end{bmatrix}$$

Examples: LDA: Two-class

Step 3: Compute eigenvector $\boldsymbol{w}^*$

Sample	Feature 1	Feature 2	Class
1	2	3	1
2	3	3	1
3	2	4	1
4	6	7	2
5	7	7	2
6	6	8	2

$$\left(\widehat{\boldsymbol{S}_B} - J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\right)\boldsymbol{w} = \boldsymbol{0}$$

$$det\left(\widehat{\boldsymbol{S}_B} - J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\right) = 0$$

$$\det\left(\begin{bmatrix}24 & 24\\ 24 & 24\end{bmatrix} - J(\boldsymbol{w})\begin{bmatrix}1.33 & -0.67\\ -0.67 & 1.33\end{bmatrix}\right) = 0$$

$$\det\left(\begin{bmatrix}24 - 1.33J(\boldsymbol{w}) & 24 + 0.67J(\boldsymbol{w})\\ 24 + 0.67J(\boldsymbol{w}) & 24 - 1.33J(\boldsymbol{w})\end{bmatrix}\right) = 0$$

$$\left(24 - 1.33J(\boldsymbol{w})\right)^2 - \left(24 + 0.67J(\boldsymbol{w})\right)^2 = 0$$

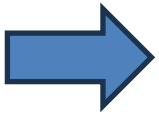
$$J_1 = 72, \qquad J_2 = 0$$

Since J(**w**) should be maximized, J(**w**)= $J_1 = 72$

Then, $\left(\widehat{\boldsymbol{S}_B} - J(\boldsymbol{w})\widehat{\boldsymbol{S}_W}\right)\boldsymbol{w} = \boldsymbol{0}$

$$\begin{bmatrix}-72 & 72\\ 72 & -72\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix} = 0$$

$$w_1 = w_2$$



After normalization, **w***= $\pm\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}$

LDA: Multi-class

- Instead of **one projection** y, we will now seek at most p=min(d, C −1) projections [y_1, y_2,…y_p] by means of p projection vectors $\boldsymbol{w}_i$ arranged by columns into a projection matrix $\boldsymbol{W}$= [$\boldsymbol{w}_1|\boldsymbol{w}_2|\ldots|\boldsymbol{w}_p$]:

$$y_i = \boldsymbol{w}_i^T \boldsymbol{x} \quad \Rightarrow \quad \boldsymbol{y} = \boldsymbol{W}^T \boldsymbol{x}$$

- From our derivation for the two-class problem, we have the scatter matrices <u>for the projected samples</u>

$$\tilde{\boldsymbol{S}}_W = \boldsymbol{W}^T \widehat{\boldsymbol{S}_W} \boldsymbol{W}$$

$$\tilde{\boldsymbol{S}}_B = \boldsymbol{W}^T \widehat{\boldsymbol{S}_B} \boldsymbol{W}$$

LDA: Multi-class

- we use the determinant of the scatter matrices to obtain a scalar objective function

$$J(\boldsymbol{W}) = \frac{\det(\tilde{\boldsymbol{S}}_B)}{\det(\tilde{\boldsymbol{S}}_W)} = \frac{\det(\boldsymbol{W}^T \widehat{\boldsymbol{S}_B} \boldsymbol{W})}{\det(\boldsymbol{W}^T \widehat{\boldsymbol{S}_W} \boldsymbol{W})}$$

- The optimal projection matrix $\boldsymbol{W}^*$ that maximizes $J(\boldsymbol{W})$ is the one whose columns are the eigenvectors $\boldsymbol{w}_i^*$ corresponding to the largest eigenvalues of $\widehat{\boldsymbol{S}_W}^{-1} \widehat{\boldsymbol{S}_B}$ (if $\widehat{\boldsymbol{S}_W}$ is invertible)

$$\boldsymbol{W}^* = argmax \frac{\det(\boldsymbol{W}^T \widehat{\boldsymbol{S}_B} \boldsymbol{W})}{\det(\boldsymbol{W}^T \widehat{\boldsymbol{S}_W} \boldsymbol{W})} \Rightarrow \widehat{\boldsymbol{S}_W}^{-1} \widehat{\boldsymbol{S}_B} \boldsymbol{w}_i^* = J(\boldsymbol{w}_i^*)\, \boldsymbol{w}_i^*$$

OR

$$\widehat{\boldsymbol{S}_B} \boldsymbol{w}_i^* = J(\boldsymbol{w}_i^*)\, \widehat{\boldsymbol{S}_W}\, \boldsymbol{w}_i^*$$

Classification with LDA

- Compute LDA projection vectors using training dataset
- Project both training and testing data into the obtained LDA space
- For each testing data point, use K-Nearest-Neighbor (KNN) or other classifiers for classification

Basis Visualization

Sample Images from YALE database

(a)

(b)

The first 10 (a) Eigenfaces, (b) Fisherfaces, calculated from the face images in the YALE database.

PCA LDA

Some Remarks

- Unlike principal component analysis (PCA), the linear discriminant transformation $\boldsymbol{W}$ is not necessarily orthogonal.
  - Because $\widehat{\boldsymbol{S}_W}^{-1}\widehat{\boldsymbol{S}_B}$ may not be symmetric.

- Why is the number of the LDA projections at most min(d,C-1)?

Some Remarks

- Why is the number of the LDA projections at most min(d, C-1)?

⇩

meaningful generalized eigenvectors of ($\widehat{\boldsymbol{S}_B}$, $\widehat{\boldsymbol{S}_w}$) is at most min($d$,$C$-1)
(Nonzero eigenvalues)

$$\widehat{\boldsymbol{S}_B}\boldsymbol{w}_i^* = J(\boldsymbol{w}_i^*)\,\widehat{\boldsymbol{S}_W}\,\boldsymbol{w}_i^*$$

meaningful generalized eigenvectors is limited by Rank($\widehat{\boldsymbol{S}_B}$)

$$\widehat{\boldsymbol{S}_B} = \sum_{i=1}^{C} n_i(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

rank <= 1

Rank subadditivity rule

$rank(A + B) \le rank(A) + rank(B)$

rank($\widehat{\boldsymbol{S}_B}$) <= C

Rank can't exceed the matrix dimension d

$$\sum_{i=1}^{C} n_i(\boldsymbol{\mu}_i - \boldsymbol{\mu}) = \boldsymbol{0} \Rightarrow \sum_{i=1}^{C} n_i\boldsymbol{\mu}_i = \boldsymbol{0}$$

Rank($\widehat{\boldsymbol{S}_B}$)<= C-1

Rank($\widehat{\boldsymbol{S}_B}$)<= min(d, C-1)

Question

- Shall LDA always be better than PCA for classification task?

Is LDA always better than PCA?

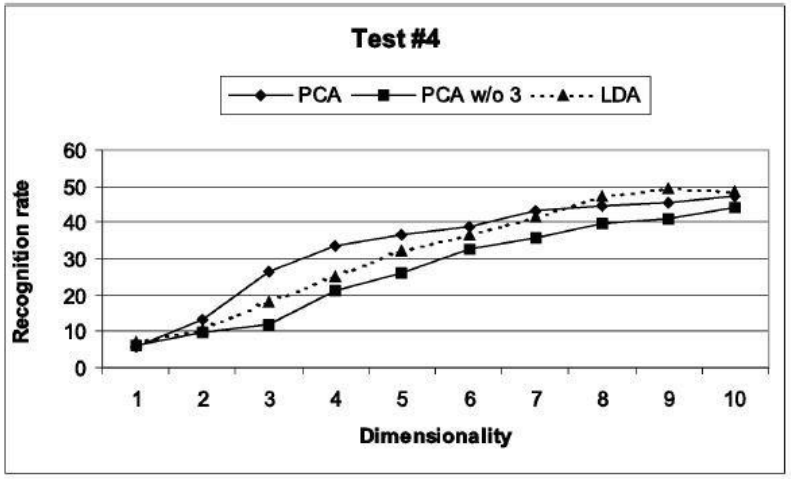
- Case Study: PCA versus LDA
  - A. Martinez, A. Kak, "PCA versus LDA", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228-233, 2001.

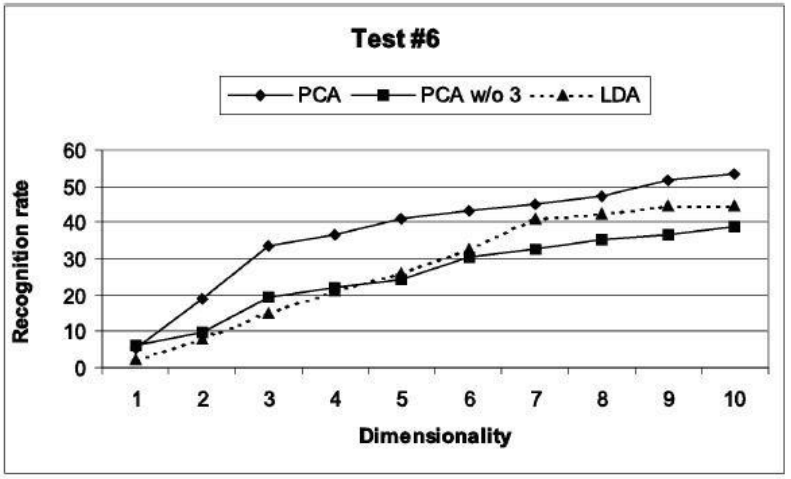
One subject in AR database

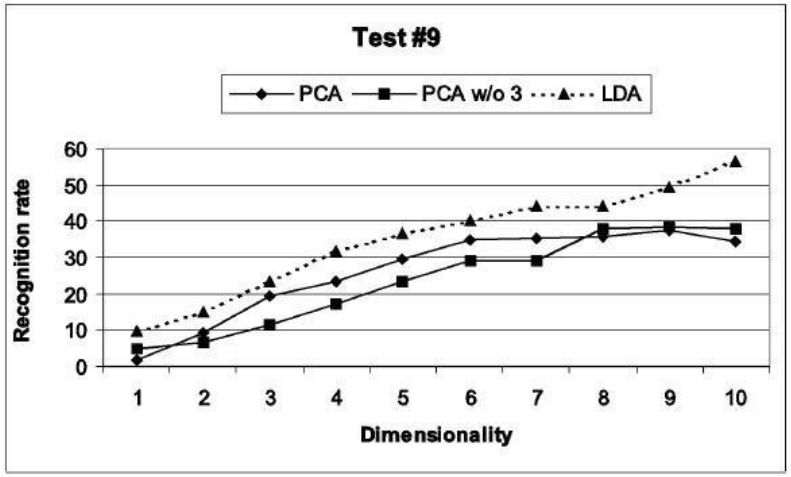
Is LDA always better than PCA?

- Small training dataset

  (2 images/person for training and 5 images/person for testing)







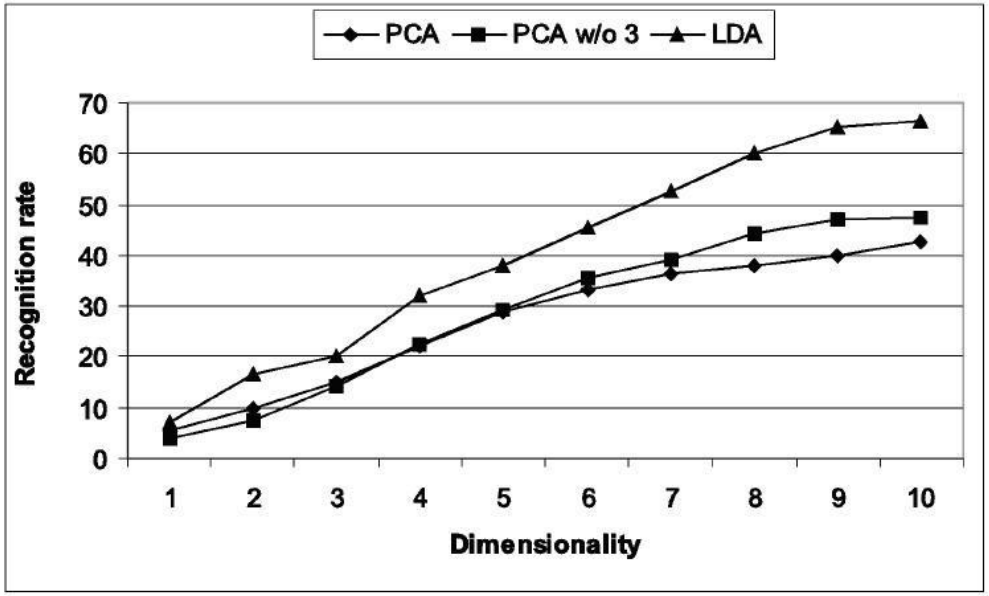
LDA is not always better when the sample size is small

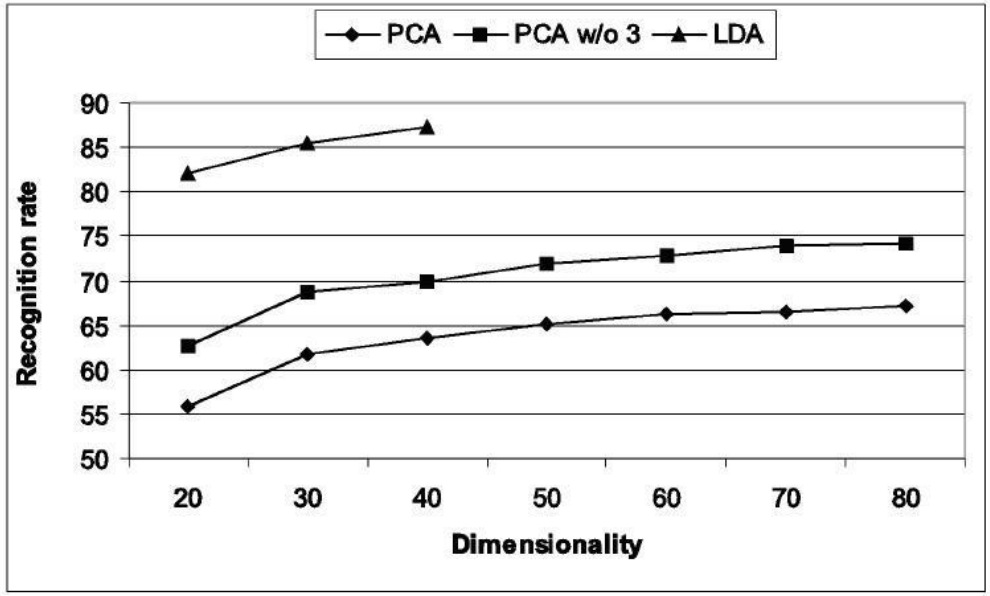
PCA w/o 3: PCA without using the top 3 eigenvectors. #i index train/test splits

Is LDA always better than PCA?

- Large training dataset

  (13 images/person for training and 13 images/person for testing)

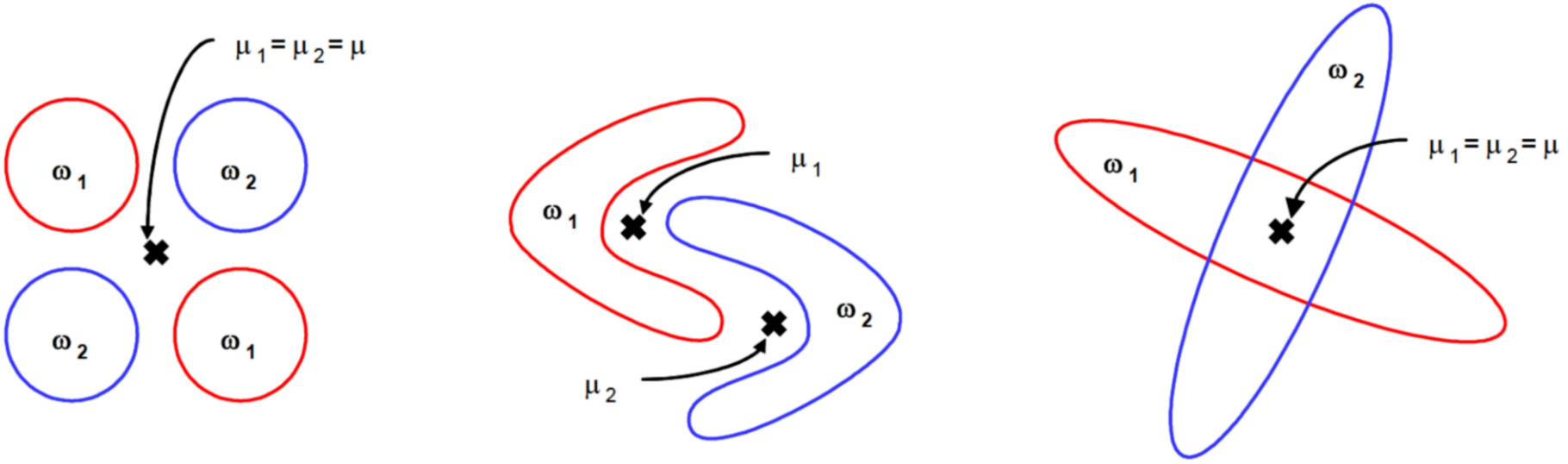




LDA outperforms PCA when the sample size is large

Limitations

- **LDA produces at most min(d, $C - 1$) feature projections**
  - If more features are needed, some other method must be employed to provide those additional features
- **LDA is a parametric method (it assumes unimodal Gaussian likelihoods)**
  - If the distributions are non-Gaussian, LDA projections may not preserve complex structure in the data needed for classification



A general framework:

Graph Embedding (GE)

Shuicheng Yan, Dong Xu et al.: Graph Embedding and Extensions: A General Framework for Dimensionality Reduction. T-PAMI 2007.

GE: Problem Formulation of Feature Learning

- For a general feature learning problem

  Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

  Goal: Find a mapping function F: $\mathbf{x} \rightarrow \boldsymbol{y}$, where $\boldsymbol{y}_i \in \mathbb{R}^{d'}, d' \ll d$

  (i.e., $\boldsymbol{y} = \mathbf{F}(\boldsymbol{x})$)

- From the perspective of **Graph Embedding**

  ✓ Undirected Weighted Graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

  Vertex set; Similarity (edge) matrix $\in \mathbb{R}^{N \times N}$

  Dissimilar: negative

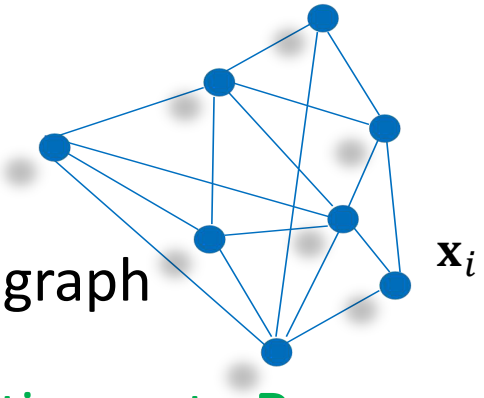
  $\mathbf{S}_{i,i} = 0$

  Similar: positive

  ✓ Diagonal matrix **D** and the Laplacian matrix **L** of G

  Sum of similarities between sample $\mathbf{x}_i$ and the other samples $\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j}$

  $\mathbf{L} = \mathbf{D} - \mathbf{S}$



G: intrinsic graph

✓ Goal: Find the desired low-dimensional vector representations **y** to **Preserve** these graph similarities.

GE: Problem Formulation of Feature Learning

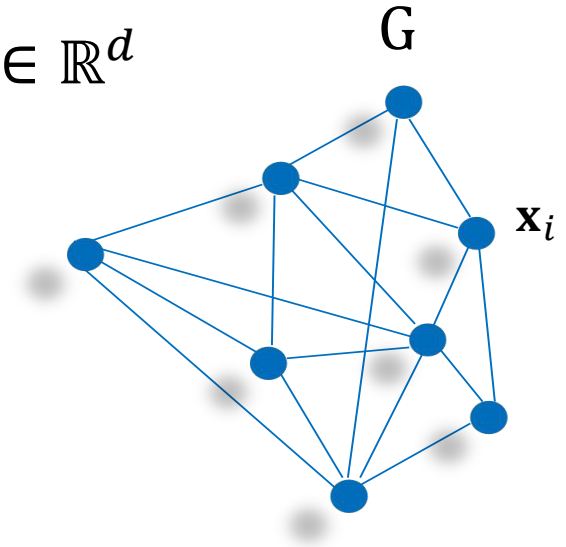
- **Assume a simpler case (y is 1D)**

Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

Intrinsic graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

Diagonal matrix **D** and the Laplacian matrix **L** of G

$$\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j} \qquad \mathbf{L} = \mathbf{D} - \mathbf{S}$$



Goal: Find y $\in \mathbb{R}$ **to preserve graph similarity** ($\mathbf{y} = [y_1, y_2, \dots, y_N]^T$)

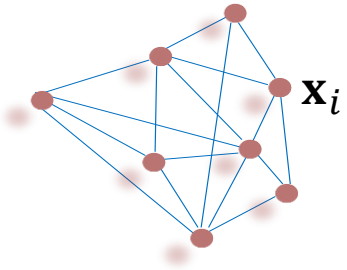
$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}_{i,j}$$

c: constant,
B: constraint matrix to avoid trivial solution

B=
- A diagonal matrix**: Scale normalization**
- Laplacian of a penalty graph G^p:
  $\mathbf{B} = \mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$

$G^p = \{\mathbf{X}, \mathbf{S}^p\}$

Strength of a relationship you want to penalize



GE: Problem Formulation of Feature Learning

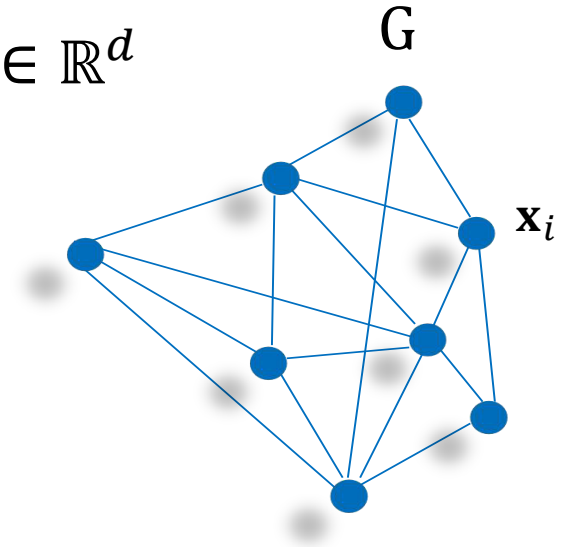
- **Assume a simpler case (y is 1D)**

Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

Intrinsic graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

The Diagonal matrix **D** and the Laplacian matrix **L** of G

$$\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j} \qquad \mathbf{L} = \mathbf{D} - \mathbf{S}$$



Goal: Find y $\in \mathbb{R}$ **to preserve graph similarity** ($\mathbf{y} = [y_1, y_2, \dots, y_N]^T$)

$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}_{i,j}$$

c: constant,
B: constraint matrix to avoid trivial solution

B=

- A diagonal matrix: **Scale normalization** — $\mathbf{y}^T\mathbf{B}\mathbf{y} = c \Leftrightarrow \sum_i \|y_i\|^2 \mathbf{B}_{i,i} = c$
- Laplacian of a penalty graph G^p: $\mathbf{B} = \mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$ — $\mathbf{y}^T\mathbf{B}\mathbf{y} = c \Leftrightarrow \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}^p_{i,j} = 2c$

Proof of Equivalence in Constraint

$$\mathbf{y}^T\mathbf{B}\mathbf{y} = c \Leftrightarrow \sum_{i\neq j}\|y_i - y_j\|^2 \mathbf{S}^p_{i,j} = 2c$$

$$\mathbf{D}^p_{i,i} = \sum_{i\neq j} S^p_{i,j}$$

$$\mathbf{B} = \mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$$

$$\mathbf{y}^T\mathbf{B}\mathbf{y} = \mathbf{y}^T\mathbf{D}^p\mathbf{y} - \mathbf{y}^T\mathbf{S}^p\mathbf{y}$$

$$= \sum_i \mathbf{D}^p_{i,i} y_i^2 - \sum_{i,j} y_i S^p_{i,j} y_j$$

$$\sum_{i\neq j}\|y_i - y_j\|^2 \mathbf{S}^p_{i,j} = \sum_{i\neq j}(y_i^2 - 2y_iy_j + y_j^2)\mathbf{S}^p_{i,j}$$

$$= \sum_{i\neq j} y_i^2 \mathbf{S}^p_{i,j} - \sum_{i\neq j} 2y_iy_j\mathbf{S}^p_{i,j} + \sum_{i\neq j} y_j^2\mathbf{S}^p_{i,j}$$

$$= 2\sum_i y_i^2 \sum_{j\neq i} S^p_{i,j} - \sum_{i\neq j} 2y_iy_j\mathbf{S}^p_{i,j}$$

$$= 2\left(\sum_i y_i^2\mathbf{D}^p_{i,i} - \sum_{i\neq j} y_iy_j\mathbf{S}^p_{i,j}\right)$$

Since typically, no penalty should be posed when i=j (i.e., $\mathbf{S}^p_{i,i} = 0$)

$$= 2\left(\sum_i y_i^2\mathbf{D}^p_{i,i} - \sum_{i,j} y_iy_j\mathbf{S}^p_{i,j}\right)$$

GE: Problem Formulation of Feature Learning

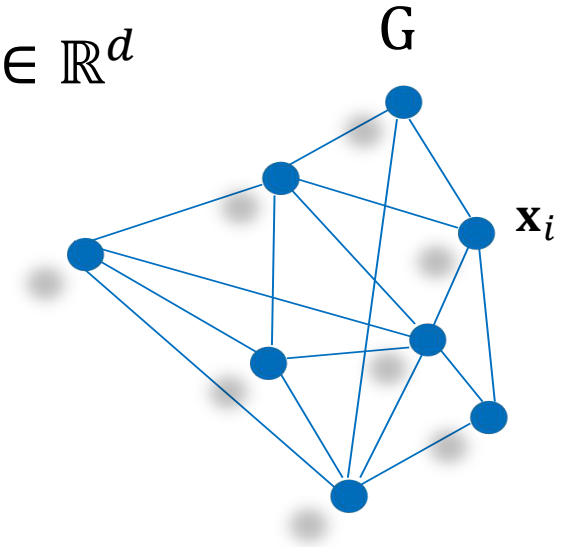
- **Assume a simpler case (y is 1D)**

Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

Intrinsic graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

The Diagonal matrix **D** and the Laplacian matrix **L** of G

$$\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j} \qquad \mathbf{L} = \mathbf{D} - \mathbf{S}$$



Goal: Find y $\in \mathbb{R}$ **to preserve graph similarity** ($\mathbf{y} = [y_1, y_2, \dots, y_N]^T$)

$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}_{i,j}$$

Why this preserves graph similarity?

c: constant,

B: constraint matrix to avoid trivial solution

GE: Problem Formulation of Feature Learning

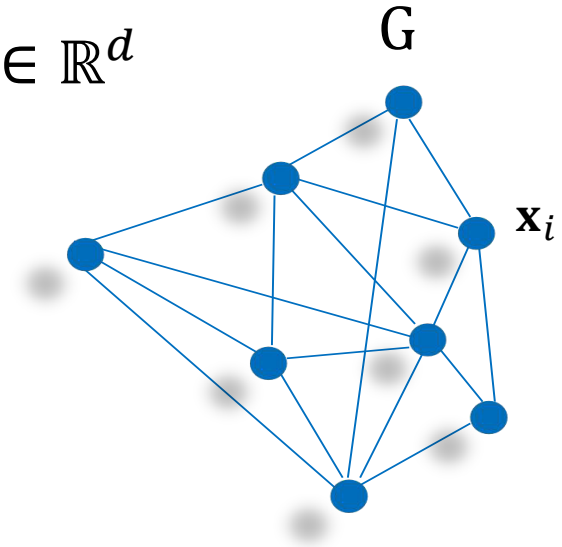
- **Assume a simpler case (y is 1D)**

Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

Intrinsic graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

The Diagonal matrix **D** and the Laplacian matrix **L** of G

$$\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j} \qquad \mathbf{L} = \mathbf{D} - \mathbf{S}$$



Goal: Find y $\in \mathbb{R}$ **to preserve graph similarity** ($\mathbf{y} = [y_1, y_2, \dots, y_N]^T$)

$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}_{i,j} = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \mathbf{y}^T\mathbf{L}\mathbf{y}$$

c: constant,

B: constraint matrix to avoid trivial solution

GE: Linearization

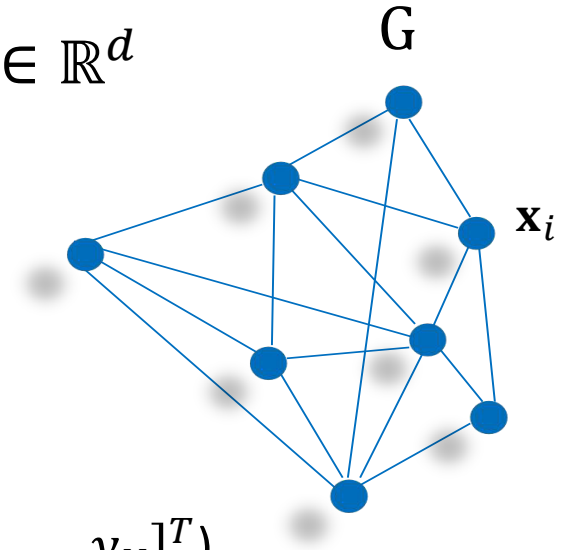
- **Assume $\mathbf{y} = \mathbf{X}^T\mathbf{w}$**

Data matrix for training: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$, where $\mathbf{x}_i \in \mathbb{R}^d$

Intrinsic graph: $\mathrm{G} = \{\mathbf{X}, \mathbf{S}\}$

The Diagonal matrix **D** and the Laplacian matrix **L** of G

$$\mathbf{D}_{i,i} = \sum_{i \neq j} \mathbf{S}_{i,j} \qquad \mathbf{L} = \mathbf{D} - \mathbf{S}$$



Goal: Find $y \in \mathbb{R}$ **to preserve graph similarity** $(\mathbf{y} = [y_1, y_2, \dots, y_N]^T)$

$$\mathbf{y}^* = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \sum_{i \neq j} \|y_i - y_j\|^2 \mathbf{S}_{i,j} = \underset{\mathbf{y}^T\mathbf{B}\mathbf{y}=c}{\operatorname{argmin}} \mathbf{y}^T\mathbf{L}\mathbf{y}$$

$$\mathbf{w}^* = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \sum_{i \neq j} \|\mathbf{w}^T\mathbf{x}_i - \mathbf{w}^T\mathbf{x}_j\|^2 \mathbf{S}_{i,j} = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}$$

GE: Linearization

- **Assume $\mathbf{y} = \mathbf{X}^T\mathbf{w}$**

$$\mathbf{w}^* = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}$$

Lagrange Multipliers:
Given f to be optimized, subject to the constraint g=0, then, solve: $\nabla f = \lambda \nabla g$

$$f = \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}, \qquad g = \mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w} - \mathrm{c}$$

Let λ be a Lagrange multiplier,

$$\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}) = \lambda\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w} - \mathrm{c})$$

$$\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}$$

Generalized eigenvalue decomposition problem:

$$\tilde{\mathbf{L}}\mathbf{w} = \lambda\tilde{\mathbf{B}}\mathbf{w}$$

Where $\tilde{\mathbf{L}} = \mathbf{X}\mathbf{L}\mathbf{X}^T$ and $\tilde{\mathbf{B}} = \mathbf{X}\mathbf{B}\mathbf{X}^T$

GE: Kernelization

- **Extend to Nonlinear Cases**

Map the data to a higher dimensional space: $\mathbf{x}_i \rightarrow \boldsymbol{\varphi}(\mathbf{x}_i)$

A *kernel function* K: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\varphi}(\boldsymbol{x}_i)^T \boldsymbol{\varphi}(\boldsymbol{x}_j)$

e.g., data **x** is 2-D, i.e., $\boldsymbol{x} = (x_1, x_2)^T$

We map it to a 6-D space: $\boldsymbol{\varphi}(\mathbf{x})=(x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1)^{\mathsf{T}}$

$$\boldsymbol{\varphi}(\boldsymbol{x}_i)^T \boldsymbol{\varphi}(\boldsymbol{x}_j) = (x_{i,1}^2, x_{i,2}^2, \sqrt{2}x_{i,1}x_{i,2}, \sqrt{2}x_{i,1}, \sqrt{2}x_{i,2}, 1) \begin{pmatrix} x_{j,1}^2 \\ x_{j,2}^2 \\ \sqrt{2}x_{j,1}x_{j,2} \\ \sqrt{2}x_{j,1} \\ \sqrt{2}x_{j,2} \\ 1 \end{pmatrix}$$

d-dimensional→*m*-dimensional

No *K*(): O(*m*)

With *K*(): O(*d*)

Kernel saves computation!

$$= x_{i,1}^2 x_{j,1}^2 + x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2} + 1$$

$$= (x_{i,1}x_{j,1} + x_{i,2}x_{j,2} + 1)^2$$

$$= (\boldsymbol{x}_i{}^T \boldsymbol{x}_j + 1)^2 = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

GE: Kernelization

- **Extend to Nonlinear Cases**

Map the data to a higher dimensional space: $\mathbf{x}_i \rightarrow \boldsymbol{\varphi}(\mathbf{x}_i)$

A *kernel function* K: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\varphi}(\boldsymbol{x}_i)^T \boldsymbol{\varphi}(\boldsymbol{x}_j)$

$$\mathbf{w}^* = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} \Rightarrow \mathbf{w}^* = \underset{\mathbf{w}^T\boldsymbol{\Phi}\mathbf{B}\boldsymbol{\Phi}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\boldsymbol{\Phi}\mathbf{L}\boldsymbol{\Phi}^T\mathbf{w}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\varphi}(\mathbf{x}_1), \dots, \boldsymbol{\varphi}(\mathbf{x}_N)]$

Assume projection vector $\mathbf{w} = \sum_i \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) = \boldsymbol{\Phi}\boldsymbol{\alpha}$

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{B}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\alpha}=c}{\operatorname{argmin}} \boldsymbol{\alpha}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{L}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\alpha} \Leftarrow \boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}^T\mathbf{G}\mathbf{B}\mathbf{G}\boldsymbol{\alpha}=c}{\operatorname{argmin}} \boldsymbol{\alpha}^T\mathbf{G}\mathbf{L}\mathbf{G}\boldsymbol{\alpha}$$

GLG·$\vec{z}$ = λ GBG$\vec{z}$

What is the solution?

Since $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\varphi}(\boldsymbol{x}_1)^T\boldsymbol{\varphi}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\varphi}(\boldsymbol{x}_1)^T\boldsymbol{\varphi}(\boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ \boldsymbol{\varphi}(\boldsymbol{x}_N)^T\boldsymbol{\varphi}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\varphi}(\boldsymbol{x}_N)^T\boldsymbol{\varphi}(\boldsymbol{x}_N) \end{bmatrix}$

Kernel Gram matrix **G**

A New Algorithm using GE: Marginal Fisher Analysis (MFA)

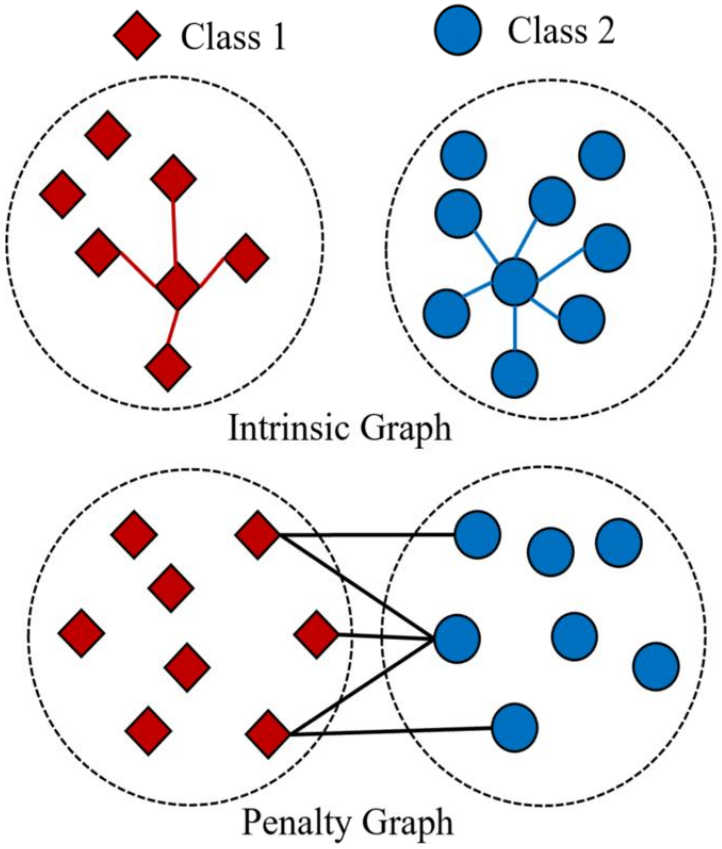
Shuicheng Yan, Dong Xu et al.: Graph Embedding and Extensions: A General Framework for Dimensionality Reduction. T-PAMI 2007.

New Algorithm: Marginal Fisher Analysis

- **Motivation**

  -Avoid certain limitations of LDA in terms of the data distribution assumption and available projection direction

- **Utilize GE**



✓ **Intrinsic Graph *G*: Intraclass compactness**

(Each sample is connected to its k_1–nearest neighbors of the same class)

$\mathbf{S}_{i,j} = \mathbf{S}_{j,i} = 1$: if for each sample $\mathbf{x}_i$, $\mathbf{x}_j$ is among the k_1–nearest neighbors of $\mathbf{x}_i$ in the same class

✓ **Penalty Graph G^p : Interclass separability**

(Marginal point pairs of different classes are connected)

$\mathbf{S}_{i,j}^{p} = 1$: if for each class *cl*, the pair $(\mathbf{x}_i, \mathbf{x}_j)$ is among the k_2–shortest pairs in the set $\{(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_i \in \mathbf{X}_{cl}, \mathbf{x}_j \notin \mathbf{X}_{cl}\}$

New Algorithm: Marginal Fisher Analysis

- **Intraclass Compactness following Linearization of GE**

$$\widehat{S_c} = \sum_{i \neq j} \left\| \mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j \right\|^2 \mathbf{S}_{i,j} = 2\mathbf{w}^T \mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{w}$$

$$= \sum_i \sum_{i \in N_{k_1}^+(j) \; or \; j \in N_{k_1}^+(i)} \left\| \mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j \right\|^2 = 2\mathbf{w}^T \mathbf{X} (\mathbf{D} - \mathbf{S}) \mathbf{X}^T \mathbf{w}$$

$N_{k_1}^+(i)$ is the index set of the k_1-nearest neighbors of the sample $\mathbf{x}_i$ in the same class

$$\mathbf{S}_{i,j} = \begin{cases} 1, \text{If } i \in N_{k_1}^+(j) \; or \; j \in N_{k_1}^+(i) \\ 0, \text{else} \end{cases}$$

- **Interclass Separability following Linearization of GE**

$$\widehat{S_p} = \sum_{i \neq j} \left\| \mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j \right\|^2 \mathbf{S}_{i,j}^p = 2\mathbf{w}^T \mathbf{X} \mathbf{L}^p \mathbf{X}^T \mathbf{w}$$

$$= \sum_i \sum_{(i,j) \in P_{k_2}(cl_i) \; or (i,j) \in P_{k_2}(cl_j)} \left\| \mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j \right\|^2 = 2\mathbf{w}^T \mathbf{X} (\mathbf{D}^p - \mathbf{S}^p) \mathbf{X}^T \mathbf{w}$$

$P_{k_2}(cl)$ is a set of data pairs that are the k_2 nearest pairs among the set $\{(i,j), i \in class\ cl, j \notin class\ cl\}$

$$\mathbf{S}_{i,j}^p = \begin{cases} 1, \text{ If } (i,j) \in P_{k_2}(cl_i) \; or (i,j) \in P_{k_2}(cl_j) \\ 0, \text{ else} \end{cases}$$

New Algorithm: Marginal Fisher Analysis

- **Marginal Fisher Criterion**

  Minimize Intraclass Compactness $\widehat{S_c}$, Maximize Interclass Separability $\widehat{S_p}$

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{\mathbf{w}^T\mathbf{X}(\mathbf{D}-\mathbf{S})\mathbf{X}^T\mathbf{w}}{\mathbf{w}^T\mathbf{X}(\mathbf{D}^p-\mathbf{S}^p)\mathbf{X}^T\mathbf{w}}$$

Which is a special linearization of GE with **B**= $\mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$

New Algorithm: Marginal Fisher Analysis

General Linearization:

$$\mathbf{w}^* = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}$$

Solve $\tilde{\mathbf{L}}\mathbf{w} = \lambda\tilde{\mathbf{B}}\mathbf{w}$,

where $\tilde{\mathbf{L}} = \mathbf{X}\mathbf{L}\mathbf{X}^T$ and $\tilde{\mathbf{B}} = \mathbf{X}\mathbf{B}\mathbf{X}^T$

MFA:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}}{\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w}}$$

$$\frac{d}{d\boldsymbol{w}}J(\boldsymbol{w}) = \mathbf{0}$$

$$\frac{(\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w})\frac{d}{d\boldsymbol{w}}(\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}) - (\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w})\frac{d}{d\boldsymbol{w}}(\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w})}{(\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w})^2} = \mathbf{0}$$

$$2(\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w})\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} - 2\left(\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}\right)\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w} = \mathbf{0}$$

Divide by $\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w}$

$$\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} - \frac{\left(\mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}\right)}{(\mathbf{w}^T\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w})}\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w} = \mathbf{0}$$

$$\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} - J(\boldsymbol{w})\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w} = \mathbf{0}$$

$$\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w} = J(\boldsymbol{w})\mathbf{X}\mathbf{L}^p\mathbf{X}^T\mathbf{w}$$

New Algorithm: Marginal Fisher Analysis

- **Marginal Fisher Criterion**

  Minimize Intraclass Compactness $\widehat{S_c}$, Maximize Interclass Separability $\widehat{S_p}$

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{\mathbf{w}^T\mathbf{X}(\mathbf{D}-\mathbf{S})\mathbf{X}^T\mathbf{w}}{\mathbf{w}^T\mathbf{X}(\mathbf{D}^p-\mathbf{S}^p)\mathbf{X}^T\mathbf{w}}$$

Which is a special linearization of GE with $\mathbf{B}= \mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$

General Linearization:

$$\mathbf{w}^* = \underset{\mathbf{w}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{w}=c}{\operatorname{argmin}} \mathbf{w}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{w}$$

If $\mathbf{B}= \mathbf{L}^p = \mathbf{D}^p - \mathbf{S}^p$, the same as marginal fisher criterion!

Experiments: Face Recognition

Sample Images from CMU PIE database (after cropping)

	G4/P16	G5/P15	G6/P14
PCA	43.4% (252)	49.2% (264)	49.2% (341)
Fisherface	76.9% (62)	83.6% (61)	88.9% (62)
MFA	82.5% (70)	87.3% (117)	90.5% (69)
PCA+LDA	78.8% (94, 44)	84.3% (93, 36)	91.2% (91, 20)
PCA+MFA	83.9% (99, 71)	87.5% (99, 104)	90.8% (91, 32)

G*m*/P*n*: *m* images per person are randomly selected for training and the remaining *n* images are used for testing.

Papers to Read and Study

- Shuicheng Yan, Dong Xu et al.: Graph Embedding and Extensions: A General Framework for Dimensionality Reduction. T-PAMI 2007.

- Jieping Ye, Ravi Janardan, Cheonghee Park, and Haesun Park. An optimization criterion for generalized discriminant analysis on undersampled problems. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 26, No. 8, pp. 982—994, 2004.

- A. Martinez, A. Kak, "PCA versus LDA", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 2, pp. 228-233, 2001.