LEC 3: Linear Discriminant Analysis (LDA)

— A Supervised Dimensionality Reduction Approach

Guangliang Chen

September 24, 2018

Outline

- Motivation:
  - PCA is unsupervised which does not use training labels
  - Variance is not always useful for classification
- LDA: a supervised dimensionality reduction approach
  - 2-class LDA
  - Multiclass extension
- Comparison between PCA and LDA

Acknowledgment

The first few slides of this presentation are based on Prof. Olga Veksler's slides at

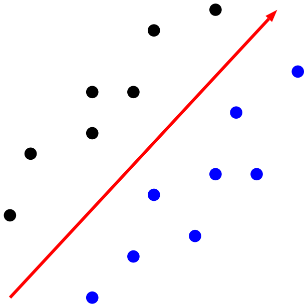
http://www.csd.uwo.ca/~olga/Courses/CS434a_541a/Lecture8.pdf

Data representation vs data classification

PCA finds the most accurate data representation in a lower dimensional space spanned by the maximum-variance directions.

However, such directions may not work well for classification (see right plot).

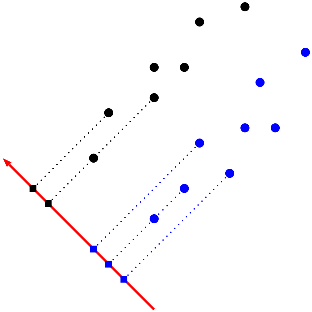
Thus, in the classification setting, we need a new projection method that is based on the discriminatory information between the different classes.



Representative but not discriminative

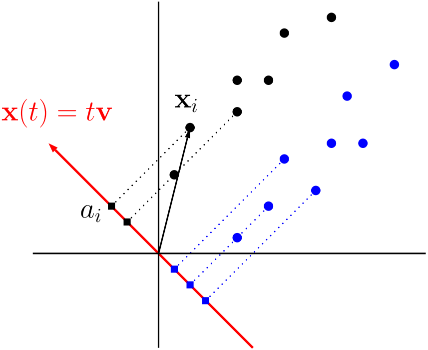
The two-class LDA problem

Given a training data set $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ consisting of two classes C_1, C_2, find a direction that "best" discriminates between the two classes.



Mathematical setup

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



First, observe that projections of the two classes onto parallel lines always have the same amount of separation.

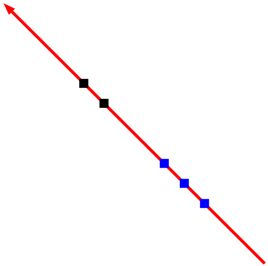
But this time we are going to focus on the lines that pass through the origin.

The 1D projections of the points are

$$a_i = \mathbf{v}^T \mathbf{x}_i, \quad i = 1, \ldots, n$$

Note that they also carry the labels of the original data.

Now the data look like this:



How do we quantify the separation between the two classes (in order to compare different directions $\mathbf{v}$ and select the best one)?

One (naive) idea is to measure the distance between the two class means in the 1D projection space: $|\mu_1 - \mu_2|$, where

$$\mu_1 = \frac{1}{n_1} \sum_{\mathbf{x}_i \in C_1} a_i = \frac{1}{n_1} \sum_{\mathbf{x}_i \in C_1} \mathbf{v}^T \mathbf{x}_i$$
$$= \mathbf{v}^T \cdot \frac{1}{n_1} \sum_{\mathbf{x}_i \in C_1} \mathbf{x}_i = \mathbf{v}^T \mathbf{m}_1$$

and similarly,

$$\mu_2 = \mathbf{v}^T \mathbf{m}_2, \quad \mathbf{m}_2 = \frac{1}{n_2} \sum_{\mathbf{x}_i \in C_2} \mathbf{x}_i.$$

That is, we solve the following problem

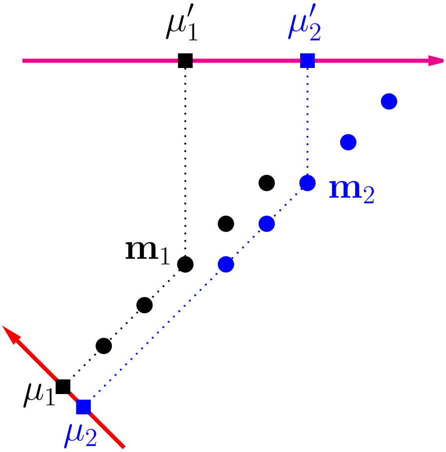
$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} |\mu_1 - \mu_2|$$

where

$$\mu_j = \mathbf{v}^T \mathbf{m}_j, \; j = 1, 2.$$

However, this criterion does not always work (as shown in the right plot).

What else do we need to control?



We should also consider the variances of the projected classes:

$$s_1^2 = \sum_{\mathbf{x}_i \in C_1} (a_i - \mu_1)^2, \quad s_2^2 = \sum_{\mathbf{x}_i \in C_2} (a_i - \mu_2)^2$$

Ideally, the projected classes have both faraway means and small variances.

This can be achieved through the following modified formulation:

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \frac{(\mu_1 - \mu_2)^2}{s_1^2 + s_2^2}.$$

where

$$\mu_1 = \mathbf{v}^T \mathbf{m}_1, \quad \mu_2 = \mathbf{v}^T \mathbf{m}_2.$$

Mathematical derivation

First, we can rewrite the distance between the two centroids as follows:

$$
\begin{aligned}
(\mu_1 - \mu_2)^2 &= (\mathbf{v}^T\mathbf{m}_1 - \mathbf{v}^T\mathbf{m}_2)^2 = (\mathbf{v}^T(\mathbf{m}_1 - \mathbf{m}_2))^2 \\
&= \mathbf{v}^T(\mathbf{m}_1 - \mathbf{m}_2) \cdot (\mathbf{m}_1 - \mathbf{m}_2)^T\mathbf{v} \\
&= \mathbf{v}^T\mathbf{S}_b\mathbf{v},
\end{aligned}
$$

where

$$
\mathbf{S}_b = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T \in \mathbb{R}^{d\times d}
$$

is called the **between-class scatter matrix**.

Remark. Clearly, $\mathbf{S}_b$ is square, symmetric and positive semidefinite. Moreover, $\mathrm{rank}(\mathbf{S}_b) = 1$, which implies that it only has 1 positive eigenvalue!

Next, for each class $j = 1, 2$, the variance of the projection (onto $\mathbf{v}$) is

$$\begin{aligned} s_j^2 &= \sum_{\mathbf{x}_i \in C_j} (a_i - \mu_j)^2 = \sum_{\mathbf{x}_i \in C_j} (\mathbf{v}^T \mathbf{x}_i - \mathbf{v}^T \mathbf{m}_j)^2 \\ &= \sum_{\mathbf{x}_i \in C_j} \mathbf{v}^T (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \mathbf{v} \\ &= \mathbf{v}^T \left[\sum_{\mathbf{x}_i \in C_j} (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \right] \mathbf{v} \\ &= \mathbf{v}^T \mathbf{S}_j \mathbf{v}, \end{aligned}$$

where

$$\mathbf{S}_j = \sum_{\mathbf{x}_i \in C_j} (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \in \mathbb{R}^{d \times d}$$

is called the **within-class scatter matrix** for class j.

The total within-class scatter of the two classes in the projection space is

$$s_1^2 + s_2^2 = \mathbf{v}^T\mathbf{S}_1\mathbf{v} + \mathbf{v}^T\mathbf{S}_2\mathbf{v} = \mathbf{v}^T(\mathbf{S}_1 + \mathbf{S}_2)\mathbf{v} = \mathbf{v}^T\mathbf{S}_w\mathbf{v}$$

where

$$\mathbf{S}_w = \mathbf{S}_1 + \mathbf{S}_2 = \sum_{\mathbf{x}_i \in C_1} (\mathbf{x}_i - \mathbf{m}_1)(\mathbf{x}_i - \mathbf{m}_1)^T + \sum_{\mathbf{x}_i \in C_2} (\mathbf{x}_i - \mathbf{m}_2)(\mathbf{x}_i - \mathbf{m}_2)^T$$

is called the **total within-class scatter matrix** of the (original) training data.

Remark. $\mathbf{S} \in \mathbb{R}^{d\times d}$ is also square, symmetric, and positive semidefinite.

Putting everything together, we have arrived at the following optimization problem:

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \frac{\mathbf{v}^T\mathbf{S}_b\mathbf{v}}{\mathbf{v}^T\mathbf{S}_w\mathbf{v}} \quad \longleftarrow \text{Where did we see this?}$$

Result

Theorem 0.1. *Suppose* $\mathbf{S}_w$ *is nonsingular. The maximizer of the problem is given by the largest eigenvector* $\mathbf{v}_1$ *of* $\mathbf{S}_w^{-1}\mathbf{S}_b$, *i.e.,* $\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v}_1 = \lambda_1\mathbf{v}_1$.

Proof. Left as homework. □

Remark.

- λ_1 is the maximal amount of separation between the two classes along any single direction.
- $\mathrm{rank}(\mathbf{S}_w^{-1}\mathbf{S}_b) = \mathrm{rank}(\mathbf{S}_b) = 1$, so λ_1 is the only nonzero (positive) eigenvalue that can be found.

Computing

The following are different ways of finding the optimal direction $\mathbf{v}_1$:

- Slowest way (via three expensive steps):
  1. First, work really hard to invert the $d \times d$ matrix $\mathbf{S}_w$,
  2. then do the matrix multiplication $\mathbf{S}_w^{-1}\mathbf{S}_b$,
  3. and finally solve the eigenvalue problem $\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v}_1 = \lambda_1\mathbf{v}_1$.
- A slight better way: Rewrite as a **generalized eigenvalue problem**

$$\mathbf{S}_b\mathbf{v}_1 = \lambda_1\mathbf{S}_w\mathbf{v}_1,$$

  and then solve it through functions like *eigs(A,B)* in MATLAB.

- The smartest way is to rewrite as

$$\lambda_1 \mathbf{v}_1 = \mathbf{S}_w^{-1} \underbrace{(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T}_{\mathbf{S}_b} \mathbf{v}_1$$
$$= \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \cdot \underbrace{(\mathbf{m}_1 - \mathbf{m}_2)^T \mathbf{v}_1}_{\text{scalar}}$$

This implies that

$$\mathbf{v}_1 \propto \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$$

and it can be computed from $\mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$ through rescaling!

Remark. Here, inverting $\mathbf{S}_w$ should still be avoided; instead, one should implement this by solving a linear system $\mathbf{S}_w \mathbf{x} = \mathbf{m}_1 - \mathbf{m}_2$. This can be done through $\mathbf{S}_w \setminus (\mathbf{m}_1 - \mathbf{m}_2)$ in MATLAB.

Two-class LDA: summary

The optimal discriminatory direction is

$$\mathbf{v}^* = \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \quad \text{(plus normalization)}$$

It is the solution of

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \frac{\mathbf{v}^T\mathbf{S}_b\mathbf{v}}{\mathbf{v}^T\mathbf{S}_w\mathbf{v}} \quad \longleftarrow \quad \frac{(\mu_1-\mu_2)^2}{s_1^2+s_2^2}$$

where

$$\mathbf{S}_b = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$$
$$\mathbf{S}_w = \mathbf{S}_1 + \mathbf{S}_2, \quad \mathbf{S}_j = \sum_{\mathbf{x}\in C_j}(\mathbf{x}-\mathbf{m}_j)(\mathbf{x}-\mathbf{m}_j)^T$$

A small example

Data

- Class 1 has three points (1,2), (2,3), (3, 4.9), with mean $\mathbf{m}_1 = (2, 3.3)^T$
- Class 2 has three points (2,1), (3,2), (4, 3.9), with mean $\mathbf{m}_2 = (3, 2.3)^T$

Within-class scatter matrix

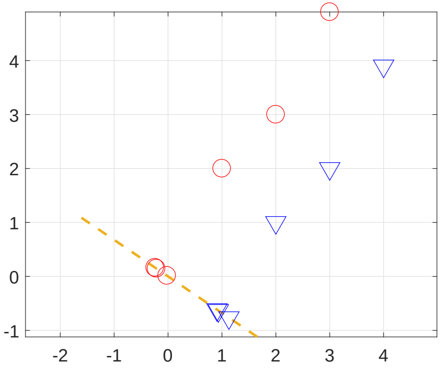
$$\mathbf{S}_w = \begin{pmatrix} 4 & 5.8 \\ 5.8 & 8.68 \end{pmatrix}$$

Thus, the optimal direction is

$$\mathbf{v} = \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2) = (-13.4074, 9.0741)^T \xrightarrow{\text{normalizing}} (-0.8282, 0.5605)^T$$

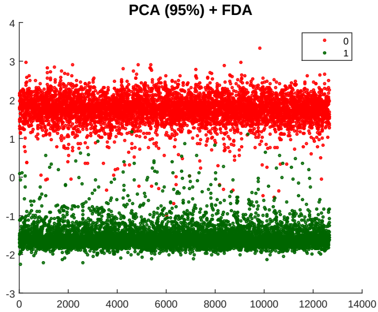
and the projection coordinates are

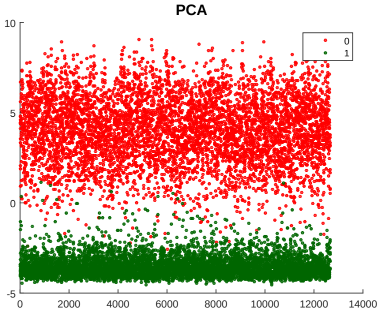
$$Y = [0.2928, 0.0252, 0.2619, -1.0958, -1.3635, -1.1267]$$



Experiment (2 digits)

MNIST handwritten digits 0 and 1 (left: LDA, right: PCA)



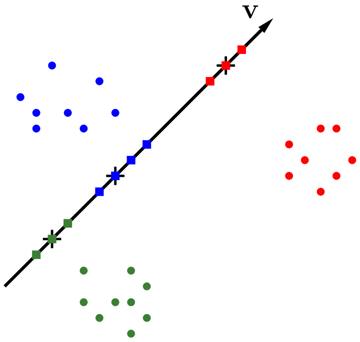


Multiclass extension

The previous procedure only applies to 2 classes. When there are $c \geq 3$ classes, what is the "most discriminatory" direction?

It will be based on the same intuition that the optimal direction $\mathbf{v}$ should project the different classes such that

- each class is as tight as possible;
- their centroids are as far from each other as possible.



Both are actually about variances.

Mathematical derivation

For any unit vector $\mathbf{v}$, the tightness of the projected classes (of the training data) is still described by the total within-class scatter:

$$\sum_{j=1}^{c} s_j^2 = \sum \mathbf{v}^T \mathbf{S}_j \mathbf{v} = \mathbf{v}^T \left(\sum \mathbf{S}_j \right) \mathbf{v} = \mathbf{v}^T \mathbf{S}_w \mathbf{v}$$

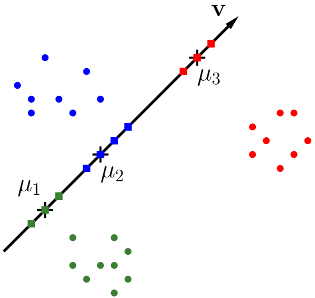
where the $\mathbf{S}_j, 1 \leq j \leq c$ are defined in the same way as before:

$$\mathbf{S}_j = \sum_{\mathbf{x} \in C_j} (\mathbf{x} - \mathbf{m}_j)(\mathbf{x} - \mathbf{m}_j)^T$$

and $\mathbf{S}_w = \sum \mathbf{S}_j$ is the total within-class scatter matrix.

To make the class centroids μ_j (in the projection space) as far from each other as possible, we can just maximize the variance of the centroids set $\{\mu_1, \ldots, \mu_k\}$:

$$\sum_{j=1}^{c} (\mu_j - \bar{\mu})^2 = \frac{1}{c} \sum_{j<\ell} (\mu_j - \mu_\ell)^2, \qquad \text{where} \quad \bar{\mu} = \frac{1}{c} \sum_{j=1}^{c} \mu_j \longleftarrow \text{simple average}.$$



We actually use a weighted mean of the projected centroids to define the between-class scatter:

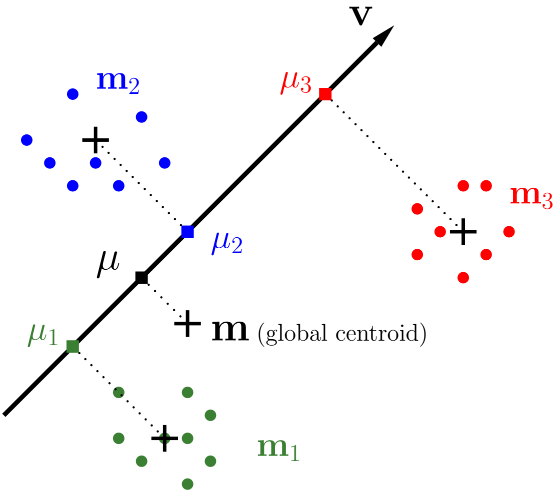
$$\sum_{j=1}^{c} n_j(\mu_j - \mu)^2, \qquad \text{where} \quad \mu = \frac{1}{n}\sum_{j=1}^{c} n_j \mu_j \longleftarrow \text{weighted average}$$

because the weighted mean (μ) is the projection of the global centroid ($\mathbf{m}$) of the training data onto $\mathbf{v}$:

$$\mathbf{v}^T\mathbf{m} = \mathbf{v}^T\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\right) = \mathbf{v}^T\left(\frac{1}{n}\sum_{j=1}^{c} n_j\mathbf{m}_j\right) = \frac{1}{n}\sum_{j=1}^{c} n_j\mu_j = \mu.$$

In contrast, the simple mean does not have such a geometric interpretation:

$$\bar{\mu} = \frac{1}{c}\sum_{j=1}^{c}\mu_j = \frac{1}{c}\sum_{j=1}^{c}\mathbf{v}^T\mathbf{m}_j = \mathbf{v}^T\left(\frac{1}{c}\sum_{j=1}^{c}\mathbf{m}_j\right)$$


$\mathbf{v}$
$\mathbf{m}_2$
μ_3
$\mathbf{m}_3$
μ_2
μ
μ_1
$\mathbf{m}$ (global centroid)
$\mathbf{m}_1$

We simplify the between-class scatter (in the $\mathbf{v}$ space) as follows:

$$\begin{aligned}
\sum_{j=1}^{c} n_j(\mu_j - \mu)^2 &= \sum n_j (\mathbf{v}^T(\mathbf{m}_j - \mathbf{m}))^2 \\
&= \sum n_j \, \mathbf{v}^T(\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T \mathbf{v} \\
&= \mathbf{v}^T \left(\sum n_j (\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T \right) \mathbf{v} \\
&= \mathbf{v}^T \mathbf{S}_b \mathbf{v}.
\end{aligned}$$

We have thus arrived at the same kind of problem

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \frac{\mathbf{v}^T \mathbf{S}_b \mathbf{v}}{\mathbf{v}^T \mathbf{S}_w \mathbf{v}} \quad \longleftarrow \quad \frac{\sum n_j (\mu_j - \mu)^2}{\sum s_j^2}$$

Remark. When $c = 2$, it can be verified that

$$\sum_{j=1}^{2} n_j(\mu_j - \mu)^2 = \frac{n_1 n_2}{n}(\mu_1 - \mu_2)^2, \qquad \text{where} \quad \mu = \frac{1}{n}(n_1\mu_1 + n_2\mu_2)$$

and

$$\sum_{j=1}^{2} n_j(\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T = \frac{n_1 n_2}{n}(\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T, \ \mathbf{m} = \frac{1}{n}(n_1\mathbf{m}_1 + n_2\mathbf{m}_2)$$

This shows that when there are only two classes, the weighted definitions are just a scalar multiple of the unweighted definitions.

Therefore, the multiclass LDA $\frac{\sum n_j(\mu_j - \mu)^2}{\sum s_j^2}$ is a natural generalization of the two-class LDA $\frac{(\mu_1 - \mu_2)^2}{s_1^2 + s_2^2}$.

Computing

The solution is given by the largest eigenvector of $\mathbf{S}_w^{-1}\mathbf{S}_b$ (when $\mathbf{S}_w$ is nonsingular):

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v}_1 = \lambda_1\mathbf{v}_1.$$

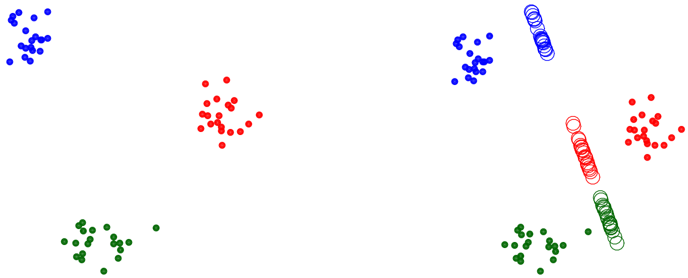
However, the formula $\mathbf{v}_1 \propto \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$ is no longer valid:

$$\lambda_1\mathbf{v}_1 = \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v}_1 = \mathbf{S}_w^{-1}\sum_j n_j(\mathbf{m}_j - \mathbf{m})\underbrace{(\mathbf{m}_j - \mathbf{m})^T\mathbf{v}_1}_{\text{scalar}}$$

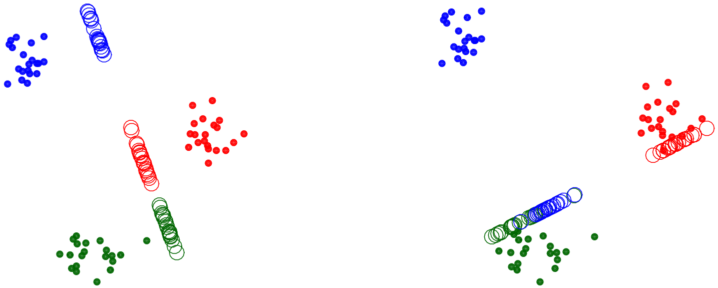
So we have to find $\mathbf{v}_1$ by solving a generalized eigenvalue problem:

$$\mathbf{S}_b\mathbf{v}_1 = \lambda_1\mathbf{S}_w\mathbf{v}_1.$$

Simulation



What about the second eigenvector $\mathbf{v}_2$?



How many discriminatory directions can we find?

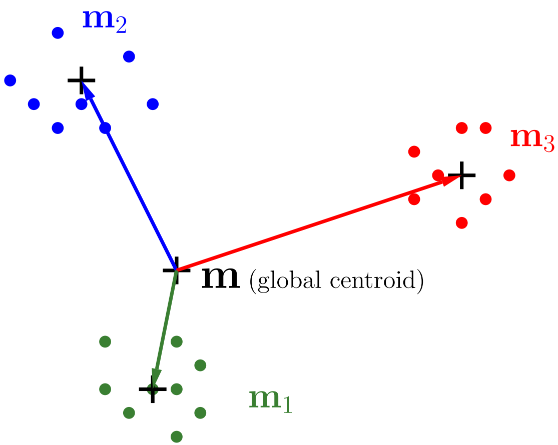
To answer this question, we just need to count the number of nonzero eigenvalues

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{v} = \lambda\mathbf{v}$$

since only the nonzero eigenvectors will be used as the discriminatory directions.

In the above equation, the within-class scatter matrix $\mathbf{S}_w$ is *assumed to be* nonsingular. However, the between-class scatter matrix $\mathbf{S}_b$ is of low rank:

$$\begin{aligned}\mathbf{S}_b &= \sum n_i(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \\ &= \left[\sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) \cdots \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})\right] \cdot \begin{bmatrix}\sqrt{n_1}(\mathbf{m}_1 - \mathbf{m})^T \\ \vdots \\ \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})^T\end{bmatrix}\end{aligned}$$


$\mathbf{m}_2$
$\mathbf{m}_3$
$\mathbf{m}$ (global centroid)
$\mathbf{m}_1$

Observe that the columns of the matrix

$$[\sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) \cdots \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})]$$

are linearly dependent:

$$\begin{aligned} &\sqrt{n_1} \cdot \sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) + \cdots + \sqrt{n_c} \cdot \sqrt{n_c}(\mathbf{m}_c - \mathbf{m}) \\ &= (n_1\mathbf{m}_1 + \cdots n_c\mathbf{m}_c) - (n_1 + \cdots + n_c)\mathbf{m} \\ &= n\mathbf{m} - n\mathbf{m} \\ &= \mathbf{0}. \end{aligned}$$

The shows that $\mathrm{rank}(\mathbf{S}_b) \leq c - 1$ (where c is the number of training classes).

Therefore, one can only find at most $c - 1$ discriminatory directions.

Multiclass LDA algorithm

Input: Training data $\mathbf{X} \in \mathbb{R}^{n\times d}$ (with c classes)

Output: At most $c-1$ discriminatory directions and projections of $\mathbf{X}$ onto them

1. Compute

$$\mathbf{S}_w = \sum_{j=1}^{c} \sum_{\mathbf{x}\in C_j} (\mathbf{x}-\mathbf{m}_j)(\mathbf{x}-\mathbf{m}_j)^T, \quad \mathbf{S}_b = \sum_{j=1}^{c} n_j(\mathbf{m}_j-\mathbf{m})(\mathbf{m}_j-\mathbf{m})^T.$$

2. Solve the generalized eigenvalue problem $\mathbf{S}_b\mathbf{v} = \lambda\mathbf{S}_w\mathbf{v}$ to find all nonzero eigenvectors $\mathbf{v}_1,\ldots,\mathbf{v}_k$ (for some $k \le c-1$)
3. Project the data $\mathbf{X}$ onto them $\mathbf{Y} = \mathbf{X}\cdot[\mathbf{v}_1\ldots\mathbf{v}_k] \in \mathbb{R}^{n\times k}$.

The singularity issue of $\mathbf{S}_w$

So far, we have assumed that the total within-class scatter matrix

$$\mathbf{S}_w = \sum_{j=1}^{c} \mathbf{S}_j, \qquad \text{where} \quad \mathbf{S}_j = \sum_{\mathbf{x}_i \in C_j} (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T$$

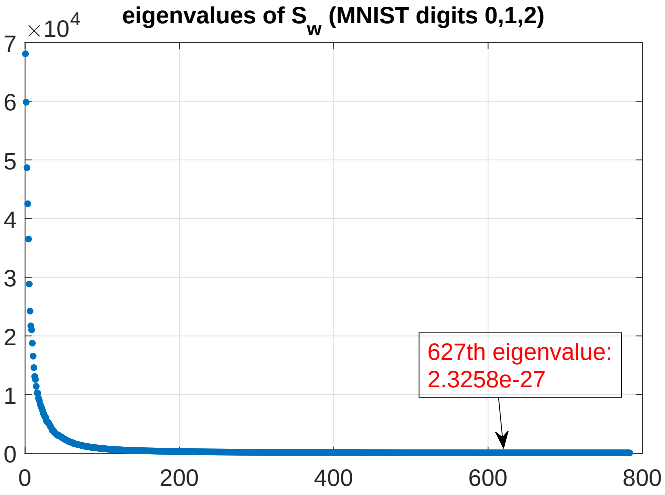
is nonsingular, so that we can solve the LDA problem

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \frac{\mathbf{v}^T \mathbf{S}_b \mathbf{v}}{\mathbf{v}^T \mathbf{S}_w \mathbf{v}}$$

as an eigenvalue problem

$$\mathbf{S}_w^{-1} \mathbf{S}_b \mathbf{v} = \lambda \mathbf{v}.$$

However, in many cases (especially when having high dimensional data), the matrix $\mathbf{S}_w \in \mathbb{R}^{d\times d}$ is singular (or very close to being singular).


eigenvalues of S_w (MNIST digits 0,1,2)
×10^4
7
6
5
4
3
2
1
0
0
200
400
600
800
627th eigenvalue:
2.3258e-27

How does this happen?

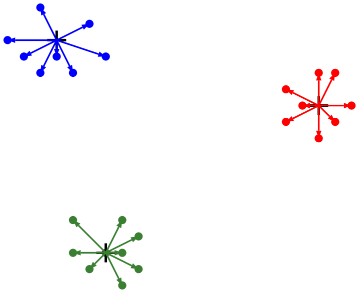
Let $\widetilde{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{m}_j$ for each $i = 1, 2 \ldots, n$ be the centered data points **using its own class centroid**.

Define

$$\widetilde{\mathbf{X}} = [\widetilde{\mathbf{x}}_1 \ldots \widetilde{\mathbf{x}}_n]^T \in \mathbb{R}^{n \times d}.$$

Then

$$\mathbf{S}_w = \widetilde{\mathbf{X}}^T \widetilde{\mathbf{X}} \in \mathbb{R}^{d \times d}.$$



Important issue: For high dimensional data (i.e., d is large), the centered data often do not fully span all d dimensions, thus making $\mathrm{rank}(\mathbf{S}_w) = \mathrm{rank}(\widetilde{\mathbf{X}}) < d$ (which implies that $\mathbf{S}_w$ is singular).

Common fixes:

- **Apply global PCA** to reduce the dimensionality of training data

$$\mathbf{Y}_{\text{train}} = \left(\mathbf{X}_{\text{train}} - [\mathbf{m}_{\text{train}} \ldots \mathbf{m}_{\text{train}}]^T\right) \cdot \mathbf{V}_{\text{train}}$$

  and then perform LDA on the reduced data:

$$\mathbf{Z}_{\text{train}} = \mathbf{Y}_{\text{train}} \cdot \mathbf{V}_{\text{lda}} \longleftarrow \text{learned from } \mathbf{Y}_{\text{train}}$$

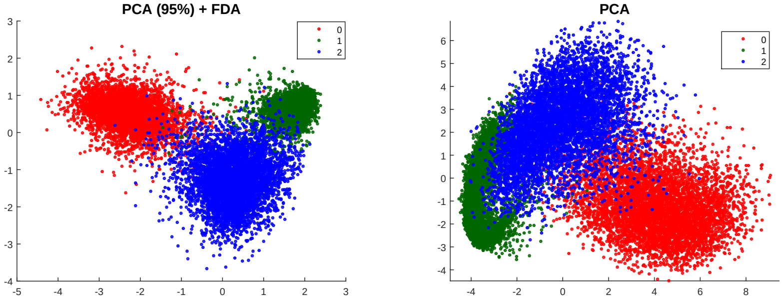
- **Regularize** $\mathbf{S}_w$:

$$\mathbf{S}_w' = \mathbf{S}_w + \beta \mathbf{I}_d = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T + \beta \mathbf{I}_d = \mathbf{Q}(\mathbf{\Lambda} + \beta \mathbf{I}_d)\mathbf{Q}^T$$

  where $\mathbf{\Lambda} + \beta \mathbf{I}_d = \operatorname{diag}(\lambda_1 + \beta, \ldots, \lambda_d + \beta)$.

Experiment (3 digits)

MNIST handwritten digits 0, 1, and 2



LDA for classification

First, we can extend LDA (plus PCA beforehand) to the test data as follows:

$$\text{PCA} \longrightarrow \quad \mathbf{Y}_{\text{test}} = \left(\mathbf{X}_{\text{test}} - [\mathbf{m}_{\text{train}} \ldots \mathbf{m}_{\text{train}}]^T\right) \cdot \mathbf{V}_{\text{train}}$$

$$\text{LDA} \longrightarrow \quad \mathbf{Z}_{\text{test}} = \mathbf{Y}_{\text{test}} \cdot \mathbf{V}_{\text{lda}}$$

Next, just select a classifier to work in the reduced space:

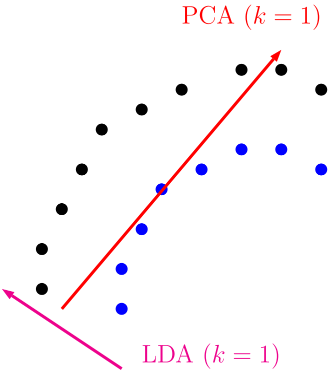
- (PCA +) LDA + kNN
- (PCA +) LDA + nearest local centroid
- (PCA +) LDA + other classifiers

Comparison between PCA and LDA

	PCA	LDA
Use labels?	no (unsupervised)	yes (supervised)
Criterion	variance	discrimination
#dimensions (k)	any	$\leq c-1$
Computing	SVD	generalized eigenvectors
Linear projection?	yes $((\mathbf{x}-\mathbf{m})^T\mathbf{V})$	yes $(\mathbf{x}^T\mathbf{V})$
Nonlinear boundary	can handle*	cannot handle

*In the case of nonlinear separation between the classes, PCA often works better than LDA as the latter can only find at most $c-1$ directions (which are insufficient to preserve all the discriminatory information in the training data).

- LDA with $k=1$: does not work well
- PCA with $k=1$: does not work well
- PCA with $k=2$: preserves all the nonlinear separation which can be handled by nonlinear classifiers.



HW3 (Due: Saturday noon, Oct. 13)

1 Apply each of PCA and LDA with $k = 2$ directly to the *iris* data and plot the two dimensional projections.

2 Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a positive semidefinite matrix whose spectral decomposition is $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$. Define its square root as

$$\mathbf{A}^{\frac{1}{2}} = \mathbf{Q}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{Q}^T$$

where $\mathbf{\Lambda}^{\frac{1}{2}}$ is defined in HW0. When $\mathbf{A} \in \mathbb{R}^{n\times n}$ is strictly positive definite (i.e., all its eigenvalues are positive), we can also define

$$\mathbf{A}^{-\frac{1}{2}} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T$$

Use such definitions to do the following.

- Find $\mathbf{A}^{\frac{1}{2}}$ and $\mathbf{A}^{-\frac{1}{2}}$ when $\mathbf{A} = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$.

- Prove the theorem on Slide 13 of this presentation (about the solution of the LDA problem).

3 Apply PCA 95% + LDA to the following subsets of digits in the USPS training set: (a) 0, 1 (b) 4, 9 (c) 1, 2, 3 (d) 3, 5, 8 and plot the projected data in each case.

4 First apply PCA 95% + LDA to all 10 classes of the USPS digits data and then apply the plain kNN classifier to the reduced data with $k = 1, \ldots, 10$ and display the test errors curve. Compare with that of PCA 95% + kNN (for each k). What is your conclusion?

5 Repeat Question 4 with nearest local centroid instead of kNN (everything else being the same).