Outline

- LDA objective
- Recall … PCA
- Now … LDA
- LDA … Two Classes
  - Counter example
- LDA … C Classes
  - Illustrative Example
- LDA vs PCA Example
- Limitations of LDA

LDA Objective

- The objective of LDA is to perform dimensionality reduction …
  - So what, PCA does this ☹…
- However, we want to preserve as much of the class discriminatory information as possible.
  - OK, that's new, let dwell deeper ☺ …

Recall … PCA

- **In PCA, the main idea to re-express the available dataset to extract the relevant information by reducing the redundancy and minimize the noise.**

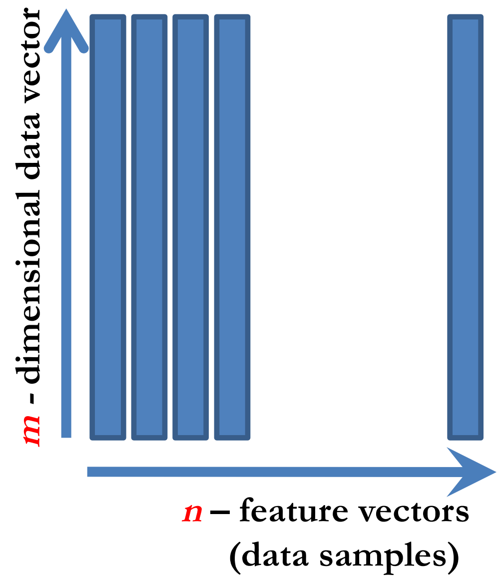
- We didn't care about whether this dataset represent features from one or more classes, i.e. the discrimination power was not taken into consideration while we were talking about PCA.

- In PCA, we had a dataset matrix **X** with dimensions *mxn*, where columns represent different data samples.

- We first started by subtracting the mean to have a zero mean dataset, then we computed the covariance matrix $\mathbf{S_x} = \mathbf{XX^T}$.

- Eigen values and eigen vectors were then computed for $\mathbf{S_x}$. Hence the new basis vectors are those eigen vectors with highest eigen values, where the number of those vectors was our choice.

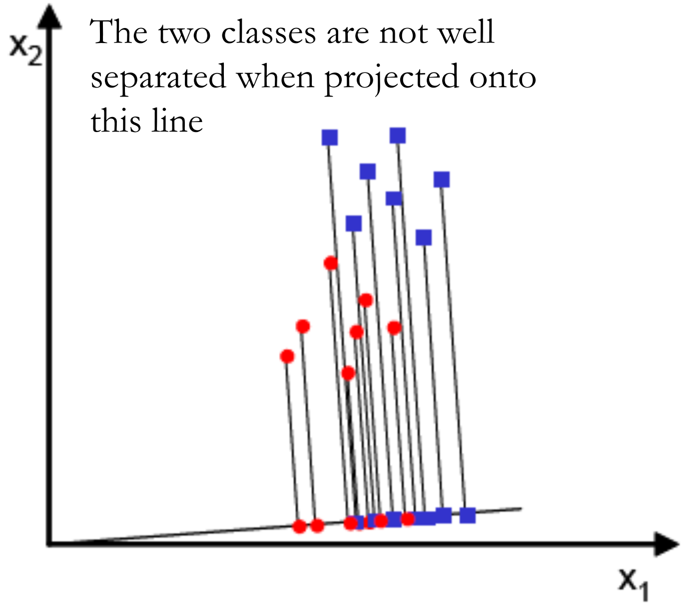
- Thus, using the new basis, we can project the dataset onto a less dimensional space with more powerful data representation.

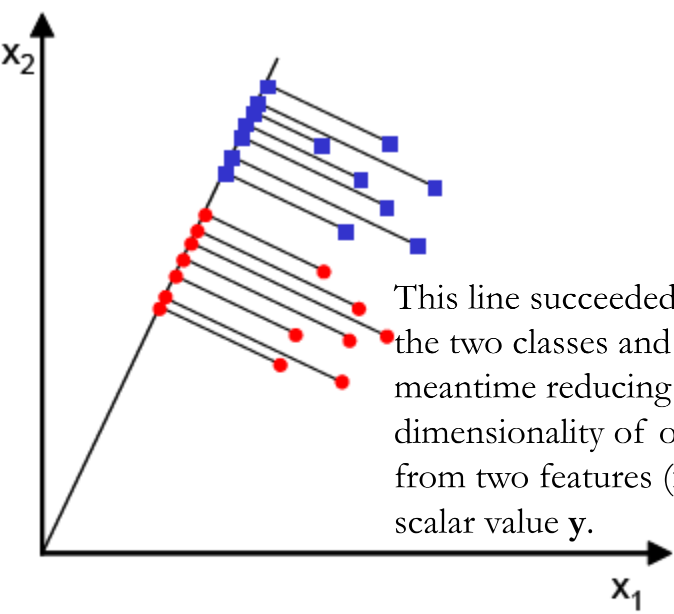


Now … LDA

- Consider a pattern classification problem, where we have C-classes, e.g. seabass, tuna, salmon …
- Each class has $\mathbf{N_i}$ m-dimensional samples, where $i = 1,2, \ldots, C$.
- Hence we have a set of m-dimensional samples $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{Ni}\}$ belong to class $\boldsymbol{\omega}_\mathbf{i}$.
- Stacking these samples from different classes into one big fat matrix **X** such that each column represents one sample.
- **We seek to obtain a transformation of X to Y through projecting the samples in X onto a hyperplane with dimension *C-1*.**
- **Let's see what does this mean?**

LDA … Two Classes





- Assume we have *m*-dimensional samples $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N\}$, N_1 of which belong to $\boldsymbol{\omega}_1$ and N_2 belong to $\boldsymbol{\omega}_2$.

- We seek to obtain a scalar **y** by projecting the samples **x** onto a line (C-1 space, C = 2).

$$y = w^T x \quad \textit{where} \quad x = \begin{bmatrix} x_1 \\ . \\ . \\ x_m \end{bmatrix} \quad \textit{and} \quad w = \begin{bmatrix} w_1 \\ . \\ . \\ w_m \end{bmatrix}$$

  – where **w** is the projection vectors used to project **x** to **y**.

- **Of all the possible lines we would like to select the one that maximizes the separability of the scalars.**

LDA … Two Classes

- In order to find a good projection vector, we need to <u>define a measure of separation between the projections</u>.

- The mean vector of each class in **x** and **y** feature space is:

$$\mu_i = \frac{1}{N_i}\sum_{x\in\omega_i} x \quad and \quad \tilde{\mu}_i = \frac{1}{N_i}\sum_{y\in\omega_i} y = \frac{1}{N_i}\sum_{x\in\omega_i} w^T x$$

$$= w^T \frac{1}{N_i}\sum_{x\in\omega_i} x = w^T \mu_i$$

  – i.e. projecting **x** to **y** will lead to projecting the mean of **x** to the mean of **y.**

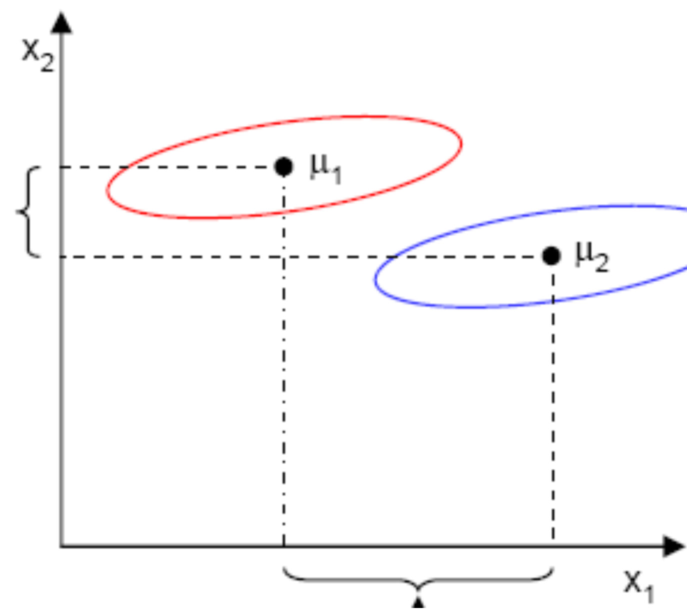
- We could then choose the <u>distance between the projected means</u> as our objective function

$$J(w) = \left|\tilde{\mu}_1 - \tilde{\mu}_2\right| = \left|w^T \mu_1 - w^T \mu_2\right| = \left|w^T\left(\mu_1 - \mu_2\right)\right|$$

LDA … Two Classes

- However, the distance between the projected means is <u>not a very good measure</u> since it does not take into account the standard deviation within the classes.

This axis yields better class separability



This axis has a larger distance between means

LDA … Two Classes

- The solution proposed by Fisher is to maximize a function that represents the difference between the means, normalized by a measure of the within-class variability, or the so-called *scatter*.

- For each class we define the **scatter**, an equivalent of the variance, as; (sum of square differences between the projected samples and their class mean).

$$\tilde{s}_i^2 = \sum_{y \in \omega_i} \left(y - \tilde{\mu}_i \right)^2$$

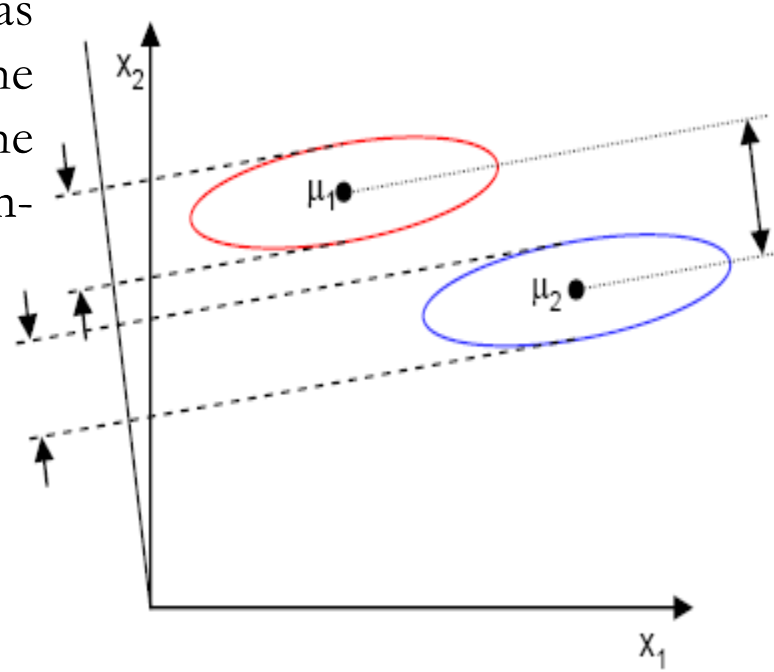
- $\tilde{s}_i^2$ measures the variability within class $\boldsymbol{\omega_i}$ after projecting it on the y-space.

- Thus $\tilde{s}_1^2 + \tilde{s}_2^2$ measures the variability within the two classes at hand after projection, hence it is called *within-class scatter* of the projected samples.

LDA … Two Classes

- The Fisher linear discriminant is defined as the linear function $\mathbf{w}^\mathbf{T}\mathbf{x}$ that maximizes the criterion function: (the distance between the projected means normalized by the within-class scatter of the projected samples.

$$J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$



- Therefore, we will be looking for a projection where examples from the same class are projected very close to each other and, at the same time, the projected means are as farther apart as possible

LDA … Two Classes

- In order to find the optimum projection w^*, we need to express $J(w)$ as an explicit function of w.

- We will define a measure of the scatter in multivariate feature space **x** which are denoted as <u>*scatter matrices*</u>;

$$J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

$$S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T$$

$$S_w = S_1 + S_2$$

- Where $\mathbf{S_i}$ is the <u>covariance matrix</u> of class $\boldsymbol{\omega_i}$, and $\mathbf{S_w}$ is called the <u>*within-class scatter matrix*</u>.

LDA … Two Classes

- Now, the scatter of the projection **y** can then be expressed as a function of the scatter matrix in feature space **x**.

$$J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

$$\tilde{s}_i^2 = \sum_{y \in \omega_i} \left(y - \tilde{\mu}_i\right)^2 = \sum_{x \in \omega_i} \left(w^T x - w^T \mu_i\right)^2$$

$$= \sum_{x \in \omega_i} w^T \left(x - \mu_i\right)\left(x - \mu_i\right)^T w$$

$$= w^T \left(\sum_{x \in \omega_i} \left(x - \mu_i\right)\left(x - \mu_i\right)^T \right) w = w^T S_i w$$

$$\tilde{s}_1^2 + \tilde{s}_2^2 = w^T S_1 w + w^T S_2 w = w^T \left(S_1 + S_2\right) w = w^T S_W w = \tilde{S}_W$$

Where $\tilde{S}_W$ is the within-class scatter matrix of the projected samples **y**.

LDA … Two Classes

- Similarly, the difference between the projected means (in y-space) can be expressed in terms of the means in the original feature space (x-space).

$$J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

$$\left(\tilde{\mu}_1 - \tilde{\mu}_2\right)^2 = \left(w^T \mu_1 - w^T \mu_2\right)^2$$

$$= w^T \underbrace{\left(\mu_1 - \mu_2\right)\left(\mu_1 - \mu_2\right)^T}_{S_B} w$$

$$= w^T S_B w = \tilde{S}_B$$

- The matrix $\mathbf{S_B}$ is called the *between-class scatter* of the original samples/feature vectors, while $\tilde{S}_B$ is the between-class scatter of the projected samples **y**.
- Since $\mathbf{S_B}$ is the outer product of two vectors, its rank is at most one.

LDA … Two Classes

- We can finally express the Fisher criterion in terms of $\mathbf{S_W}$ and $\mathbf{S_B}$ as:

$$\boxed{J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} = \frac{w^T S_B w}{w^T S_W w}}$$

- Hence ***J(w)*** **is a measure of the difference between class means (encoded in the between-class scatter matrix) normalized by a measure of the within-class scatter matrix.**

LDA … Two Classes

- To find the maximum of *J(w)*, we differentiate and equate to zero.

$$\frac{d}{dw}J(w) = \frac{d}{dw}\left(\frac{w^T S_B w}{w^T S_W w}\right) = 0$$

$$\Rightarrow \left(w^T S_W w\right)\frac{d}{dw}\left(w^T S_B w\right) - \left(w^T S_B w\right)\frac{d}{dw}\left(w^T S_W w\right) = 0$$

$$\Rightarrow \left(w^T S_W w\right)2S_B w - \left(w^T S_B w\right)2S_W w = 0$$

$$\textit{Dividing by } 2w^T S_W w:$$

$$\Rightarrow \left(\frac{w^T S_W w}{w^T S_W w}\right)S_B w - \left(\frac{w^T S_B w}{w^T S_W w}\right)S_W w = 0$$

$$\Rightarrow S_B w - J(w)S_W w = 0$$

$$\Rightarrow S_W^{-1}S_B w - J(w)w = 0$$

LDA … Two Classes

- Solving the generalized eigen value problem

$$S_W^{-1} S_B w = \lambda w \qquad where \quad \lambda = J(w) = scalar$$

yields

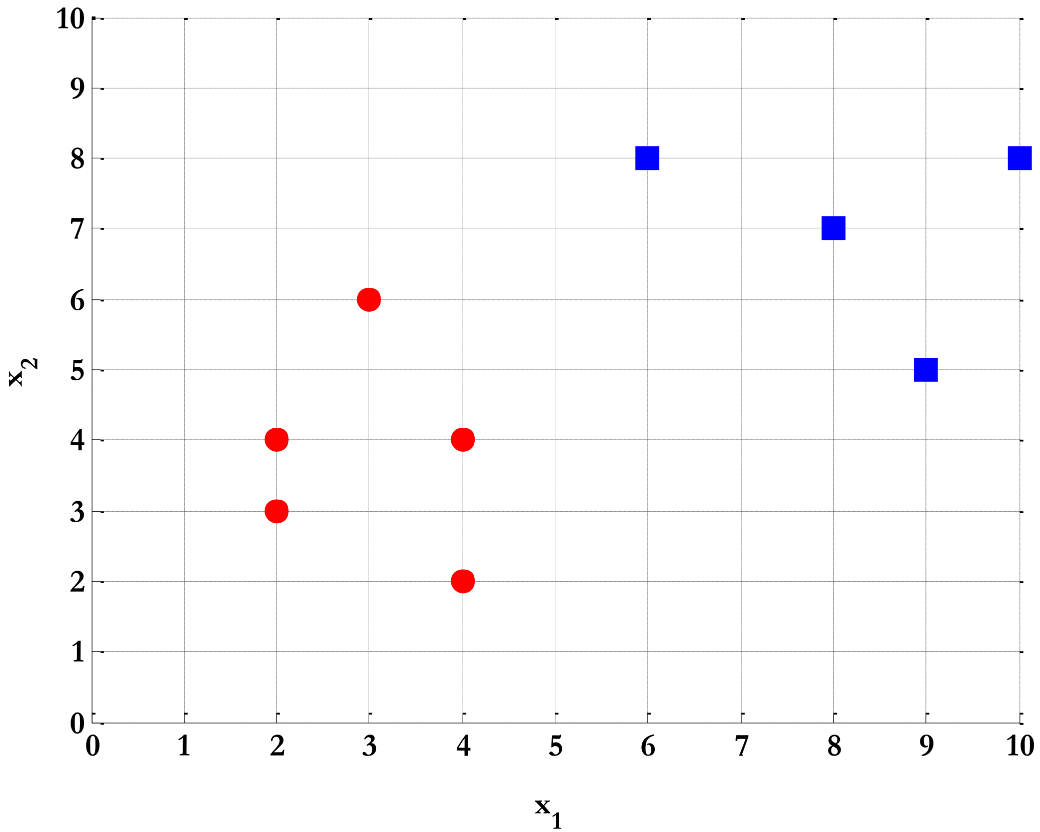
$$w^* = \arg\max_w J(w) = \arg\max_w \left(\frac{w^T S_B w}{w^T S_W w} \right) = S_W^{-1} (\mu_1 - \mu_2)$$

- This is known as Fisher's Linear Discriminant, although it is not a discriminant but rather a specific choice of direction for the projection of the data down to one dimension.

- Using the same notation as PCA, **the solution will be the eigen vector(s) of** $S_X = S_W^{-1} S_B$

LDA … Two Classes - Example

- Compute the Linear Discriminant projection for the following two-dimensional dataset.
  - Samples for class $\boldsymbol{\omega_1}$: $\mathbf{X_1}=(x_1,x_2)=\{(4,2),(2,4),(2,3),(3,6),(4,4)\}$
  - Sample for class $\boldsymbol{\omega_2}$: $\mathbf{X_2}=(x_1,x_2)=\{(9,10),(6,8),(9,5),(8,7),(10,8)\}$




```
% samples for class 1
X1 = [4,2;
      2,4;
      2,3;
      3,6;
      4,4];


% samples for class 2
X2 = [9,10;
      6,8;
      9,5;
      8,7;
      10,8];
```

LDA … Two Classes - Example

- The classes mean are :

$$\mu_1 = \frac{1}{N_1}\sum_{x\in\omega_1} x = \frac{1}{5}\left[\begin{pmatrix}4\\2\end{pmatrix}+\begin{pmatrix}2\\4\end{pmatrix}+\begin{pmatrix}2\\3\end{pmatrix}+\begin{pmatrix}3\\6\end{pmatrix}+\begin{pmatrix}4\\4\end{pmatrix}\right]=\begin{pmatrix}3\\3.8\end{pmatrix}$$

$$\mu_2 = \frac{1}{N_2}\sum_{x\in\omega_2} x = \frac{1}{5}\left[\begin{pmatrix}9\\10\end{pmatrix}+\begin{pmatrix}6\\8\end{pmatrix}+\begin{pmatrix}9\\5\end{pmatrix}+\begin{pmatrix}8\\7\end{pmatrix}+\begin{pmatrix}10\\8\end{pmatrix}\right]=\begin{pmatrix}8.4\\7.6\end{pmatrix}$$

```
% class means
Mu1 = mean(X1)';
Mu2 = mean(X2)';
```

LDA … Two Classes - Example

- Covariance matrix of the first class:

$$S_1 = \sum_{x \in \omega_1} (x - \mu_1)(x - \mu_1)^T = \left[\begin{pmatrix} 4 \\ 2 \end{pmatrix} - \begin{pmatrix} 3 \\ 3.8 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 2 \\ 4 \end{pmatrix} - \begin{pmatrix} 3 \\ 3.8 \end{pmatrix}\right]^2$$

$$+ \left[\begin{pmatrix} 2 \\ 3 \end{pmatrix} - \begin{pmatrix} 3 \\ 3.8 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 3 \\ 6 \end{pmatrix} - \begin{pmatrix} 3 \\ 3.8 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 3 \\ 3.8 \end{pmatrix}\right]^2$$

$$= \begin{pmatrix} 1 & -0.25 \\ -0.25 & 2.2 \end{pmatrix}$$

```
% covariance matrix of the first class
S1 = cov(X1);
```

LDA … Two Classes - Example

- Covariance matrix of the second class:

$$S_2 = \sum_{x \in \omega_2} (x-\mu_2)(x-\mu_2)^T = \left[\begin{pmatrix} 9 \\ 10 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 6 \\ 8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^2$$

$$+ \left[\begin{pmatrix} 9 \\ 5 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 8 \\ 7 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^2 + \left[\begin{pmatrix} 10 \\ 8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^2$$

$$= \begin{pmatrix} 2.3 & -0.05 \\ -0.05 & 3.3 \end{pmatrix}$$

```
% covariance matrix of the first class
S2 = cov(X2);
```

LDA … Two Classes - Example

- Within-class scatter matrix:

$$S_w = S_1 + S_2 = \begin{pmatrix} 1 & -0.25 \\ -0.25 & 2.2 \end{pmatrix} + \begin{pmatrix} 2.3 & -0.05 \\ -0.05 & 3.3 \end{pmatrix}$$

$$= \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}$$

```
% within-class scatter matrix
Sw = S1 + S2 ;
```

LDA … Two Classes - Example

- Between-class scatter matrix:

$$
\begin{aligned}
S_B &= (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T \\
&= \left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]\left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^T \\
&= \begin{pmatrix} -5.4 \\ -3.8 \end{pmatrix}\begin{pmatrix} -5.4 & -3.8 \end{pmatrix} \\
&= \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix}
\end{aligned}
$$

```
% between-class scatter matrix
SB = (Mu1-Mu2)*(Mu1-Mu2)';
```

LDA … Two Classes - Example

- The LDA projection is then obtained as the solution of the generalized eigen value problem $S_W^{-1}S_B w = \lambda w$

$$\Rightarrow \left| S_W^{-1}S_B - \lambda I \right| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}^{-1} \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 0.3045 & 0.0166 \\ 0.0166 & 0.1827 \end{pmatrix} \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 9.2213 - \lambda & 6.489 \\ 4.2339 & 2.9794 - \lambda \end{pmatrix} \right|$$

$$= (9.2213 - \lambda)(2.9794 - \lambda) - 6.489 \times 4.2339 = 0$$

$$\Rightarrow \lambda^2 - 12.2007\lambda = 0 \Rightarrow \lambda(\lambda - 12.2007) = 0$$

$$\Rightarrow \lambda_1 = 0, \lambda_2 = 12.2007$$

LDA … Two Classes - Example

```
% computing the LDA projection
invSw = inv(Sw);

invSw_by_SB = invSw * SB;

% getting the projection vector
[V,D] = eig(invSw_by_SB)

% the projection vector
W = V(:,1);
```

- Hence

$$\begin{pmatrix} 9.2213 & 6.489 \\ 4.2339 & 2.9794 \end{pmatrix} w_1 = \underbrace{0}_{\lambda_1} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

and

$$\begin{pmatrix} 9.2213 & 6.489 \\ 4.2339 & 2.9794 \end{pmatrix} w_2 = \boxed{\underbrace{12.2007}_{\lambda_2}} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

Thus;

$$w_1 = \begin{pmatrix} -0.5755 \\ 0.8178 \end{pmatrix} \quad and \quad \boxed{w_2 = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix} = w^*}$$

- The optimal projection is the one that given maximum $\lambda = J(w)$

LDA … Two Classes - Example

Or directly;

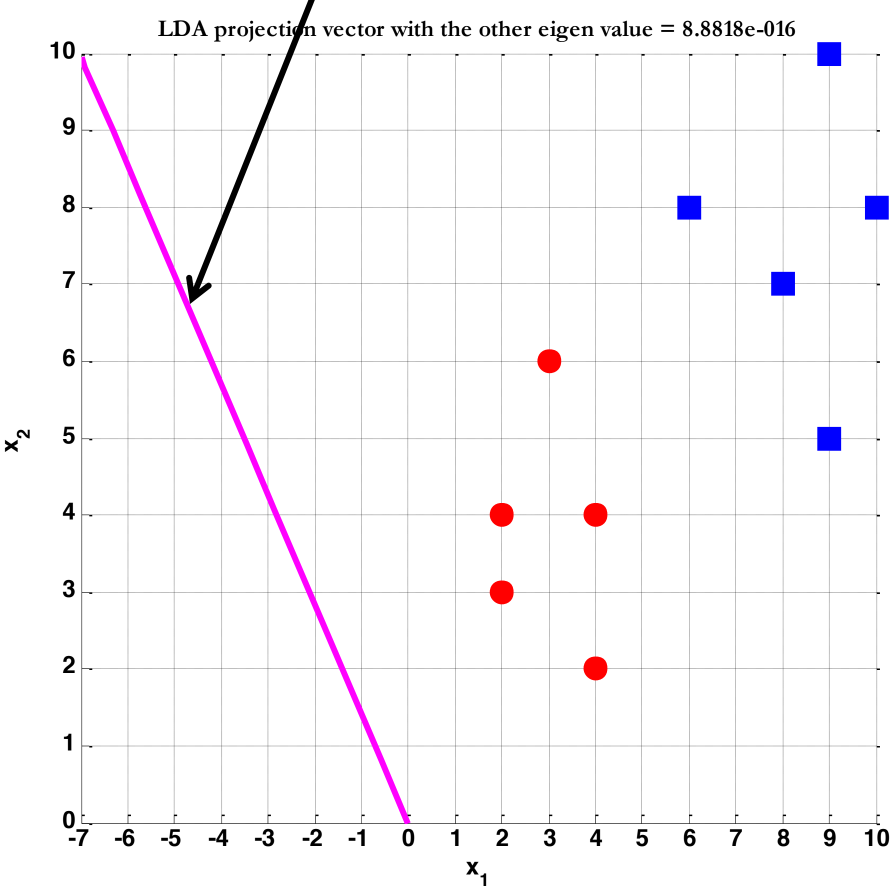
$$w^* = S_W^{-1}(\mu_1 - \mu_2) = \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}^{-1} \left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix} \right]$$

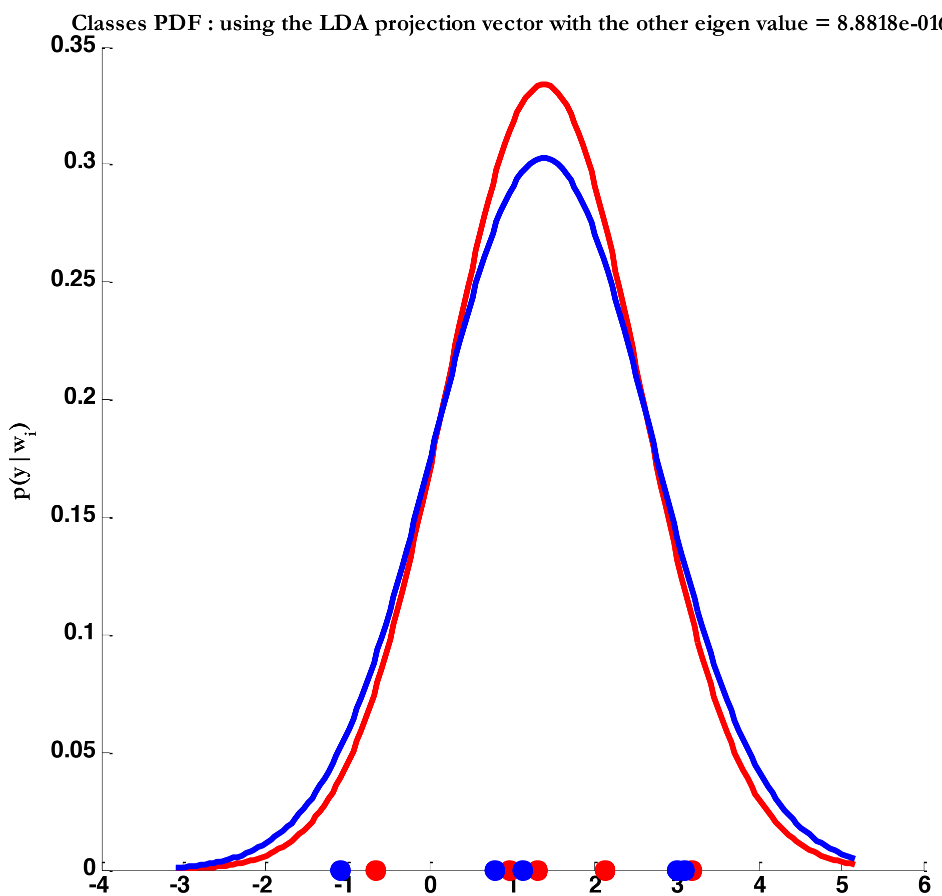
$$= \begin{pmatrix} 0.3045 & 0.0166 \\ 0.0166 & 0.1827 \end{pmatrix} \begin{pmatrix} -5.4 \\ -3.8 \end{pmatrix}$$

$$= \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix}$$

LDA - Projection

The projection vector corresponding to the **smallest** eigen value

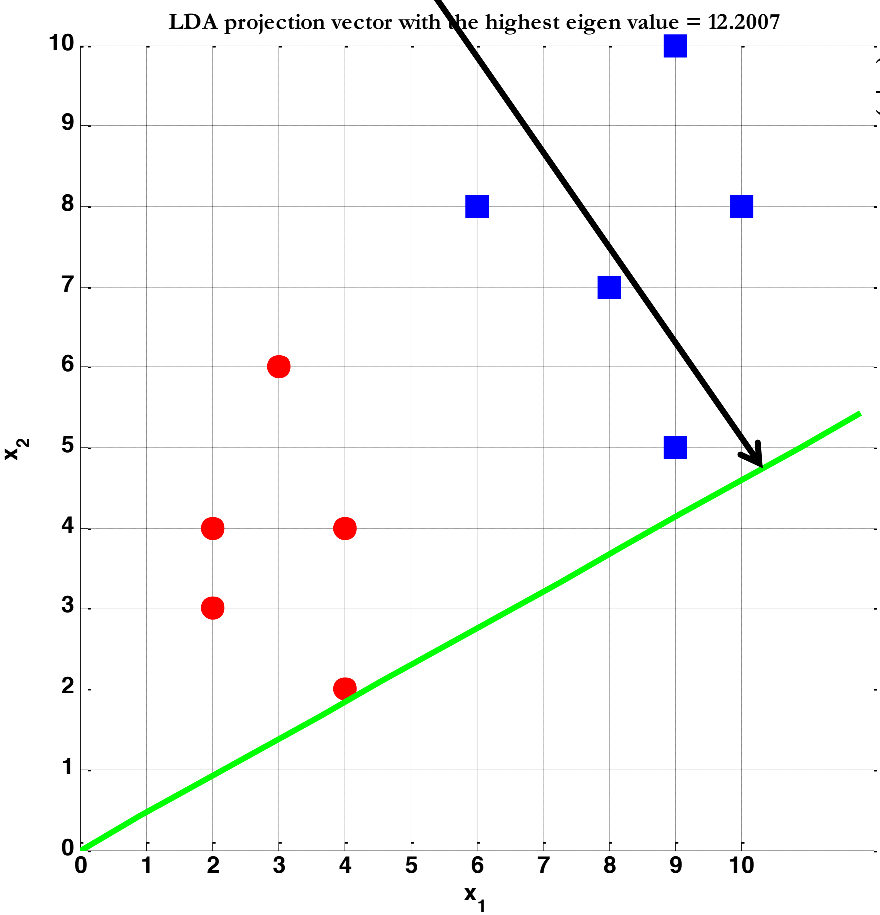


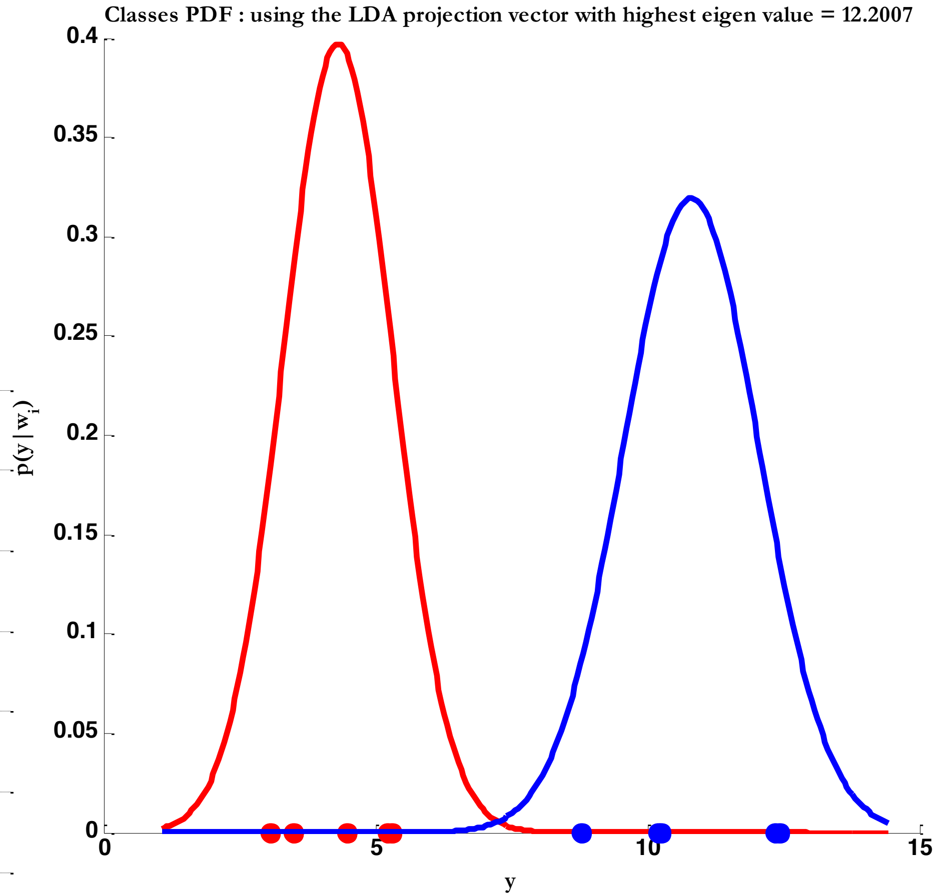


Using this vector leads to **bad separability** between the two classes

LDA - Projection

The projection vector corresponding to the **highest** eigen value





Using this vector leads to **good separability** between the two classes

LDA … C-Classes

- Now, we have *C*-classes instead of just two.
- We are now seeking (*C-1*) projections [$\mathbf{y_1}$, $\mathbf{y_2}$, …, $\mathbf{y_{C-1}}$] by means of (*C-1*) projection vectors $\mathbf{w_i}$.
- $\mathbf{w_i}$ can be arranged by *columns* into a projection matrix $\mathbf{W}$ = [$\mathbf{w_1}$|$\mathbf{w_2}$|…|$\mathbf{w_{C-1}}$] such that:

$$y_i = w_i^T x \quad \Rightarrow \quad y = W^T x$$

$$where \quad x_{m\times 1} = \begin{bmatrix} x_1 \\ . \\ . \\ x_m \end{bmatrix} \quad , \quad y_{C-1\times 1} = \begin{bmatrix} y_1 \\ . \\ . \\ y_{C-1} \end{bmatrix}$$

$$and \quad W_{m\times C-1} = \left[w_1 | \quad w_2 | \quad ... \quad | w_{C-1} \right]$$

LDA … C-Classes

- If we have *n*-feature vectors, we can stack them into one matrix as follows;

$$Y = W^T X$$

$$where \quad X_{m\times n} = \begin{bmatrix} x_1^1 & x_1^2 & . & x_1^n \\ . & . & . & . \\ . & . & . & . \\ x_m^1 & x_m^2 & . & x_m^n \end{bmatrix} \quad , \quad Y_{C-1\times n} = \begin{bmatrix} y_1^1 & y_1^2 & . & y_1^n \\ . & . & . & . \\ . & . & . & . \\ y_{C-1}^1 & y_{C-1}^2 & . & y_{C-1}^n \end{bmatrix}$$

$$and \quad W_{m\times C-1} = \left[w_1 \mid \quad w_2 \mid \quad \ldots \quad \mid w_{C-1} \right]$$

LDA – C-Classes

- Recall the two classes case, the *within-class scatter* was computed as:

$$S_w = S_1 + S_2$$

- This can be generalized in the *C*-classes case as:

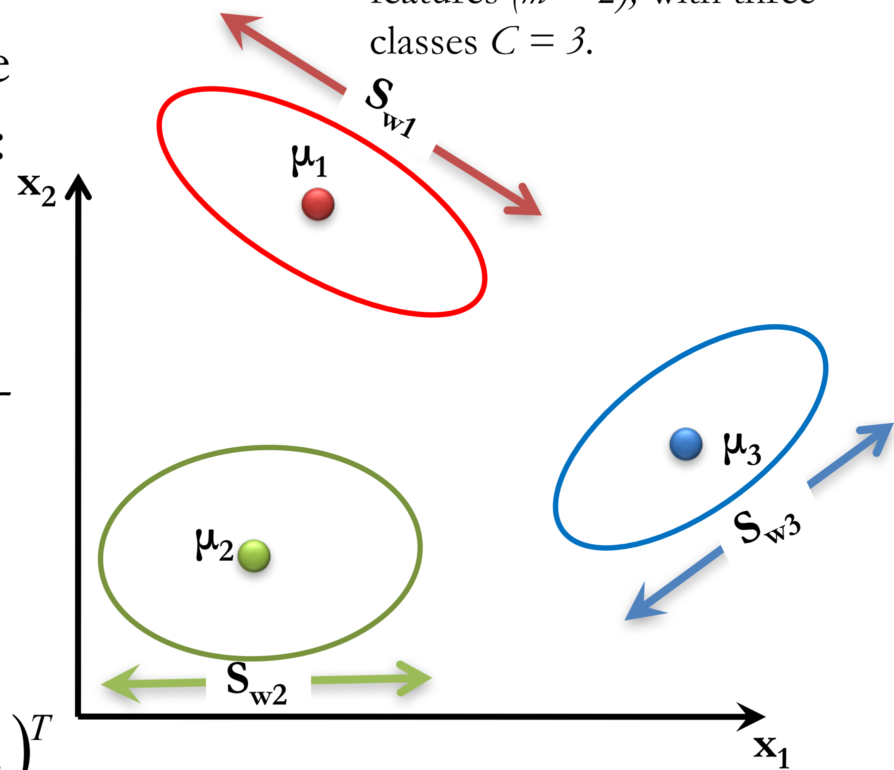
$$S_W = \sum_{i=1}^{C} S_i$$

where $$S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T$$

and $$\mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

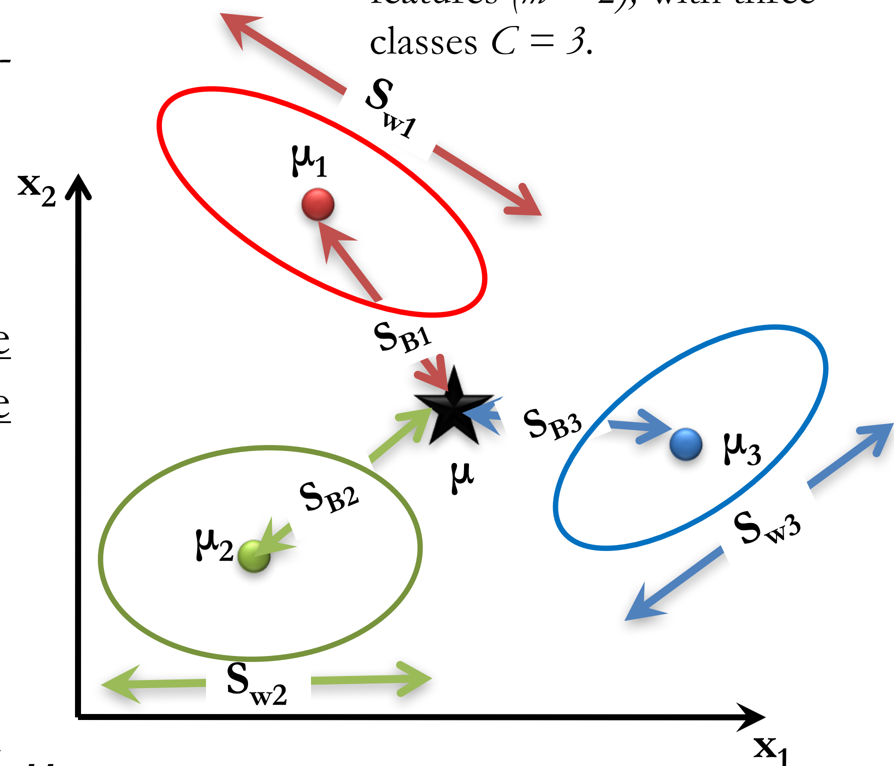
N_i : number of data samples in class **ω_i**.

Example of two-dimensional features *(m = 2)*, with three classes *C = 3*.



LDA – C-Classes

Example of two-dimensional features *(m = 2)*, with three classes *C = 3*.



- Recall the two classes case, the *between-class scatter* was computed as:

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

- For *C*-classes case, we will measure the between-class scatter with respect to the mean of all classes as follows:

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$where \qquad \mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{\forall x} N_i \mu_i$$

N: number of all data .

$$and \qquad \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

$\mathbf{N_i}$: number of data samples in class $\boldsymbol{\omega_i}$.

LDA – C-Classes

- Similarly,
  - We can define the mean vectors for the projected samples **y** as:

$$\tilde{\mu}_i = \frac{1}{N_i} \sum_{y \in \omega_i} y \qquad and \qquad \tilde{\mu} = \frac{1}{N} \sum_{\forall y} y$$

  - While the scatter matrices for the projected samples **y** will be:

$$\tilde{S}_W = \sum_{i=1}^{C} \tilde{S}_i = \sum_{i=1}^{C} \sum_{y \in \omega_i} (y - \tilde{\mu}_i)(y - \tilde{\mu}_i)^T$$

$$\tilde{S}_B = \sum_{i=1}^{C} N_i (\tilde{\mu}_i - \tilde{\mu})(\tilde{\mu}_i - \tilde{\mu})^T$$

LDA – C-Classes

- Recall in two-classes case, we have expressed the scatter matrices of the projected samples in terms of those of the original samples as:

$$\widetilde{S}_W = W^T S_W W$$

$$\widetilde{S}_B = W^T S_B W$$

This still hold in *C*-classes case.

- Recall that we are looking for a projection that maximizes the ratio of between-class to within-class scatter.

- Since the projection is no longer a scalar (it has *C-1* dimensions), we then use the determinant of the scatter matrices to obtain a scalar objective function:

$$J(W) = \frac{\left|\widetilde{S}_B\right|}{\left|\widetilde{S}_W\right|} = \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

- And we will seek the projection $\mathbf{W}^*$ that maximizes this ratio.

LDA – C-Classes

- To find the maximum of *J(W)*, we differentiate with respect to **W** and equate to zero.

- Recall in two-classes case, we solved the eigen value problem.

$$S_W^{-1} S_B w = \lambda w \qquad where \quad \lambda = J(w) = scalar$$

- For *C*-classes case, we have *C-1* projection vectors, hence the eigen value problem can be generalized to the *C*-classes case as:

$$S_W^{-1} S_B w_i = \lambda_i w_i \qquad where \quad \lambda_i = J(w_i) = scalar \qquad and \quad i = 1,2,\ldots C-1$$

- Thus, It can be shown that the optimal projection matrix $\mathbf{W}^*$ is the one whose columns are the eigenvectors corresponding to the largest eigen values of the following generalized eigen value problem:

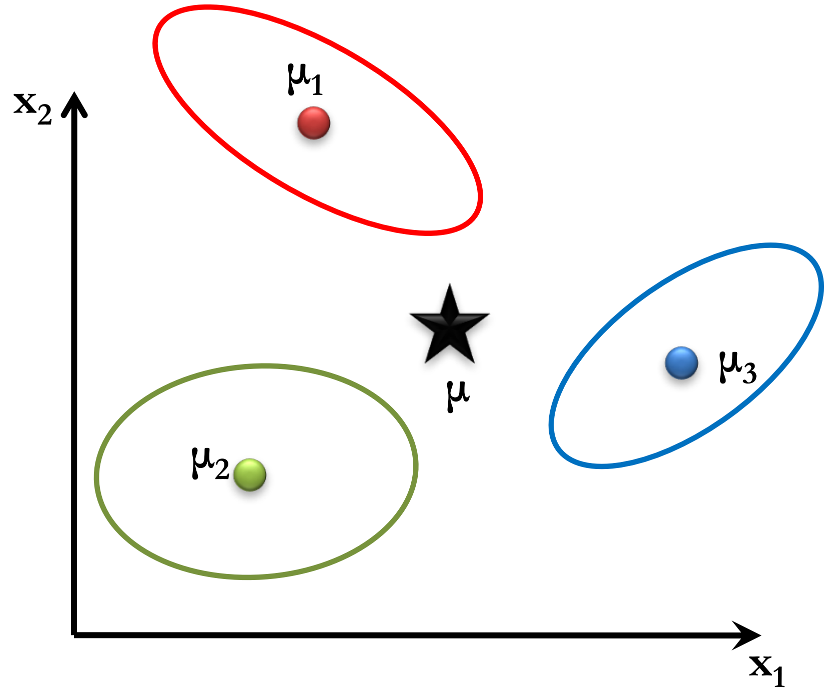
$$S_W^{-1} S_B W^* = \lambda W^*$$

$$where \quad \lambda = J(W^*) = scalar \qquad and \qquad W^* = \left[w_1^* \mid \quad w_2^* \mid \quad \ldots \quad \mid w_{C-1}^* \right]$$

Illustration – 3 Classes

- Let's generate a dataset for each class to simulate the three classes shown

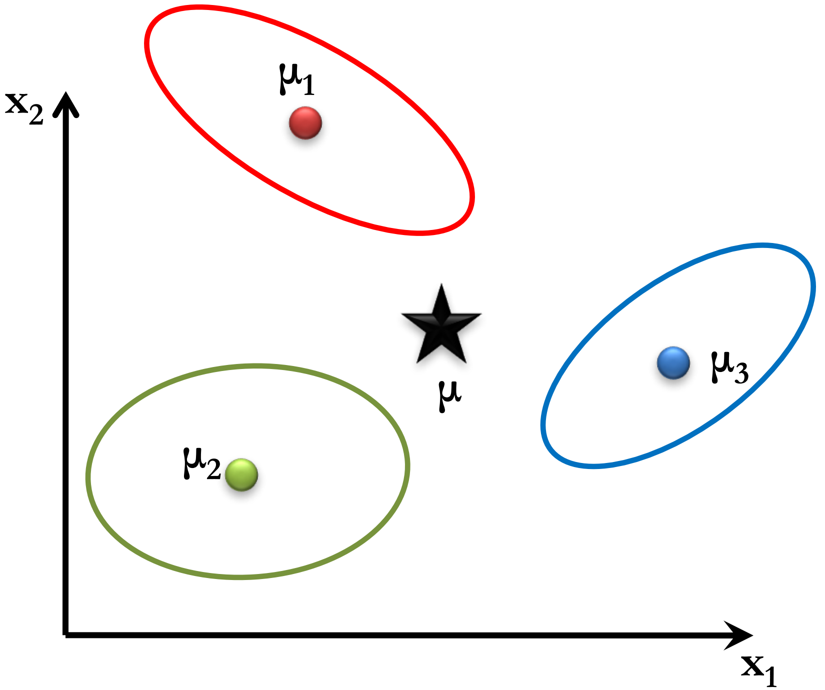
- For each class do the following,
  - Use the random number generator to generate a uniform stream of 500 samples that follows U(0,1).
  - Using the Box-Muller approach, convert the generated uniform stream to N(0,1).



  - Then use the method of eigen values and eigen vectors to manipulate the standard normal to have the required mean vector and covariance matrix .
  - Estimate the mean and covariance matrix of the resulted dataset.

Dataset Generation

- By visual inspection of the figure, classes parameters (means and covariance matrices can be given as follows:



Overall mean $\mu = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$

$$\mu_1 = \mu + \begin{bmatrix} -3 \\ 7 \end{bmatrix}, \quad \mu_2 = \mu + \begin{bmatrix} -2.5 \\ -3.5 \end{bmatrix}, \quad \mu_3 = \mu + \begin{bmatrix} 7 \\ 5 \end{bmatrix}$$

$$S_1 = \begin{pmatrix} 5 & -1 \\ -3 & 3 \end{pmatrix}$$ → Negative covariance to lead to data samples distributed along the $y = -x$ line.

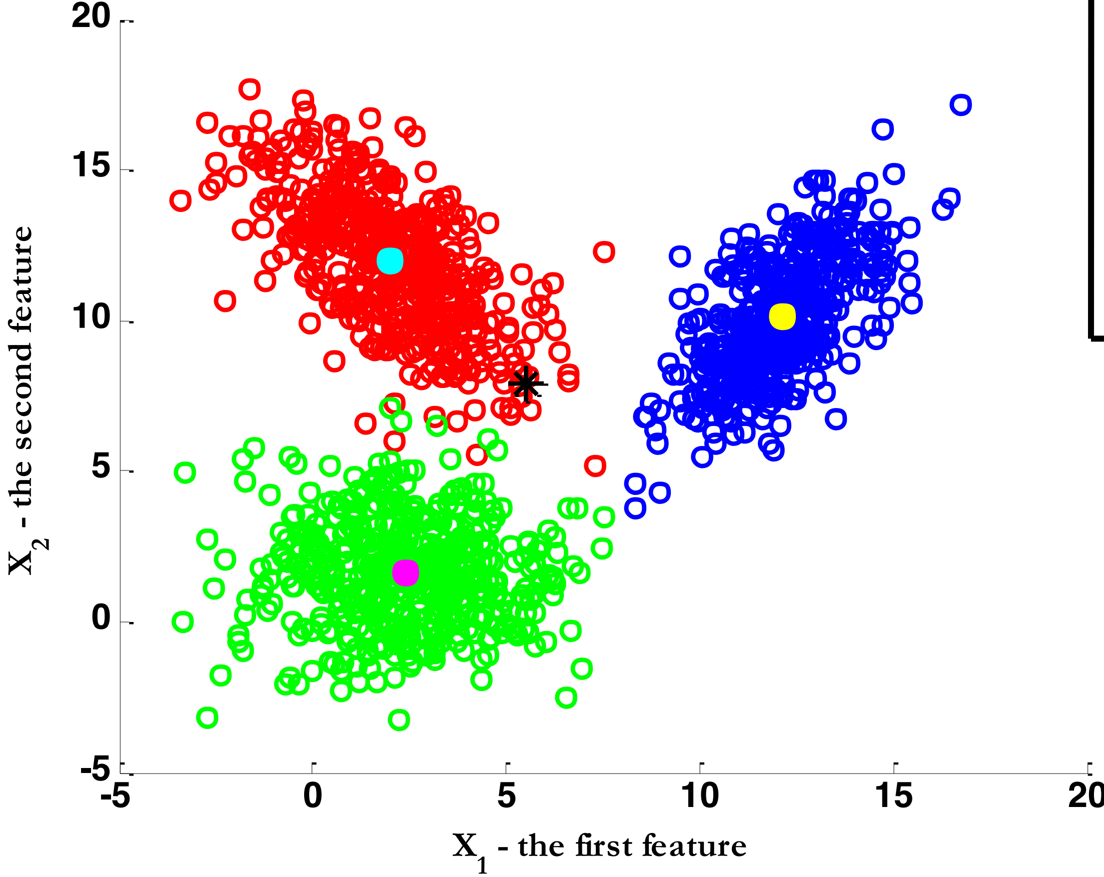
$$S_2 = \begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}$$ → Zero covariance to lead to data samples distributed *horizontally*.

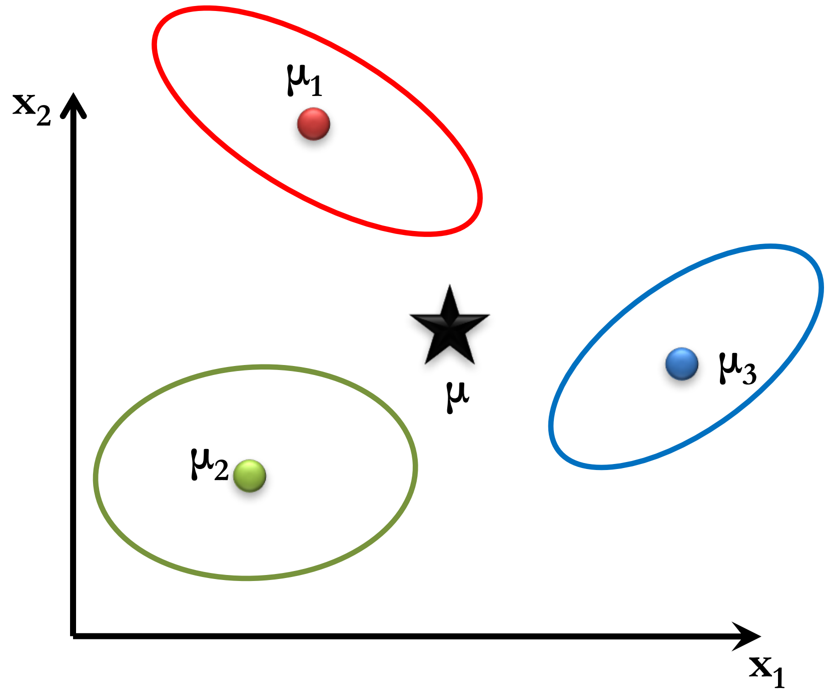
$$S_3 = \begin{pmatrix} 3.5 & 1 \\ 3 & 2.5 \end{pmatrix}$$ → Positive covariance to lead to data samples distributed along the $y = x$ line.

In Matlab ☺

```
% let the center of all classes be
Mu = [ 5;5];

%% for the first class
Mu1 = [Mu(1)-3; Mu(2)+7];
CovM1 = [5 -1; -3 3];

% Generating feature vectors using Box-Muller approach
% Generate a random variable following uniform(0,1) having two features and
% 1000 feature vectors
U = rand(2,1000);

% Extracting from the generated uniform random variable two independent
% uniform random variables
u1 = U(:,1:2:end);
u2 = U(:,2:2:end);

% Using u1 and u2, we will use Box-Muller method to generate the feature
% vectors to follow standard normal
X = sqrt((-2).*log(u1))  .* (cos(2*pi.*u2));
clear u1 u2 U;

% Now ... Manipulating the generated Features N(0,1) to following certain
% mean and covariance other than the standard normal

% First we will change its variance then we will change its mean

% Getting the eigen vectors and values of the covariance matrix
[V,D] = eig(CovM1); % D is the eigen values matrix and V is the eigen vectors
matrix
newX = X;
for j = 1 : size(X,2)
    newX(:,j) = V * sqrt(D) * X(:,j);
end

% changing its mean
newX = newX + repmat(Mu1,1,size(newX,2));

% now our dataset for the first class matrix will be
X1 = newX ; % each column is a feature vector, each row is a single feature


% ... do the same for the other two classes with difference means and
covariance matrices
```

It's Working … ☺


20
15
10
5
0
-5
-5
0
5
10
15
20
X_2 - the second feature
X_1 - the first feature


x_2
x_1
μ_1
μ_2
μ_3
μ

Computing LDA Projection Vectors

```matlab
%% computing the LDA
% class means
Mu1 = mean(X1')';
Mu2 = mean(X2')';
Mu3 = mean(X3')';

% overall mean
Mu = (Mu1 + Mu2 + Mu3)./3;

% class covariance matrices
S1 = cov(X1');
S2 = cov(X2');
S3 = cov(X3');

% within-class scatter matrix
Sw = S1 + S2 + S3 ;

% number of samples of each class
N1 = size(X1,2);
N2 = size(X2,2);
N3 = size(X3,2);

% between-class scatter matrix
SB1 = N1 .* (Mu1-Mu)*(Mu1-Mu)';
SB2 = N2 .* (Mu2-Mu)*(Mu2-Mu)';
SB3 = N3 .* (Mu3-Mu)*(Mu3-Mu)';

SB = SB1 + SB2 + SB3;

% computing the LDA projection
invSw = inv(Sw);
invSw_by_SB = invSw * SB;

% getting the projection vectors
%[V,D] = EIG(X) produces a diagonal matrix D of eigenvalues and a
%full matrix V whose columns are the corresponding eigenvectors
[V,D] = eig(invSw_by_SB);

% the projection vectors - we will have at most C-1 projection vectors,
% from which we can choose the most important ones ranked by their
% corresponding eigen values ... lets investigate the two projection
% vectors
W1 = V(:,1);
W2 = V(:,2);
```

$$\boxed{S_W^{-1} S_B}$$

Recall …

$$S_W = \sum_{i=1}^{C} S_i$$

$$\text{where} \qquad S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T$$

$$\text{and} \qquad \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

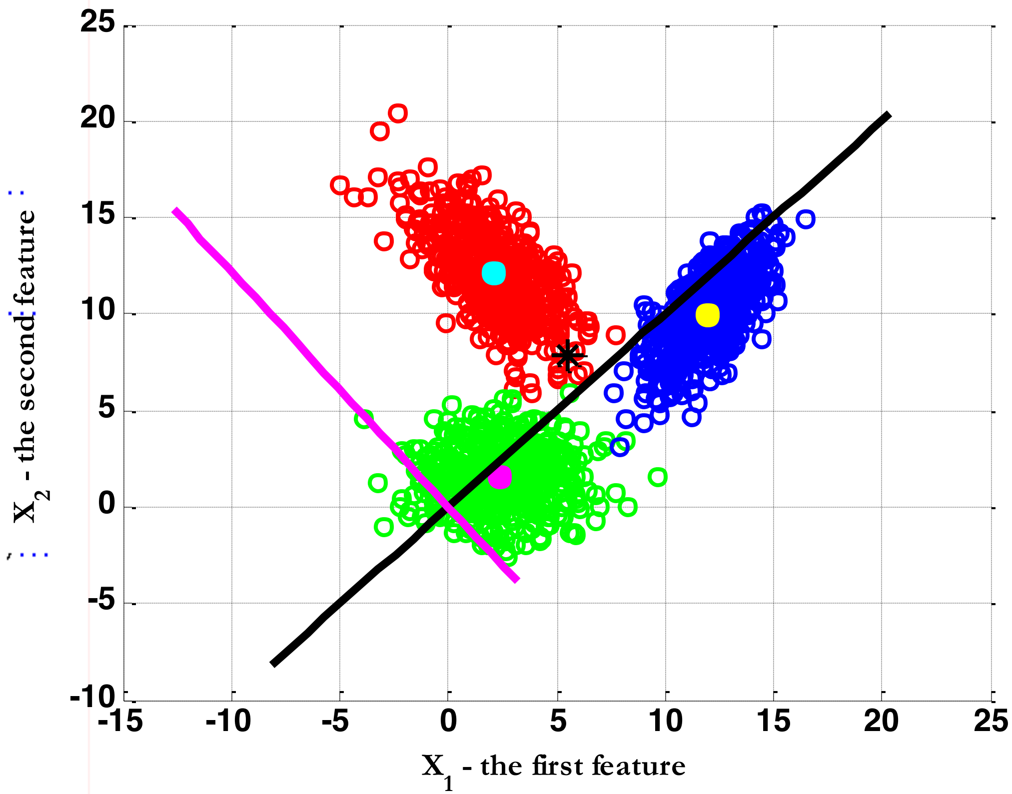
$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$\text{where} \qquad \mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{\forall x} N_i \mu_i$$

$$\text{and} \qquad \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

Let's visualize the projection vectors *W*

```
%% lets visualize them ...
% we will plot the scatter plot to better visualize the features
hfig = figure;
axes1 = axes('Parent',hfig,'FontWeight','bold','FontSize',12);
hold('all');

% Create xlabel
xlabel('X_1 - the first feature','FontWeight','bold','FontSize',12,...
    'FontName','Garamond');

% Create ylabel
ylabel('X_2 - the second feature','FontWeight','bold','FontSize',12,...
    'FontName','Garamond');

% the first class
scatter(X1(1,:),X1(2,:), 'r','LineWidth',2,'Parent',axes1);
hold on

% class's mean
plot(Mu1_est(1),Mu1_est(2),'co','MarkerSize',8,'MarkerEdgeColor','c',...
    'Color','c','LineWidth',2,'MarkerFaceColor','c','Parent',axes1);
hold on

% the second class
scatter(X2(1,:),X2(2,:), 'g','LineWidth',2,'Parent',axes1);
hold on

% class's mean
plot(Mu2_est(1),Mu2_est(2),'mo','MarkerSize',8,'MarkerEdgeColor','m',...
    'Color','m','LineWidth',2,'MarkerFaceColor','m','Parent',axes1);
hold on

% the third class
scatter(X3(1,:),X3(2,:), 'b','LineWidth',2,'Parent',axes1);
hold on

% class's mean
plot(Mu3_est(1),Mu3_est(2),'yo','LineWidth',2,'MarkerSize',8,'MarkerEdgeColor',...
    'y','Color','y','MarkerFaceColor','y','Parent',axes1);
hold on
```

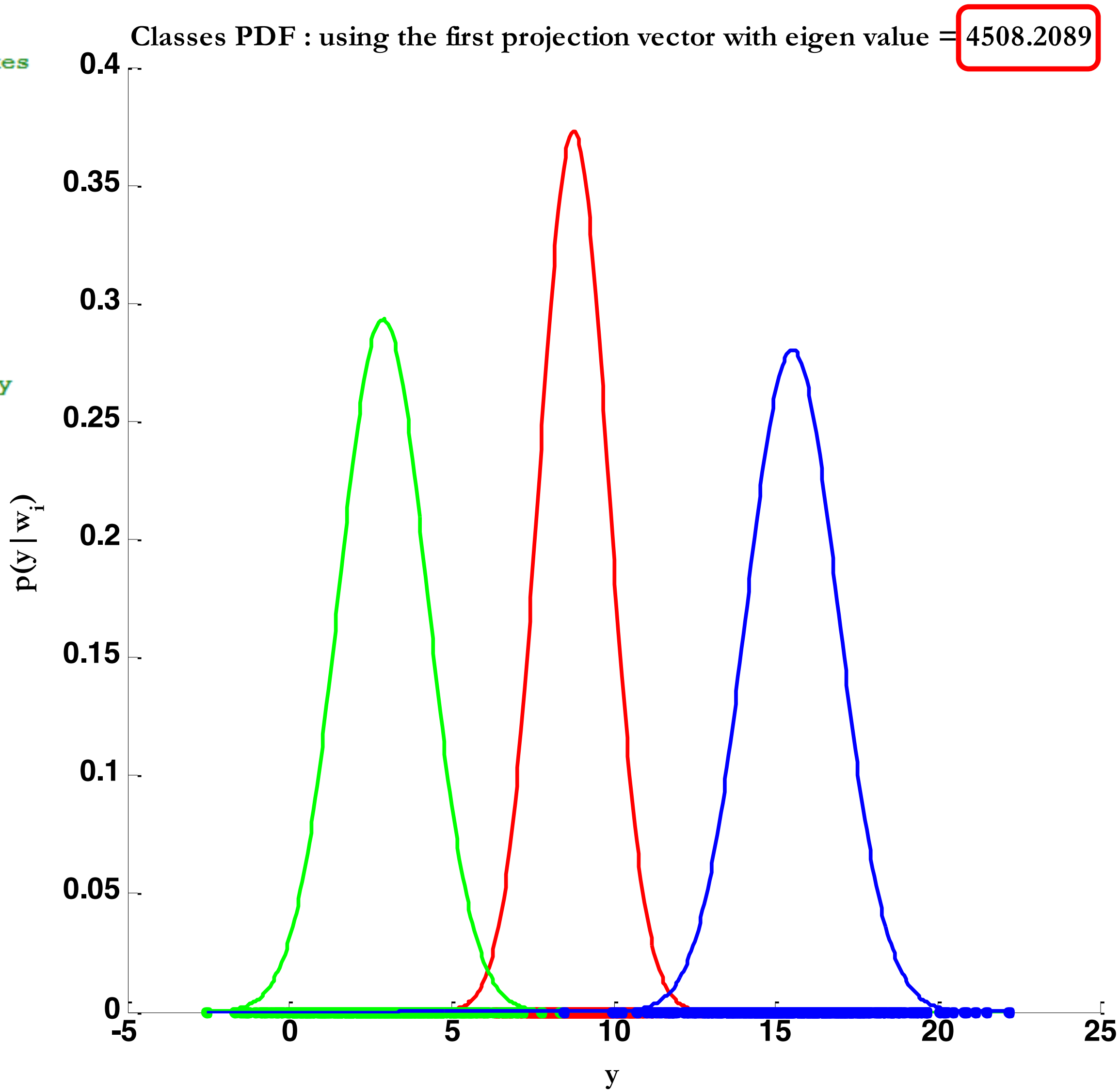



```
% drawing the projection vectors
% the first vector
t = -10:25;
line_x1 = t .* W1(1);
line_y1 = t .* W1(1);

% the second vector
t = -5:20;
line_x2 = t .* W2(1);
line_y2 = t .* W2(2);

plot(line_x1,line_y1,'k-','LineWidth',3);
hold on
plot(line_x2,line_y2,'m-','LineWidth',3);
grid on
```

Projection … $y = W^T x$

Along first projection vector

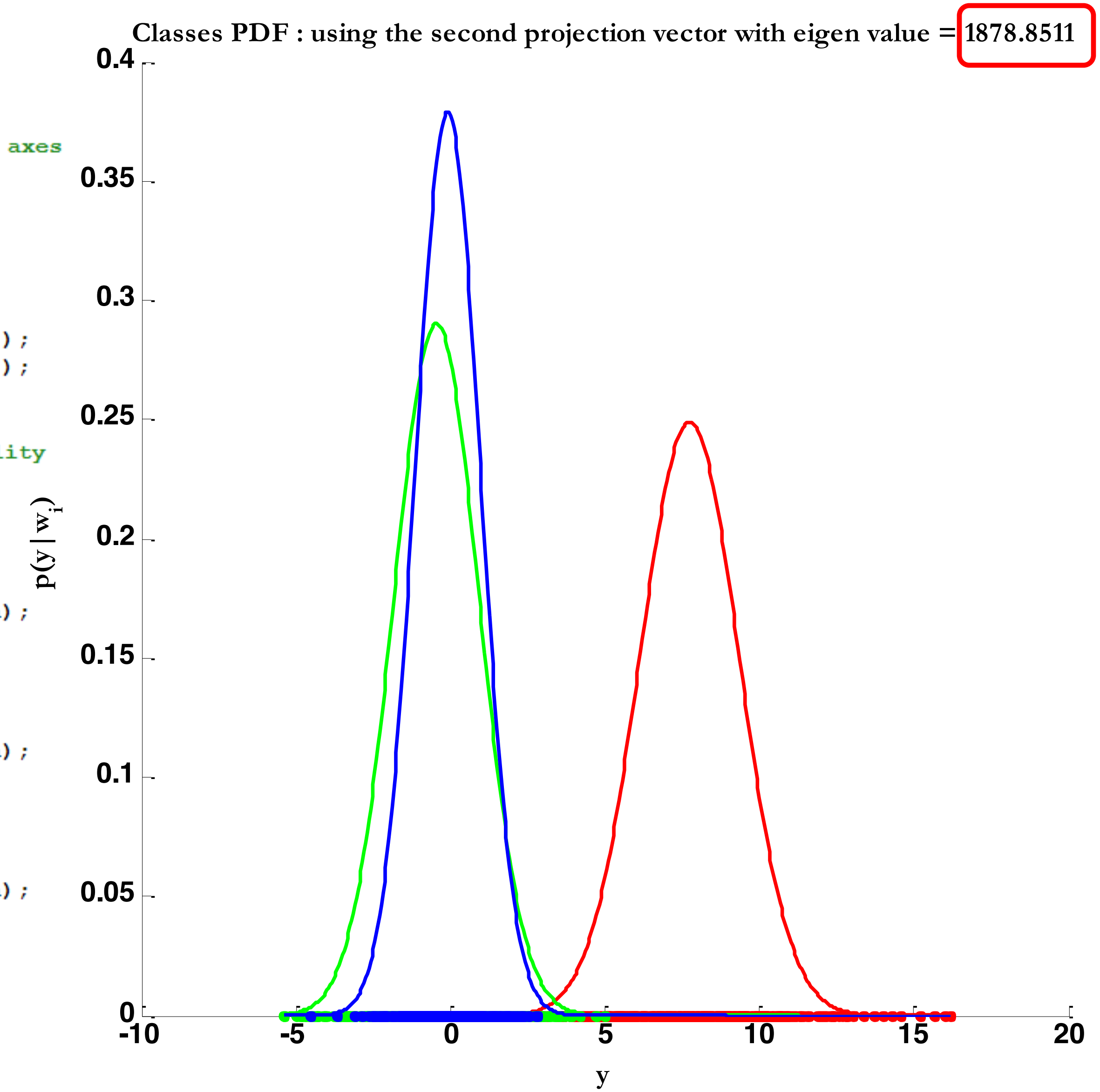
```
% project data samples along the projections axes
% the first projection vector
y1_w1 = W1'*X1;
y2_w1 = W1'*X2;
y3_w1 = W1'*X3;

% projection limits
minY = min([min(y1_w1),min(y2_w1),min(y3_w1)]);
maxY = max([max(y1_w1),max(y2_w1),max(y3_w1)]);
y_w1 = minY:0.05:maxY;

% for visualization lets compute the probability
% density function of the
% classes after projection
% the first class
y1_w1_Mu = mean(y1_w1);
y1_w1_sigma = std(y1_w1);
y1_w1_pdf = mvnpdf(y_w1',y1_w1_Mu,y1_w1_sigma);

% the second class
y2_w1_Mu = mean(y2_w1);
y2_w1_sigma = std(y2_w1);
y2_w1_pdf = mvnpdf(y_w1',y2_w1_Mu,y2_w1_sigma);

% the third class
y3_w1_Mu = mean(y3_w1);
y3_w1_sigma = std(y3_w1);
y3_w1_pdf = mvnpdf(y_w1',y3_w1_Mu,y3_w1_sigma);
```



Projection … $y = W^T x$

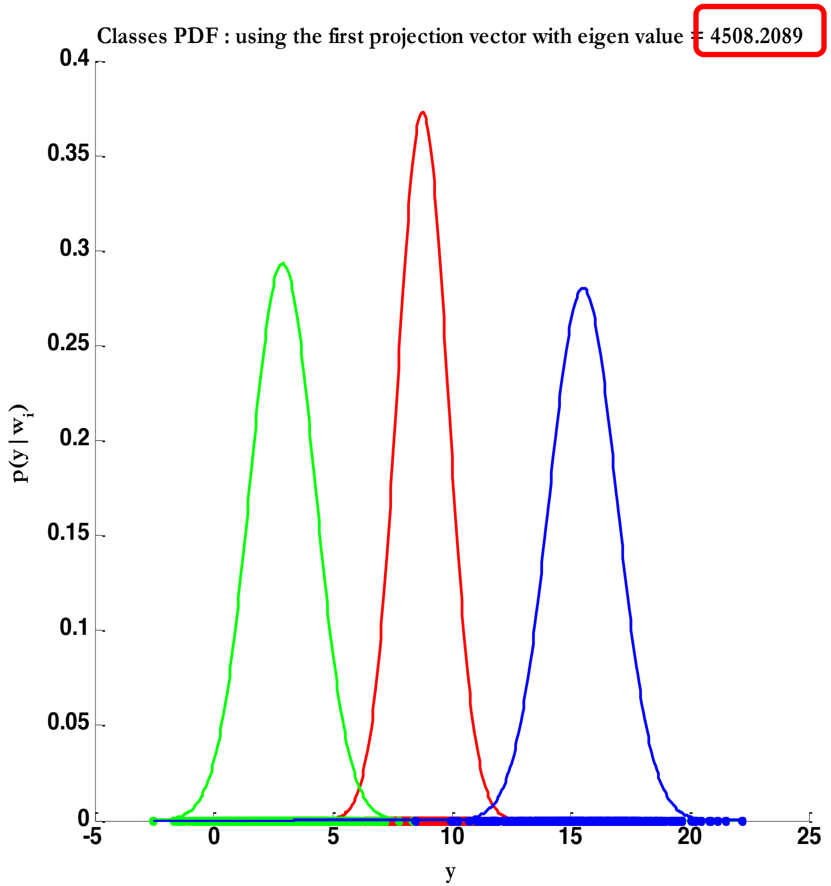
Along second projection vector

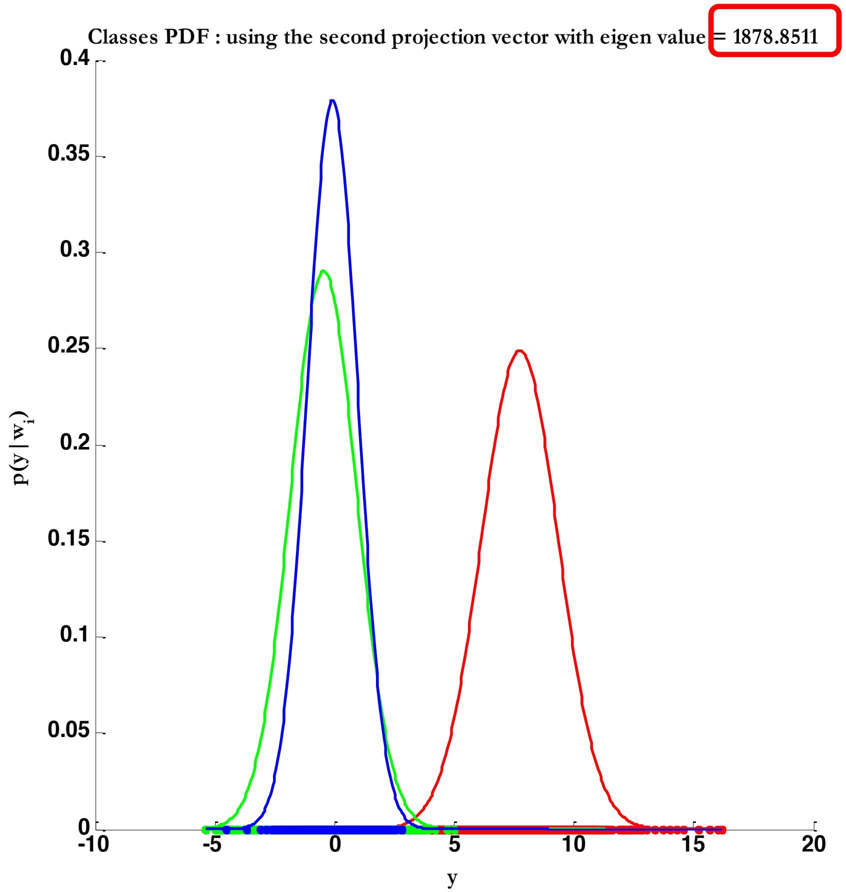
```
% project data samples along the projections axes
% the second projection vector
y1_w2 = W2'*X1;
y2_w2 = W2'*X2;
y3_w2 = W2'*X3;

% projection limits
minY = min([min(y1_w2),min(y2_w2),min(y3_w2)]);
maxY = max([max(y1_w2),max(y2_w2),max(y3_w2)]);
y_w2 = minY:0.05:maxY;

% for visualization lets compute the probability
% density function of the
% classes after projection
% the first class
y1_w2_Mu = mean(y1_w2);
y1_w2_sigma = std(y1_w2);
y1_w2_pdf = mvnpdf(y_w2',y1_w2_Mu,y1_w2_sigma);

% the second class
y2_w2_Mu = mean(y2_w2);
y2_w2_sigma = std(y2_w2);
y2_w2_pdf = mvnpdf(y_w2',y2_w2_Mu,y2_w2_sigma);

% the third class
y3_w2_Mu = mean(y3_w2);
y3_w2_sigma = std(y3_w2);
y3_w2_pdf = mvnpdf(y_w2',y3_w2_Mu,y3_w2_sigma);
```



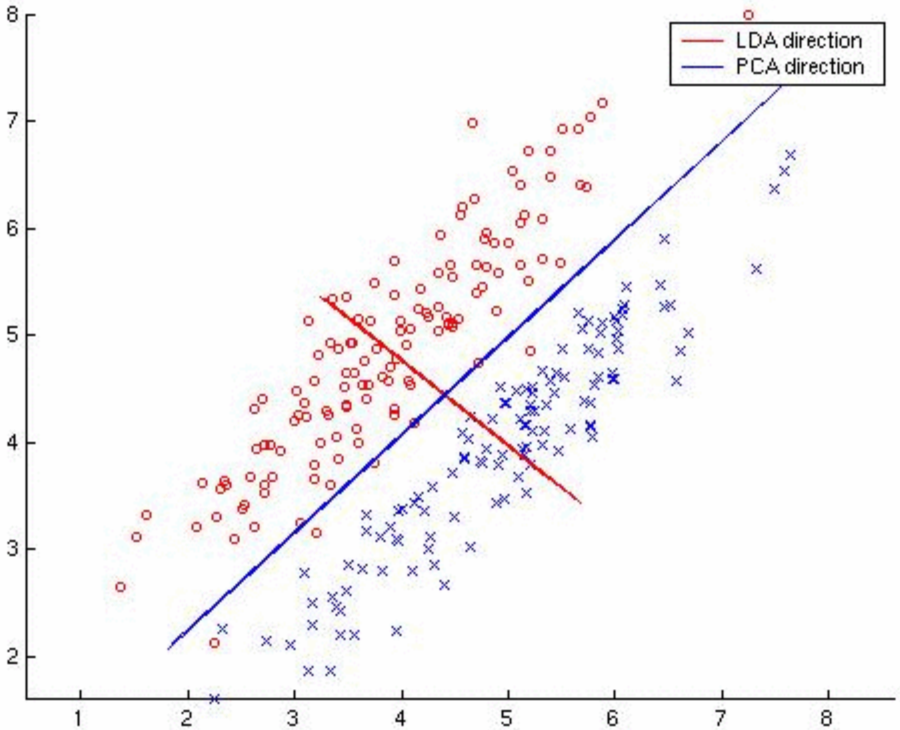
Which is Better?!!!

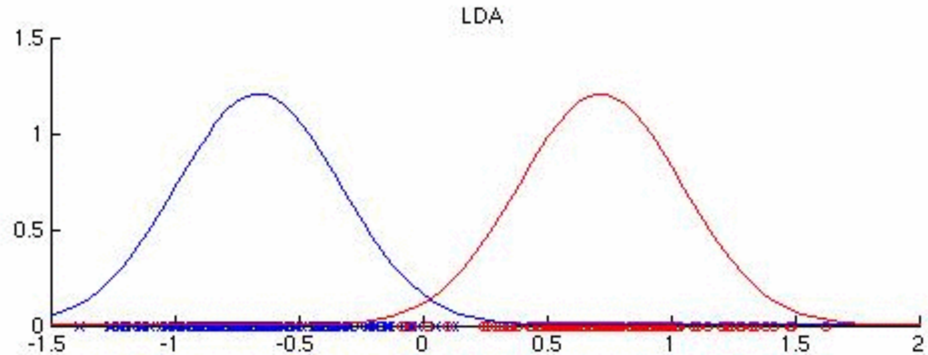
- Apparently, the projection vector that has the **highest eigen value** provides higher discrimination power between classes

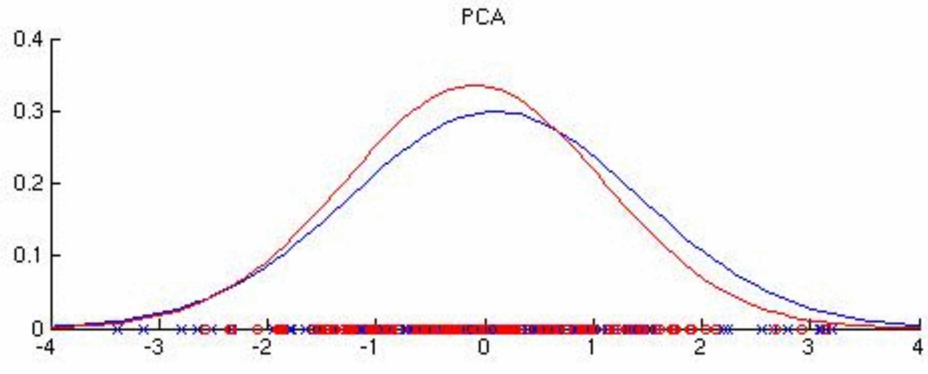




PCA vs LDA

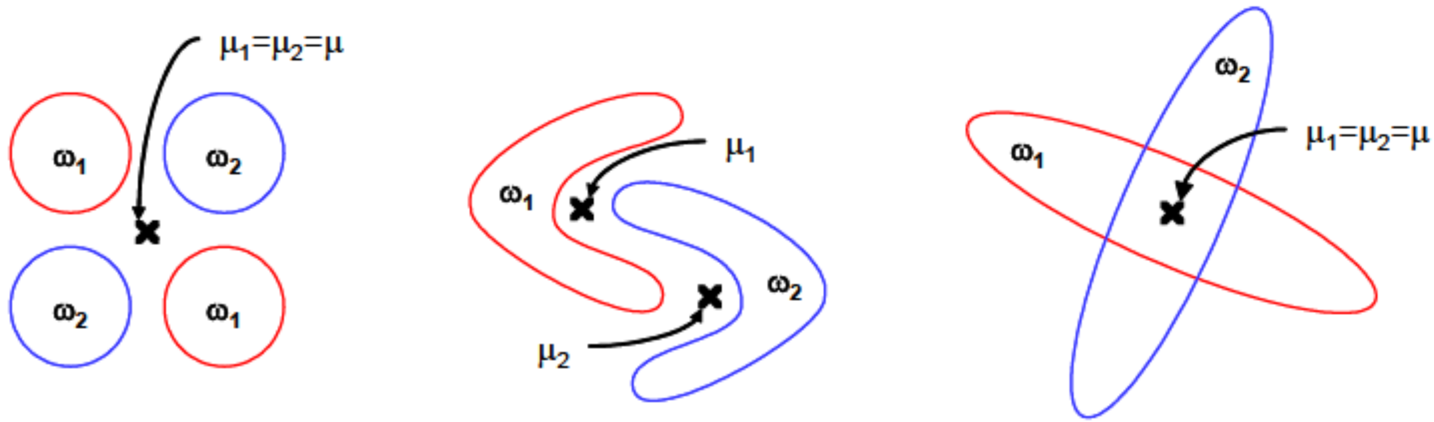






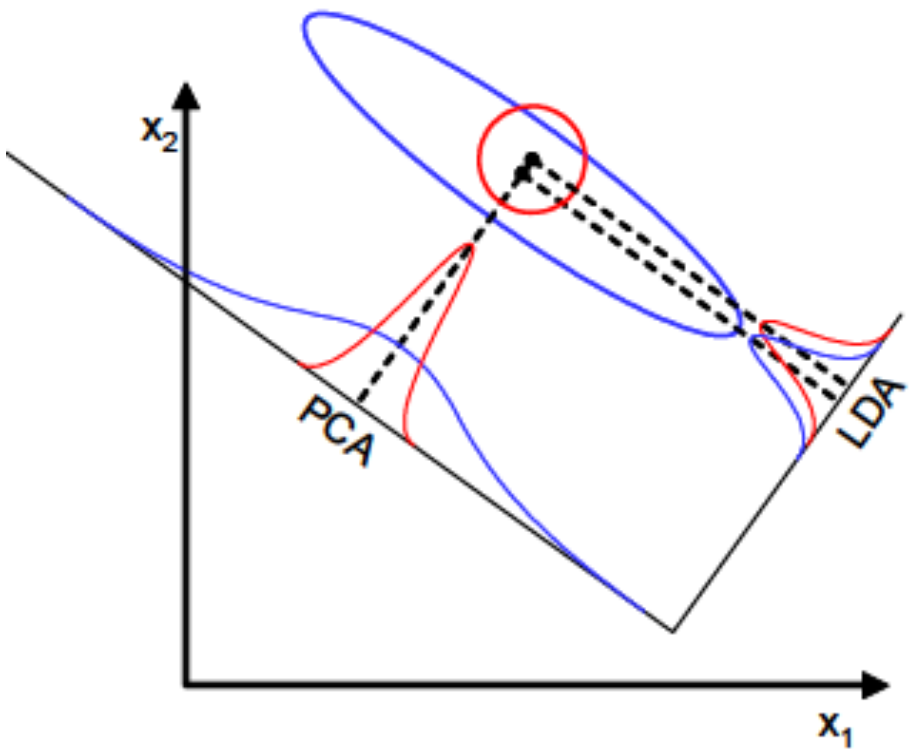
Limitations of LDA ☹

- **LDA produces at most C-1 feature projections**
  - If the classification error estimates establish that more features are needed, some other method must be employed to provide those additional features
- **LDA is a parametric method since it assumes unimodal Gaussian likelihoods**
  - If the distributions are significantly non-Gaussian, the LDA projections will not be able to preserve any complex structure of the data, which may be needed for classification.



Limitations of LDA ☹

- LDA will fail when the discriminatory information is not in the mean but rather in the variance of the data



Thank You