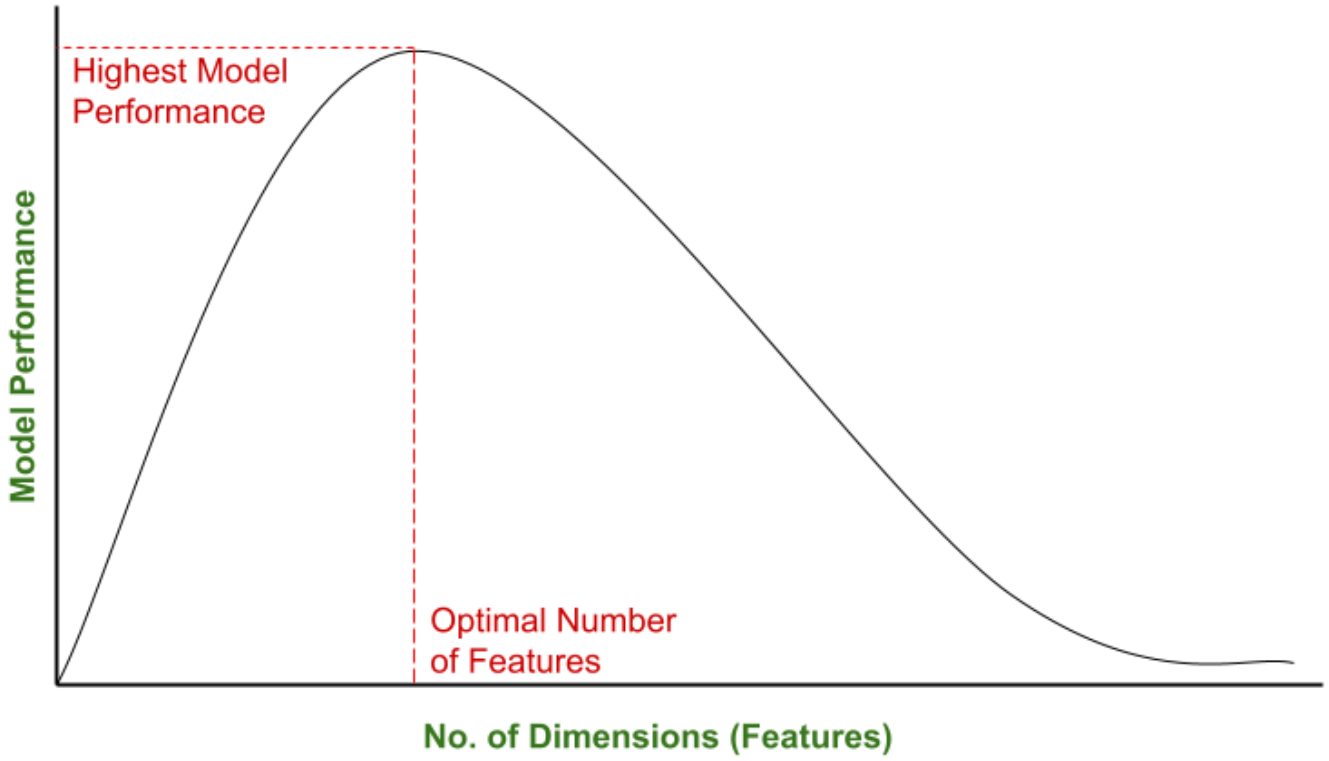
Features Dimensionality Reduction

PLAKSHA UNIVERSITY

The Curse of Dimensionality

High number of features may not always be optimal to train an ML model.



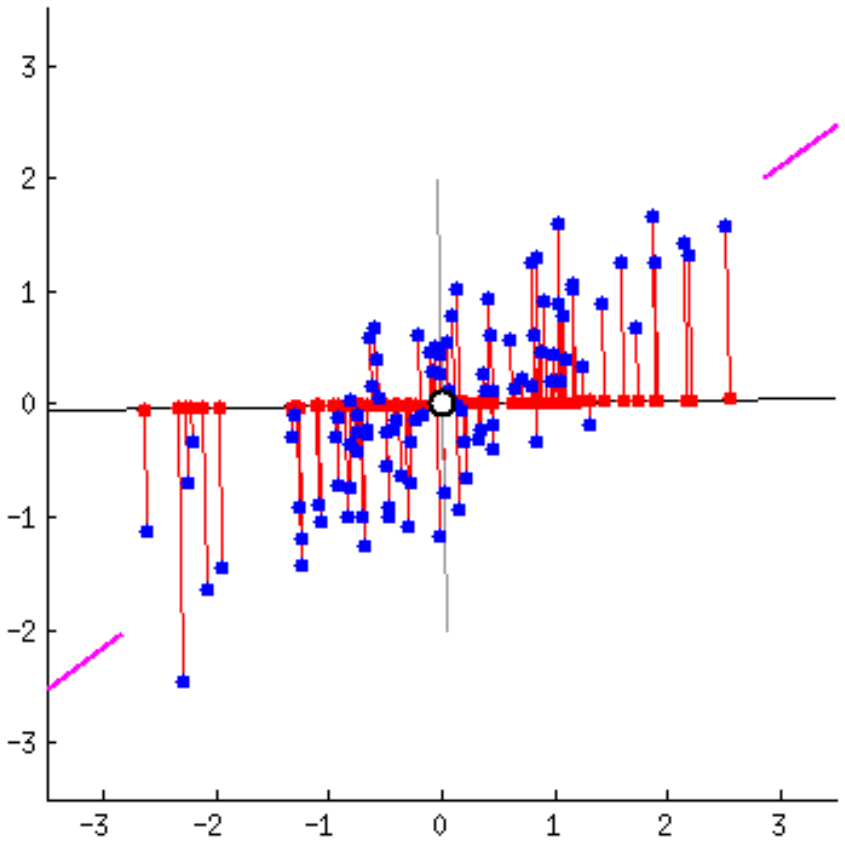
In general, the highest model performance may not be achieved by using a large number of features.

This is the curse of dimensionality.

Training a model with a large number of features is also more time-consuming and may lead to overfitting.

Thus, there is need to reduce the number of features for "optimal" model performance.

PCA - Recap



Principal components are constructed to account for the variance in features.

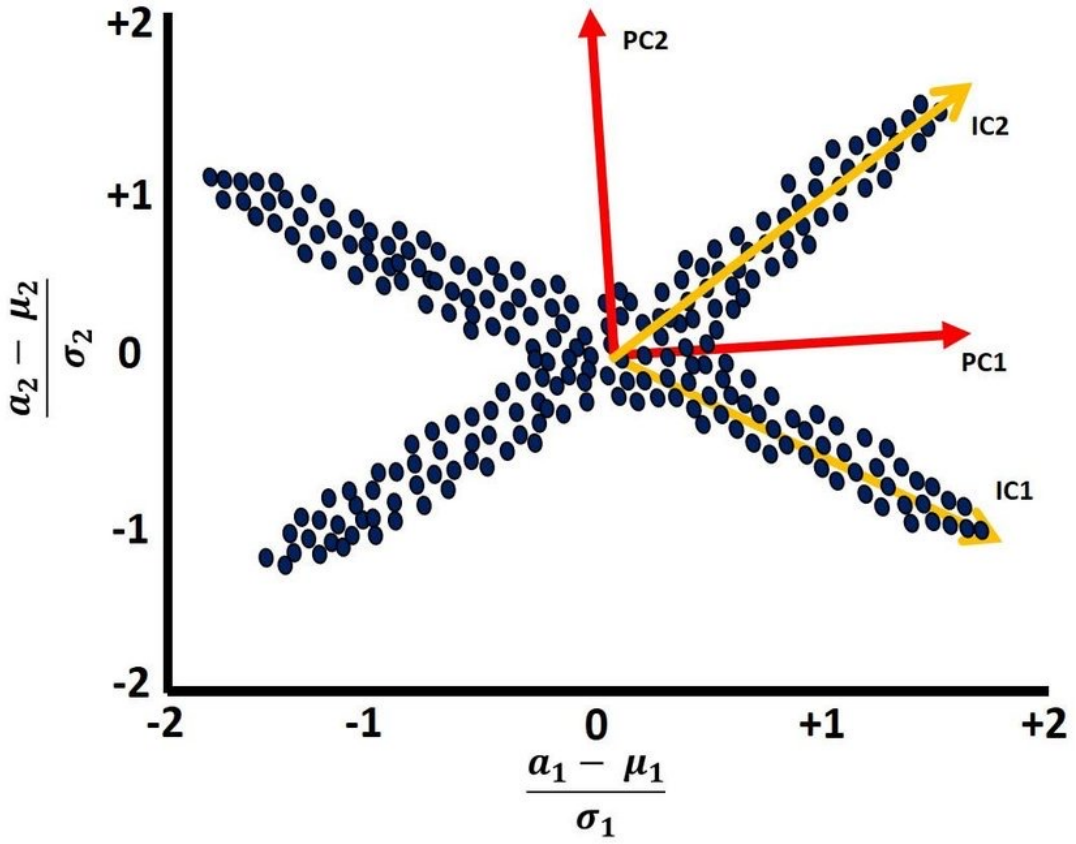
The first principal component account for the largest possible variance in the dataset.

In this graphic, the first principal component is the line that matches the purple marks.

The second principal component is calculated similarly with the condition that it is orthogonal to the first, and so on!

PCA is an unsupervised algorithm!

ICA - Recap



In PCA, we find the principal components that explain the maximum variance in the data. The PCs are orthogonal to each other.

In ICA, we find the independent components that form the data. They are non-orthogonal.

What are independent components?
If one component cannot be estimated from the other component, then it is independent.

ICA is an unsupervised algorithm!

Linear Discriminant Analysis (LDA)

- The goal of LDA is to perform features dimensionality reduction.
- Come on! That's what PCA was already doing quite well. Why do we need LDA?
- LDA does features dimensionality reduction by preserving as much of class discriminatory information as possible.
- Well, that's kind of new!

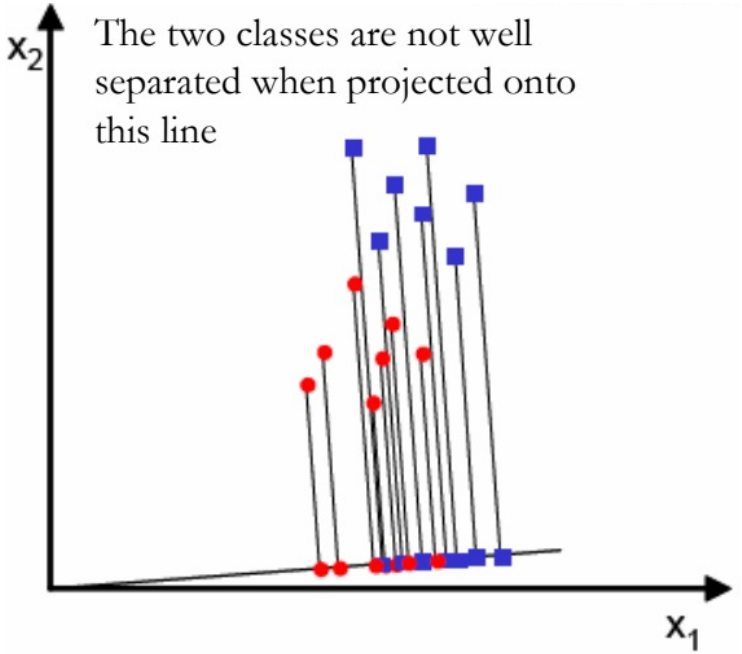
Linear Discriminant Analysis (LDA)

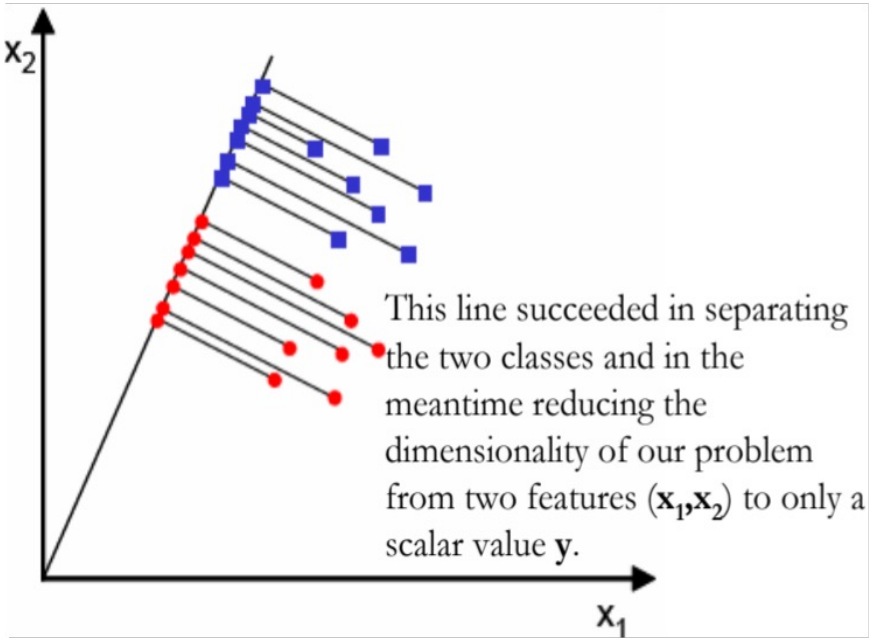
- Let's say we have a features classification problem where we have C number of classes for movies such as action, drama, comedy, thriller, etc.
- Each class has N_i samples with m number of features, where i = 1, 2, ..., C.
- Thus, we have a set of m-dimensional samples $\{x_1, x_2, ..., x_{Ni}\}$ belonging to class ω_i.
- Our matrix X (rows N_i*C, columns m) consists of these samples so that each column has features and each row has the data samples from each class.
- Our goal is to find that projection component of this dataset which allows us to discriminate between the C classes as much as possible.

LDA – Two Features and Two Classes Example

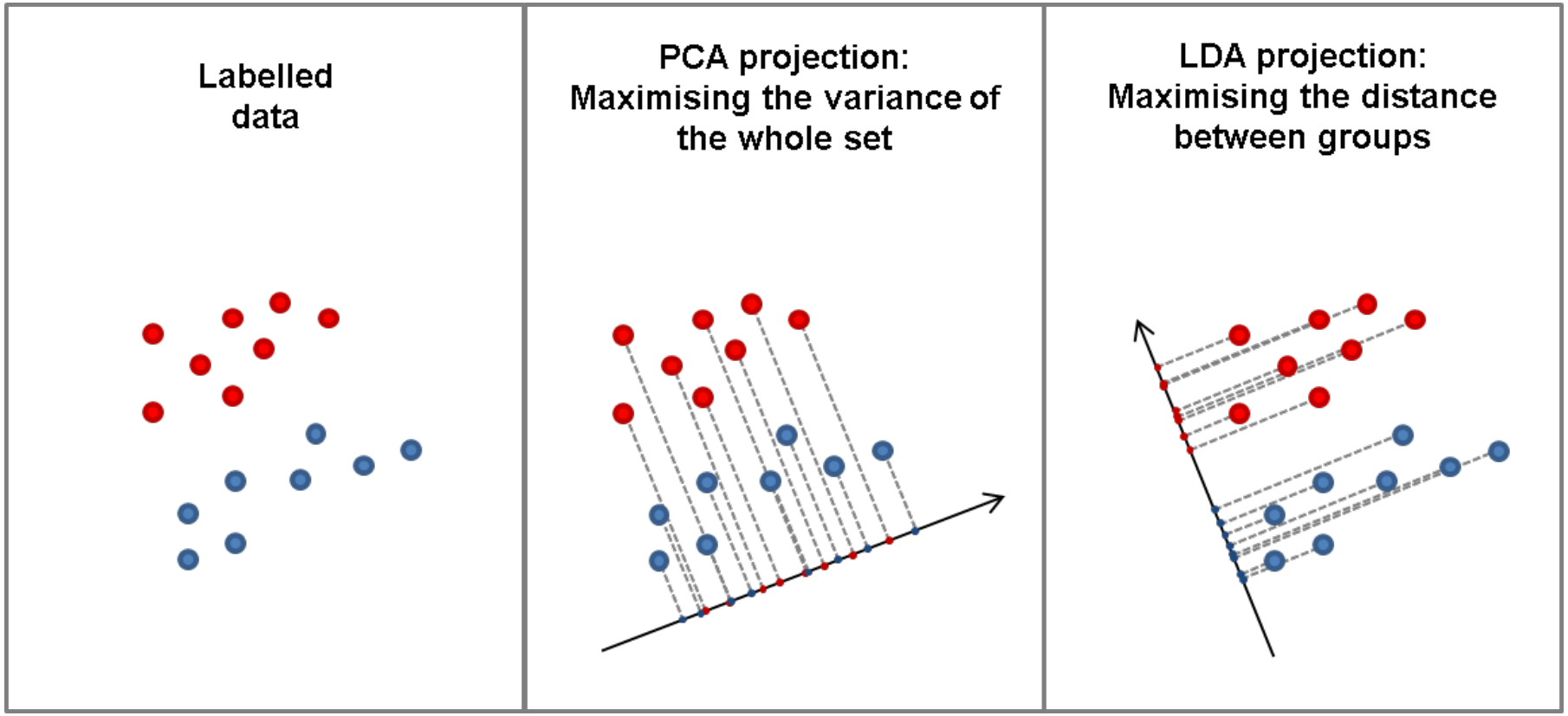
- Let's say we have m-dimensional (m=2 because two features) samples $\{x_1, x_2,..., x_N\}$, N_1 of which belong to the class ω_1 and N_2 belong to the class ω_2.

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





LDA – Two Features and Two Classes Example



Since LDA does depend on class labels, it is a supervised method!

LDA – Two Features and Two Classes Example

- Let's say we have m-dimensional samples $\{x_1, x_2, \ldots, x_N\}$, N_1 of which belong to the class ω_1 and N_2 belong to the class ω_2, m=2 because two features.

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

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

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

LDA – Two Features and Two Classes Example

- So, to find the maximum separability, we need to define what does separability mean for LDA?

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

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

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

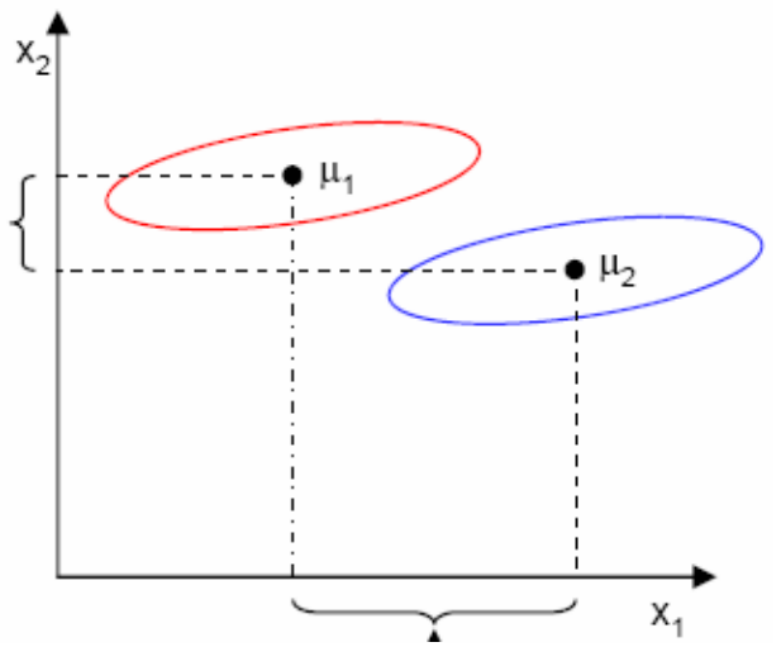
- We could then choose the distance between the projected means as our objective function.

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

LDA – Two Features and Two Classes Example

- However, the distance between the projected means may not always be a very good measure since it does not consider the variance within the classes.

This axis yields better class separability



This axis has a larger distance between means

LDA – Two Features and Two Classes Example

- One solution 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 as

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

i.e., $\tilde{s}_i^2$ measures the variability within class ω_i after projecting it on the y-space.

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

LDA – Two Features and Two Classes Example

So, taking both mean and scatter into account, we want to maximize

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

This is called the Fisher linear discriminant. It was given my Sir Ronald Fisher.

British polymath who was active as a mathematician, statistician, biologist, geneticist, and academic.

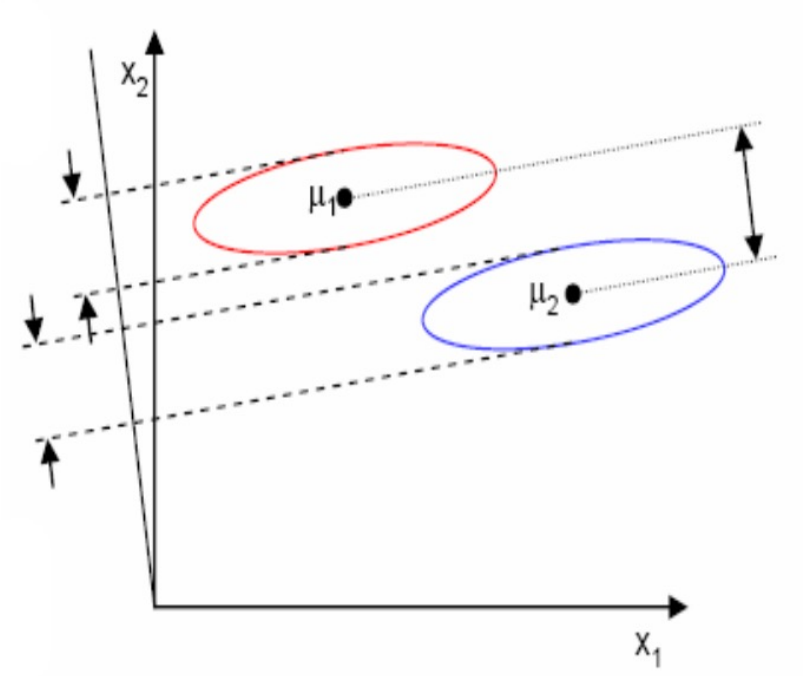
"a genius who almost single-handedly created the foundations for modern statistical science" –Hald Angers

Thus, LDA is often also called Fischer Discriminant Analysis.

LDA – Two Features and Two Classes Example

So, taking both mean and scatter into account, we want to maximize

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



Fisher Linear Discriminant allows us to look for a projection where samples from the same class are projected very close to each other and, at the same time, the projection means are as farther as possible.

LDA Algorithm

Step 1: For each class C, and for each datapoint in the class, find the means of the features.

Step 2: For each class C, compute the covariance matrix

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

and add each such covariance matrix to get the *within-class scatter* matrix.

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

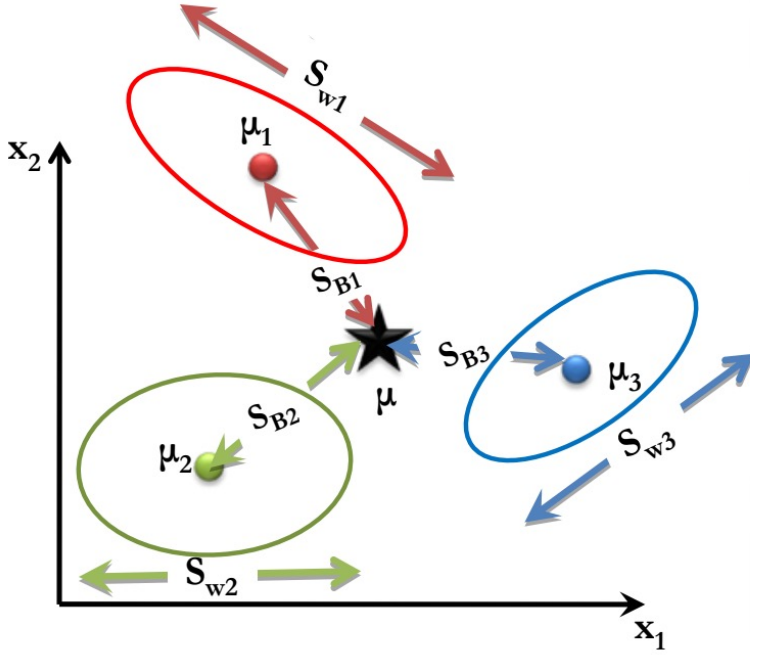
LDA Algorithm

Step 3: Compute the *between-class scatter matrix*

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

where, $\mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{\forall x} N_i \mu_i$ — **N**: number of all data

$\mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$ — $\mathbf{N_i}$: number of data samples in class $\boldsymbol{\omega_i}$.

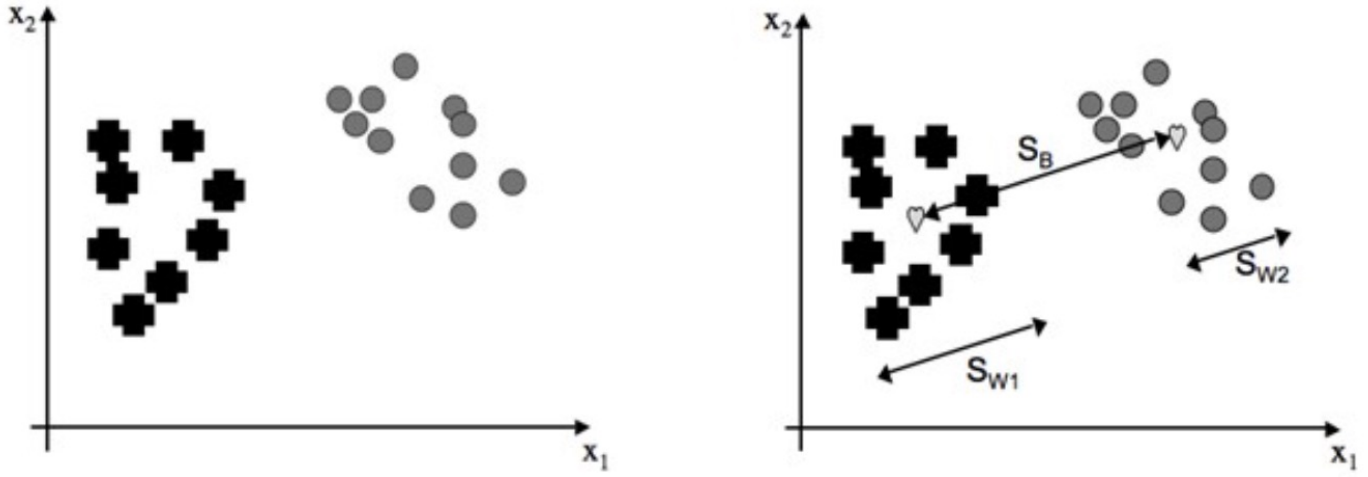


LDA Algorithm

Step 4: Maximize the between-class scatter and minimize the within-class scatter. In effect, seek the project W* that maximizes this ratio.

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

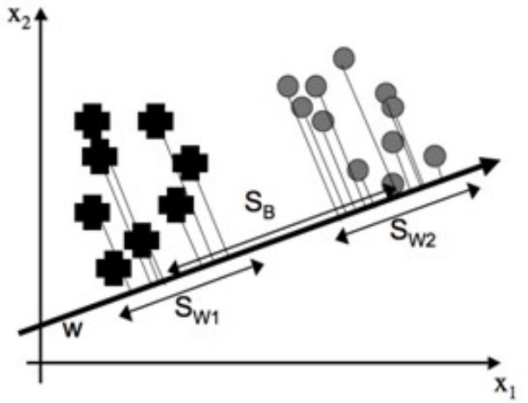
Why?



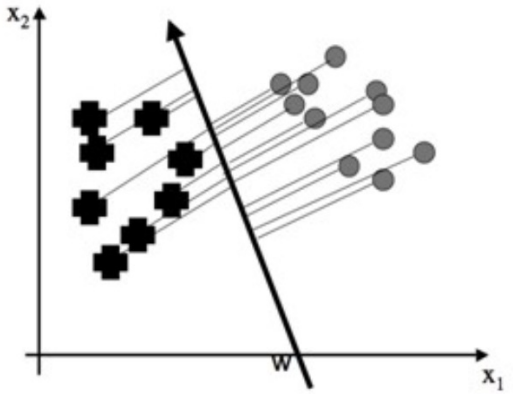
LDA Algorithm

Step 4: Maximize the between-class scatter and minimize the within-class scatter. In effect, seek the project W* that maximizes this ratio.

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



Good Projection



Bad Projection

LDA Algorithm

Step 4: Maximize the between-class scatter and minimize the within-class scatter. In effect, seek the project W* that maximizes this ratio.

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

Just like PCA, we solve the eigen value problem

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

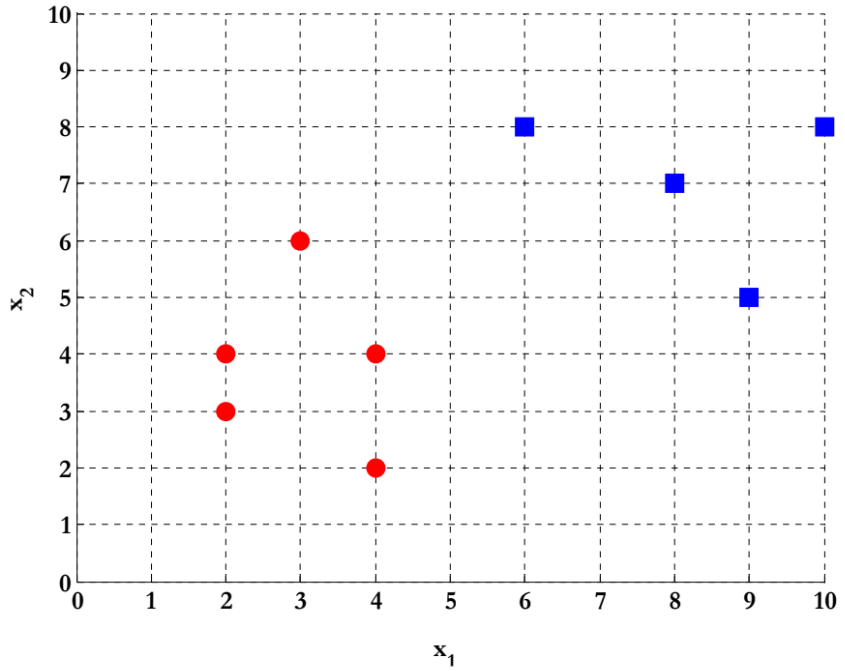
to get eigenvalues that give us the projections from corresponding w vectors.

LDA Example

Compute the Linear Discriminant projection for the following two-dimensional dataset.

Samples for class ω1 : X1=(x1,x2)={(4,2),(2,4),(2,3),(3,6),(4,4)}

Sample for class ω2 : X2=(x1,x2)={(9,10),(6,8),(9,5),(8,7),(10,8)}



LDA Example

Step 1: For each class C, and for each datapoint in the class, find the means of the features.

Samples for class ω1 : X1=(x1,x2)={(4,2),(2,4),(2,3),(3,6),(4,4)}

Sample for class ω2 : X2=(x1,x2)={(9,10),(6,8),(9,5),(8,7),(10,8)}

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

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

LDA Example

Step 2: For each class C, compute the covariance matrix and add each such covariance matrix to get the *within-class scatter* matrix.

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

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

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

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

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

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

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

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

LDA Example

Step 3: Compute the *between-class scatter matrix*.

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

$$= \left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]\left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^T$$

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

$$= \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix}$$

LDA Example

Step 4: Solve the eigen value problem and get the projections

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

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

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

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

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

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

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

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

LDA Example

Step 4: Solve the eigen value problem and get the projections

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

and

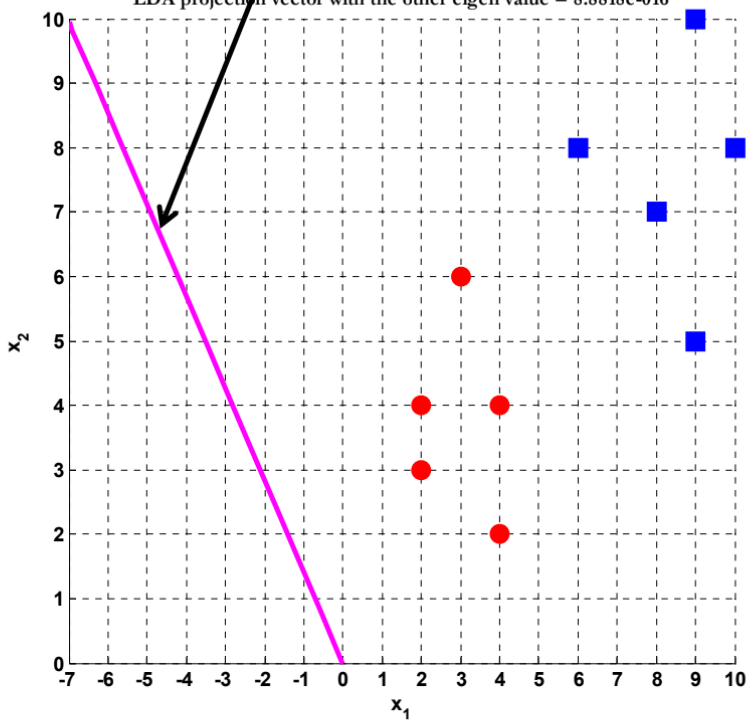
$$\begin{pmatrix} 9.2213 & 6.489 \\ 4.2339 & 2.9794 \end{pmatrix} w_2 = \underbrace{12.2007}_{\lambda_2} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

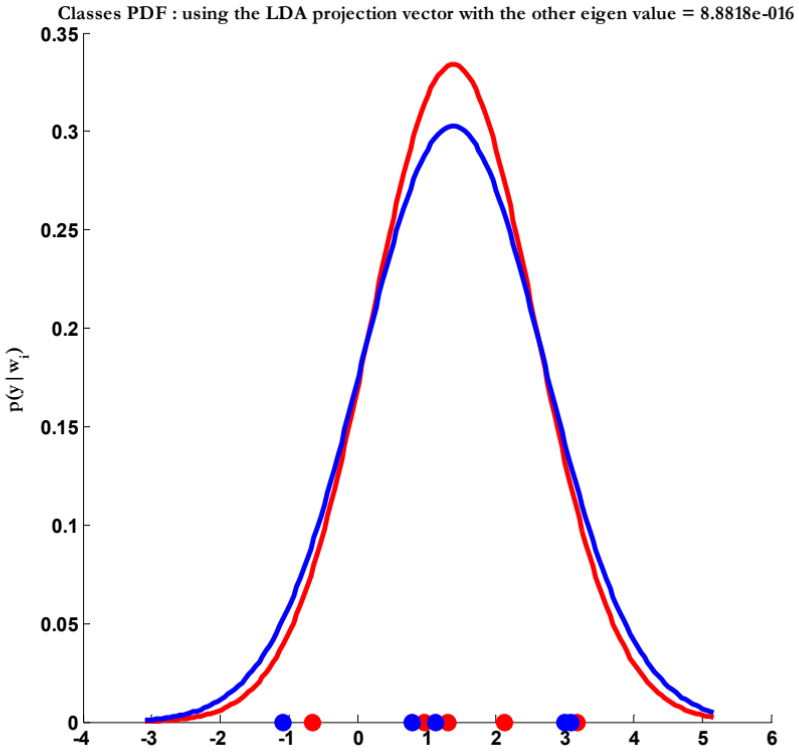
Thus;

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

LDA Example

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

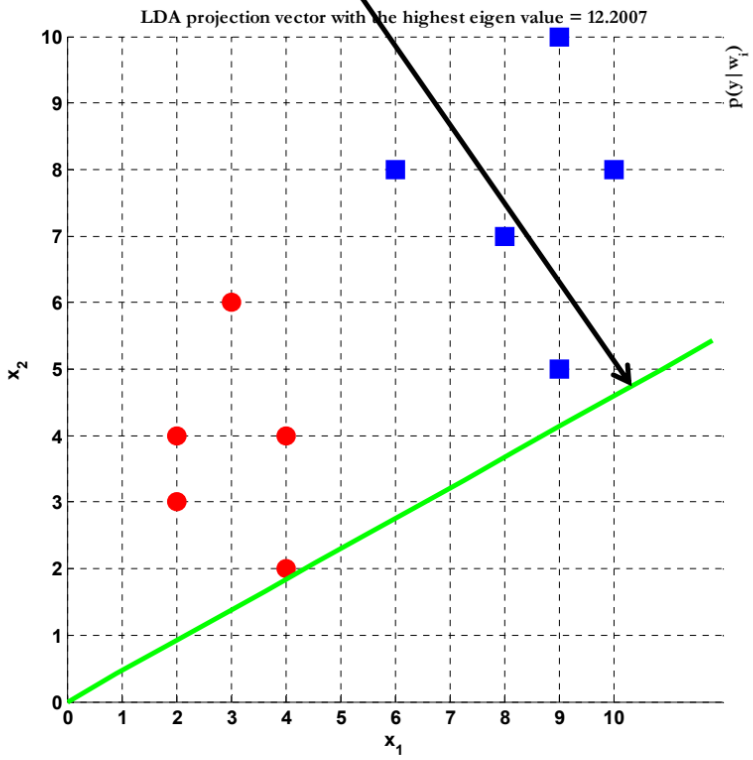


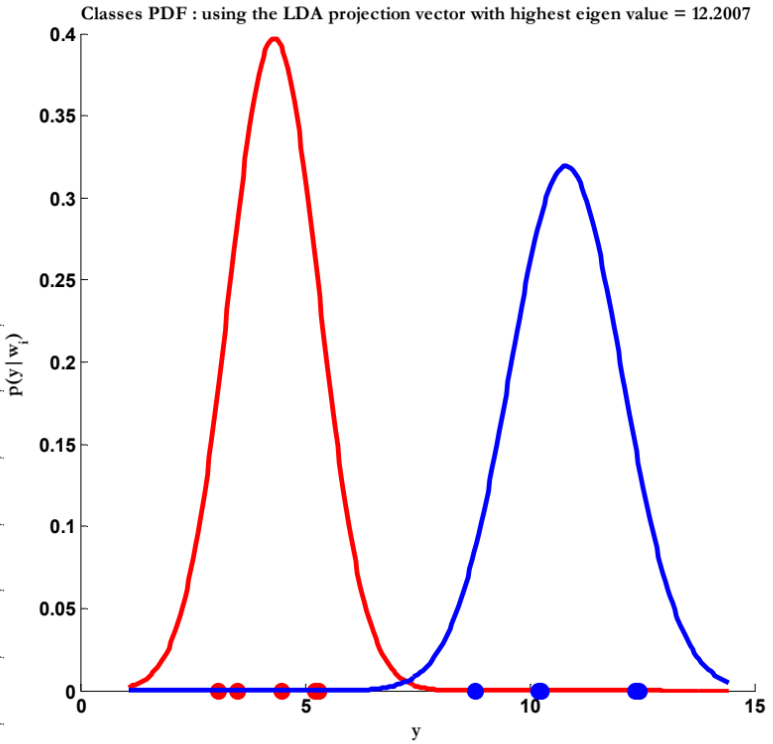


Using this vector leads to **bad separability** between the two classes

LDA Example

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





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

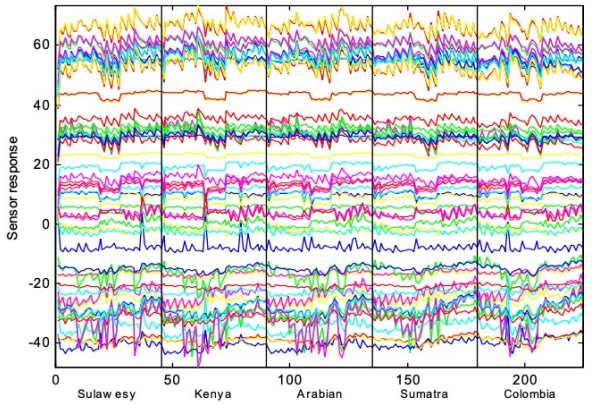
PCA vs. LDA

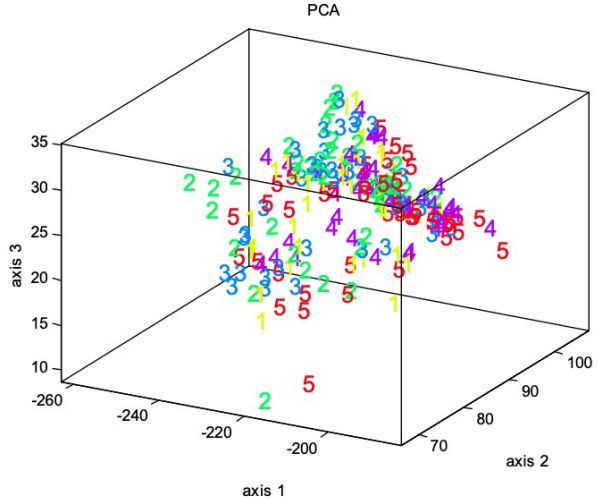
- PCA's objective is to identify the directions that capture the most variation in the data, while LDA aims at maximizing the separation between classes of data.
- Thus, PCA is an unsupervised technique while LDA is a supervised technique.
- PCA reduces dimensionality by project data onto a lower-dimensional space while LDA reduces dimensionality by creating a linear combination of features maximizing the separation between the classes.
- PCA's output are orthogonal principal components. LDA's output are discriminant functions maximizing the separation between the classes.
- PCA is mostly used for exploratory data analysis and data visualization. LDA is generally used for classification tasks.

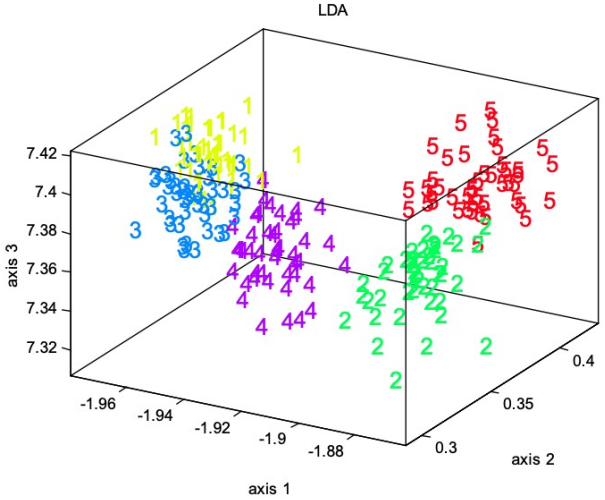
PCA vs. LDA

LDA Vs. PCA: Coffee discrimination with a gas sensor array

- **These figures show the performance of PCA and LDA on an odor recognition problem**
  - Five types of coffee beans were presented to an array of chemical gas sensors
  - For each coffee type, 45 “sniffs” were performed and the response of the gas sensor array was processed in order to obtain a 60-dimensional feature vector
- **Results**
  - From the 3D scatter plots it is clear that LDA outperforms PCA in terms of class discrimination
  - This is one example where the discriminatory information is not aligned with the direction of maximum variance

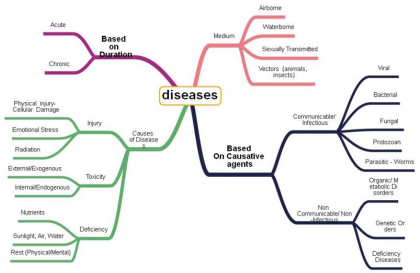






LDA Applications

Customer Identification
Select features that can specify the customers who are likely to purchase a specific product



Disease Classification on patients' data
Classifying diseases as mild, moderate, or severe using various parameters of patient health

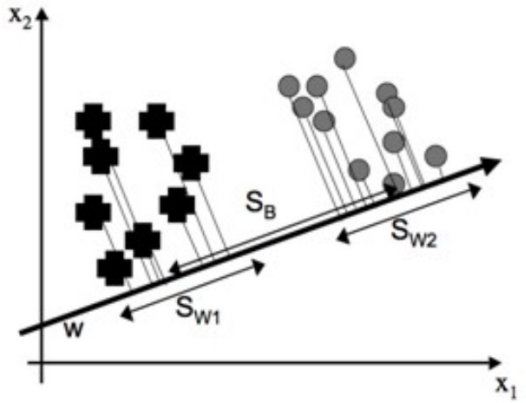
Bankruptcy Prediction
Edward Altman's 1968 model is still a leading model to predict if a bank will go under

Spam Detection
Select the "optimal" features from hundreds of features to detect spam emails

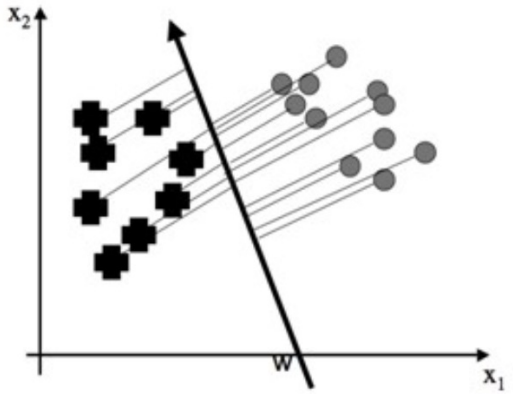
LDA Limitations

- LDA produces at most C-1 projections i.e., it reduces original features dimensionality to at most C-1 dimensions.

  Why?



Good Projection



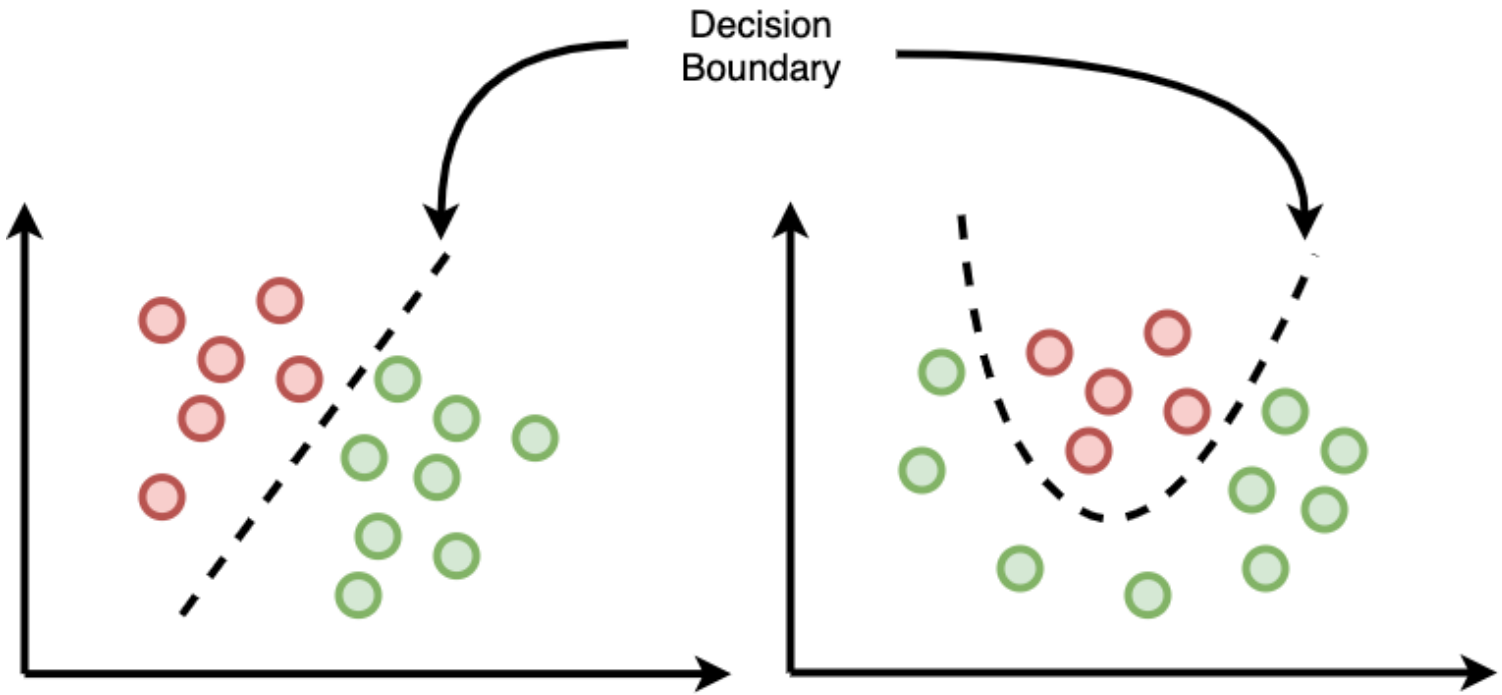
Bad Projection

The Between Class Scatter Matrix metric will always have C-1 rank at most.

Thus, if the classification error after LDA is high and more features are needed, some other method must be employed to provide those additional features.

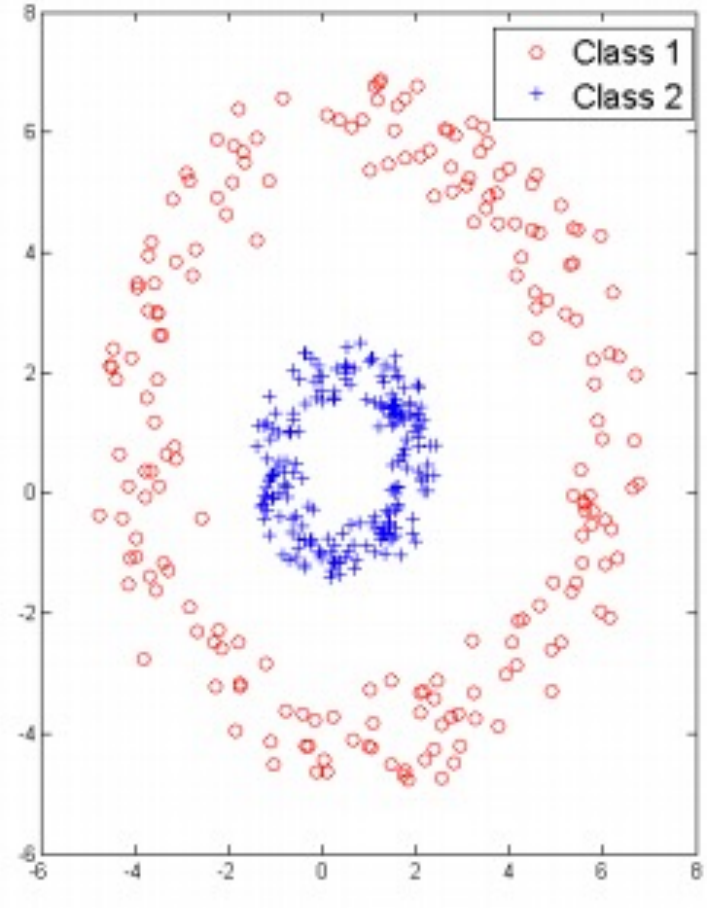
LDA Limitations

- The classes may not be linearly separable.

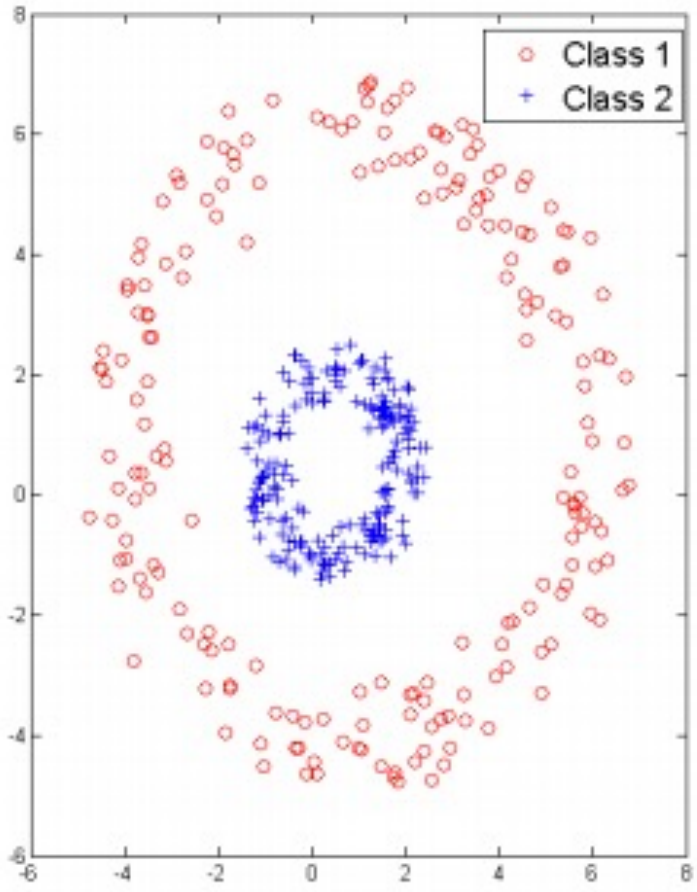


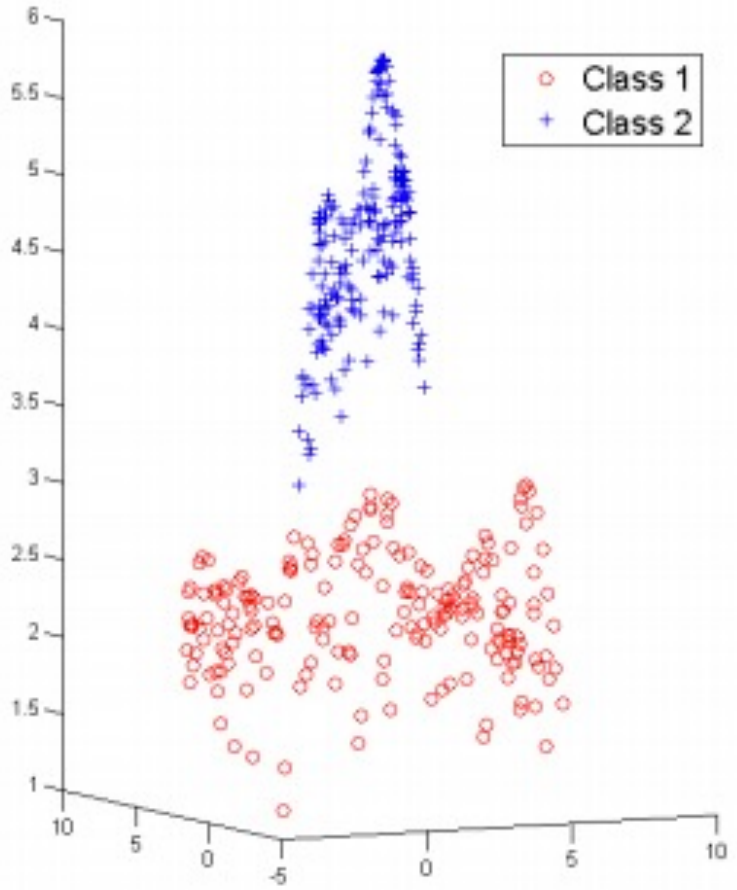
So, what should we do?

- The classes may not be linearly separable.



Kernel LDA





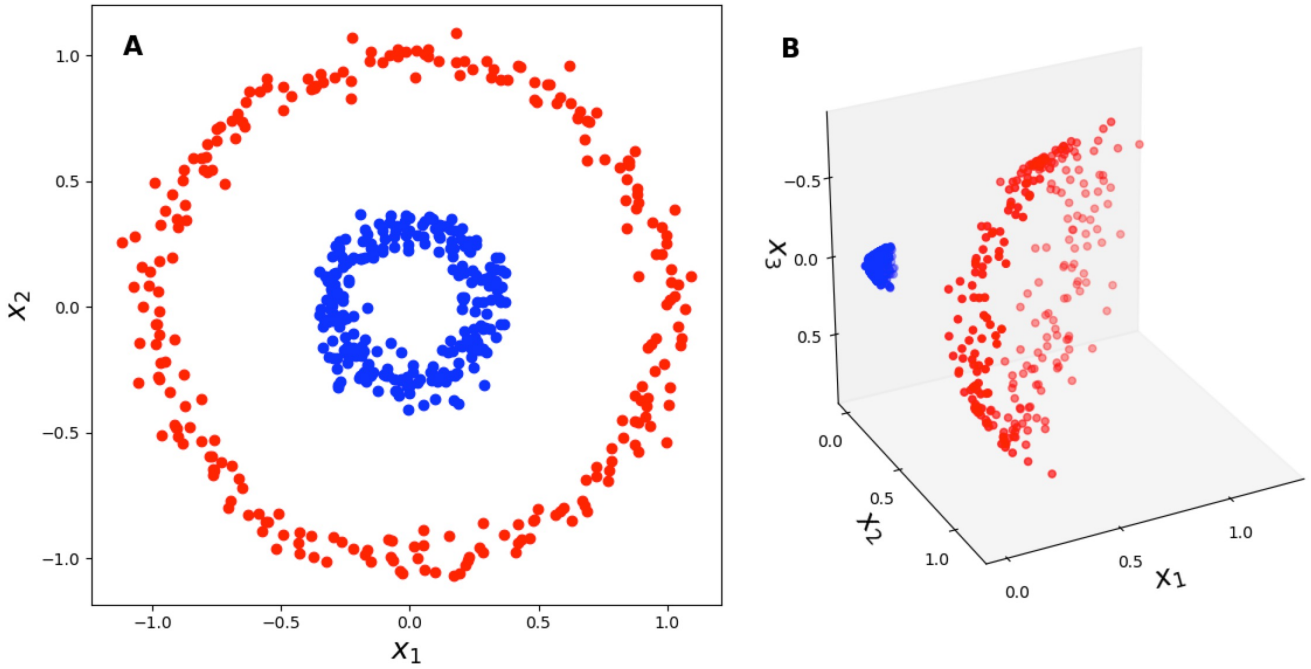
In this example, we used Radial Basis Function (RBF) kernel to project 2D data into 3D where it is easily possible to use LDA for classification.

Commonly Used Kernels Functions

- linear $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \cdot \mathbf{x}_2$
- polynomial $k(\mathbf{x}_1, \mathbf{x}_2) = (\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)^d$
- Gaussian or radial basis $k(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma \left\|\mathbf{x}_1 - \mathbf{x}_2\right\|^2\right)$
- sigmoid $k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)$

Polynomial Kernel

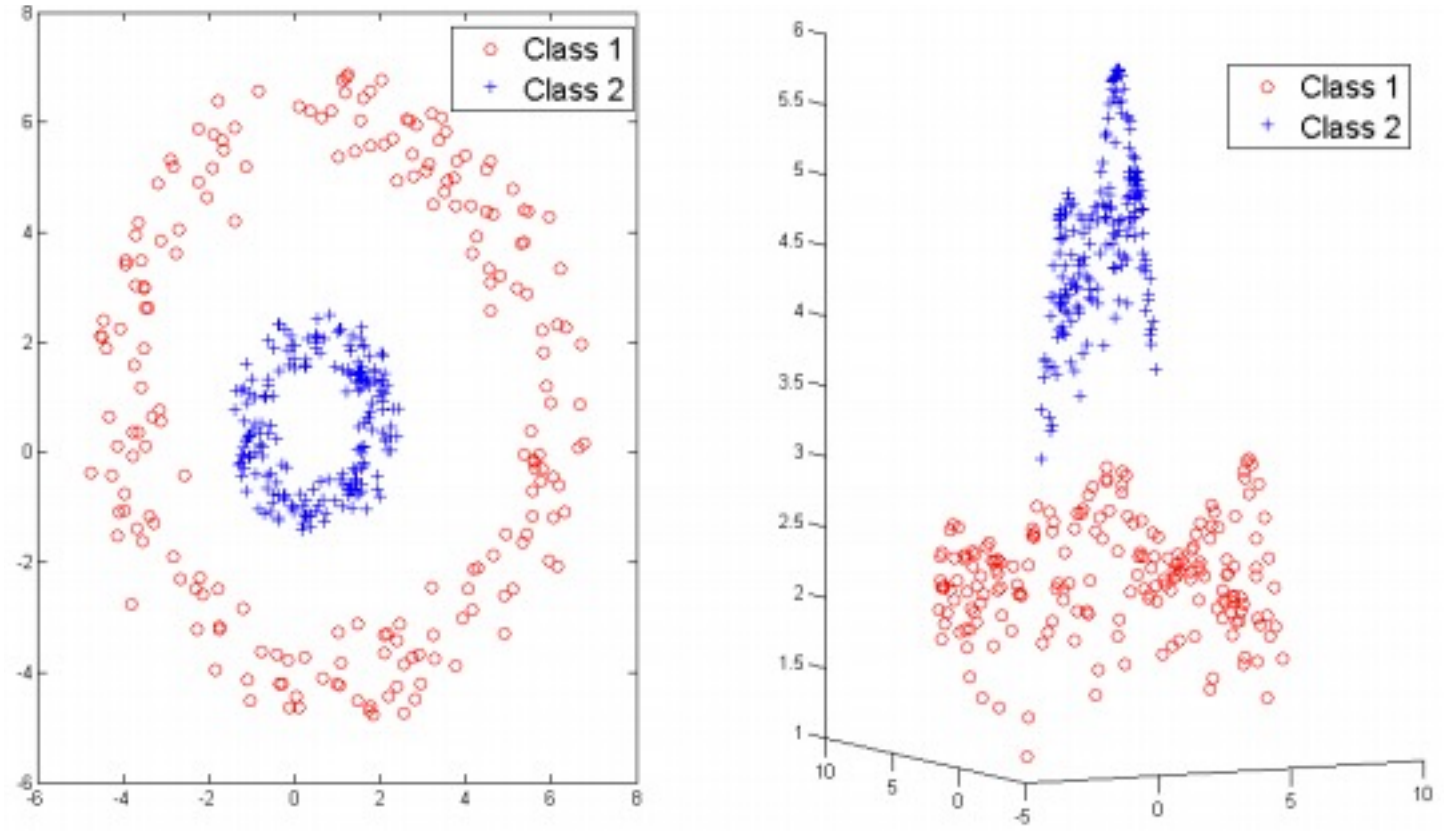
- Used when the data has polynomial features or contains interaction effects between the features.



Kernel Function: $\varphi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{bmatrix}$

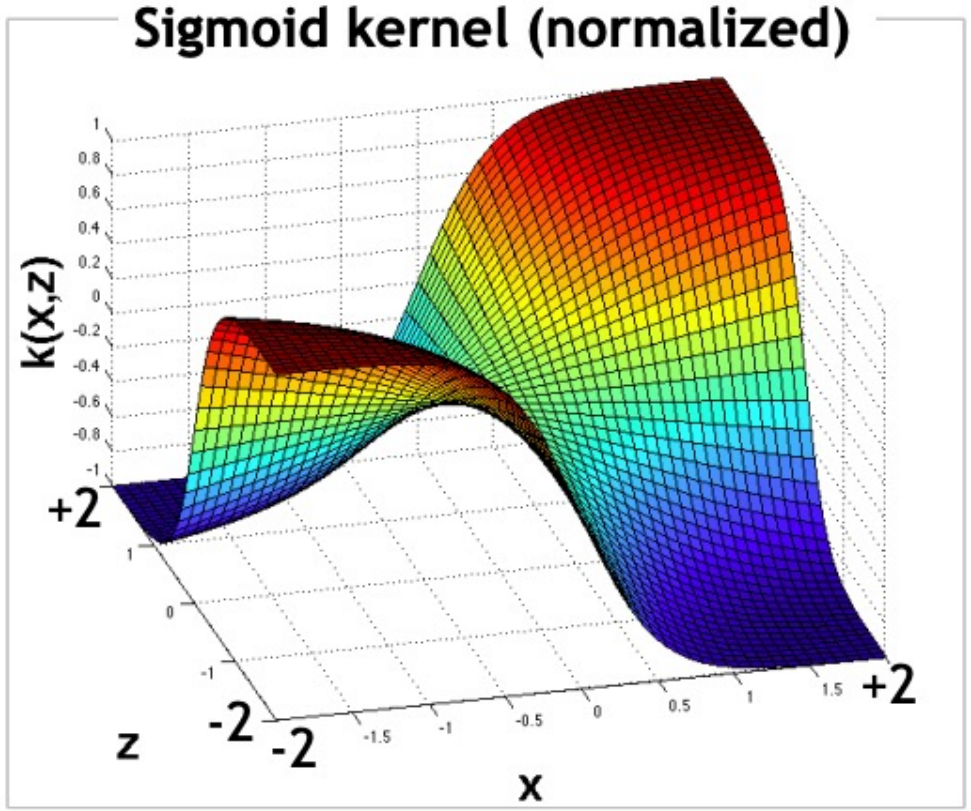
RBF Kernel

- Used if the data cannot be well-separated by a linear or polynomial decision boundaries.



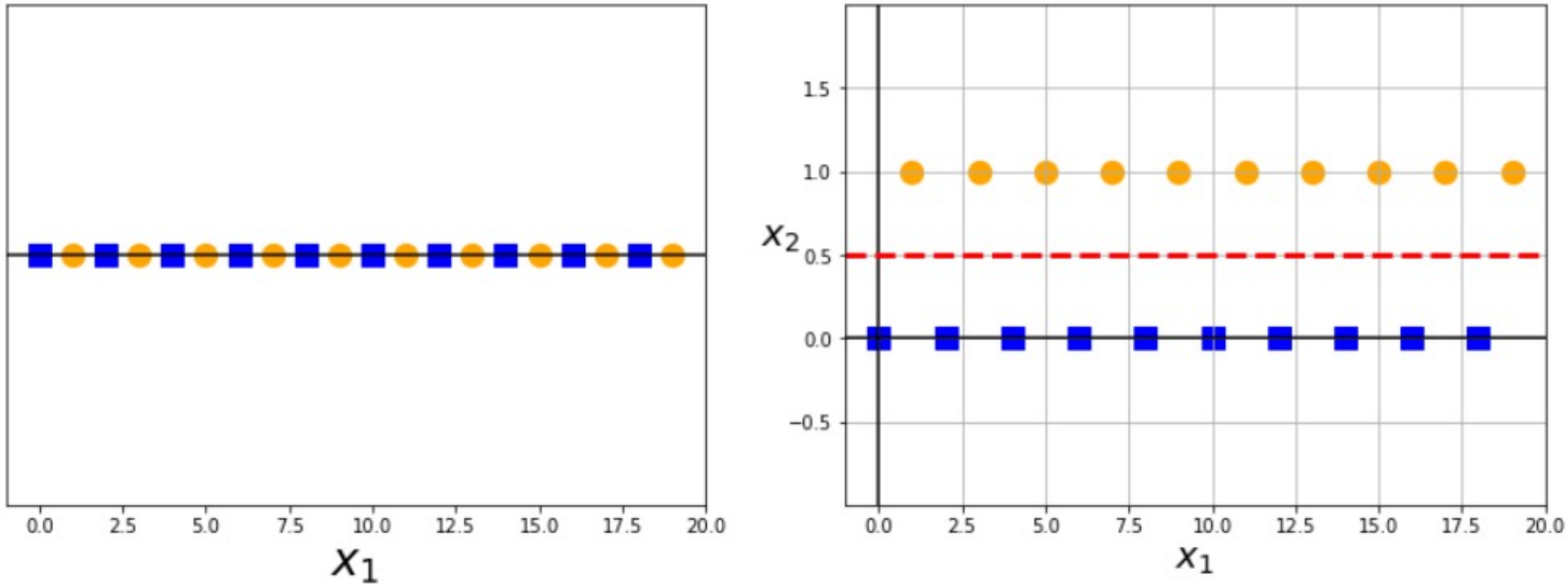
Sigmoid Kernel

- Used if the data has a sigmoidal shape or exhibits strong nonlinearities.



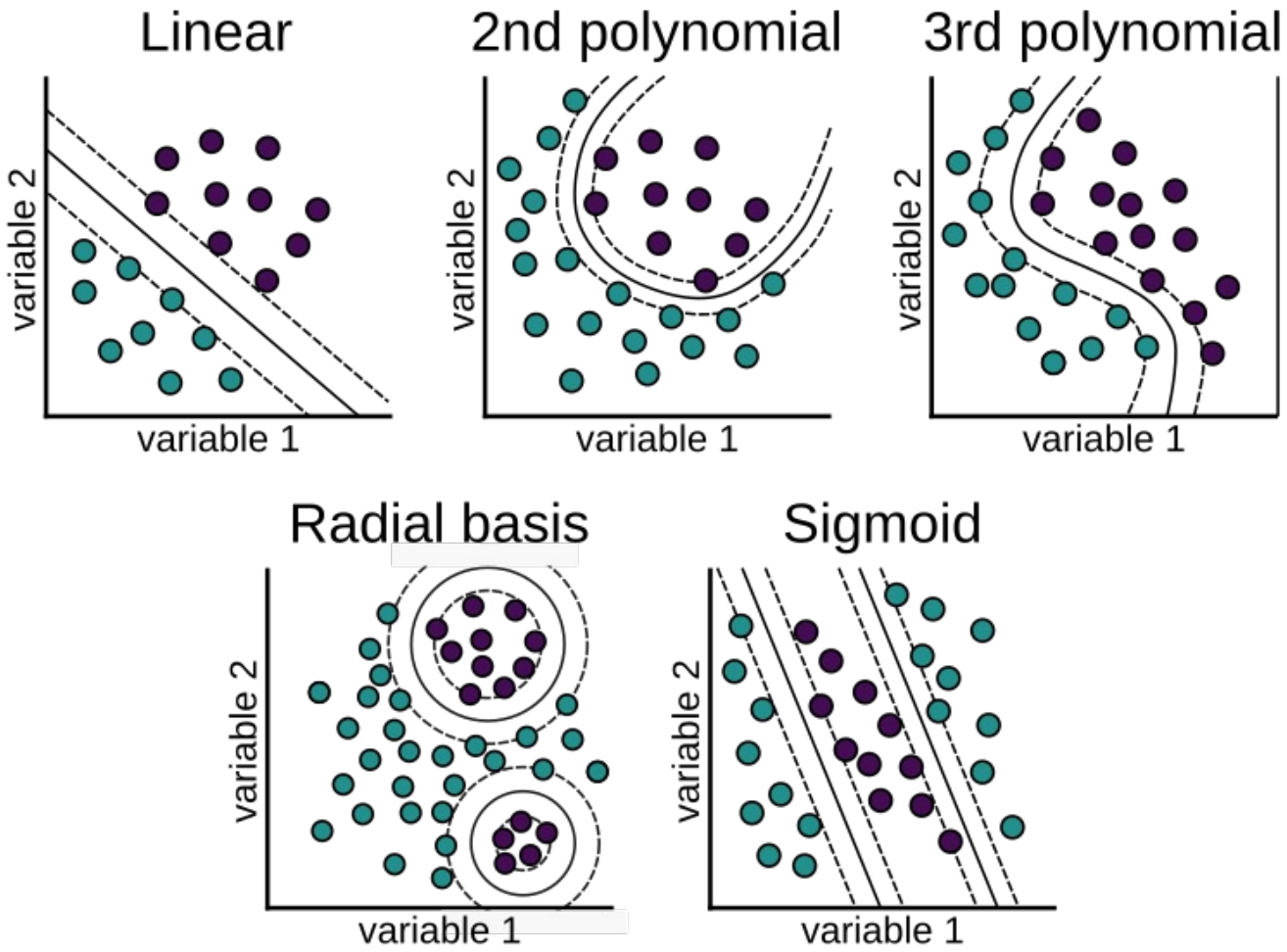
Custom Kernels

- Feel free to create your own kernels!

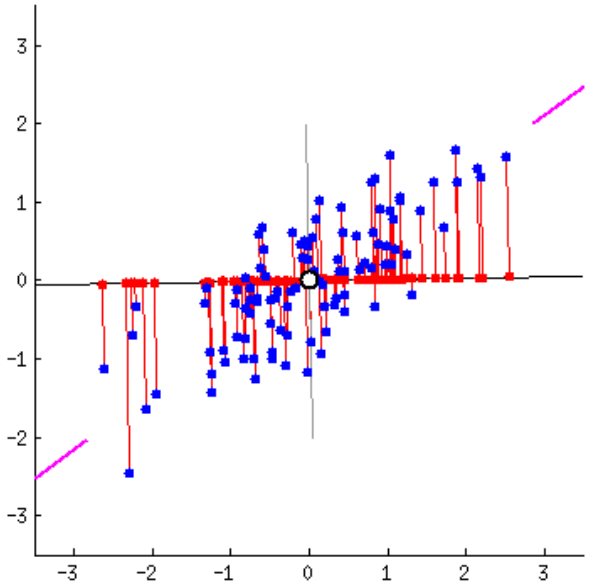


Kernel Function: $\phi(x) = \text{mod}(x,2)$

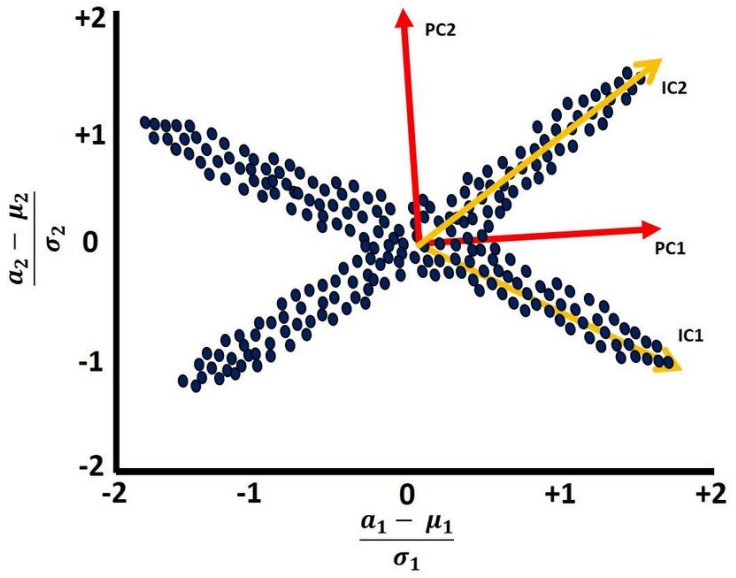
Kernel Trick is very useful in Features Classification



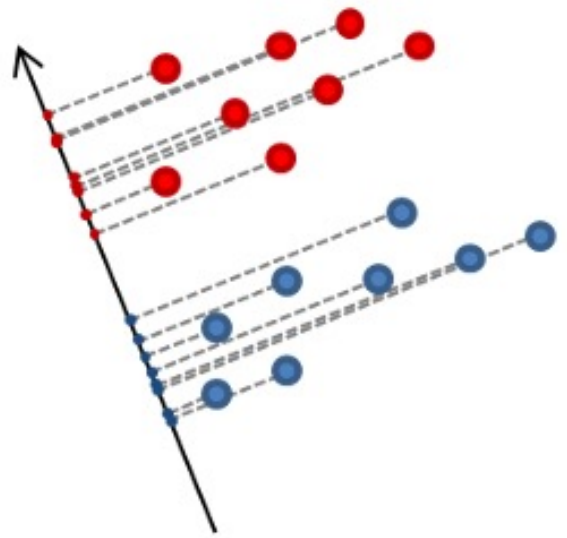
Features Dimensionality Reduction Summary



PCA
Finds orthogonal components explaining variance
Unsupervised Method



ICA
Finds original sources of information
Unsupervised Method



LDA
Finds projections maximizing distinction between classes
Supervised Method

Summary

This lecture

- LDA and its applications

- Kernel trick

Next lecture

- Features Classification using distance-based techniques

- K-Nearest Neighbors algorithm

Questions?


Intro to ML
Features Extraction
Features Clustering
Features Dim Reduction
Features Classification
ML Systems Challenges
ML Real-world Applications
Intro to NNs
Multi-layer NNs
“Deep” NNs
Other ML Methods