Dimensionality Reduction

Dimensionality reduction

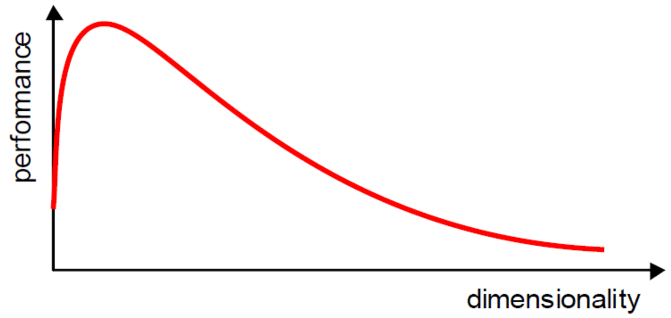
- PCA (Principal Component Analysis):
  - Find projection that maximize the variance
- ICA (Independent Component Analysis):
  - Very similar to PCA except that it assumes non-Guassian features
- Multidimensional Scaling:
  - Find projection that best preserves inter-point distances
- LDA(Linear Discriminant Analysis):
  - Maximizing the component axes for class-separation
- …
- …

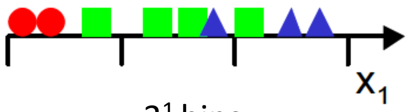
Dimensionality

- Increasing the number of features will not always improve classification accuracy.

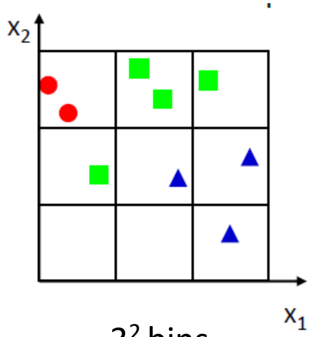
- In practice, the inclusion of more features might actually lead to worse performance.

- The number of training examples required increases **exponentially** with dimensionality **d** (i.e., $k^{\mathbf{d}}$).

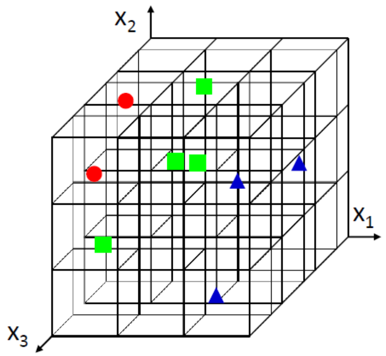




3^1 bins



3^2 bins

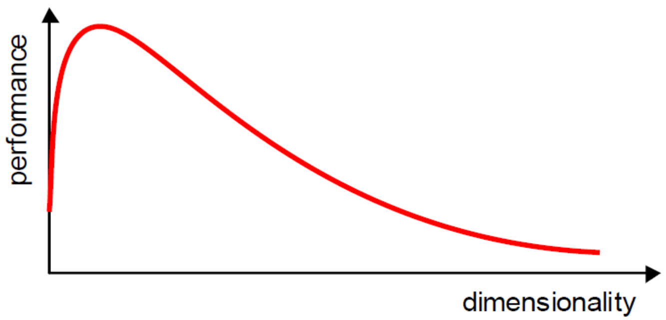


k: number of bins per feature

3^3 bins

Motivation

- Dimensionality reduction
  - A way to simplify complex high-dimensional data
  - Summarize data with a lower dimensional real valued vector



- Given data points in *d* dimensions
- Convert them to data points in *r*<*d* dimensions
- With minimal loss of information

Dimensionality Reduction (cont'd)

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features.

The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_K \end{bmatrix}$$

K<<N

Feature selection: chooses a subset of the original features.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \cdot \\ \cdot \\ x_{i_K} \end{bmatrix}$$

K<<N

Feature Extraction

- Linear combinations are particularly attractive because they are simpler to compute and analytically tractable.

- Given $x \in R^N$, find an N x K matrix U such that:

$$y = U^T x \in R^K \text{ where } K<N$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \overset{U^T}{\xrightarrow{f(\mathbf{x})}} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_K \end{bmatrix}$$

This is a projection from the N-dimensional space to a K-dimensional space.

Feature Extraction (cont'd)

- From a mathematical point of view, finding an optimum mapping $\mathbf{y}=f(\mathbf{x})$ is equivalent to optimizing an **objective** function.

- Different methods use different objective functions, e.g.,
  - Information Loss: The goal is to represent the data as accurately as possible (i.e., no loss of information) in the lower-dimensional space.
  - Discriminatory Information: The goal is to enhance the class-discriminatory information in the lower-dimensional space.

Feature Extraction (cont'd)

- Commonly used linear feature extraction methods:
  - Principal Components Analysis (PCA): Seeks a projection that **preserves** as much **information** in the data as possible.
  - Linear Discriminant Analysis (LDA): Seeks a projection that **best discriminates** the data.

- Some other interesting methods:
  - Retaining interesting directions (Projection Pursuit),
  - Making features as independent as possible (Independent Component Analysis or ICA),
  - Embedding to lower dimensional manifolds (Isomap, Locally Linear Embedding or LLE).

Eigenvector and Eigenvalue

$$Ax = \lambda x$$

A: Square Matirx

λ: Eigenvalue or characteristic value

X: Eigenvector or characteristic vector

- ***The zero vector can not be an eigenvector***
- ***The value zero can be eigenvalue***

Eigenvector and Eigenvalue

Ax = λx

A: Square Matirx

λ: Eigenvalue or characteristic value

X: Eigenvector or characteristic vector

Example

$$Show\ \ x=\begin{bmatrix}2\\1\end{bmatrix}\ is\,an\,eigenvector\ for\ \ A=\begin{bmatrix}2 & -4\\3 & -6\end{bmatrix}$$

$$Solution: Ax=\begin{bmatrix}2 & -4\\3 & -6\end{bmatrix}\begin{bmatrix}2\\1\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

$$But\quad for\quad \lambda=0,\quad \lambda x=0\begin{bmatrix}2\\1\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

$$Thus, x\,is\,an\,eigenvector\,of\ A, and\,\lambda=0\ is\,an\,eigenvalue.$$

Eigenvector and Eigenvalue

$$Ax = \lambda x \longrightarrow \begin{array}{l} Ax - \lambda x = 0 \\ (A - \lambda I)x = 0 \end{array}$$

If we define a new matrix B: $\longrightarrow$ $B = A - \lambda I$, $Bx = 0$

If B has an inverse: $\longrightarrow$ $x = B^{-1}0 = 0$ ✗ **BUT! an eigenvector cannot be zero!!**

x will be an eigenvector of A if and only if B does not have an inverse, or equivalently det(B)=0 :

$$\boxed{\det(A - \lambda I) = 0}$$

Eigenvector and Eigenvalue

Example 1: Find the eigenvalues of

$$A = \begin{bmatrix} 2 & -12 \\ 1 & -5 \end{bmatrix}$$

$$|\lambda I - A| = \begin{vmatrix} \lambda - 2 & 12 \\ -1 & \lambda + 5 \end{vmatrix} = (\lambda - 2)(\lambda + 5) + 12$$

$$= \lambda^2 + 3\lambda + 2 = (\lambda + 1)(\lambda + 2)$$

two eigenvalues: $-1, -2$

Note: The roots of the characteristic equation can be repeated. That is, $\lambda_1 = \lambda_2 = \ldots = \lambda_k$. If that happens, the eigenvalue is said to be of multiplicity k.

Example 2: Find the eigenvalues of

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

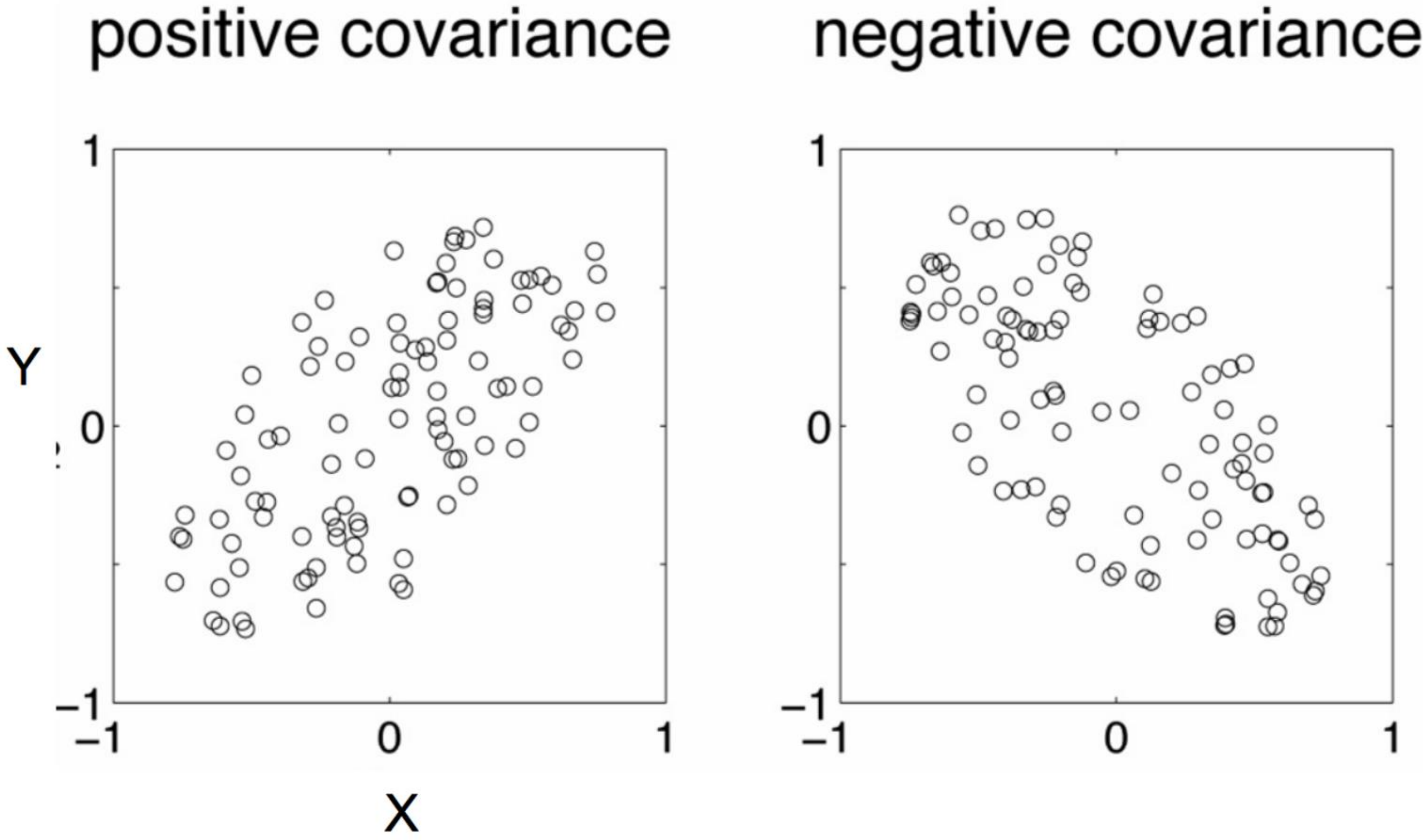
$$|\lambda I - A| = \begin{vmatrix} \lambda - 2 & -1 & 0 \\ 0 & \lambda - 2 & 0 \\ 0 & 0 & \lambda - 2 \end{vmatrix} = (\lambda - 2)^3 = 0$$

$\lambda = 2$ is an eigenvector of multiplicity 3.

Covariance

- Variance and Covariance:
  - Measure of the “spread” of a set of points around their center of mass(mean)
- Variance:
  - Measure of the deviation from the mean for points in one dimension
- Covariance:
  - Measure of how much each of the dimensions vary from the mean with **respect to each other**

- **Covariance is measured between two dimensions**
- **Covariance sees if there is a relation between two dimensions**
- **Covariance between one dimension is the variance**

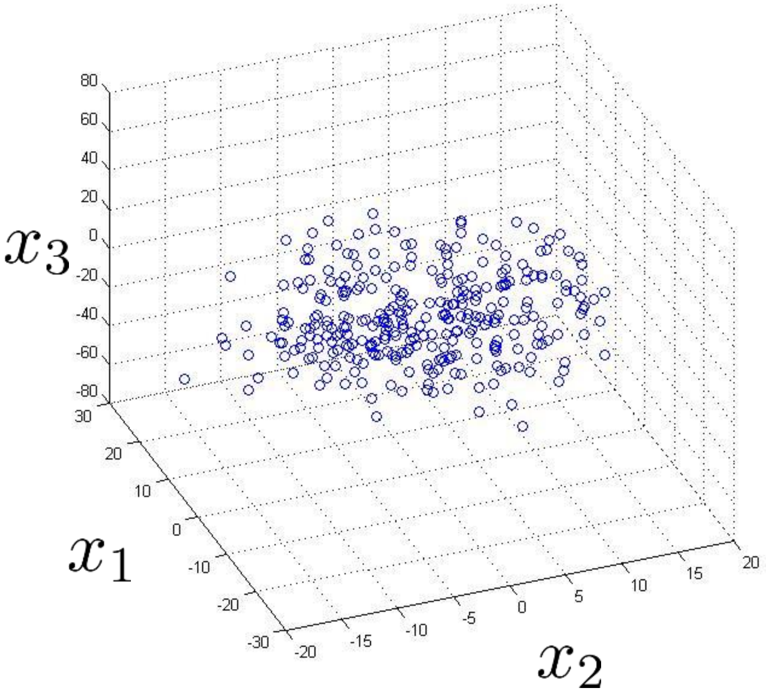


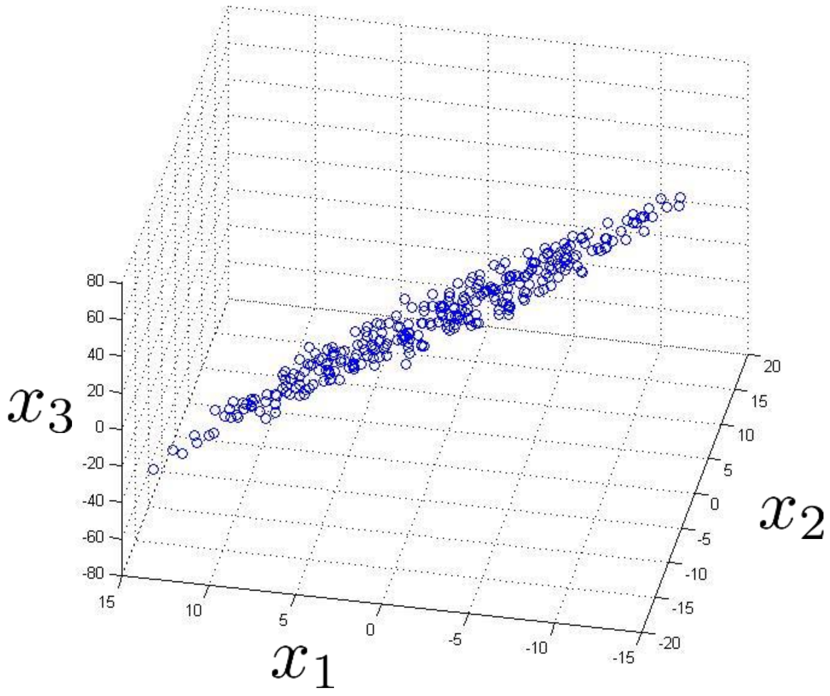
Positive: Both dimensions increase or decrease together

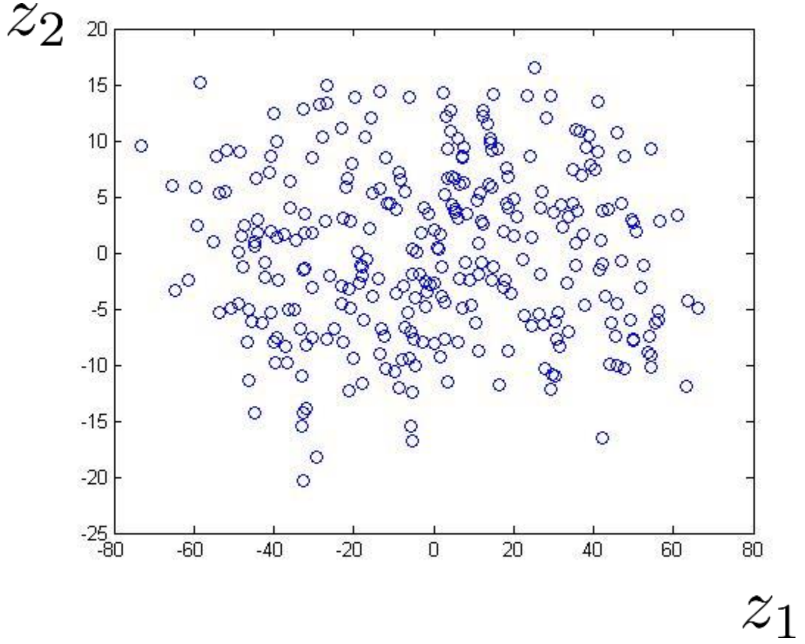
Negative: While one increase the other decrease

Data Compression

Reduce data from 3D to 2D







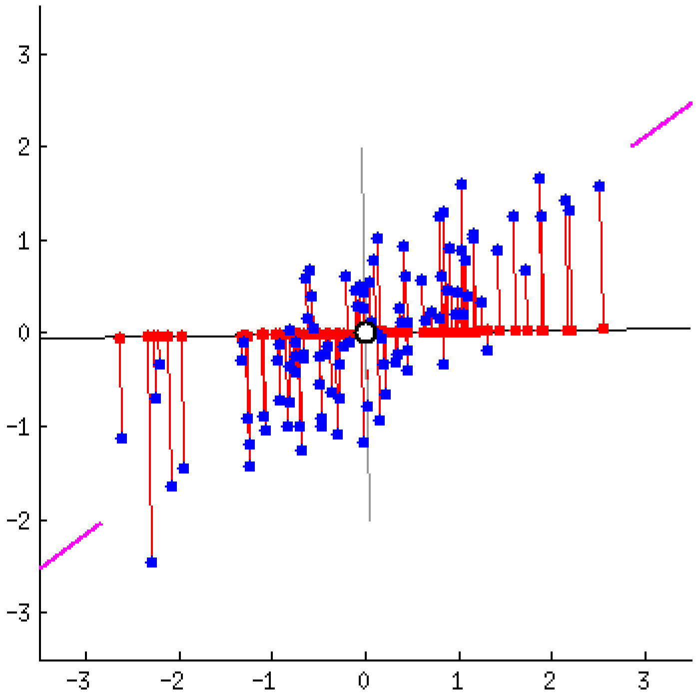
Principal Component Analysis

Goal: Find r-dim projection that best preserves variance

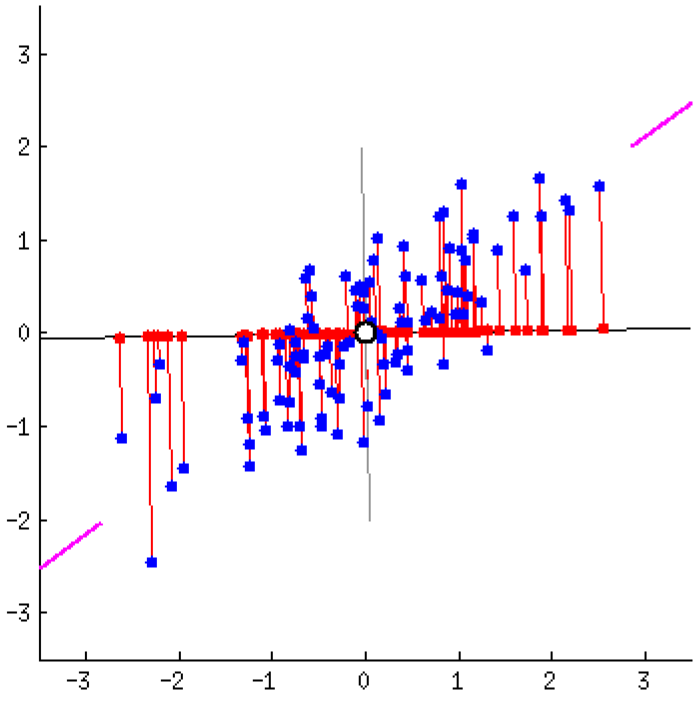
1. Compute mean vector μ and covariance matrix Σ of original points
2. Compute eigenvectors and eigenvalues of Σ
3. Select top r eigenvectors
4. Project points onto subspace spanned by them:

$$y = A(x - \mu)$$

   where y is the new point, x is the old one, and the rows of A are the eigenvectors



3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3


3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3

Principal Component Analysis (PCA)

- Let's represent $x \in R^N$ as a linear combination of an orthonormal set of N basis vectors $<v_1, v_2, \ldots, v_N>$ in R^N :

$$v_i^T v_j = \begin{cases} 1 & if \;\; i = j \\ 0 & otherwise \end{cases} \qquad \mathbf{x} = \sum_{i=1}^{N} x_i v_i = x_1 v_1 + x_2 v_2 + \ldots + x_N v_N$$

$$where \quad x_i = \frac{\mathbf{x}^T v_i}{v_i^T v_i} = \mathbf{x}^T v_i$$

- PCA seeks to represent x in a new space of lower dimensionality using only K basis vectors (K < N)

$$\hat{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K$$

such that $\| \mathbf{x} - \hat{\mathbf{x}} \|$ is minimized

(i.e., minimize information loss)

Principal Component Analysis (PCA)

- How should we determine the "optimal" lower dimensional space basis vectors <$u_1, u_2, \ldots, u_K$> ?

The optimal space of lower dimensionality can be defined by:

(1) Finding the eigenvectors u_i of the covariance matrix of the data Σ_x

$$\Sigma_x u_i = \lambda_i u_i$$

(2) Choosing the K "largest" eigenvectors u_i (i.e., corresponding to the K "largest" eigenvalues λ_i)

We refer to "largest" eigenvectors u_i as principal components.

PCA - Steps

- Suppose we are given $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$ N x 1 vectors

Step 1: compute sample mean

$$\bar{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i$$

Step 2: subtract sample mean (i.e., center data at zero)

$$\Phi_i = \mathbf{x}_i - \bar{\mathbf{x}}$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

Step 3: compute the sample covariance matrix Σ_x

$$\Sigma_{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T = \frac{1}{M}AA^T$$ where $A=[\Phi_1\ \Phi_2\ \ldots\ \Phi_M]$

(N x M matrix)

PCA - Steps

Step 4: compute the eigenvalues/eigenvectors of Σ_x

$$\Sigma_x u_i = \lambda_i u_i$$

we assume $\lambda_1 > \lambda_2 > \ldots > \lambda_N \quad and \quad u_1, u_2, \ldots, u_N$ are the corresponding eigenvectors

Note : most software packages return the eigenvalues (and corresponding eigenvectors) is decreasing order – if not, you need to do it yourself)

Since Σ_x is symmetric, <u_1,u_2,…,u_N> form an orthogonal basis in R^N, therefore:

$$\mathbf{x} - \bar{\mathbf{x}} = \sum_{i=1}^{N} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_N u_N$$

$$y_i = \frac{(\mathbf{x} - \bar{\mathbf{x}})^T u_i}{u_i^T u_i} = (\mathbf{x} - \bar{\mathbf{x}})^T u_i \qquad if \ \ \| u_i \| = 1$$

Note : most software packages normalize u_i to unit length to simplify calculations; if not, you need to do it yourself):

PCA - Steps

Step 5: dimensionality reduction step – approximate **x** using only the first K eigenvectors (K<N) (i.e., corresponding to the K largest eigenvalues where K is a parameter):

$$\mathbf{x} - \bar{\mathbf{x}} = \sum_{i=1}^{N} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_N u_N$$

approximate using first K terms

$$\hat{\mathbf{x}} - \bar{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K$$

or

$$(\hat{\mathbf{x}} - \bar{\mathbf{x}}) = U \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

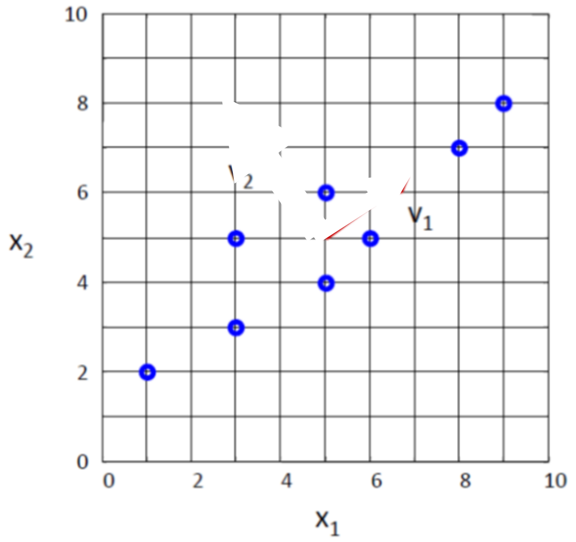
$$where \;\; U = [u_1\, u_2 \ldots u_K] \quad N\, x\, K$$

i.e., the columns of U are the the first K eigenvectors of Σ_x

Example

- Compute the PCA for dataset

(1,2),(3,3),(3,5),(5,4),(5,6),(6,5),(8,7),(9,8)



- Compute the sample covariance matrix is:

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

$$\Sigma_x = \begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}$$

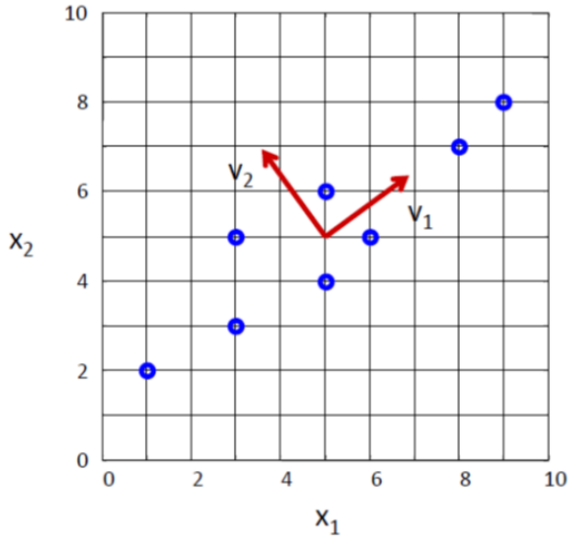
- The eigenvalues can be computed by finding the roots of the characteristic polynomial:

$$\Sigma_x v = \lambda v \Rightarrow |\Sigma_x - \lambda I| = 0$$
$$\Rightarrow \begin{vmatrix} 6.25 - \lambda & 4.25 \\ 4.25 & 3.5 - \lambda \end{vmatrix} = 0$$
$$\Rightarrow \boldsymbol{\lambda_1 = 9.34; \lambda_2 = 0.41}$$

Example (cont'd)

- The eigenvectors are the solutions of the systems:

$$\Sigma_{\mathbf{x}} u_i = \lambda_i u_i$$



$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} \lambda_1 v_{11} \\ \lambda_1 v_{12} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} 0.81 \\ 0.59 \end{bmatrix}$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} \lambda_2 v_{21} \\ \lambda_2 v_{22} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} -0.59 \\ 0.81 \end{bmatrix}$$

- Normalize the eigenvectors vectors to unit-length.

Note: if u_i is a solution, then (cu_i) is also a solution where c is any constant.

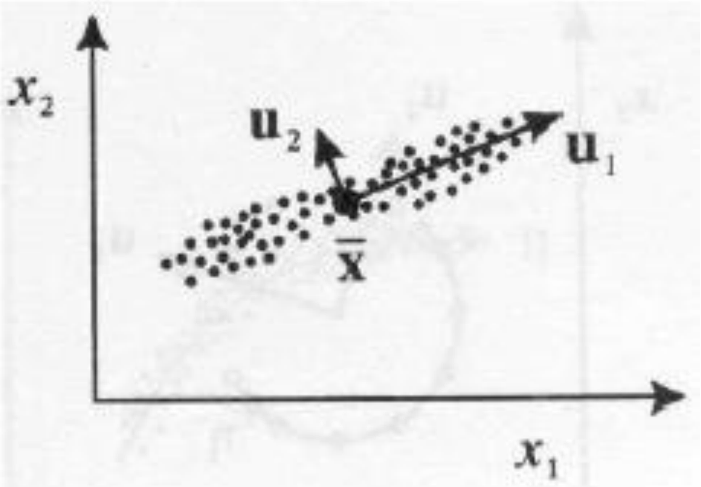
$$\boldsymbol{\lambda_1 = 9.34; \lambda_2 = 0.41}$$

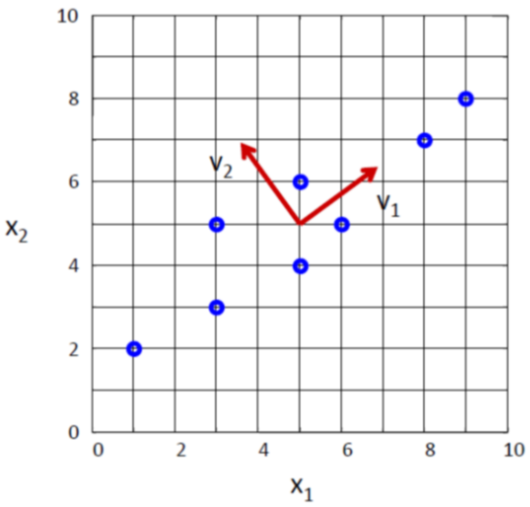
9.34/(9.34+0.41)
=0.958

Geometric interpretation of PCA

- PCA chooses the eigenvectors of the covariance matrix corresponding to the largest eigenvalues.
- The eigenvalues correspond to the variance of the data along the eigenvector directions.
- Therefore, PCA projects the data along the directions where the data varies most.

u_1: direction of max variance
u_2: orthogonal to u_1





How do we choose K ?

- *K* is typically chosen based on how much information (variance) we want to preserve:

$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > T \quad where\ T\ is\ a\ threshold\ (e.g., 0.9)$$

- If T=0.9, for example, we say that we "preserve" 90% of the information (variance) in the data.

- If K=N, then we "preserve" 100% of the information in the data (i.e., just a change of basis)

Approximation Error

- The approximation error (or reconstruction error) can be computed as:

$$\| \mathbf{x} - \hat{\mathbf{x}} \|$$

where $\hat{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K + \bar{\mathbf{x}}$ (reconstruction)

- It can also be shown that the approximation error can be computed as follows:

$$\| \mathbf{x} - \hat{\mathbf{x}} \| = \frac{1}{2} \sum_{i=K+1}^{N} \lambda_i$$

Data Normalization

- The principal components are dependent on the *units* used to measure the original variables as well as on the *range* of values they assume.
- Data should always be normalized prior to using PCA.
- A common normalization method is to transform all the data to have zero mean and unit standard deviation:

$$\frac{x_i - \mu}{\sigma}$$

where μ and σ are the mean and standard deviation of the i-th feature x_i

Comments

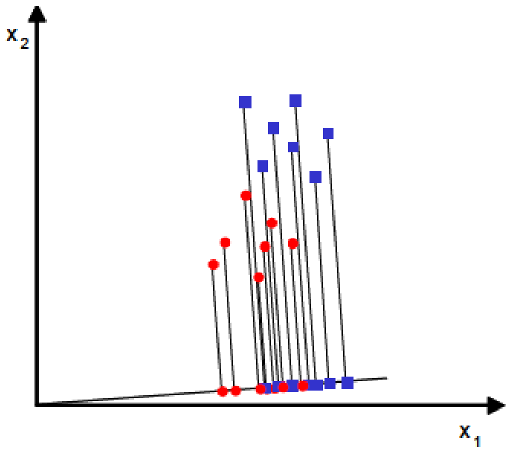
- PCA simply performs a coordinate rotation that aligns the transformed axes with the directions of maximum variance.

- The new covariance matrix $\Sigma_{\mathbf{y}}$ is diagonal (i.e., PCA simply de-correlates the variables).
  - Note that if the data follow a Gaussian distribution, then the variables also become independent.

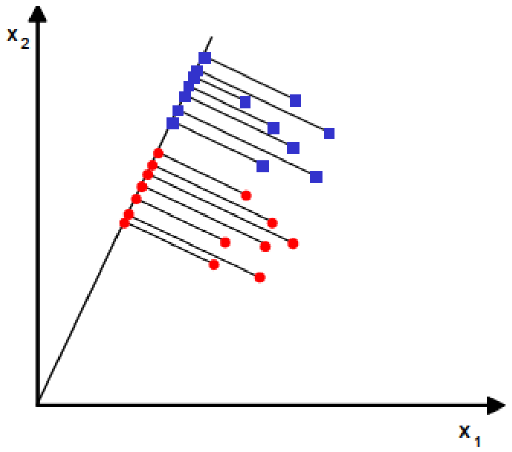
- The main limitation of PCA is that it does not consider class separability since it does not take into account the class information.
  - i.e., there is no guarantee that the directions of maximum variance will contain good features for discrimination.

Linear Discriminant Analysis (LDA)

- What is the goal of LDA?
  - Seeks to find directions along which the classes are best separated (i.e., increase discriminatory information).
  - It takes into consideration the scatter within-classes and between-classes.



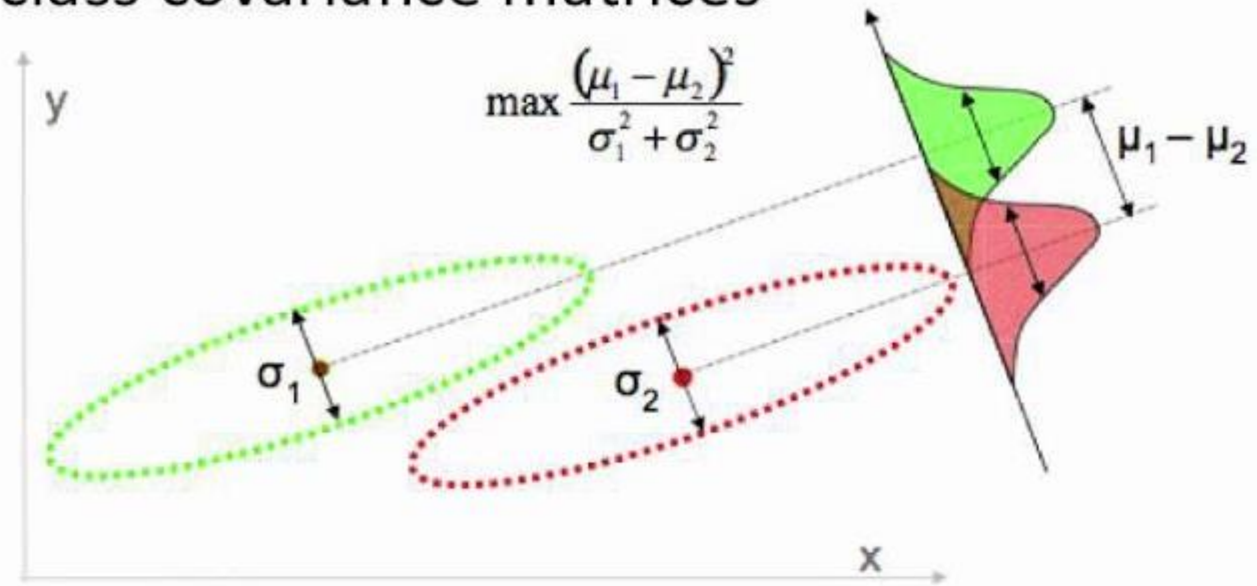
Bad separability



Good separability

Linear Discriminant Analysis

- LDA: pick a new dimension that gives:
  - maximum separation between means of projected classes
  - minimum variance within each projected class
- Solution: eigenvectors based on between-class and within-class covariance matrices



Case of C classes

- Let us assume C classes
- Let is assume each class contains M_i samples, i=1,2,..,C and that

$$M = \sum_{i=1}^{C} M_i$$

- Let μ_i is the mean of the i-th class, i=1,2,…,C

Within-class scatter matrix

$$S_w = \sum_{i=1}^{C} \sum_{j=1}^{M_i} (x_{ij} - \boldsymbol{\mu}_i)(x_{ij} - \boldsymbol{\mu}_i)^T$$

Between-class scatter matrix

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

$$where \quad \mu = \frac{1}{C} \sum_{i=1}^{C} \mu_i$$

(i.e., μ is the mean of means)

Case of C classes (cont'd)

- Suppose the desired projection transformation is:

$$\mathbf{y} = U^T \mathbf{x}$$

- Suppose the scatter matrices of the projected data y are:

$$\tilde{S}_b, \tilde{S}_w$$

- **LDA seeks projections that maximize the between-class scatter and minimize the within-class scatter:**

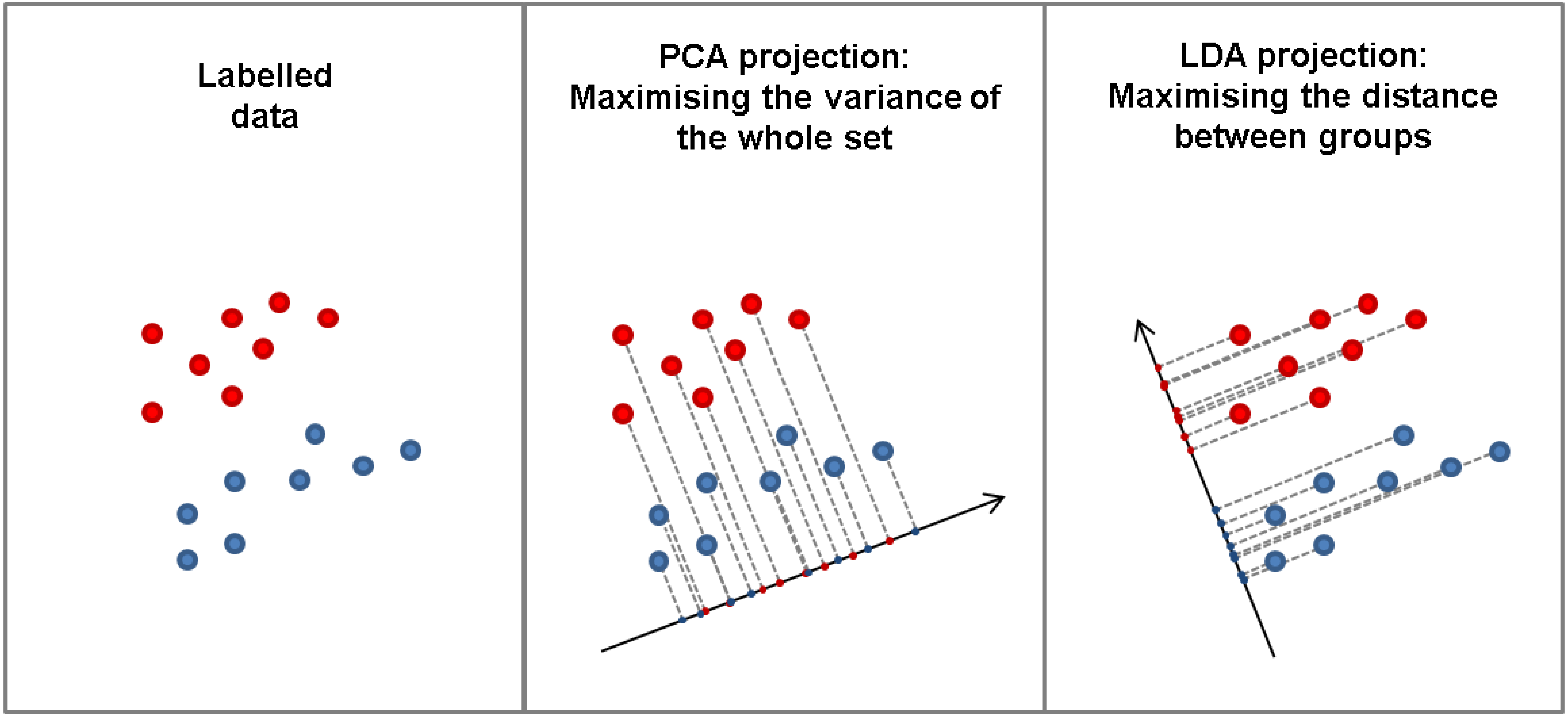
$$\max \frac{|\tilde{S}_b|}{|\tilde{S}_w|} = \max \frac{|U^T S_b U|}{|U^T S_w U|}$$

Case of C classes (cont'd)

- It can be shown that the columns of the matrix *U* are the eigenvectors (i.e., called *Fisherfaces*) corresponding to the largest eigenvalues of the following *generalized eigen-problem:*

$$S_b \boldsymbol{u}_k = \lambda_k S_w \boldsymbol{u}_k$$

- Note: S_b has at most rank C-1; therefore, the max number of eigenvectors with non-zero eigenvalues is C-1 (i.e., max dimensionality of sub-space is C-1)


Labelled data
PCA projection: Maximising the variance of the whole set
LDA projection: Maximising the distance between groups

Case of C classes (cont'd)

- If S_w is non-singular, we can solve a *conventional* eigenvalue problem as follows:

$$S_b \boldsymbol{u}_k = \lambda_k S_w \boldsymbol{u}_k$$

$$\Downarrow$$

$$S_w^{-1} S_b \boldsymbol{u}_k = \lambda_k \boldsymbol{u}_k$$

- In practice, S_w is singular due to the high dimensionality of the data (e.g., images) and the much smaller number of data ($M << N$)

Does S_w^{-1} always exist? (cont'd)

- To alleviate this problem, PCA could be applied first:

1) Apply PCA to reduce data dimensionality:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{PCA} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_M \end{bmatrix}$$

2) Apply LDA to find the most discriminative directions:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_M \end{bmatrix} \xrightarrow{LDA} \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ . \\ . \\ z_K \end{bmatrix}$$