Machine Learning

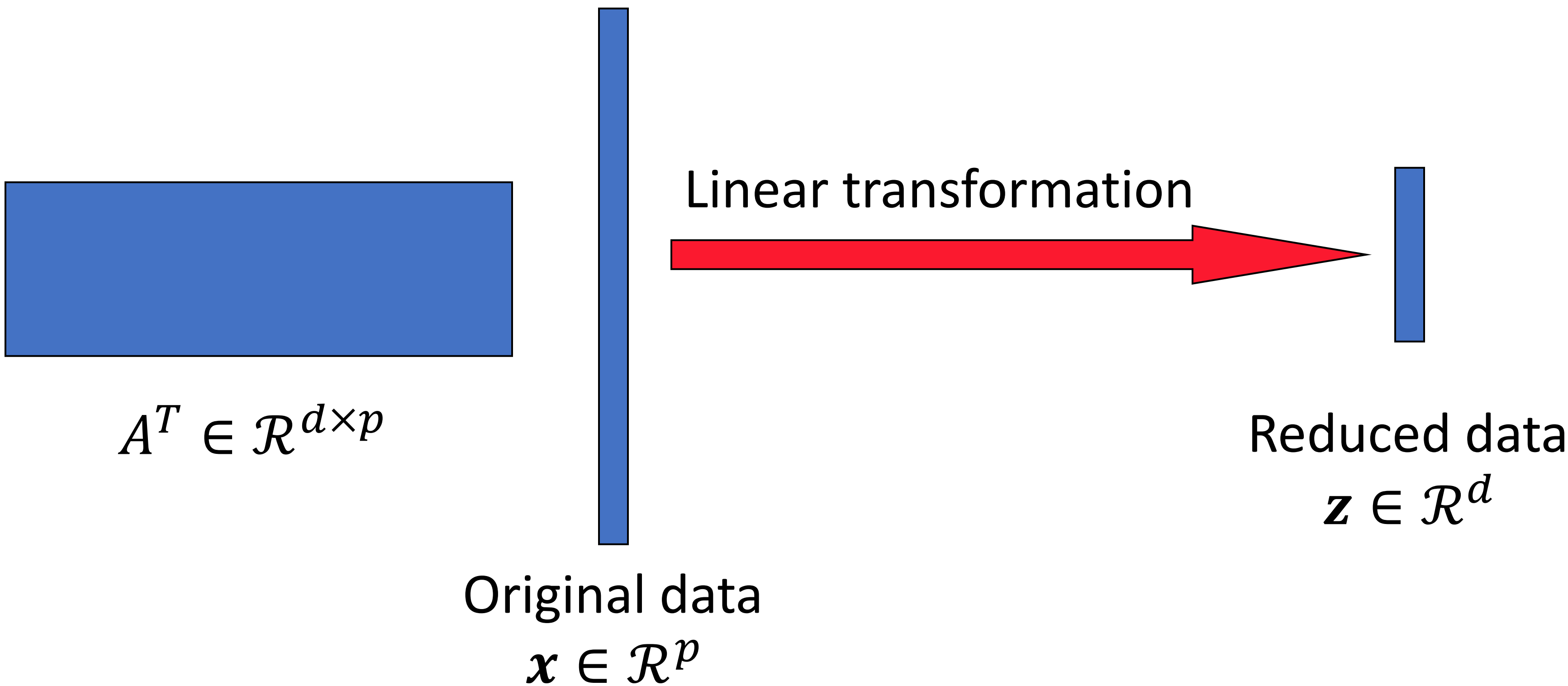
Lecture 17: Dimension Reduction

杨思蓓

SIST

Email: yangsb@shanghaitech.edu.cn

What is Dimensionality Reduction?



$$A \in \mathcal{R}^{p \times d}: \boldsymbol{x} \in \mathcal{R}^p \rightarrow \boldsymbol{z} = A^T \boldsymbol{x} \in \mathcal{R}^d$$

Relation to Matrix Factorization

$$X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_n] = UV = U[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots \boldsymbol{v}_n]$$

$$\boldsymbol{x}_i = U\boldsymbol{v}_i$$

$$\boldsymbol{x}_i \in \mathcal{R}^m, \qquad \boldsymbol{v}_i \in \mathcal{R}^k$$

- If there is a matrix $A \in \mathcal{R}^{k \times m}$ which satisfies:

$$AU = I$$

$$A\boldsymbol{x}_i = \boldsymbol{v}_i$$

What is Dimensionality Reduction (Feature Reduction)?

- Feature reduction refers to the mapping of the original high-dimensional data onto a lower-dimensional space.
  - Criterion for feature reduction can be different based on different problem settings.
    - Unsupervised setting: minimize the information loss
    - Supervised setting: maximize the class discrimination

- Given a set of data points of *p* variables $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_2\}$
- Compute the linear transformation (projection)

$$A \in \mathcal{R}^{p \times d} : \boldsymbol{x} \in \mathcal{R}^p \rightarrow \boldsymbol{z} = A^T \boldsymbol{x} \in \mathcal{R}^d$$

Feature Extraction vs Feature Selection

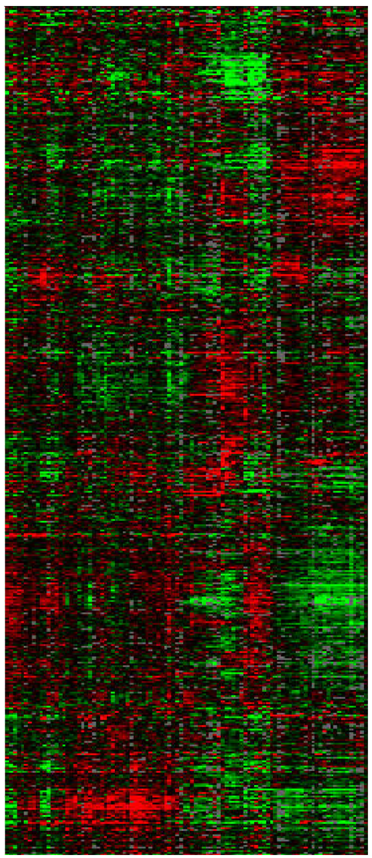
- Dimensionality reduction (Feature reduction)
  - Feature extraction
  - Feature selection

- Selection: choose a best subset of size *d* from the available *p* features
- Extraction: given *p* features (set X), extract *d* new features (set Z) by linear or non-linear combination of all the p features

$$A \in \mathcal{R}^{p\times d}: \boldsymbol{x} \in \mathcal{R}^{p} \rightarrow \boldsymbol{z} = A^{T}\boldsymbol{x} \in \mathcal{R}^{d}$$

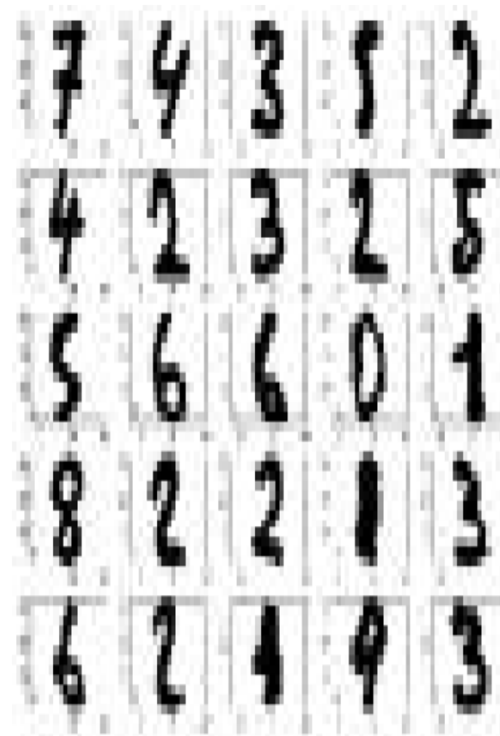
- Selection: $A \in [0,1]^{p\times d}$, every column of *A* has only one 1.
- Extraction: $A \in \mathcal{R}^{p\times d}$
  - Non-linear: $\boldsymbol{z} = f(\boldsymbol{x})$

High-dimensional Data



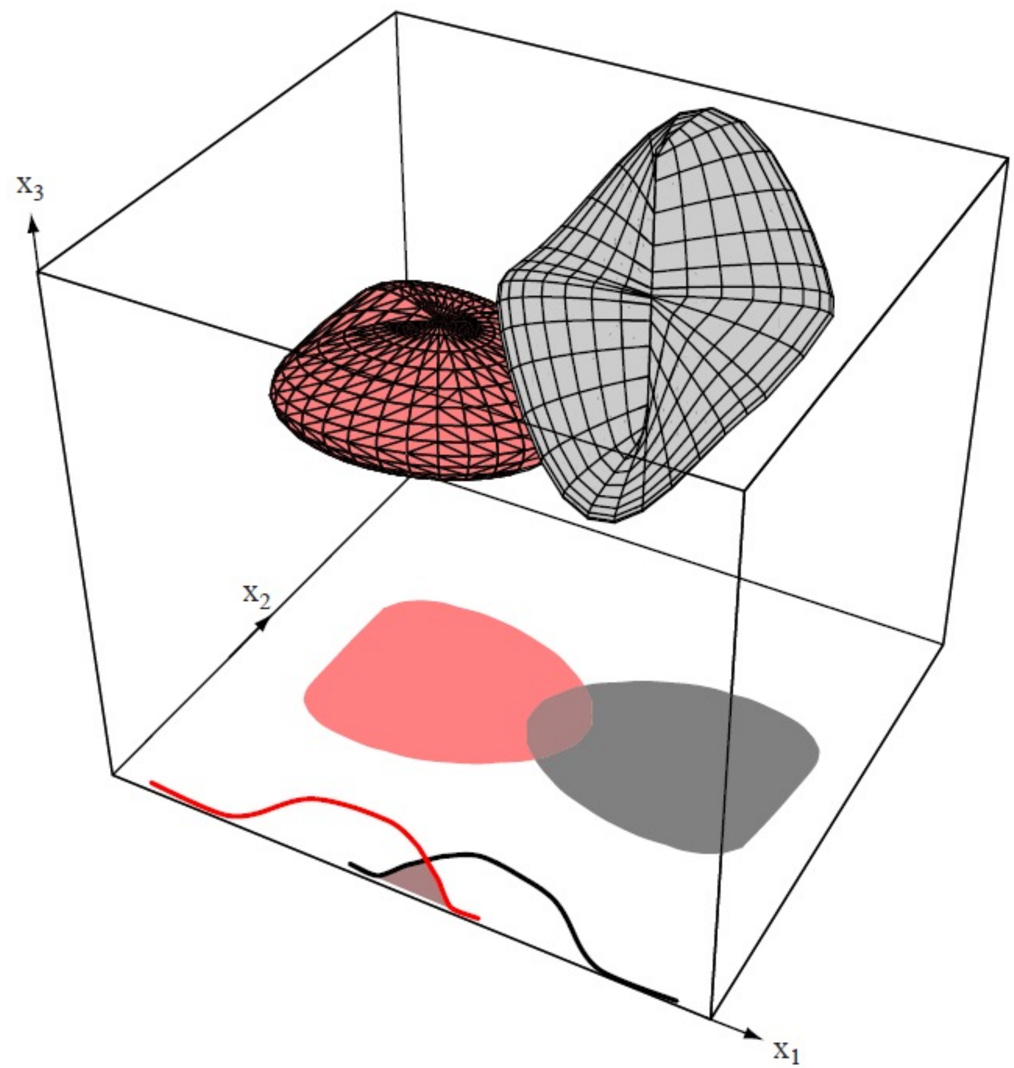
Gene expression

Face images



Handwritten digits

Why Dimensionality Reduction?



Why Dimensionality Reduction?

- Intuition: More the number of features, the better the classification performance?
  - Not always!

- There are two issues that must be confronted with high dimensional feature spaces
  - How does the classification accuracy depend on the dimensionality and the number of training samples
  - The computational complexity of designing a classifier

G. V. Trunk. "A Problem of Dimensionality: A Simple Example", TPAMI, July 1979

Why Dimensionality Reduction?

- Most machine learning and data mining techniques may not be effective for high-dimensional data
  - Curse of Dimensionality
  - Query accuracy and efficiency degrade rapidly as the dimension increases.

- The intrinsic dimension may be small.
  - Handwritten digit images.
  - For example, the number of genes responsible for a certain type of disease may be small.

Why Dimensionality Reduction?

- Visualization: projection of high-dimensional data onto 2D or 3D.
- Data compression: efficient storage and retrieval.
- Noise removal: positive effect on query accuracy.

Application of Dimensionality Reduction

- Face recognition
- Handwritten digit recognition
- Text mining
- Image retrieval
- Microarray data analysis
- Protein classification

Dimensionality Reduction Algorithms

- Unsupervised
  - Latent Semantic Indexing (LSI): truncated SVD
  - Principal Component Analysis (PCA)
  - Independent Component Analysis (ICA)
  - Canonical Correlation Analysis (CCA)

- Supervised
  - Linear Discriminant Analysis (LDA)

- Semi-supervised
  - Semi-supervised Discriminant Analysis (SDA)

Dimensionality Reduction Algorithms

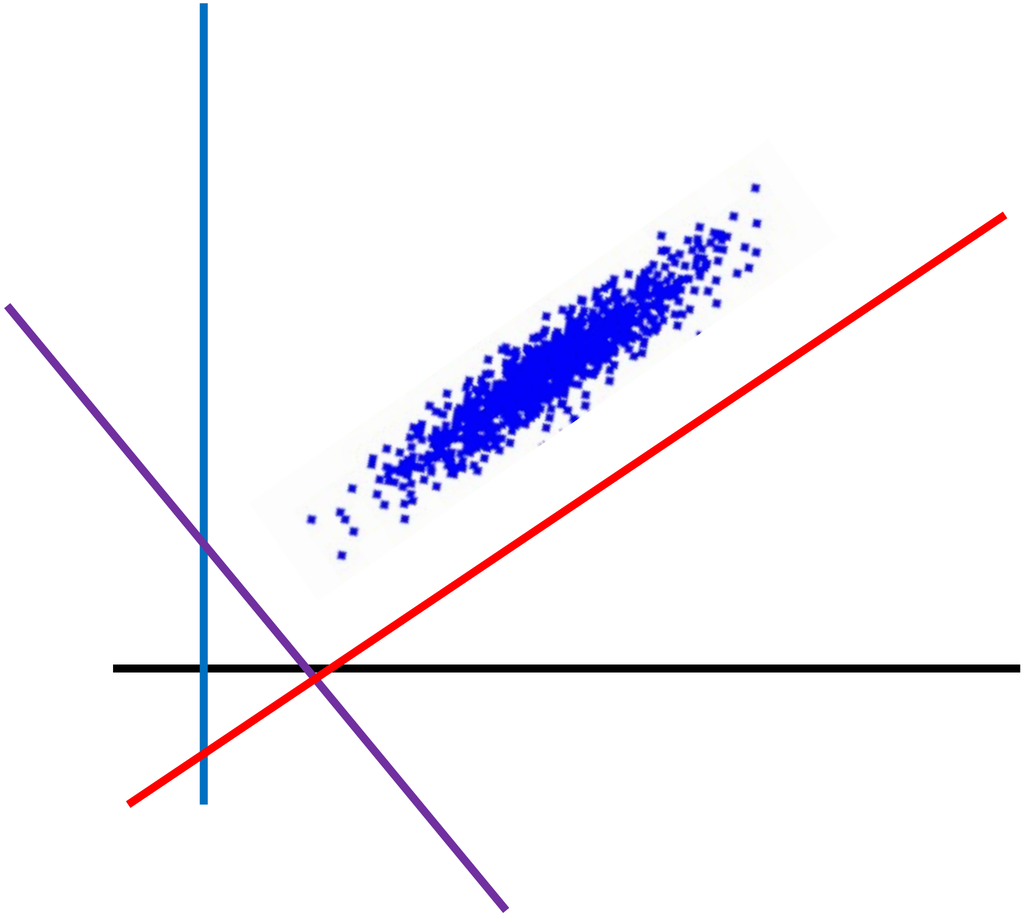
- Linear
  - Latent Semantic Indexing (LSI): truncated SVD
  - Principal Component Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - Canonical Correlation Analysis (CCA)

- Nonlinear
  - Nonlinear feature reduction using kernels
  - Manifold learning

Algorithms

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Locality Preserving Projections (LPP)
- The framework of graph based dimensionality reduction.

Principal Component Analysis

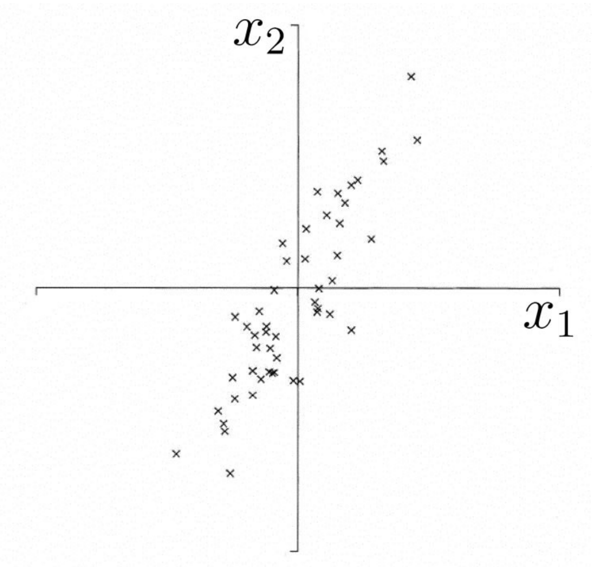


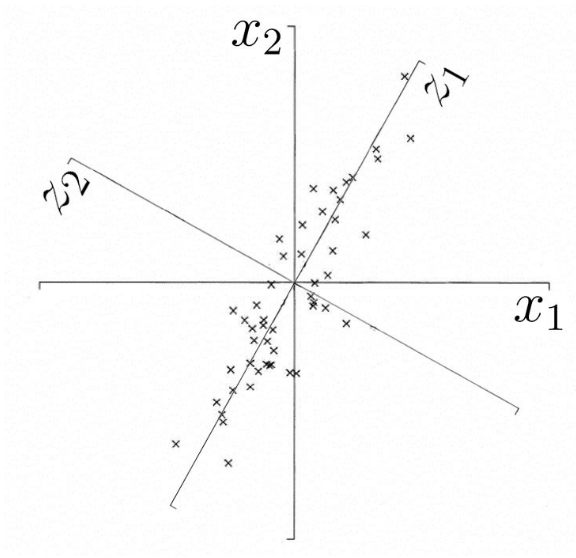
What is Principal Component Analysis?

- Principal component analysis (PCA)
  - Reduce the dimensionality of a data set by finding a new set of variables, smaller than the original set of variables
  - Retains most of the sample's information.
  - Useful for the compression and classification of data.

- By information we mean the variation present in the sample, given by the correlations between the original variables.
  - The new variables, called principal components (PCs), are uncorrelated, and are ordered by the fraction of the total information each retains.

Geometric Picture of Principal Components (PCs)





- The 1st PC z_1 is a minimum distance fit to a line in X space
- The 2nd PC z_2 is a minimum distance fit to a line in the plane perpendicular to the 1st PC
- PCs are a series of linear least squares fits to a line, each orthogonal to all the previous.

Algebraic Derivation of PCs

- Given a sample of *n* observations on a vector of *p* variables

$$\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\} \in \mathcal{R}^p$$

- Define the first principal component of the sample by the linear transformation

$$z_i^{(1)} = \boldsymbol{a}_1^T \boldsymbol{x}_i, \qquad i = 1, \cdots n$$

is chosen such that $var\left(z^{(1)}\right)$ is maximum.

Algebraic Derivation of PCs

$$var\left(z^{(1)}\right) = E\left(\left(z^{(1)} - \bar{z}^{(1)}\right)^2\right)$$

$$= \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{a}_1^T\boldsymbol{x}_i - \boldsymbol{a}_1^T\overline{\boldsymbol{x}})^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{a}_1^T(\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T\boldsymbol{a}_1 = \boldsymbol{a}_1^T S\boldsymbol{a}_1$$

Where $S = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T$

is the covariance matrix and $\overline{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i$ is the mean.

Algebraic Derivation of PCs

$$\max_{\boldsymbol{a}_1} \boldsymbol{a}_1^T S \boldsymbol{a}_1$$
$$s.t. \quad \boldsymbol{a}_1^T \boldsymbol{a}_1 = 1$$

Let λ be a Lagrange multiplier

$$L = \boldsymbol{a}_1^T S \boldsymbol{a}_1 - \lambda(\boldsymbol{a}_1^T \boldsymbol{a}_1 - 1)$$
$$\frac{\partial L}{\partial \boldsymbol{a}_1} = 2S\boldsymbol{a}_1 - 2\lambda \boldsymbol{a}_1 = 0$$
$$S\boldsymbol{a}_1 = \lambda \boldsymbol{a}_1$$

therefor, $\boldsymbol{a}_1$ is an eigenvector of S corresponding to the largest eigenvalue $\lambda = \lambda_1$.

Algebraic Derivation of PCs

$$\max_{\boldsymbol{a}_2} \boldsymbol{a}_2^T S \boldsymbol{a}_2$$

$$s.t. \quad \boldsymbol{a}_2^T \boldsymbol{a}_2 = 1, \operatorname{cov}\left(z^{(2)}, z^{(1)}\right) = 0$$

$$\operatorname{cov}\left(z^{(2)}, z^{(1)}\right) = \boldsymbol{a}_2^T S \boldsymbol{a}_1 = \lambda_1 \boldsymbol{a}_2^T \boldsymbol{a}_1$$

$$L = \boldsymbol{a}_2^T S \boldsymbol{a}_2 - \lambda(\boldsymbol{a}_2^T \boldsymbol{a}_2 - 1) - \phi(\lambda_1 \boldsymbol{a}_2^T \boldsymbol{a}_1)$$

$$S\boldsymbol{a}_2 = \lambda \boldsymbol{a}_2$$

$\boldsymbol{a}_2$ is an eigenvector of S corresponding to the second largest eigenvalue $\lambda = \lambda_2$.

Algebraic Derivation of PCs

- In general:

$$var\left(z^{(k)}\right) = \boldsymbol{a}_k^T S \boldsymbol{a}_k = \lambda_k$$

- The k^{th} largest eigenvalue of S is the variance of k^{th} PC.
- The k^{th} PC $z^{(k)}$ retains the k^{th} greatest fraction of the variation in the sample.

Principle Component Analysis

- Main steps for computing PCs:
  - Form the covariance matrix *S*.
  - Compute its eigenvectors: $\{\boldsymbol{a}_i\}_{i=1}^{p}$
  - Use the first d eigenvectors $\{\boldsymbol{a}_i\}_{i=1}^{d}$ to form the d PCs.
  - The transformation *A* is given by

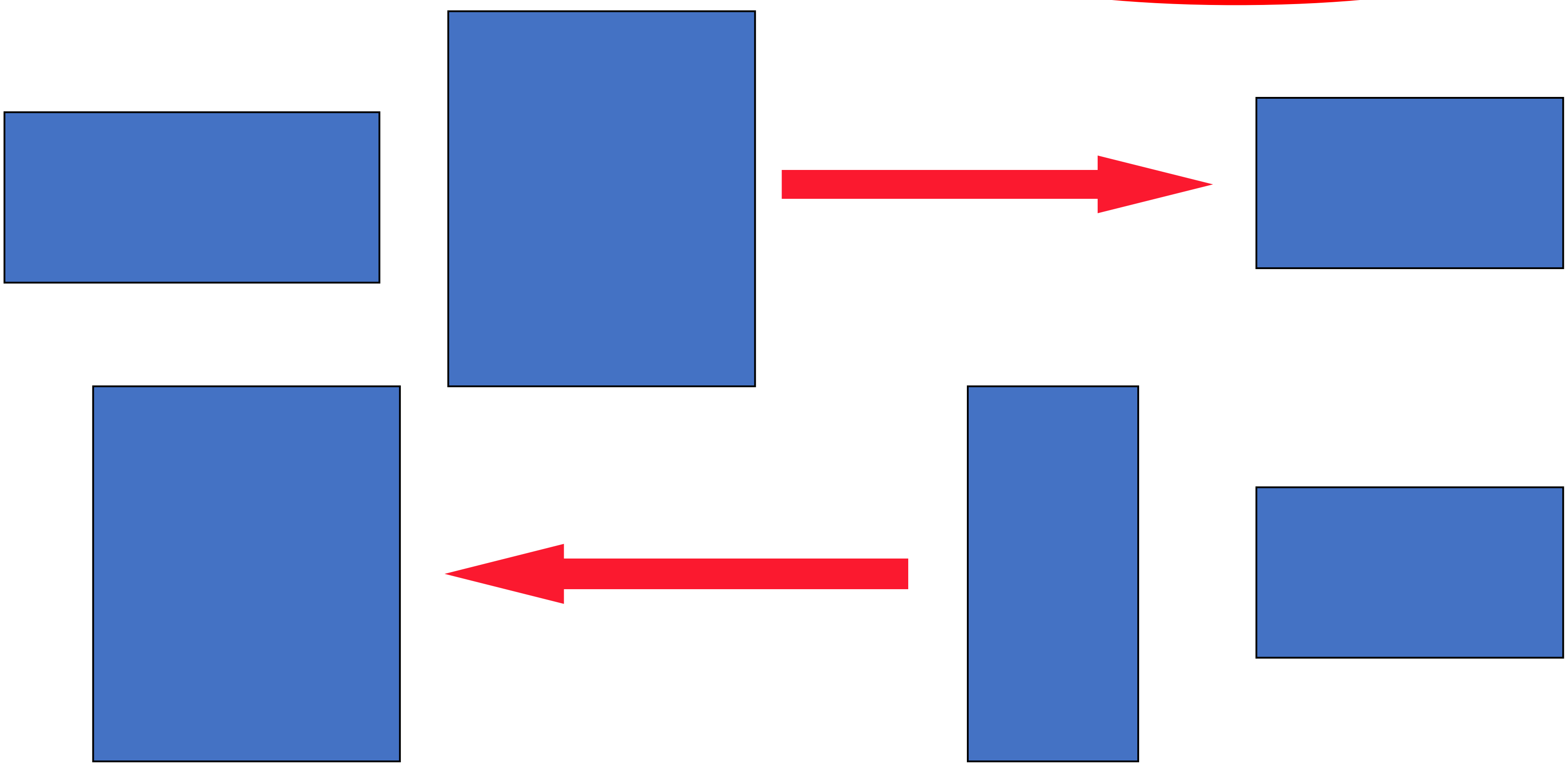
$$A = [\boldsymbol{a}_1, \cdots \boldsymbol{a}_d]$$

- A test point $\boldsymbol{x} \in \mathcal{R}^p \rightarrow A^T\boldsymbol{x} \in \mathcal{R}^d$

Reconstruction

Optimality Property of PCA

- Dimension reduction: $X \in \mathcal{R}^{p\times n} \rightarrow A^T X \in \mathcal{R}^{d\times n}$
- Original data: $A^T X \in \mathcal{R}^{d\times n} \rightarrow \bar{X} = A(A^T X) \in \mathcal{R}^{p\times n}$



Optimality Property of PCA

- Main theoretical result:
- The matrix *A* consisting of the first *d* eigenvectors of the covariance matrix *S* solves the following optimization problem:

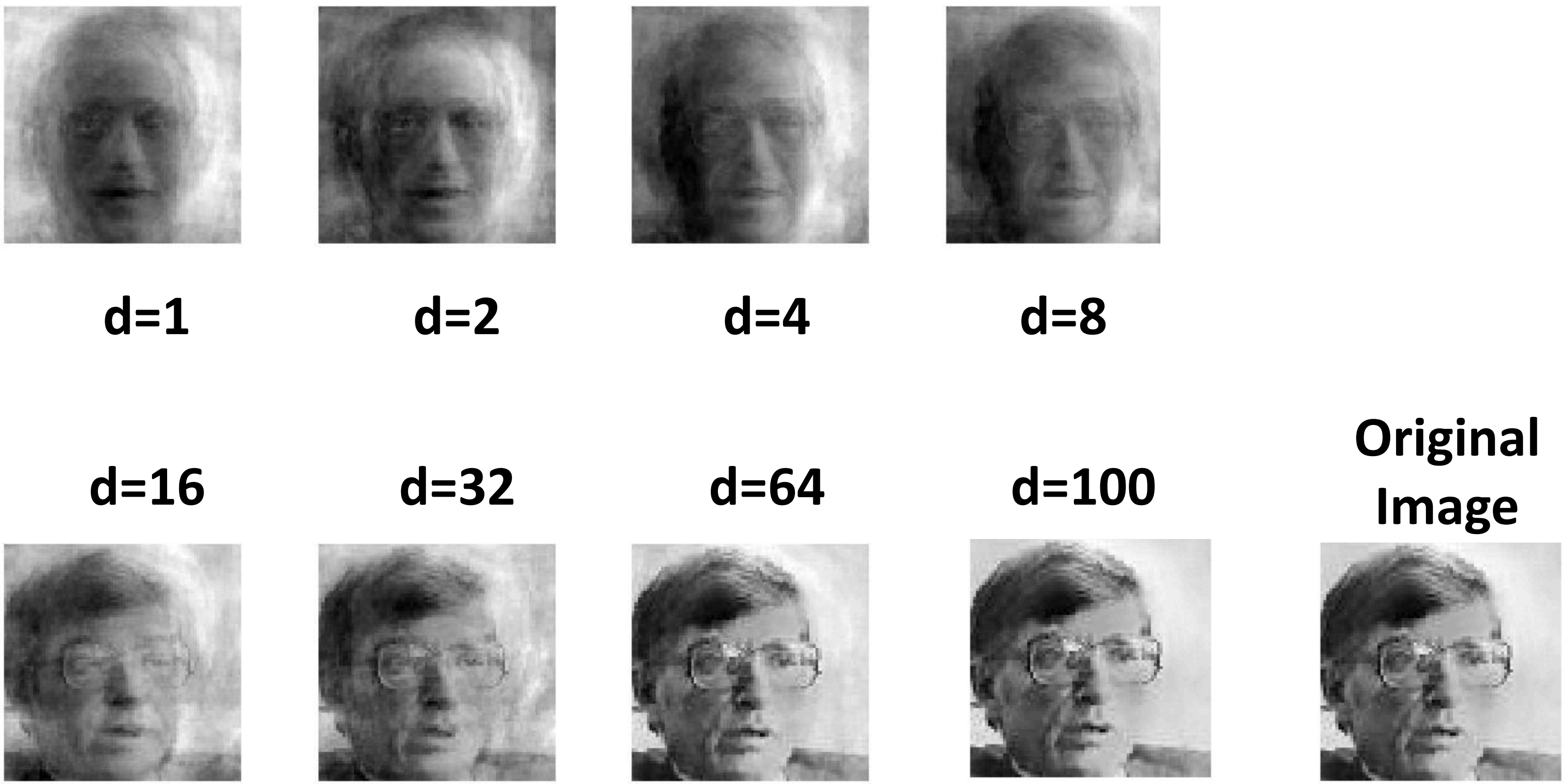
$$\min_{A \in \mathcal{R}^{p \times d}} \|X - AA^T X\|_F^2 \quad s.t.\ A^T A = I_d$$

$$\|X - \bar{X}\|_F^2$$

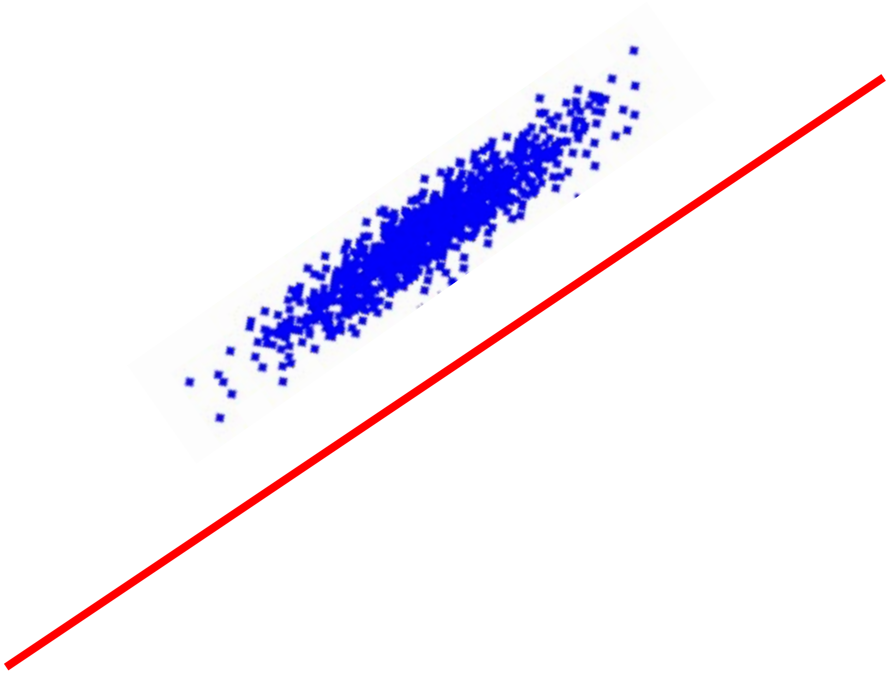
Reconstruction error

- PCA projection minimizes the reconstruction error among all linear projections of size *d*.

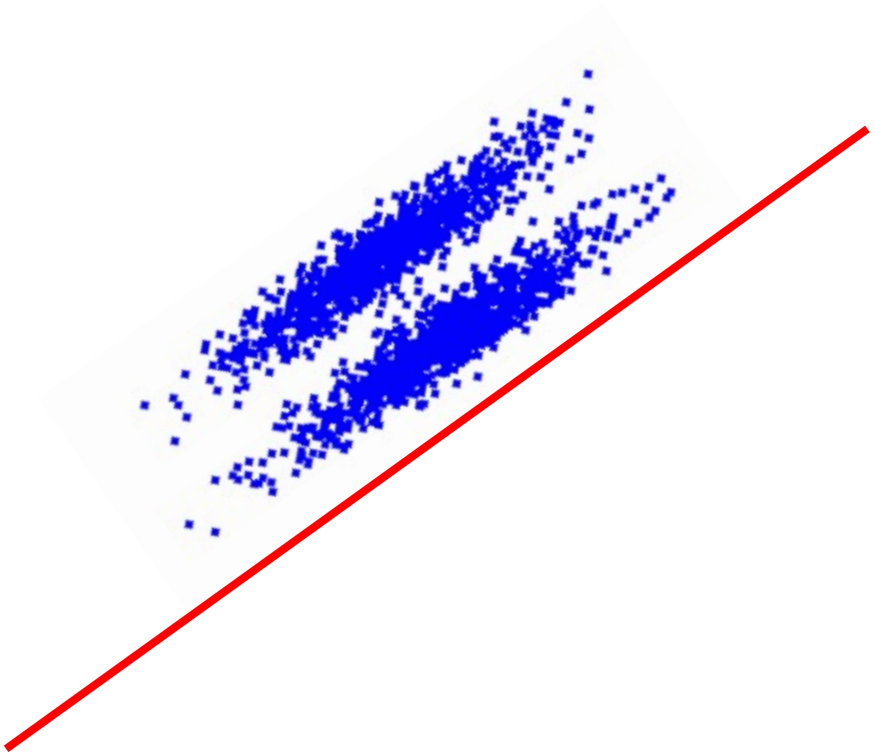
PCA for Image Compression



Principal Component Analysis

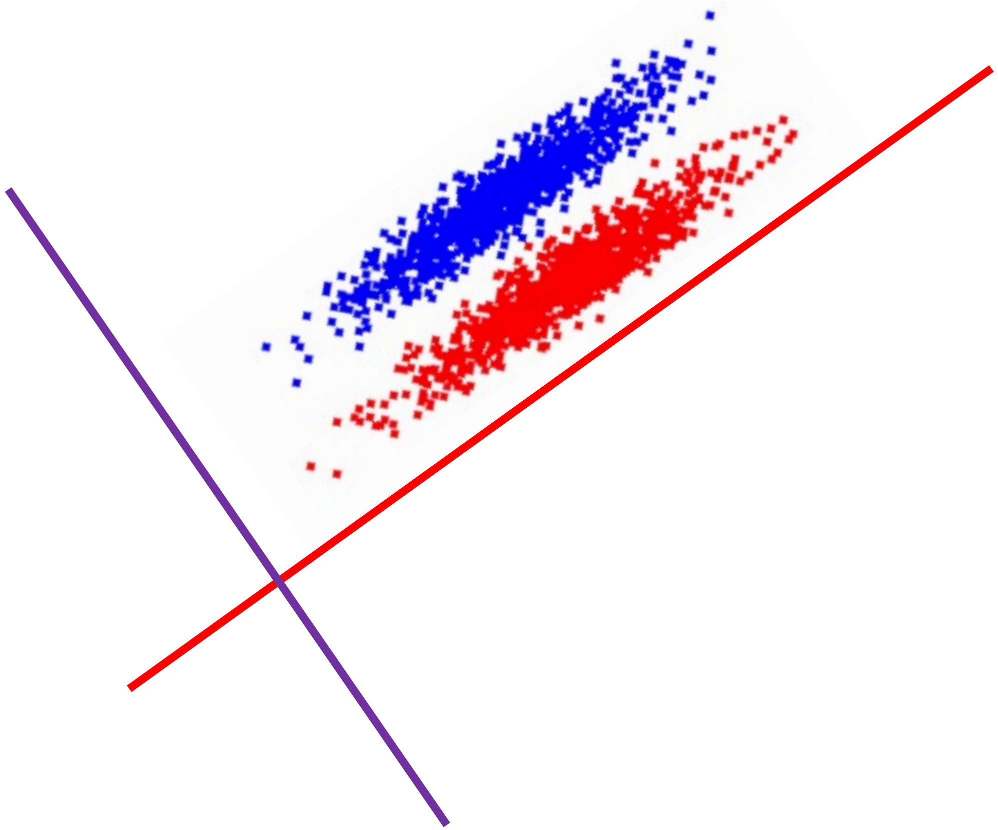


Principal Component Analysis

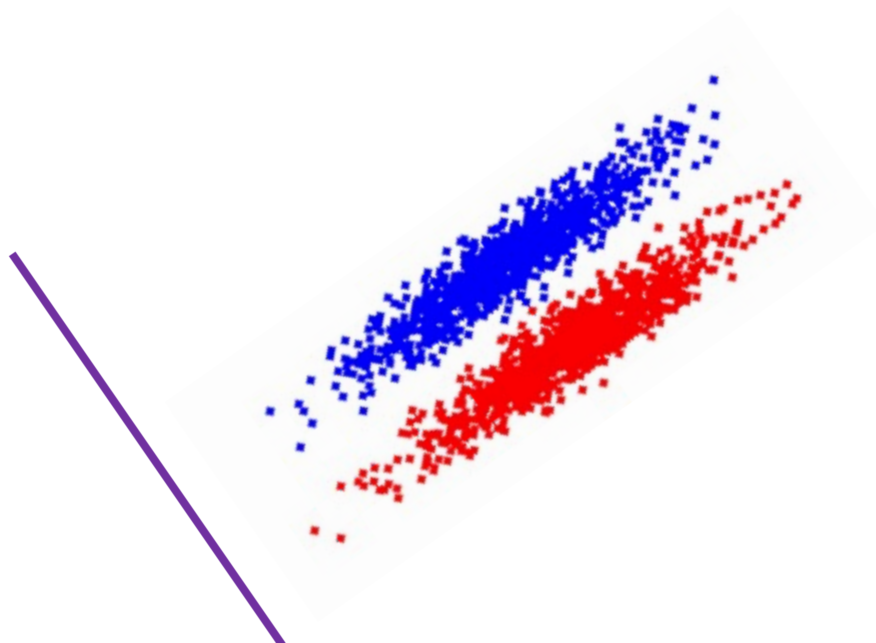


- Find a transformation $\boldsymbol{a}$, such that the $\boldsymbol{a}^T X$ $w^T x$ is dispersed the most (maximum distribution)

Principal Component Analysis



Linear Discriminant Analysis (Fisher Linear Discriminant)



- Find a transformation $\boldsymbol{a}$, such that the $\boldsymbol{a}^T X_1$ and $\boldsymbol{a}^T X_2$ are maximally separated & each class is minimally dispersed (maximum separation)

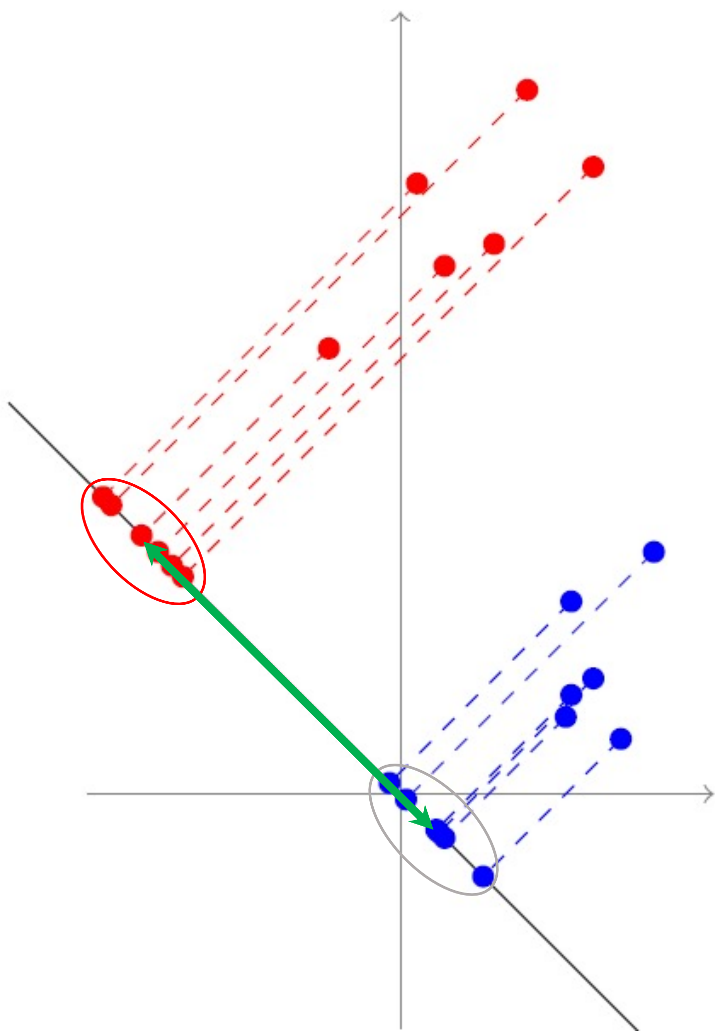
Linear Discriminant Analysis

- Perform dimensionality reduction “while preserving as much of the class discriminatory information as possible”.

- Seeks to find directions along which the classes are best separated.

- Takes into consideration the scatter within-classes but also the scatter between-classes.

Linear Discriminant Analysis



- **Two Classes $\boldsymbol{\omega_1}, \boldsymbol{\omega_2}$**

$$z = \boldsymbol{a}^T \boldsymbol{x}$$

$$\tilde{\mu}_i = \frac{1}{n_i} \sum_{z \in \omega_i} z$$

$$\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{\boldsymbol{x} \in \omega_i} \boldsymbol{x}\,, \tilde{\mu}_i = \boldsymbol{a}^T \boldsymbol{\mu}_i$$

$$|\tilde{\mu}_1 - \tilde{\mu}_2| = |\boldsymbol{a}^T (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)|$$

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

$$\frac{1}{n} (\tilde{s}_1^2 + \tilde{s}_2^2)$$

$$J(\boldsymbol{a}) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

Linear Discriminant Analysis

- **Two Classes**

$$J(\boldsymbol{a}) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2} = \frac{\boldsymbol{a}^T S_B \boldsymbol{a}}{\boldsymbol{a}^T S_W \boldsymbol{a}}$$

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

$$= \boldsymbol{a}^T \sum_{\boldsymbol{x} \in \omega_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{a} = \boldsymbol{a}^T S_i \boldsymbol{a}$$

S_i: *scatter matrix*

within-class scatter matrix: $S_W = S_1 + S_2$ $\qquad$ $\tilde{s}_1^2 + \tilde{s}_2^2 = \boldsymbol{a}^T S_W \boldsymbol{a}$

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (\boldsymbol{a}^T \boldsymbol{\mu}_1 - \boldsymbol{a}^T \boldsymbol{\mu}_2)^2 = \boldsymbol{a}^T (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \boldsymbol{a} = \boldsymbol{a}^T S_B \boldsymbol{a}$$

between-class scatter matrix: $S_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$

Linear Discriminant Analysis

- **Two Classes**

$$J(\boldsymbol{a}) = \frac{\boldsymbol{a}^T S_B \boldsymbol{a}}{\boldsymbol{a}^T S_W \boldsymbol{a}}$$

$$S_B \boldsymbol{a} = \lambda S_W \boldsymbol{a}$$

? Try it

$$\boldsymbol{a}^* = S_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

$$S_W = \sum_i^2 S_i = \sum_i^2 \sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T$$

Linear Discriminant Analysis

- Multi-classes

$$J(\boldsymbol{a}) = \frac{\boldsymbol{a}^T S_B \boldsymbol{a}}{\boldsymbol{a}^T S_W \boldsymbol{a}}$$

$$\boldsymbol{\mu} = \frac{1}{n}\sum_{\boldsymbol{x}} \boldsymbol{x} = \frac{1}{n}\sum_{i=1}^{c} n_i \boldsymbol{\mu}_i$$

$$\boxed{S_W = \sum_{i=1}^{c} S_i = \sum_{i=1}^{c}\sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{x}-\boldsymbol{\mu}_i)(\boldsymbol{x}-\boldsymbol{\mu}_i)^T}$$

$$S_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$$

$$S_T = \sum_{\boldsymbol{x}} (\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^T = \sum_{i=1}^{c}\sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{x}-\boldsymbol{\mu}_i+\boldsymbol{\mu}_i-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu}_i+\boldsymbol{\mu}_i-\boldsymbol{\mu})^T$$

$$= \sum_{i=1}^{c}\sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{x}-\boldsymbol{\mu}_i)(\boldsymbol{x}-\boldsymbol{\mu}_i)^T + \sum_{i=1}^{c}\sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{\mu}_i-\boldsymbol{\mu})(\boldsymbol{\mu}_i-\boldsymbol{\mu})^T$$

$$= S_W + \boxed{\sum_{i=1}^{c} n_i(\boldsymbol{\mu}_i-\boldsymbol{\mu})(\boldsymbol{\mu}_i-\boldsymbol{\mu})^T = S_B}$$

$$S_T = S_W + S_B$$

Linear Discriminant Analysis

- Multi-classes

$$J(\boldsymbol{a}) = \frac{\boldsymbol{a}^T S_B \boldsymbol{a}}{\boldsymbol{a}^T S_W \boldsymbol{a}}$$

$$S_W = \sum_{i=1}^{c} S_i = \sum_{i=1}^{c} \sum_{\boldsymbol{x} \in \omega_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T$$

$$S_B = \sum_{i=1}^{c} n_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

$$S_B \boldsymbol{a} = \lambda S_W \boldsymbol{a}$$

$$S_B \boldsymbol{a} = \lambda S_T \boldsymbol{a}$$

Linear Discriminant Analysis

- Main steps:
  - Form the scatter matrices S_B and S_W.
  - Compute the eigenvectors $\{\boldsymbol{a}_i\}_{i=1}^{c-1}$ corresponding to the non-zero eigenvalue of the generalized eigen-problem:
$$S_B\boldsymbol{a} = \lambda S_W\boldsymbol{a} \quad \text{or} \quad S_B\boldsymbol{a} = \lambda S_T\boldsymbol{a}$$
  - The transformation A is given by
$$A = [\boldsymbol{a}_1, \cdots \boldsymbol{a}_{c-1}]$$

- A test point $\boldsymbol{x} \in \mathcal{R}^p \rightarrow A^T\boldsymbol{x} \in \mathcal{R}^{(c-1)}$

PCA vs LDA