Dimensionality Reduction

Sandro Cumani

sandro.cumani@polito.it

Politecnico di Torino

Dimensionality Reduction

Dimensionality reduction techniques compute a mapping from the n–dimensional feature space to a m–dimensional space, with $m \ll n$

- Compress information
- Remove unwanted variability (noise)
- Simplify classification (reduced effects due to high dimensionality, reduced risk of overfitting)
- Data visualization

Dimensionality Reduction

Different goals may require different approaches

- Compress information: we want to retain the maximum amount of information for a given output size
- Improve classification: we want to retain discriminant information
- Data visualization: we want 2-D or 3-D representations that preserve as much as possible relationships between different samples

Dimensionality Reduction

We will focus on two linear methods:

- Unsupervised: Principal Component Analysis (PCA)
- Supervised: Linear Discriminant Analysis (LDA)

In both cases, we want to find a subspace of the feature space that preserves most of the “useful” information

Notes on linear algebra

We recall some linear algebra notions for real matrices that will be used in the following

For a quick reference about matrix properties, you can also refer to

Petersen, Pedersen, “The Matrix Cookbook”, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.113.6244

Notes on linear algebra

Let $\boldsymbol{A}$ be a square symmetric $n \times n$ matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$

$\boldsymbol{A}$ admits an eigen–decomposition

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^{-1} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^{T}$$

- $\boldsymbol{V}$ is an orthogonal $n \times n$ matrix whose columns are the (right) eigenvectors of $\boldsymbol{A}$
- $\boldsymbol{\Sigma}$ is a diagonal $n \times n$ matrix whose elements are the eigenvalues of $\boldsymbol{A}$

Since $\boldsymbol{V}$ is orthogonal, $\boldsymbol{V}^{T}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^{T} = \boldsymbol{I}$ and $\boldsymbol{V}^{-1} = \boldsymbol{V}^{T}$

Notes on linear algebra

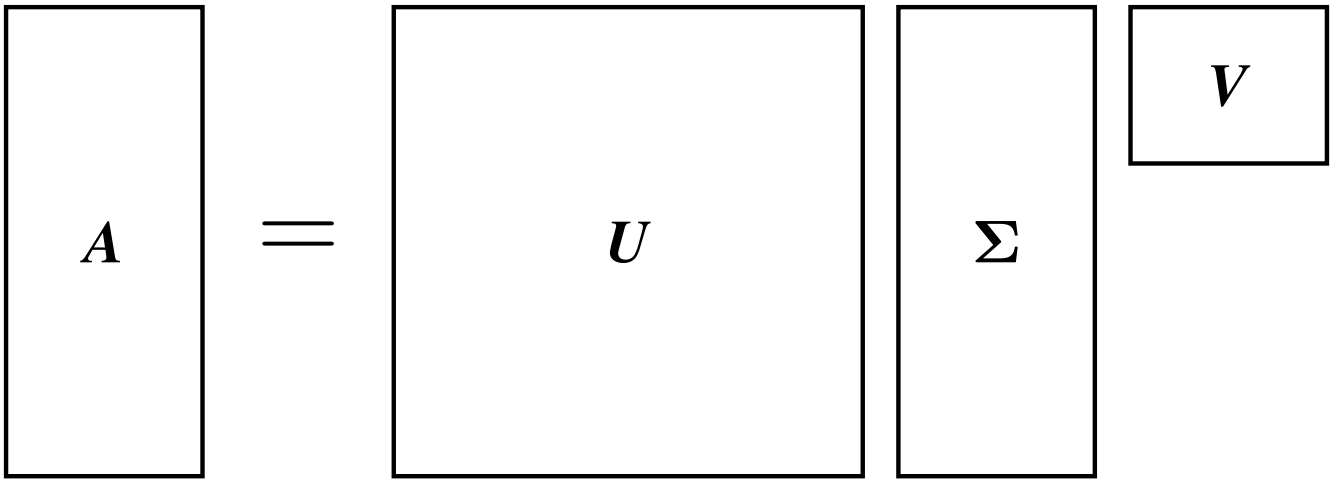
A generic rectangular matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ always admits a Singular Value Decomposition (SVD) of the form:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

where

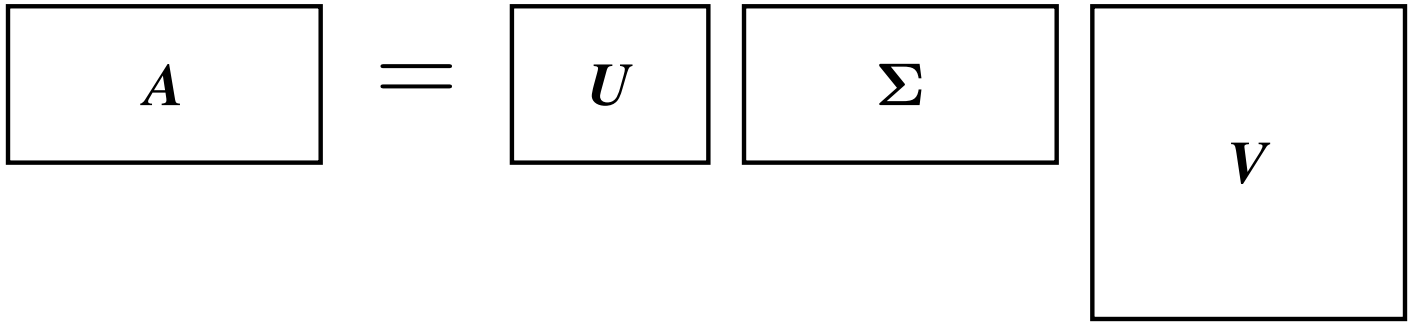
- $\boldsymbol{U}$ is an orthogonal $n \times n$ matrix of eigenvectors of $\boldsymbol{A}\boldsymbol{A}^T$
- $\boldsymbol{V}$ is an orthogonal $m \times m$ matrix of eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$
- $\boldsymbol{\Sigma}$ is a diagonal rectangular matrix containing the singular values of $\boldsymbol{A}$ (in decreasing order)

Notes on linear algebra



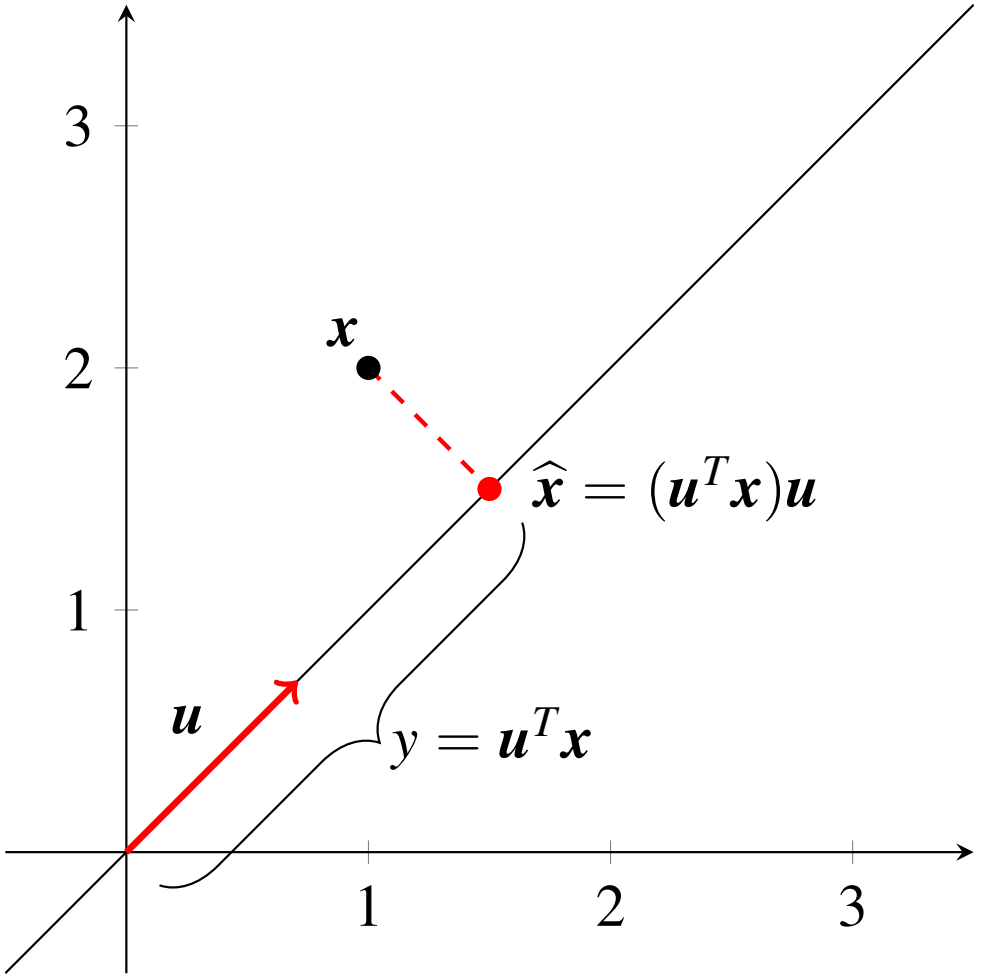
$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_m \\ & & \mathbf{0} & \end{bmatrix}$$

Notes on linear algebra



$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \mathbf{0} \\ & & \ddots & \\ & & & \sigma_m \end{bmatrix}$$

Projecting a point over a direction $\boldsymbol{u}$:

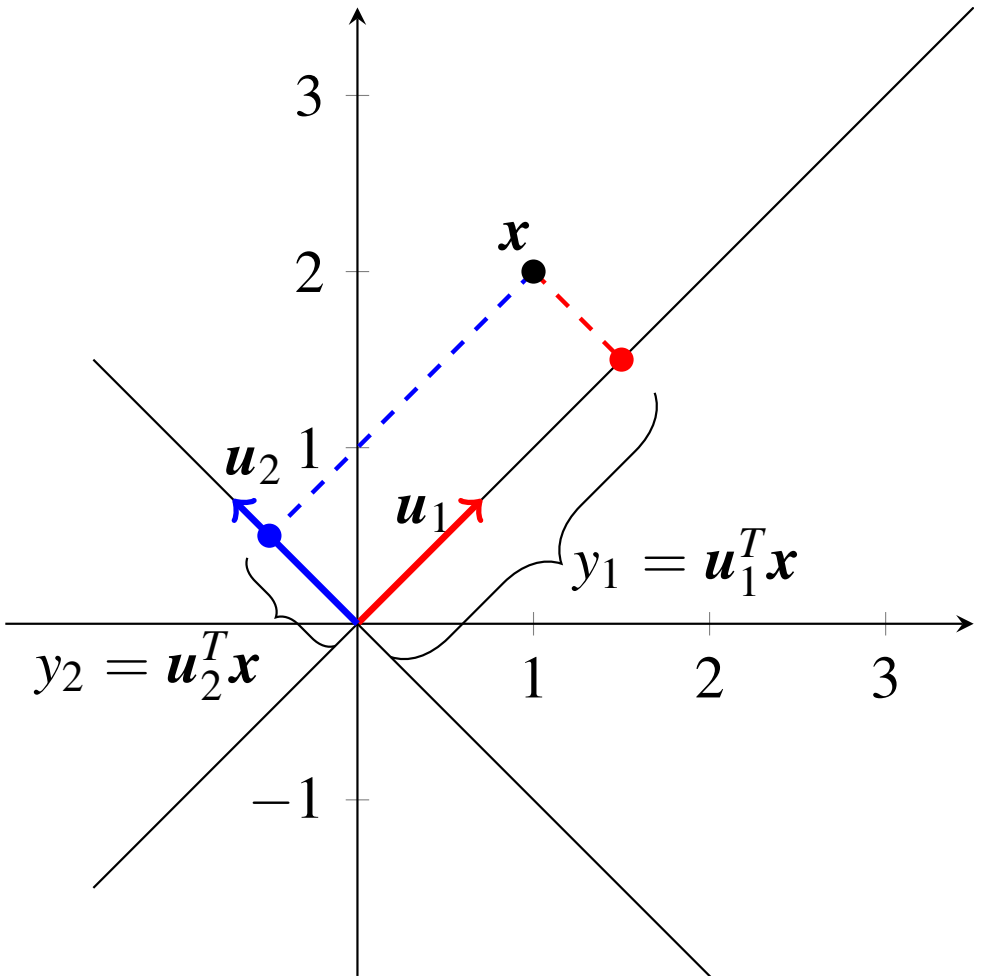


$\boldsymbol{u}$ is a unit vector representing a direction

$y = \boldsymbol{u}^T\boldsymbol{x}$ is the projection of $\boldsymbol{x}$ over $\boldsymbol{u}$

$\widehat{\boldsymbol{x}} = y\boldsymbol{u} = (\boldsymbol{u}^T\boldsymbol{x})\boldsymbol{u}$ is the representation of the projected point in the original space (reconstruction)

Projecting a point in an m-dimensional (sub)space $\boldsymbol{U} = [\boldsymbol{u}_1 \dots \boldsymbol{u}_m]$ ($m = 2$ in the example):



The columns of $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2]$ form a basis of $\mathbb{R}^2$

$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_1^T\boldsymbol{x} \\ \boldsymbol{u}_2^T\boldsymbol{x} \end{bmatrix} = \boldsymbol{U}^T\boldsymbol{x}$ is the projection of $\boldsymbol{x}$ over the column space of $\boldsymbol{U}$

The representation (reconstruction) of $\boldsymbol{y}$ in the original space can be obtained as

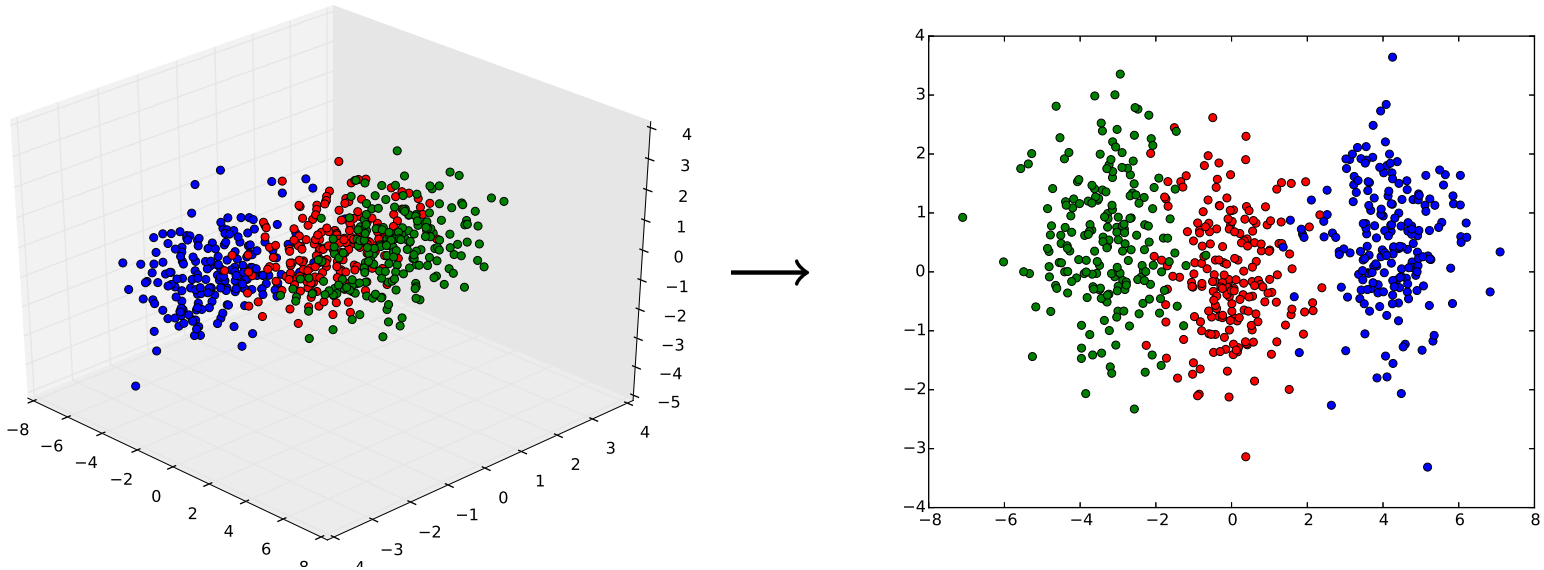
$$\widehat{\boldsymbol{x}} = y_1\boldsymbol{u}_1 + y_2\boldsymbol{u}_2 = \boldsymbol{U}\boldsymbol{y} = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{x}$$

In this example, since $\boldsymbol{U}$ is full rank, $\widehat{\boldsymbol{x}} = \boldsymbol{x}$

Principal Component Analysis

We are given a zero-mean dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots \boldsymbol{x}_K\}$, with $\boldsymbol{x}_i \in \mathbb{R}^n$

We want to find the subspace of $\mathbb{R}^n$ that allows preserving *most* of the information



Principal Component Analysis

A subspace can be represented as a matrix $\boldsymbol{P} \in \mathbb{R}^{n \times m}$ whose columns are orthonormal

The columns of $\boldsymbol{P}$ form a basis of a subspace of $\mathbb{R}^n$ with dimension m

The projection of $\boldsymbol{x}$ over the subspace is given by $\boldsymbol{y} = \boldsymbol{P}^T \boldsymbol{x}$:

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{p}_1^T \boldsymbol{x} \\ \boldsymbol{p}_2^T \boldsymbol{x} \\ \vdots \\ \boldsymbol{p}_m^T \boldsymbol{x} \end{bmatrix}$$

where $\boldsymbol{p}_1 \ldots \boldsymbol{p}_m$ are the columns of $\boldsymbol{P}$

We can compute the coordinates of the projected point $\boldsymbol{y}$ in the original space as $\hat{\boldsymbol{x}} = \boldsymbol{P}\boldsymbol{y}$

Principal Component Analysis

We need to define a criterion for estimating $\boldsymbol{P}$

A reasonable criterion may be the minimization of the average reconstruction error (K is the number of samples)

$$\frac{1}{K}\sum_{i=1}^{K}\|\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i\|^2=\frac{1}{K}\sum_{i=1}^{K}\|\boldsymbol{x}_i-\boldsymbol{P}\boldsymbol{y}_i\|^2=\frac{1}{K}\sum_{i=1}^{K}\left\|\boldsymbol{x}_i-\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i\right\|^2$$

We therefore want to solve

$$\boldsymbol{P}^*=\arg\min_{\boldsymbol{P}}\frac{1}{K}\sum_{i=1}^{K}\|\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i\|^2=\arg\min_{\boldsymbol{P}}\frac{1}{K}\sum_{i=1}^{K}\left\|\boldsymbol{x}_i-\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i\right\|^2$$

Principal Component Analysis

We can rewrite the objective function as

$$\mathcal{L}(\boldsymbol{P}) = \frac{1}{K}\sum_{i=1}^{K}\left\|\boldsymbol{x}_i - \boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i\right\|^2 = \frac{1}{K}\sum_{i=1}^{K}\left(\boldsymbol{x}_i^T\boldsymbol{x}_i - 2\boldsymbol{x}_i^T\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i + \boldsymbol{x}_i^T\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i\right)$$

$$= \frac{1}{K}\sum_{i=1}^{K}\left(\boldsymbol{x}_i^T\boldsymbol{x}_i - 2\boldsymbol{x}_i^T\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i + \boldsymbol{x}_i^T\boldsymbol{P}\boldsymbol{P}^T\boldsymbol{x}_i\right)$$

$$= \frac{1}{K}\sum_{i=1}^{K}\left(\boldsymbol{x}_i^T\boldsymbol{x}_i - \mathrm{Tr}(\boldsymbol{P}^T\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P})\right)$$

where Tr represents the trace of a matrix, and we used the fact that $\boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{I}$

Here we used the fact that, for any two vectors $\boldsymbol{v}$ and $\boldsymbol{w}$

$$\boldsymbol{v}^T\boldsymbol{w} = \mathrm{Tr}(\boldsymbol{v}^T\boldsymbol{w})$$

and the property that

$$\mathrm{Tr}(\boldsymbol{A}\boldsymbol{B}) = \mathrm{Tr}(\boldsymbol{B}\boldsymbol{A})$$

Principal Component Analysis

Since Tr is a linear operator, minimizing $\mathcal{L}$ is equivalent to maximizing

$$\widehat{\mathcal{L}}(\boldsymbol{P}) = \mathrm{Tr}\left(\boldsymbol{P}^T \left[\frac{1}{K}\sum_{i=1}^{K} \boldsymbol{x}_i \boldsymbol{x}_i^T\right] \boldsymbol{P}\right)$$

We require that $\boldsymbol{P}$ is an orthogonal matrix

It can be shown that the optimal solution is then given by the matrix $\boldsymbol{P}$ whose columns are the m eigenvectors of $\frac{1}{K}\sum_{i=1}^{K} \boldsymbol{x}_i \boldsymbol{x}_i^T$ corresponding to the m largest eigenvalues

Principal Component Analysis

Let

$$\frac{1}{K}\sum_{i=1}^{K} \boldsymbol{x}_i \boldsymbol{x}_i^T = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$$

be an eigen–decomposition of the matrix, with $\boldsymbol{\Sigma}$ containing the eigenvalues in descending order

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}, \quad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$$

and

$$\boldsymbol{U} = [\boldsymbol{u}_1 \ldots \boldsymbol{u}_m, \boldsymbol{u}_{m+1} \ldots \boldsymbol{u}_n]$$

Then

$$\boldsymbol{P}^* = [\boldsymbol{u}_1 \ldots \boldsymbol{u}_m]$$

i.e., $\boldsymbol{P}^*$ corresponds to the first m columns of $\boldsymbol{U}$

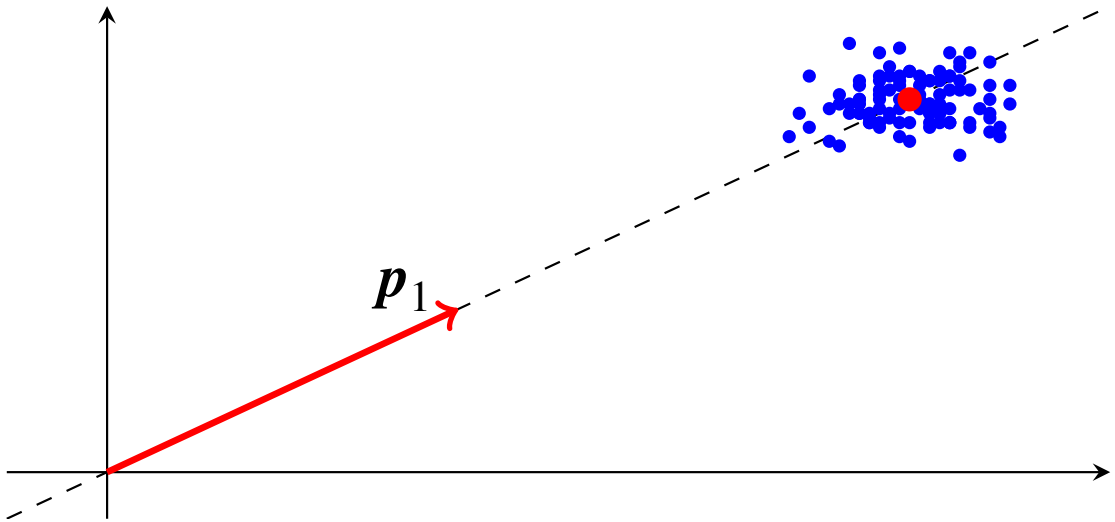
Principal Component Analysis

What happens if the dataset is not zero-mean?

$\boldsymbol{P}$ represents a subspace, whose axes pass through the origin

If the dataset is far from the origin, the first PCA direction will approximately connect the origin and the dataset mean

This direction (in most cases) is not very interesting



Principal Component Analysis

We can recast the problem as looking for the projection surface (line, plane, ...) that minimizes the reconstruction error

In practice, we can cast the problem as jointly looking for a dataset shift $\boldsymbol{m}$ and a subspace $\boldsymbol{U}$ over which we can project the shifted data

An optimal solution for the shift is given by the dataset mean

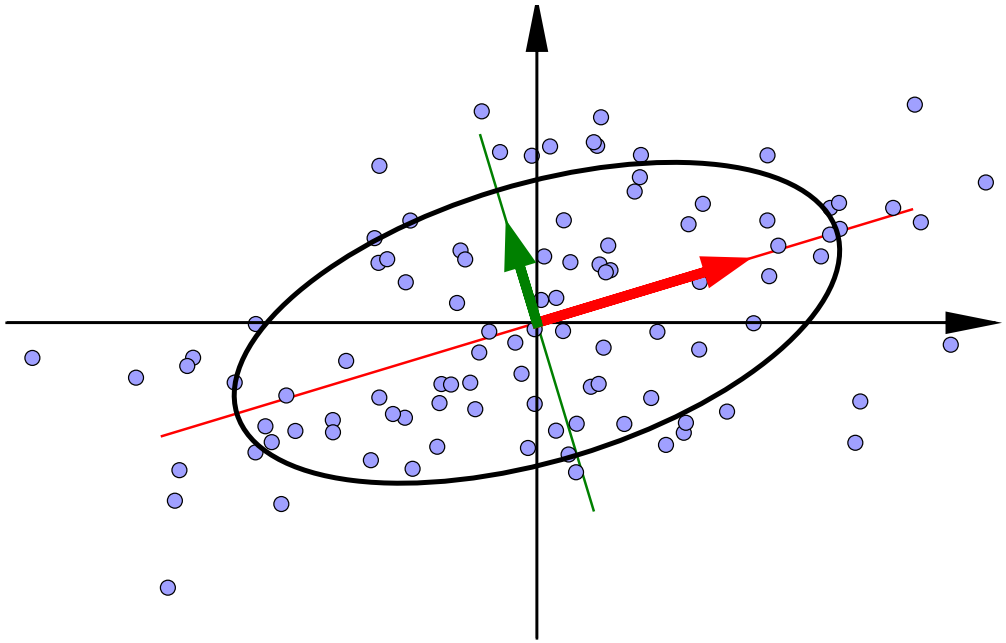
In practical terms, we remove the dataset mean before computing PCA

If the dataset mean is $\bar{\boldsymbol{x}}$, the PCA subspace is computed from the eigenvectors of the empirical covariance matrix

$$\frac{1}{K}\sum_{i}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T$$

Principal Component Analysis

PCA can be interpreted as the linear mapping that preserves the directions with highest variance



The axes of the ellipse correspond to the principal directions

Principal Component Analysis

Consider the covariance matrix

$$\boldsymbol{C} = \frac{1}{K} \sum_{i} (\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T$$

We can compute the eigen–decomposition of $\boldsymbol{C}$

$$\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$$

$\boldsymbol{U}$: Eigenvectors matrix

$\boldsymbol{\Sigma}$: Diagonal eigenvalues matrix

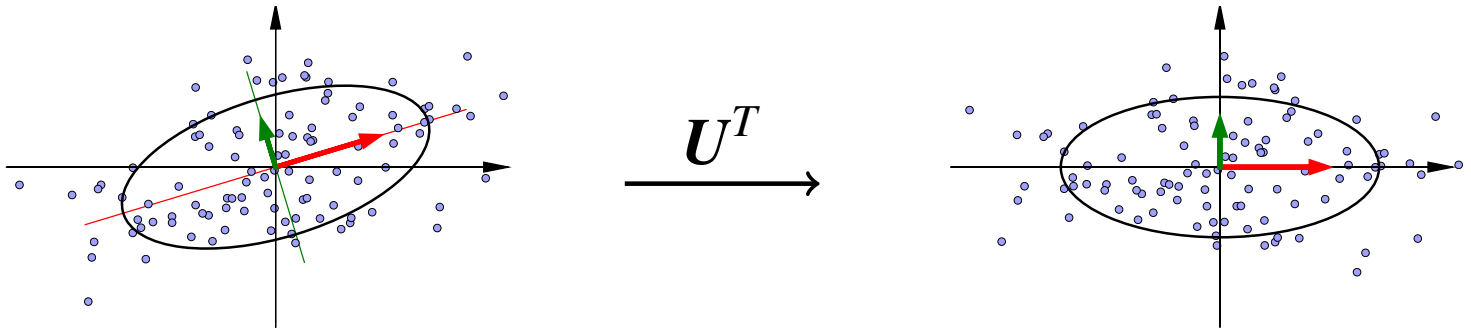
Remember that $\boldsymbol{U}^{-1} = \boldsymbol{U}^T$, thus $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$

Principal Component Analysis

Projecting our data centered points over $\boldsymbol{U}^T$ we obtain

$$\boldsymbol{C}' = \frac{1}{K}\sum_i \boldsymbol{U}^T (\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T \boldsymbol{U} = \boldsymbol{U}^T \left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T\right) \boldsymbol{U} = \boldsymbol{\Sigma}$$

Projection over $\boldsymbol{U}^T$ transforms our data so that the different directions are uncorrelated



To keep only the m directions with highest variance we keep only the first m transformed directions

Principal Component Analysis

Usually PCA is applied directly to centered data

- Compute sample mean $\bar{\boldsymbol{x}}$
- Center data: $\boldsymbol{z}_i = \boldsymbol{x}_i - \bar{\boldsymbol{x}}$
- Compute the sample covariance matrix

$$\boldsymbol{C} = \frac{1}{K} \sum_i (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T = \frac{1}{k} \sum_i \boldsymbol{z}_i \boldsymbol{z}_i^T$$

- Compute the eigen–decomposition of $\boldsymbol{C} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{U}^T$
- Project the data in the subspace spanned by the m columns of $\boldsymbol{U}$ corresponding to the m highest eigenvalues (matrix $\boldsymbol{P}$): $\boldsymbol{y}_i = \boldsymbol{P}^T \boldsymbol{z}_i = \boldsymbol{P}^T (\boldsymbol{x}_i - \bar{\boldsymbol{x}})$
- Reconstruction requires inverting the process: $\hat{\boldsymbol{x}}_i = \overline{\boldsymbol{P}} \boldsymbol{y}_i + \bar{\boldsymbol{x}}$

Principal Component Analysis

Selection of optimal m can be done by cross–validation using a validation set

We can also select m as to retain a given percentage t (e.g. 95%) of the variance of the data

Remember that each eigenvalue corresponds to the variance along the corresponding axis

We choose m as the lowest number for which the sum of the first m eigenvalues divided by the sum of all eigenvalues is larger than t

$$\min_m m \quad \text{s.t.} \quad \frac{\sum_{i=1}^{m} \sigma_i}{\sum_{i=1}^{n} \sigma_i} \geq t$$

where σ_i is the i–th largest eigenvalue (diagonal element of Σ)

Principal Component Analysis

Computing the sample covariance can be difficult when the feature space is very large

Different solutions

- Truncated Singular Value Decomposition
- Probabilistic PCA

For smaller dataset the standard approach is sufficient

MNIST Dataset

To compare different methods we consider the MNIST handwritten digits dataset

Task: classification of handwritten digits

- Classes are digits $0, 1, \ldots, 9$
- Images have already been normalized (centered and scaled to a square 28×28 shape)
- Gray-scale images (originally binary)
- 60 000 training images (6000 for each digit)
- 10 000 test images

MNIST Dataset

Train samples:

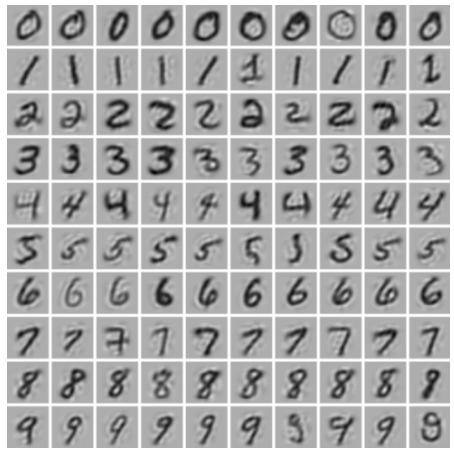
Test samples:

In general, we will consider both multi–class and pair–wise binary classification tasks

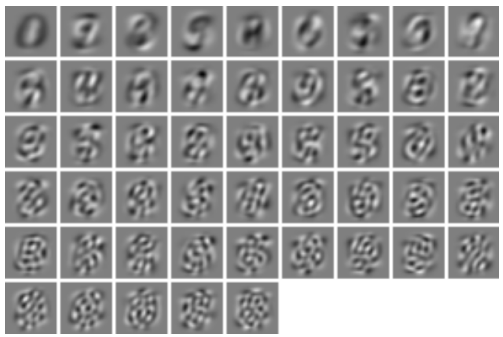
In the following we show the results of PCA applied to the images

Principal Component Analysis

50 dimensional PCA

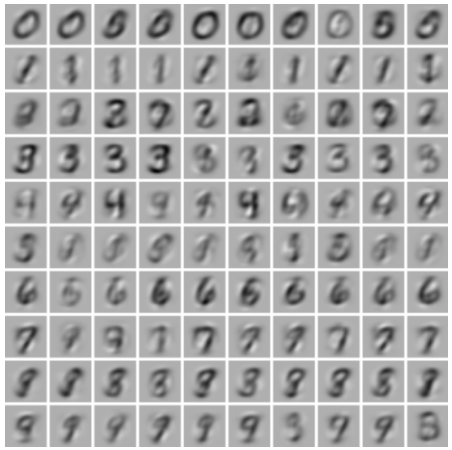


$\boldsymbol{U}$:

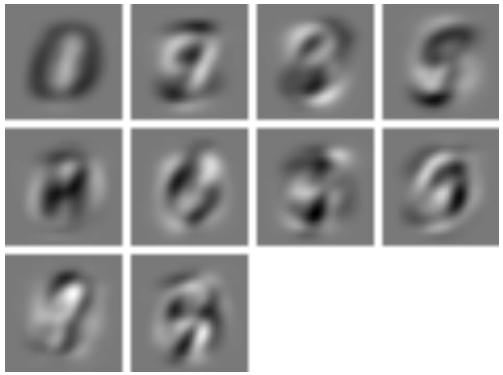


Principal Component Analysis

10 dimensional PCA



$\boldsymbol{U}$:



Dimensionality Reduction

To analyze the results of PCA as pre-processing for classification we consider a very simple classifier based on Euclidean distance

For each class, we compute the mean vector $\boldsymbol{\mu}_c = \frac{1}{n_c} \sum_{i=1}^{n_c} \boldsymbol{x}_{c,i}$

For a test sample $\boldsymbol{x}_t$ we predict its label as the label of the class whose mean is closest to the test sample itself:

$$c_t = \arg\min_c \|(\boldsymbol{x}_t - \boldsymbol{\mu}_c)\|^2$$

Dimensionality Reduction

We measure the error rate as the number of incorrect classified samples over the total number of samples (we will see better measures in the next weeks)

	m[1]			
w/o PCA	100	50	9	5
18.0%	18.1%	18.2%	25.5%	35.9%

Most of the dimensions can be safely removed — the results with 50 and 100 dimensions are very close to those of the full image

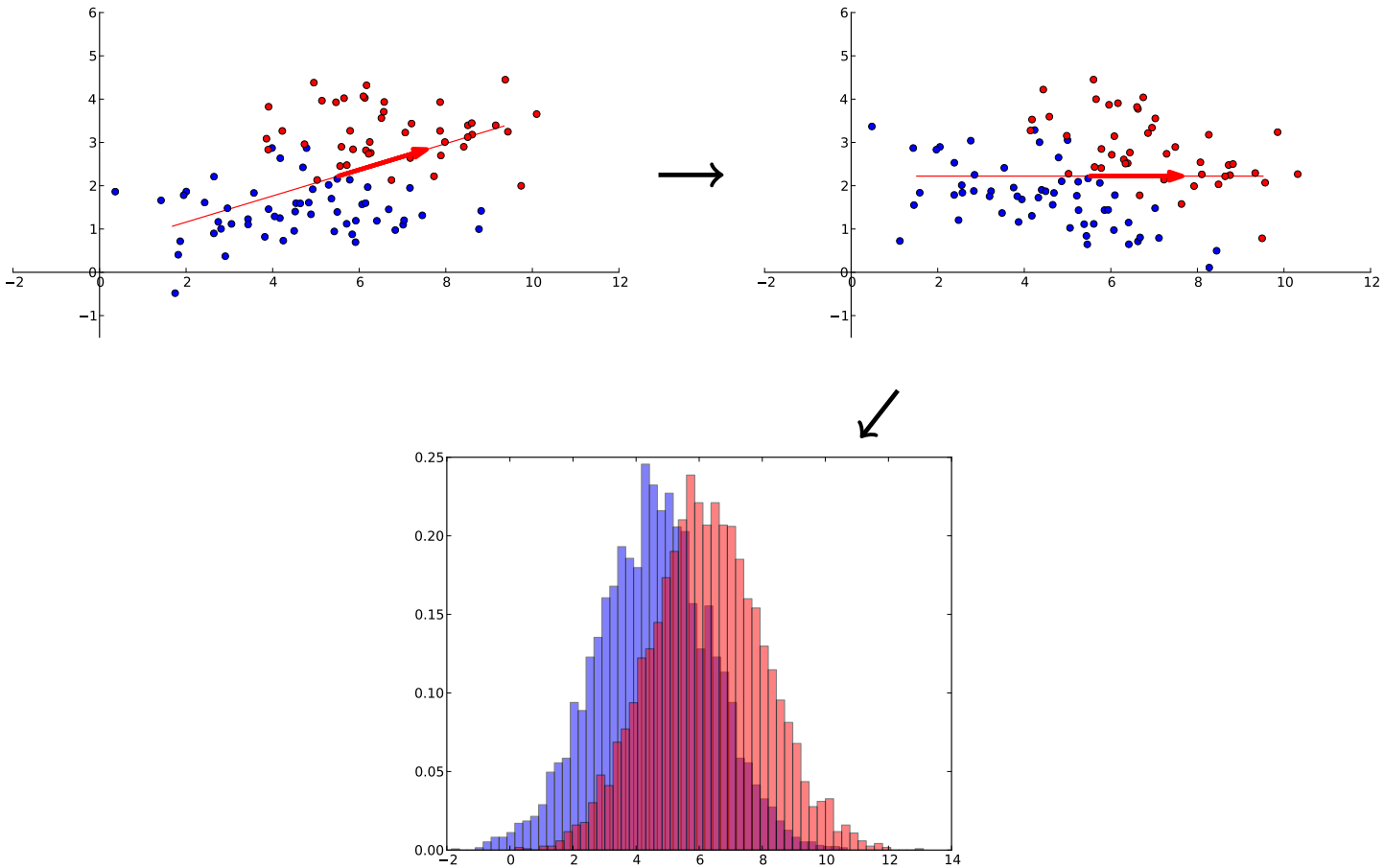
If we remove too much information, classification accuracy drops significantly

For the Euclidean classifier on MNIST, PCA is not helpful in removing unwanted variability

[1] Remember that the optimal dimensionality m should be selected according to results on a validation set!

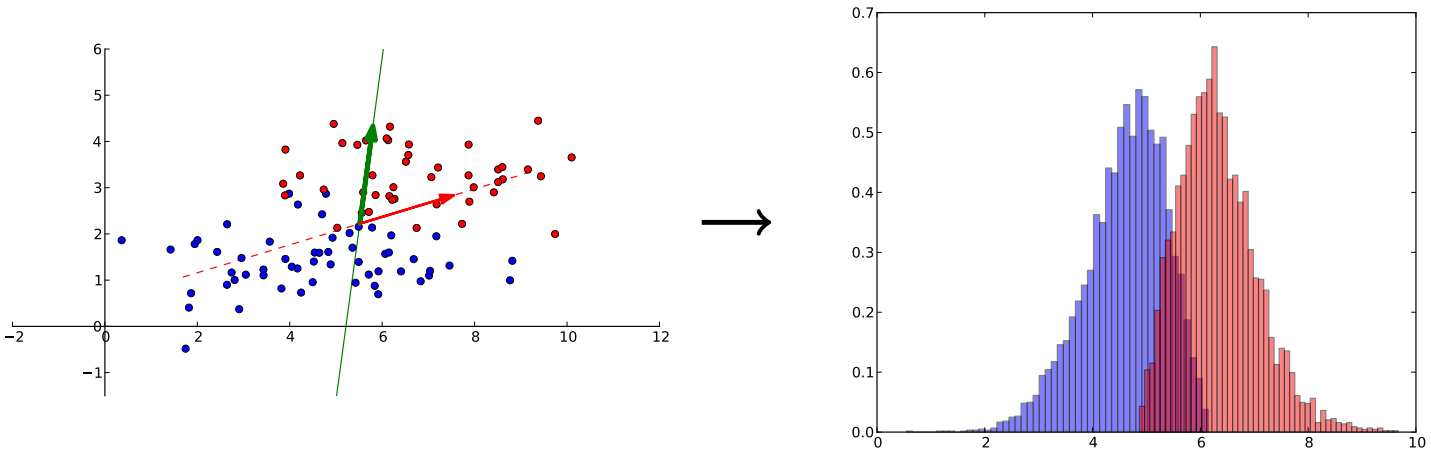
Linear Discriminant Analysis

PCA is unsupervised: no guarantee of obtaining discriminant directions



Linear Discriminant Analysis

We want a transformation that allows us to better separate the classes



We represent a direction as a unit vector $\boldsymbol{w}$

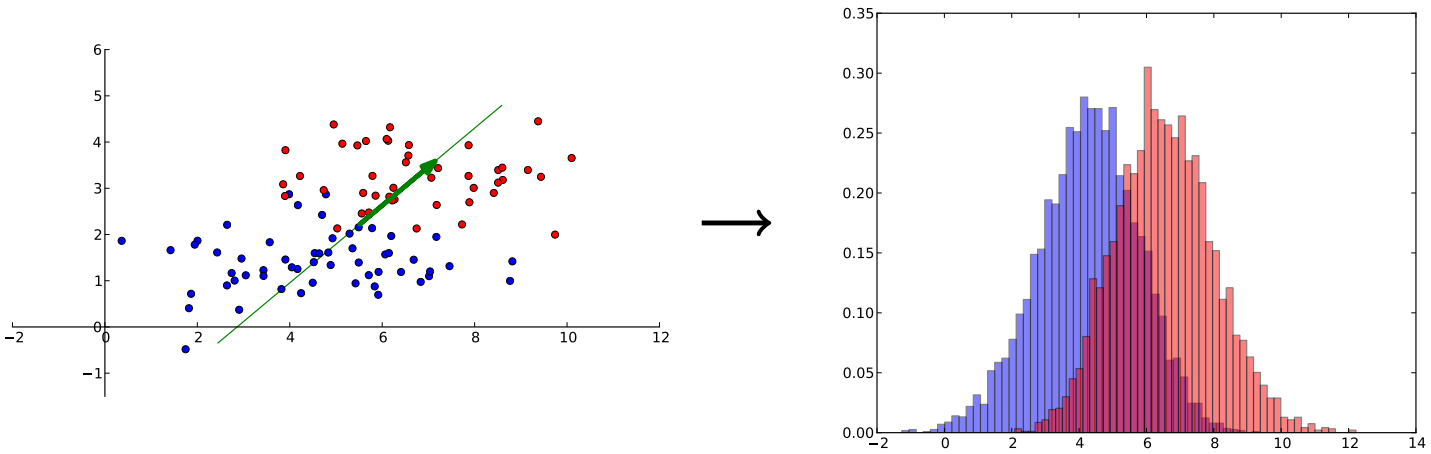
The projected point is $y = \boldsymbol{w}^T \boldsymbol{x}$ (a scalar)

Red line: PCA
Green line: Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis

We could consider the line connecting the class means

$$\boldsymbol{w} \propto \boldsymbol{\mu}_2 - \boldsymbol{\mu}_1$$



Problem: if the data points of each class are scattered along the same directions of the class mean, we still cannot properly separate the classes

Linear Discriminant Analysis

Fisher Linear Discriminant Analysis: find a direction that has a large separation between the classes and small spread inside each class

We measure spread in terms of class covariance

LDA: maximize the between-class variability over within-class variability ratio for the transformed samples:

$$\max_{\boldsymbol{w}} \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$$

Linear Discriminant Analysis

The between and within class variability matrices are defined as

$$\boldsymbol{S}_B \triangleq \frac{1}{N} \sum_{c=1}^{K} n_c \left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right) \left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right)^T$$

$$\boldsymbol{S}_W \triangleq \frac{1}{N} \sum_{c=1}^{K} \sum_{i=1}^{n_c} \left(\boldsymbol{x}_{c,i} - \boldsymbol{\mu}_c\right) \left(\boldsymbol{x}_{c,i} - \boldsymbol{\mu}_c\right)^T$$

where $\boldsymbol{x}_{c,i}$ is the i-th sample of class c, n_c is the number of samples of class c, K is the total number of classes, N is the total number of samples $N = \sum_{c=1}^{K} n_c$, and

- $\boldsymbol{\mu}$ is the dataset mean $\boldsymbol{\mu} = \frac{1}{N} \sum_{c=1}^{K} \sum_i \boldsymbol{x}_{c,i}$
- $\boldsymbol{\mu}_c$ is the mean of class c $\boldsymbol{\mu}_c = \frac{1}{n_c} \sum_{i=1}^{n_c} \boldsymbol{x}_{c,i}$

Linear Discriminant Analysis

The between class covariance matrix can be interpreted as a covariance matrix for the class means, where each class is weighted by the corresponding sample size n_c

The within class covariance matrix can be seen as a (also weighted) average of the covariance matrix of each class

We can observe that

$$\boldsymbol{S}_B + \boldsymbol{S}_W = \frac{1}{N} \sum_{c=1}^{K} \sum_{i=1}^{n_c} (\boldsymbol{x}_{c,i} - \boldsymbol{\mu}) (\boldsymbol{x}_{c,i} - \boldsymbol{\mu})^T$$

i.e., the covariance matrix of the dataset as a whole

Linear Discriminant Analysis

Since we are looking for a discriminant direction $\boldsymbol{w}$, we now consider the between and within class variance of the projected samples $\boldsymbol{w}^T\boldsymbol{x}$

The global mean and class means in the direction $\boldsymbol{w}$ are simply

$$m = \boldsymbol{w}^T\boldsymbol{\mu}, \quad m_c = \boldsymbol{w}^T\boldsymbol{\mu}_c$$

The between and within class variance over the direction $\boldsymbol{w}$ can also be computed as:

$$s_B = \frac{1}{N}\sum_{c=1}^{K} n_c \left(\boldsymbol{w}^T\boldsymbol{\mu}_c - \boldsymbol{w}^T\boldsymbol{\mu}\right)\left(\boldsymbol{w}^T\boldsymbol{\mu}_c - \boldsymbol{w}^T\boldsymbol{\mu}\right)^T = \boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w}$$

$$s_W = \frac{1}{N}\sum_{c=1}^{K}\sum_{i=1}^{n_c} n_c \left(\boldsymbol{w}^T\boldsymbol{x}_{c,i} - \boldsymbol{w}^T\boldsymbol{\mu}_c\right)\left(\boldsymbol{w}^T\boldsymbol{x}_{c,i} - \boldsymbol{w}^T\boldsymbol{\mu}_c\right)^T = \boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w}$$

Linear Discriminant Analysis

Fisher discriminant analysis defines as criterion of optimality the maximization of the ratio of between and within class variance for the projected points

We assume that $\boldsymbol{S}_W$ is full rank, thus the objective function is

$$\mathcal{L}(\boldsymbol{w}) = \frac{s_B}{s_W} = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$$

Note that the criterion does not depend on the scale of $\boldsymbol{w}$, i.e. if $\boldsymbol{w}$ is a maximizer of $\mathcal{L}$, then $\alpha \boldsymbol{w}$ is also a maximizer of $\mathcal{L}$. We can therefore select a maximizer with unit norm.

Linear Discriminant Analysis

We can find an optimum by solving $\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}) = 0$, where $\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w})$ is the gradient of $\mathcal{L}$:

$$\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}) = 2\frac{\boldsymbol{S}_B\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w}} - 2\frac{\boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w}\boldsymbol{S}_W\boldsymbol{w}}{(\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w})^2} = 0$$

The optimum is obtained for

$$(\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w})\boldsymbol{S}_B\boldsymbol{w} = (\boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w})\boldsymbol{S}_W\boldsymbol{w}$$

i.e.

$$\boldsymbol{S}_W^{-1}\boldsymbol{S}_B\boldsymbol{w} = \lambda(\boldsymbol{w})\boldsymbol{w}$$

where

$$\lambda(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) = \frac{\boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w}}$$

Linear Discriminant Analysis

We can observe that

- The optimal solution is an eigenvector of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$
- The eigenvalue corresponding to solution $\boldsymbol{w}$ is $\lambda(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w})$, i.e. the value of the ratio we want to maximize

The maximum of $\mathcal{L}$ is thus the eigenvector of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$ corresponding to the largest eigenvalue

Linear Discriminant Analysis

The method was originally introduced to solve binary problems

Indeed, once we have estimated $\boldsymbol{w}$, we can project our test samples over $\boldsymbol{w}$, and assign the class according to whether the projected value (score) is larger or lower than a given threshold[2]:

$$C(\boldsymbol{x}_t) = \begin{cases} C_1 & \text{if } \boldsymbol{w}^T \boldsymbol{x}_t \geq t \\ C_2 & \text{if } \boldsymbol{w}^T \boldsymbol{x}_t < t \end{cases}$$

[2] How to select a good threshold will be a topic for next classes

Linear Discriminant Analysis

For the binary problem, we can express the between-class covariance matrix as

$$\boldsymbol{S}_B = k(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$$

where k is a constant that depends on the number of samples of each class but is irrelevant to find the optimal direction.

In this case, we can observe that $\text{rank}(\boldsymbol{S}_B) = 1$, thus $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$ has a single non-zero eigenvalue

It can be verified that the eigenvector of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$ associated to the non-zero eigenvalue is

$$\boldsymbol{w} \propto \boldsymbol{S}_W^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)$$

Linear Discriminant Analysis

Despite being originally introduced to solve binary classification problems, LDA has found large success as a dimensionality reduction technique

In this case, we are interested in looking for the m most discriminant directions

We represent these directions as a matrix $\boldsymbol{W}$, whose columns contain the directions we want to find

Notice that we do not require that $\boldsymbol{W}$ is orthogonal

If we want, we can nevertheless find a basis for the subspace spanned by the columns of $\boldsymbol{W}$

Linear Discriminant Analysis

The projected points are computed as $\widehat{\boldsymbol{x}} = \boldsymbol{W}^T \boldsymbol{x}$

We can express the projected between and within class covariance matrices as

$$\widehat{\boldsymbol{S}_B} = \boldsymbol{W}^T \boldsymbol{S}_B \boldsymbol{W}$$
$$\widehat{\boldsymbol{S}_W} = \boldsymbol{W}^T \boldsymbol{S}_W \boldsymbol{W}$$

Different criteria can be used to generalize the 1-dimensional case

A common one looks for the maximizer of

$$\mathcal{L} = \mathrm{Tr}\left(\widehat{\boldsymbol{S}_W}^{-1} \widehat{\boldsymbol{S}_B}\right)$$

Linear Discriminant Analysis

It can be shown that the solution is given by the m (right) eigenvectors corresponding to the m largest eigenvalues of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$

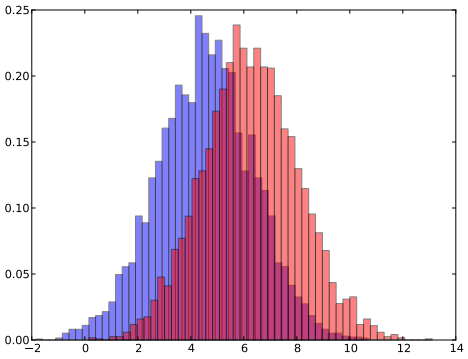
We can also compute the solution by solving the generalized eigenvalue problem $\boldsymbol{S}_B\boldsymbol{w} = \lambda\boldsymbol{S}_W\boldsymbol{w}$

Notice that, from the definition of $\boldsymbol{S}_B$, the number of non–zero eigenvalues is at most $C - 1$

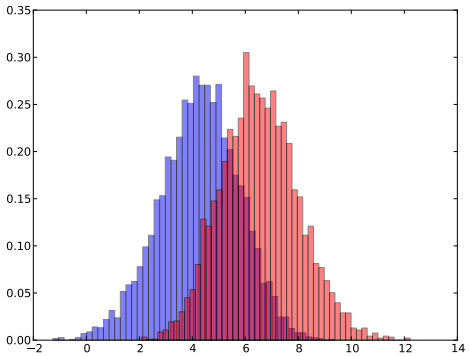
Therefore, LDA allows estimating at most $C - 1$ directions

Linear Discriminant Analysis

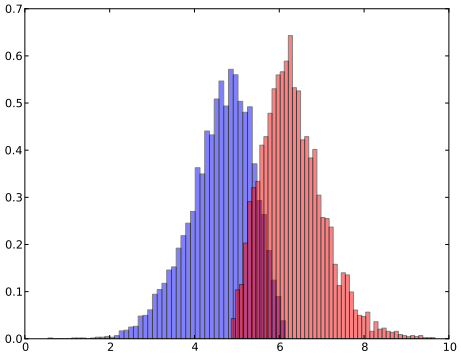
PCA



Eigenvectors of $\boldsymbol{S}_B$



LDA



Linear Discriminant Analysis

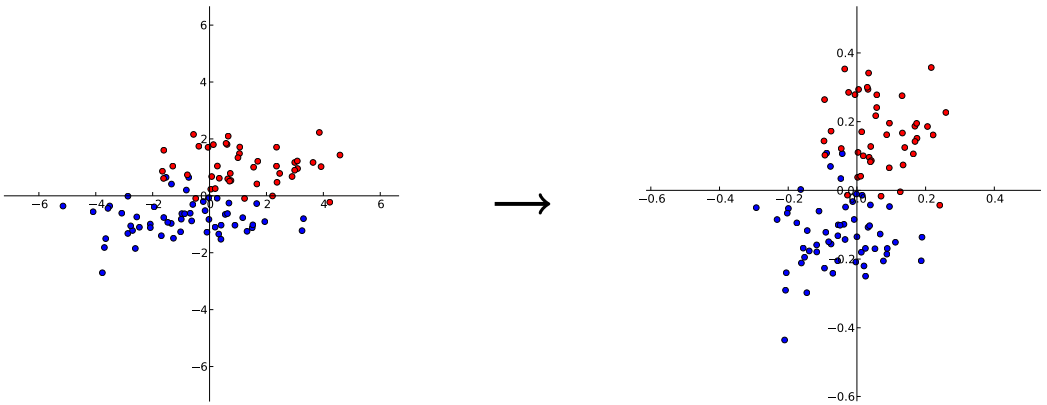
A way to solve the generalized eigenvalue problem, and thus find the LDA matrix $\boldsymbol{W}$, consists in the joint diagonalization of $\boldsymbol{S}_W$ and $\boldsymbol{S}_B$ that makes $\boldsymbol{S}_W$ become the identity matrix and $\boldsymbol{S}_B$ become a diagonal matrix

Remember that applying the linear transformation $\tilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ to our dataset, covariance matrices transform as $\tilde{\boldsymbol{\Sigma}} = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T$

Linear Discriminant Analysis

Whiten $\boldsymbol{S}_W$

- Compute the eigen–value decomposition $\boldsymbol{S}_W = \boldsymbol{U}_W \boldsymbol{\Sigma}_W \boldsymbol{U}_W^T$
- Apply the whitening transformation[3] described by $\boldsymbol{P}_W = \boldsymbol{U}_W \boldsymbol{\Sigma}_W^{-\frac{1}{2}} \boldsymbol{U}_W^T$
- $\boldsymbol{S}_W \rightarrow \boldsymbol{I}$, $\boldsymbol{S}_B \rightarrow \boldsymbol{P}\boldsymbol{S}_B\boldsymbol{P}^T$

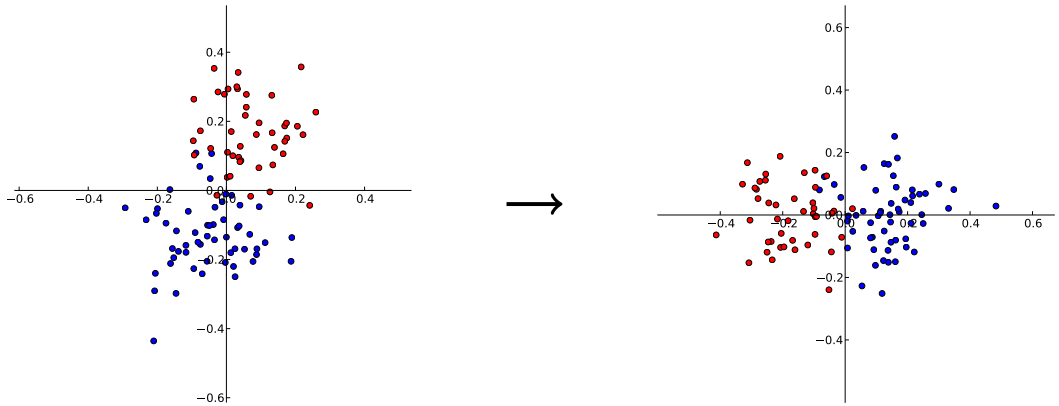


[3] There are different methods to compute the whitening transformation. Indeed, matrix $\boldsymbol{P}$ is defined up to a unitary transformation.

Linear Discriminant Analysis

Diagonalize the transformed $\boldsymbol{S}_B$

- Compute the eigenvalue decomposition $\boldsymbol{P}\boldsymbol{S}_B\boldsymbol{P}^T = \boldsymbol{U}_B\boldsymbol{\Sigma}_B\boldsymbol{U}_B^T$
- Diagonalize by projecting over $\boldsymbol{U}_B^T$
- $\boldsymbol{S}_W \rightarrow \boldsymbol{I}$, $\boldsymbol{S}_B \rightarrow \boldsymbol{\Sigma}_B$



The first m directions of the transformed samples correspond to the LDA subspace. This method will be further discussed in Laboratory 3.

Linear Discriminant Analysis

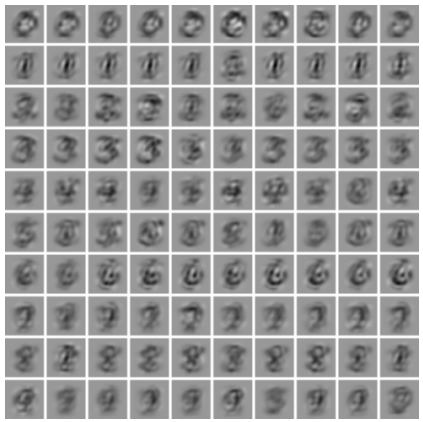
LDA assumes Gaussian–distributed noise

When the number of directions is large S_W can be singular or close to singular

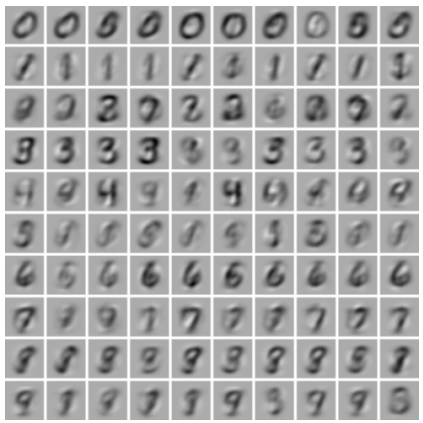
It is often helpful to pre–process our data using PCA before applying LDA

Linear Discriminant Analysis

In MNIST we have 9 classes, so we can have at most 9 directions



PCA+LDA



PCA

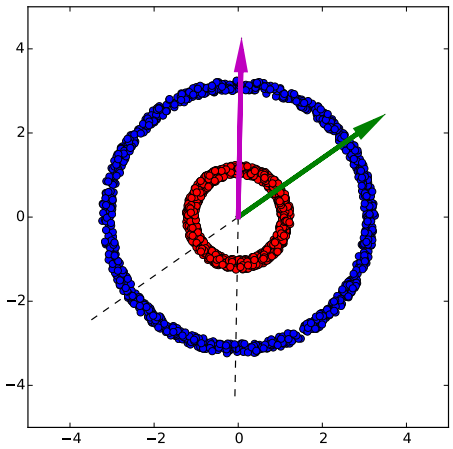
MNIST — Error rates for euclidean distance classifier[4]

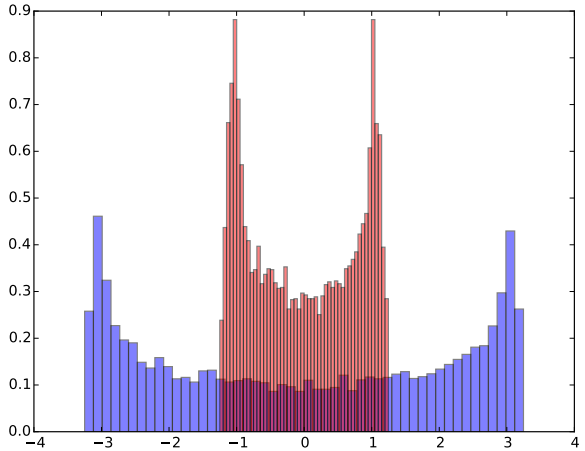
m	PCA	PCA+LDA
100	18.1%	—
50	18.2%	—
9	25.5%	12.2%
5	35.9%	17.9%

[4] Remember that the method and dimensionality m should be selected according to results on the validation set!

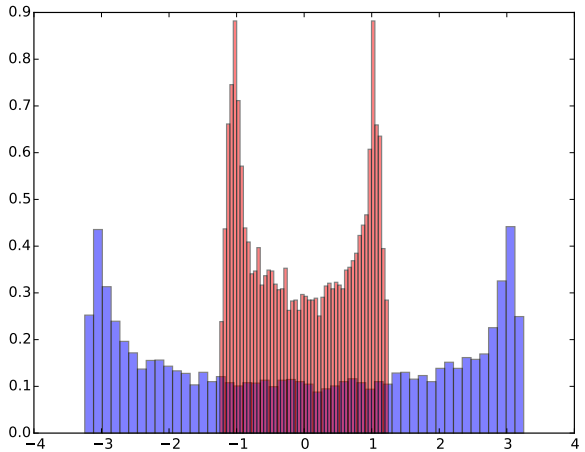
Non–linear Dimensionality Reduction

Linear transformations are not always suited for our data





PCA



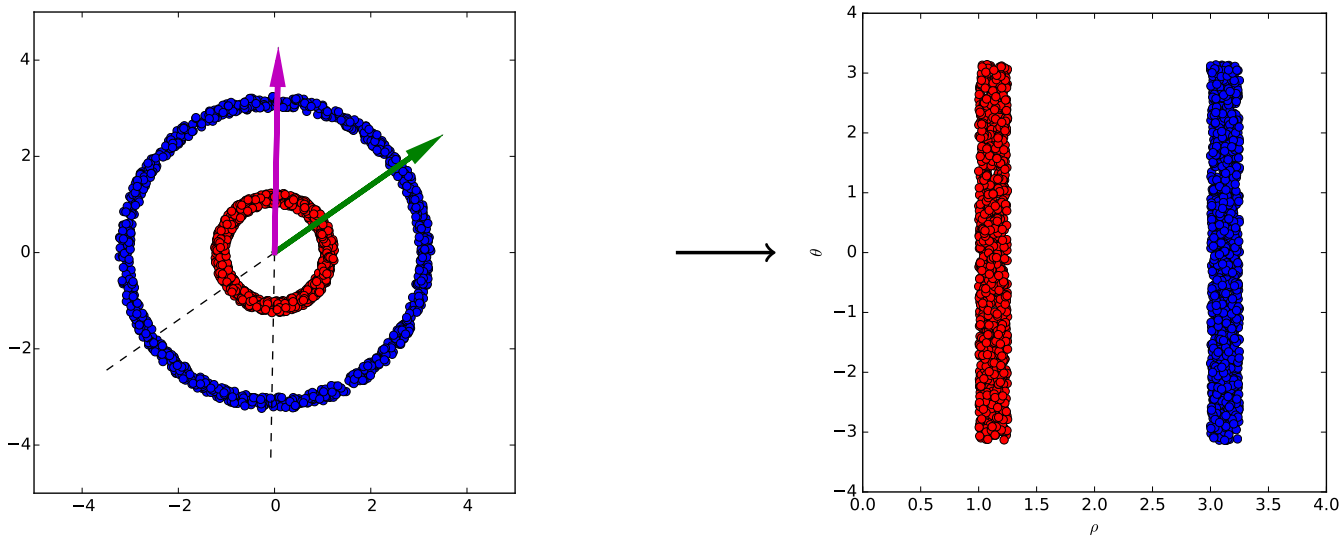
LDA

Non–linear Dimensionality Reduction

We can transform the features so that linear methods are suited for the transformed data

For example, we can represent the 2-dimensional data in the figure through polar coordinates :

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longrightarrow \boldsymbol{y} = f(\boldsymbol{x}) \begin{bmatrix} \rho(\boldsymbol{x}) \\ \theta(\boldsymbol{x}) \end{bmatrix}$$



Alternative non-linear methods will be presented in other courses