*** Applied Machine Learning Fundamentals ***
Principal Component Analysis

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2023/2024

Find all slides on GitHub (DaWe1992/Applied_ML_Fundamentals)

Lecture Overview

Agenda for this Unit

Section:

Introduction

- Why Dimensionality Reduction?
- Data Compression
- Data Visualization
- What is PCA?

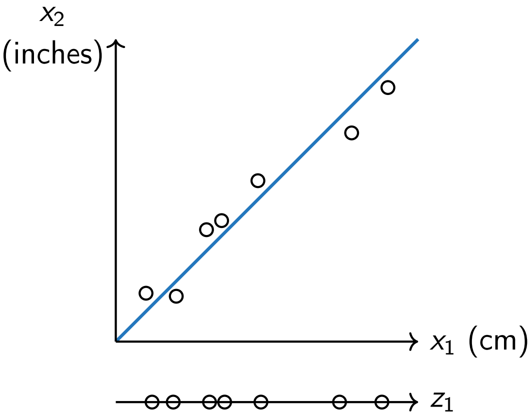
Why Dimensionality Reduction?

- Most data is high-dimensional
- Dimensionality reduction can be used for:
  - **Lossy (!)** data compression
  - Feature extraction
  - Data visualization

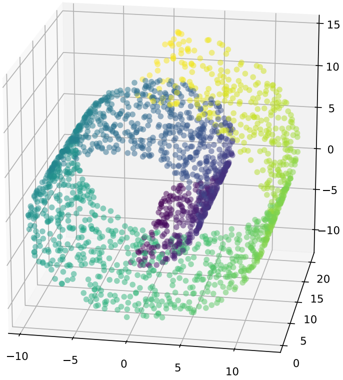
Dimensionality reduction can help to **speed up** learning algorithms substantially. Too many (correlated) features usually **decrease the performance** of the learning algorithm (cf. curse of dimensionality).

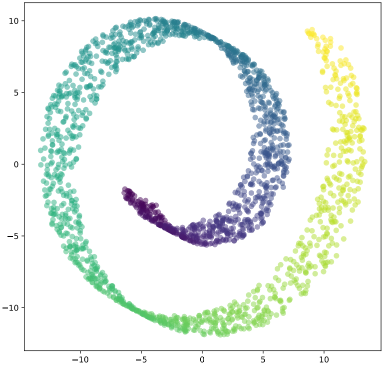
Use Case I: Data Compression / Feature Extraction

- The features *inches* and *cm* are closely related
- **Problems**:
  - Redundancy
  - More memory needed
  - Algorithms become slow
- **Solution**: Convert x_1 and x_2 into a new feature z_1 $(\mathbb{R}^2 \to \mathbb{R})$



Use Case II: Data Visualization

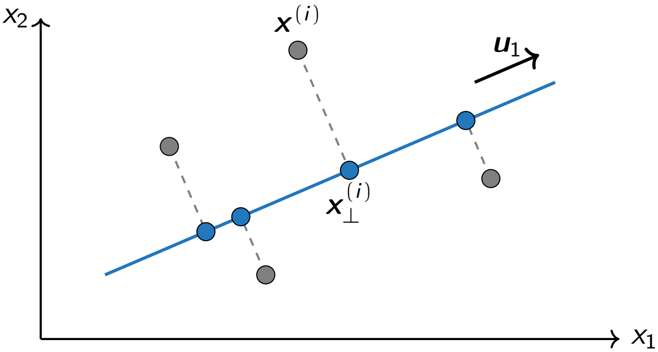




PCA: Principal Component Analysis

- PCA is an **unsupervised** algorithm
- It is known as the *Karhunen-Loève* transform
- PCA can be defined as the **orthogonal projection** of the data onto a lower dimensional **linear space** (*principal subspace*)
- Consider a data set of n observations $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n)}\}$
  - $\boldsymbol{x}^{(i)}$ is a real-valued vector in $\mathbb{R}^m$ (m-dimensional)
  - We want to project the data onto a space having dimensionality $k \ll m$, while **maximizing the variance of the projected data** ($\mathbb{R}^m \to \mathbb{R}^k$)
- **Remove dimensions which are the least informative of the data**

Orthogonal Projections

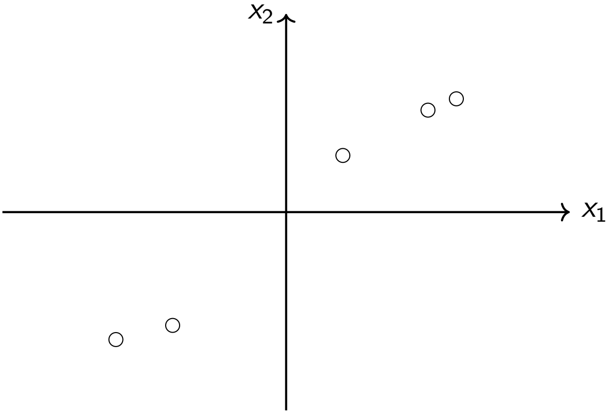


- $\boldsymbol{x}^{(i)}$ denote the original data points
- $\boldsymbol{x}_{\perp}^{(i)}$ is the orthogonal projection of $\boldsymbol{x}^{(i)}$ onto vector $\boldsymbol{u}_1$

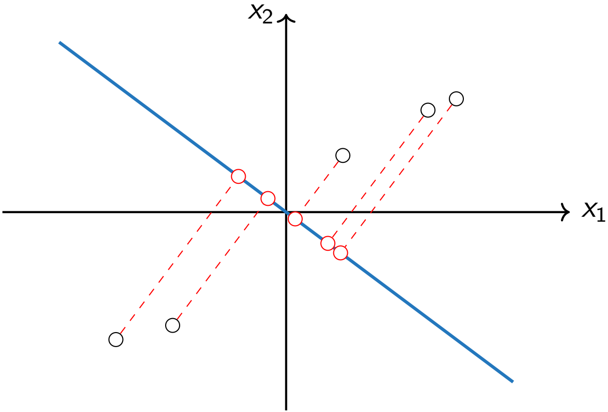
Section:

Maximum Variance Formulation

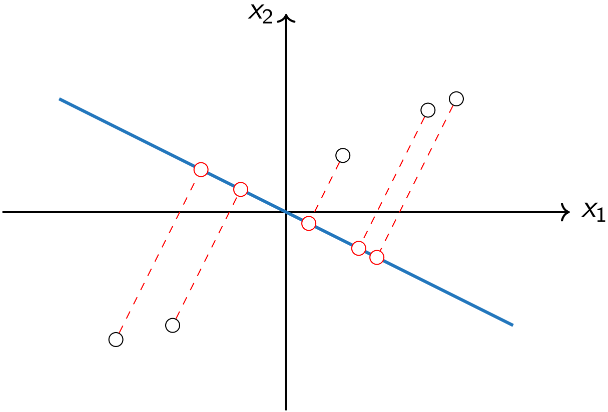
Maximum Variance Formulation



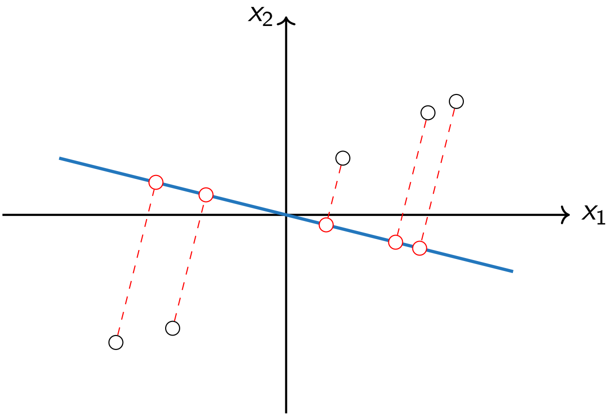
Maximum Variance Formulation



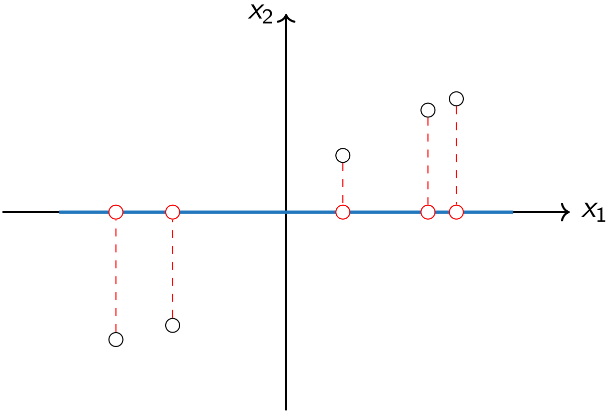
Maximum Variance Formulation



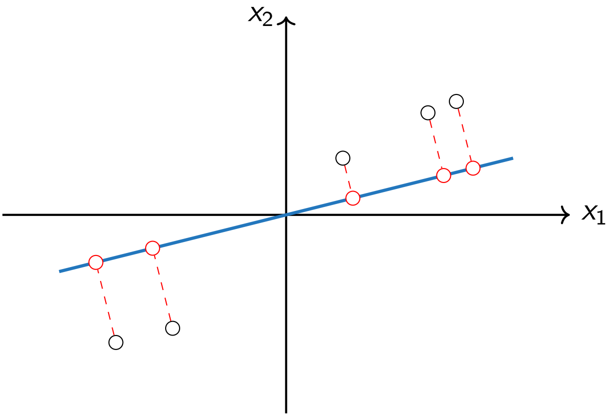
Maximum Variance Formulation



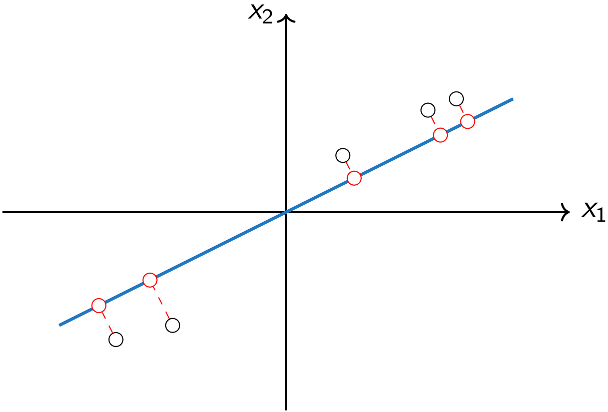
Maximum Variance Formulation



Maximum Variance Formulation



Maximum Variance Formulation



Maximum Variance Formulation (Ctd.)

- In the following we assume $k = 1$ (projection onto a line defined by a unit vector $\boldsymbol{u}_1$)
- Each data point $\boldsymbol{x}^{(i)}$ is projected onto a scalar value $\boldsymbol{u}_1^\intercal \boldsymbol{x}^{(i)}$
- The mean of the projected data is $\boldsymbol{u}_1^\intercal \overline{\boldsymbol{x}}$, where $\overline{\boldsymbol{x}}$ is the sample set mean:

$$\overline{\boldsymbol{x}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}^{(i)} \tag{1}$$

- The variance of the projected data is given by:

$$\frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{u}_1^\intercal \boldsymbol{x}^{(i)} - \boldsymbol{u}_1^\intercal \overline{\boldsymbol{x}} \right)^2 = \boldsymbol{u}_1^\intercal \boldsymbol{\Sigma} \boldsymbol{u}_1 \tag{2}$$

Maximum Variance Formulation (Ctd.)

- $\boldsymbol{\Sigma}$ is the covariance matrix defined by:

$$\boldsymbol{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} \overbrace{(\boldsymbol{x}^{(i)} - \overline{\boldsymbol{x}})(\boldsymbol{x}^{(i)} - \overline{\boldsymbol{x}})^{\intercal}}^{\textbf{Outer product} \rightarrow \textbf{matrix}} \tag{3}$$

- The projected variance $\boldsymbol{u}_1^{\intercal} \boldsymbol{\Sigma} \boldsymbol{u}_1$ is maximized with respect to $\boldsymbol{u}_1$
- Constraint: $\|\boldsymbol{u}_1\| = 1$, otherwise $\boldsymbol{u}_1$ grows unboundedly
- We have to solve the following optimization problem:

$$\max_{\boldsymbol{u}_1} \{\boldsymbol{u}_1^{\intercal} \boldsymbol{\Sigma} \boldsymbol{u}_1 + \lambda_1 (1 - \boldsymbol{u}_1^{\intercal} \boldsymbol{u}_1)\} \tag{4}$$

Maximum Variance Formulation (Ctd.)

- $\nabla_{\boldsymbol{u}_1}\{\boldsymbol{u}_1^\intercal \boldsymbol{\Sigma} \boldsymbol{u}_1 + \lambda_1(1 - \boldsymbol{u}_1^\intercal \boldsymbol{u}_1)\} \stackrel{!}{=} 0 \qquad \Longrightarrow \boldsymbol{\Sigma} \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$
- This is an **eigenvalue problem**
- The equation tells us that $\boldsymbol{u}_1$ must be an eigenvector of $\boldsymbol{\Sigma}$
- If we left-multiply by $\boldsymbol{u}_1^\intercal$ and use $\boldsymbol{u}_1^\intercal \boldsymbol{u}_1 = 1$, we see: $\boldsymbol{u}_1^\intercal \boldsymbol{\Sigma} \boldsymbol{u}_1 = \lambda_1$

The variance reaches a maximum, if we set $\boldsymbol{u}_1$ equal to the eigenvector having the largest eigenvalue λ_1. This eigenvector is the first principal component.

Maximum Variance Formulation (Ctd.)

- Additional principal components can be defined in an **incremental fashion**
- Choose each new component such that it **maximizes the remaining projected variance**
- All principal components are **orthogonal to each other**
- Projection onto k dimensions:
  - The lower-dimensional space is defined by the k eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_k$ of the covariance matrix $\boldsymbol{\Sigma}$
  - These correspond to the k largest eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$

Section:

PCA Algorithm

- The Algorithm
- An Example
- Data Reconstruction
- Choice of k

Algorithm 1: PCA Algorithm

Input: Input data $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\} \in \mathbb{R}^{n \times m}$, number of dimensions k

Output: Projected data $\boldsymbol{Z} \in \mathbb{R}^{n \times k}$

$\overline{\boldsymbol{x}} \longleftarrow \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}^{(i)}$ `// sample set mean`

$\boldsymbol{\Sigma} \longleftarrow \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{x}^{(i)} - \overline{\boldsymbol{x}})(\boldsymbol{x}^{(i)} - \overline{\boldsymbol{x}})^\intercal$ `// covariance matrix`

Perform **s**ingular **v**alue **d**ecomposition to find the eigenvectors of matrix $\boldsymbol{\Sigma}$:

$$[\boldsymbol{U}, \boldsymbol{S}, \boldsymbol{V}] = SVD(\boldsymbol{\Sigma})$$

Select first k eigenvectors: $\boldsymbol{U}_k \longleftarrow \boldsymbol{U}_{(:,:k)}$ `// eig.vecs with largest eig.vals.`

$\boldsymbol{Z} \longleftarrow \boldsymbol{U}_k^\intercal \boldsymbol{X}$

Projection of the Data

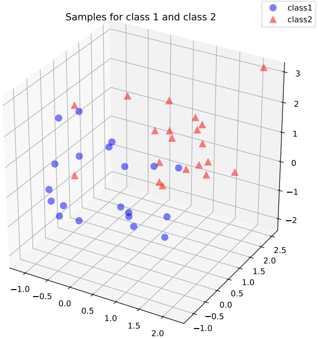
- Matrix $\boldsymbol{U}$ is obtained by applying **singular value decomposition** to $\boldsymbol{\Sigma}$

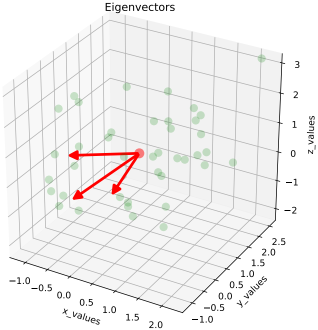
$$\boldsymbol{U} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_1 & \boldsymbol{u}_2 & \dots & \boldsymbol{u}_m \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{m \times m} \tag{5}$$

- The projection $\mathbb{R}^m \rightarrow \mathbb{R}^k (k \ll m)$ is performed as follows:

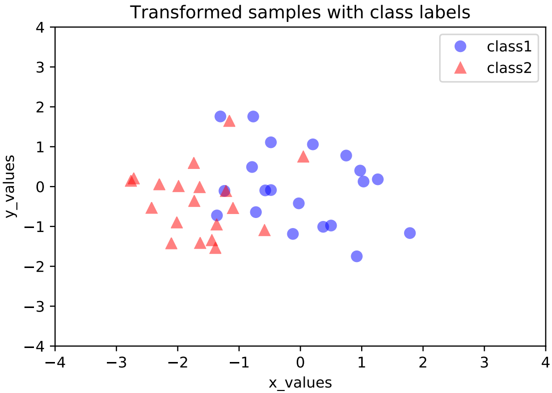
$$\begin{bmatrix} z_1^{(i)} \\ \vdots \\ z_k^{(i)} \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_1 & \boldsymbol{u}_2 & \dots & \boldsymbol{u}_k \\ | & | & & | \end{bmatrix}^\intercal \begin{bmatrix} x_1^{(i)} \\ \vdots \\ x_m^{(i)} \end{bmatrix} \tag{6}$$

PCA Result


Samples for class 1 and class 2
class1
class2


Eigenvectors
x_values
y_values
z_values

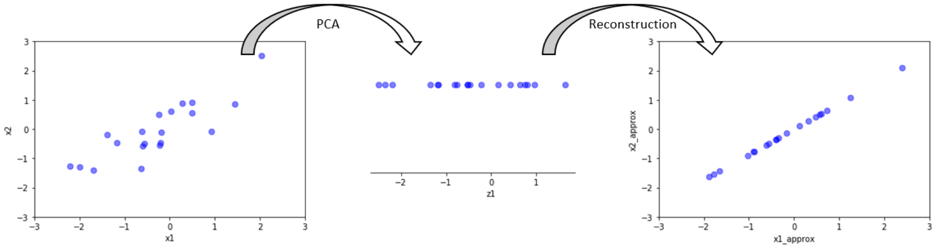
PCA Result (Ctd.)



Reconstruction from compressed Representation

It is possible to compute an approximate reconstruction of the data after having applied PCA ($\mathbb{R}^k \to \mathbb{R}^m$):

$$\boldsymbol{x}_{\approx}^{(i)} = \boldsymbol{U}_k \boldsymbol{z}^{(i)} \tag{7}$$



Choosing the Number of Components

- The goal is to preserve as much variance as possible
- Minimize the **average projection error** given by:

$$\frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{x}^{(i)} - \boldsymbol{x}_{\approx}^{(i)}\|^2 \tag{8}$$

- **Total variation** in the data is computed as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{x}^{(i)}\|^2 \tag{9}$$

Choosing the Number of Components (Ctd.)

- Typically, k is chosen to be the smallest value such that:

$$\frac{\overbrace{{}^{1}/_{n} \sum_{i=1}^{n} \|\boldsymbol{x}^{(i)} - \boldsymbol{x}_{\approx}^{(i)}\|^2}^{\text{average projection error}}}{\underbrace{{}^{1}/_{n} \sum_{i=1}^{n} \|\boldsymbol{x}^{(i)}\|^2}_{\text{total variation}}} \leqslant \gamma \tag{10}$$

- This means that $(1 - \gamma) \cdot 100\,\%$ of the variance is retained

You can be more efficient...

- The above algorithm is computationally very expensive
- The same result can be computed much more efficient, remember:

$$[\boldsymbol{U}, \boldsymbol{S}, \boldsymbol{V}] = SVD(\boldsymbol{\Sigma}) \tag{11}$$

- We can use the $(m \times m)$-matrix $\boldsymbol{S}$ (eigenvalues on the main diagonal):

$$\boldsymbol{S} = \begin{bmatrix} S_{11} & 0 & \dots & 0 \\ 0 & S_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & S_{mm} \end{bmatrix} \tag{12}$$

You can be more efficient... (Ctd.)

- For a given k, the fraction of variance retained can be computed as follows:

$$1 - \frac{\sum_{j=1}^{k} S_{jj}}{\sum_{j=1}^{m} S_{jj}} \leqslant 1 - \gamma \tag{13}$$

- The matrix has to be computed only once and can be reused for all k

Simplification:

$$\frac{\sum_{j=1}^{k} S_{jj}}{\sum_{j=1}^{m} S_{jj}} \geqslant \gamma$$

Section:

PCA Applications

Eigenfaces
Face Morphing

Application of PCA to Images: Eigenfaces

Figure: 100 images of faces

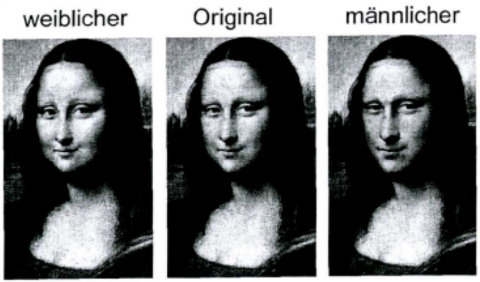
Figure: First 36 principal components

Application of PCA to Images: Eigenfaces (Ctd.)

Figure: Original images

Figure: Reconstructed images

Application of PCA to Images: Face Morphing



Section:

Wrap-Up

Summary
Self-Test Questions
Lecture Outlook

Summary

- Dimensionality reduction is important to avoid the **curse of dimensionality** ☠...
- ...or simply to **visualize high-dimensional data**
- It is defined as the **orthogonal projection** of the data onto a lower-dimensional (linear) space
- We want to **keep the dimensions with the most variance**
- These dimensions are called **principal components**
- Lots of applications: Eigenfaces, Morphing, ...

Self-Test Questions

1. How can PCA be defined?
2. What is the geometric relationship between the principal components?
3. Outline the PCA algorithm!
4. How can you recover the original data? Will you get the exact same data?
5. Explain how the number of components / dimensions can be chosen!
6. Name some use cases where PCA is useful!

What's next...?

The Exam

Just kidding... (maybe)

Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Principal Component Analysis
Term: Winter term 2023/2024

Contact:
Daniel Wehner, M.Sc.
SAP SE / DHBW Mannheim
daniel.wehner@sap.com

Do you have any questions?