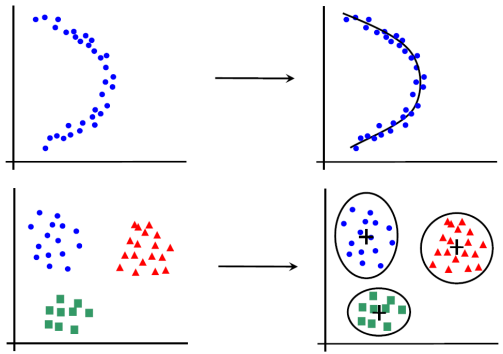


Grundlagen der Künstlichen Intelligenz

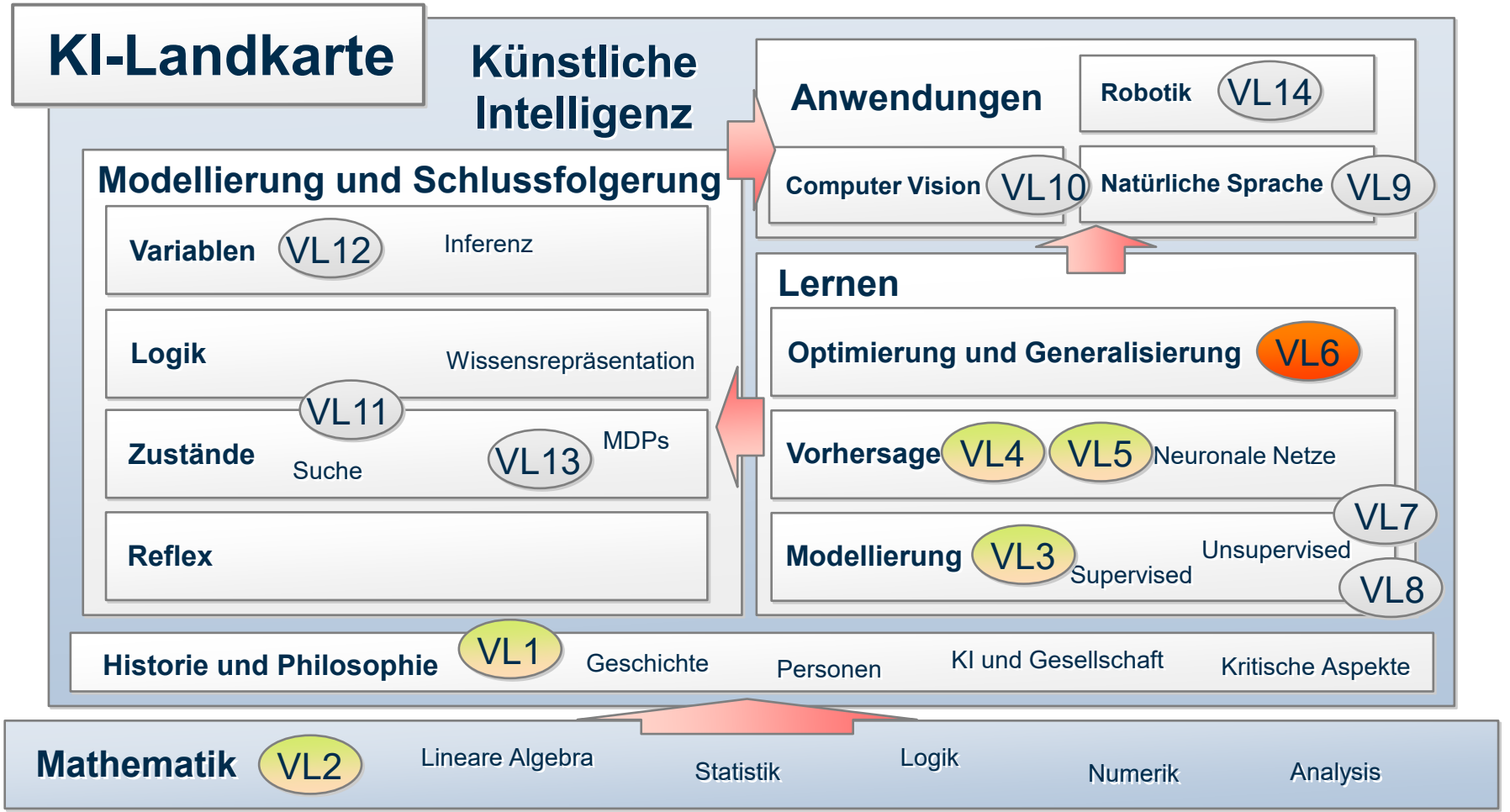
Wintersemester 25/26

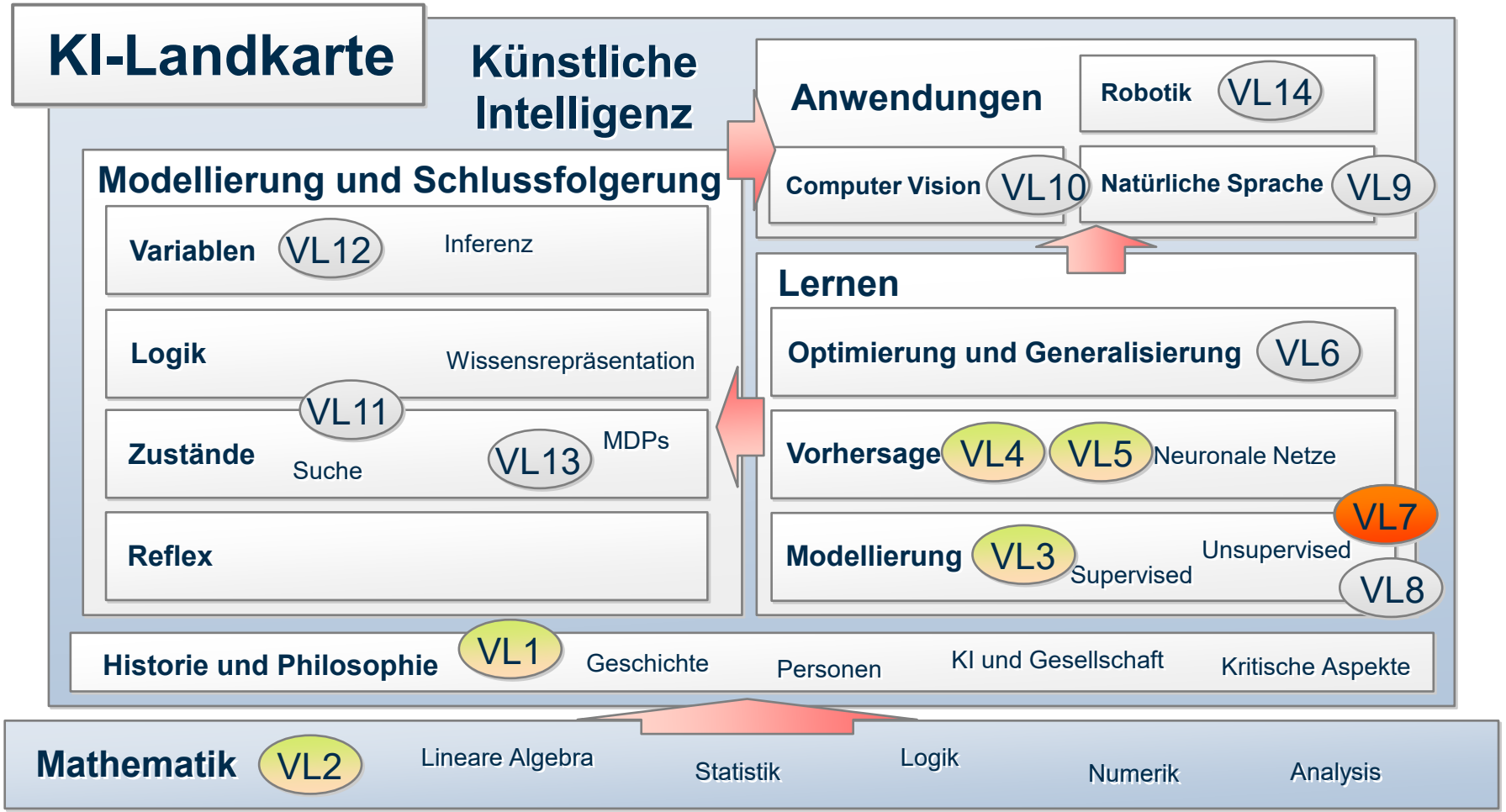
Vorlesung 7

Unüberwachtes Lernen, Dimensionalitätsreduktion, Hauptkomponentenanalyse, Autoencoder und Clustering

T.T.-Prof. Dr. Peer Nowack
Prof. Dr. Pascal Friederich

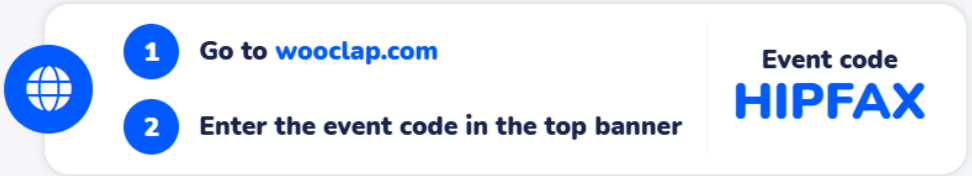
8. Dezember 2025


KI-Landkarte
Künstliche Intelligenz
Anwendungen
Robotik
VL14
Computer Vision
VL10
Natürliche Sprache
VL9
Modellierung und Schlussfolgerung
Variablen
VL12
Inferenz
Logik
Wissensrepräsentation
VL11
Zustände
Suche
VL13
MDPs
Reflex
Lernen
Optimierung und Generalisierung
VL6
Vorhersage
VL4
VL5
Neuronale Netze
VL7
Modellierung
VL3
Supervised
Unsupervised
VL8
Historie und Philosophie
VL1
Geschichte
Personen
KI und Gesellschaft
Kritische Aspekte
Mathematik
VL2
Lineare Algebra
Statistik
Logik
Numerik
Analysis


KI-Landkarte
Künstliche Intelligenz
Anwendungen
Robotik
VL14
Computer Vision
VL10
Natürliche Sprache
VL9
Modellierung und Schlussfolgerung
Variablen
VL12
Inferenz
Logik
Wissensrepräsentation
VL11
Zustände
Suche
VL13
MDPs
Reflex
Lernen
Optimierung und Generalisierung
VL6
Vorhersage
VL4
VL5
Neuronale Netze
VL7
Modellierung
VL3
Supervised
Unsupervised
VL8
Historie und Philosophie
VL1
Geschichte
Personen
KI und Gesellschaft
Kritische Aspekte
Mathematik
VL2
Lineare Algebra
Statistik
Logik
Numerik
Analysis

Wir werden immer mal wieder Umfragen machen...

Sie können sich bereits einloggen. Nutzen Sie das KIT Wi-Fi bei schlechtem Empfang.



You cannot vote anymore

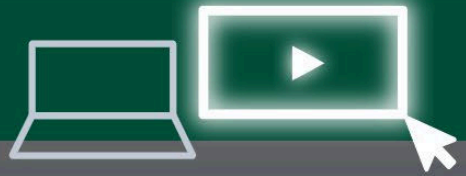
Was ist charakteristisch für unüberwachtes Lernen?

1 Wir haben stets Inputdaten und labelled Outputs. 4% 4

2 Wir haben Daten ohne Labels. 77% 72 ✓

3 Wir lernen aus Aktionen und der Akkumulation von Belohnungen. 11% 10

4 Ich weiß es (noch) nicht. 8% 7



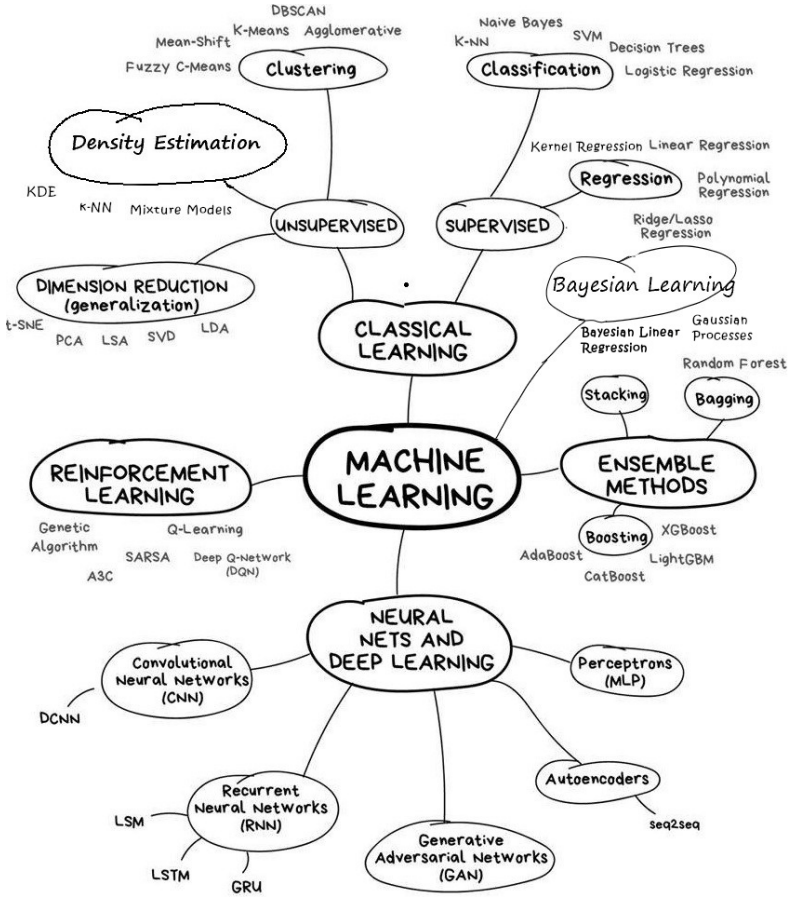
Click on the projected screen to start the question

wooclap

70%

77% correct 93 / 136

Der ML-Zoo...


MACHINE LEARNING
CLASSICAL LEARNING
UNSUPERVISED
SUPERVISED
Clustering
DBSCAN
K-Means
Agglomerative
Mean-Shift
Fuzzy C-Means
Density Estimation
KDE
K-NN
Mixture Models
DIMENSION REDUCTION (generalization)
t-SNE
PCA
LSA
SVD
LDA
Classification
Naive Bayes
K-NN
SVM
Decision Trees
Logistic Regression
Regression
Kernel Regression
Linear Regression
Polynomial Regression
Ridge/Lasso Regression
Bayesian Learning
Bayesian Linear Regression
Gaussian Processes
ENSEMBLE METHODS
Stacking
Bagging
Random Forest
Boosting
XGBoost
AdaBoost
LightGBM
CatBoost
REINFORCEMENT LEARNING
Genetic Algorithm
Q-Learning
SARSA
Deep Q-Network (DQN)
A3C
NEURAL NETS AND DEEP LEARNING
Convolutional Neural Networks (CNN)
DCNN
Perceptrons (MLP)
Recurrent Neural Networks (RNN)
LSM
LSTM
GRU
Autoencoders
seq2seq
Generative Adversarial Networks (GAN)

Heute – Unüberwachtes Lernen (Unsupervised Learning)

Lernmethoden und Datentypen

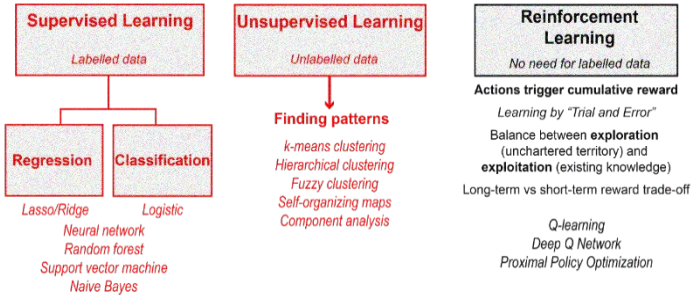
Überwachtes Lernen *Daten mit Labels*

- Regression
- Klassifikation

Unüberwachtes Lernen *Daten ohne Labels*

- Dimensionalitätsreduktion
- Clustering
- Schätzung der Dichte

Reinforcement Learning *Daten aus Interaktion*

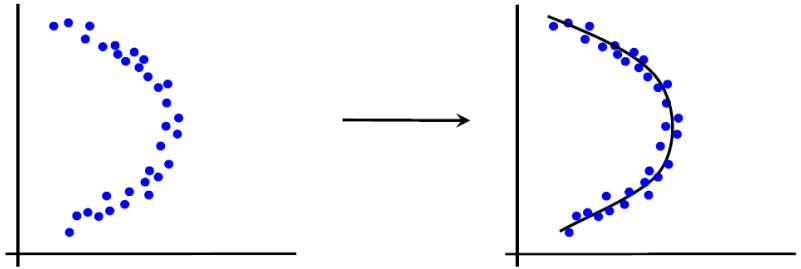


Anwendungsbereiche des unüberwachten Lernens

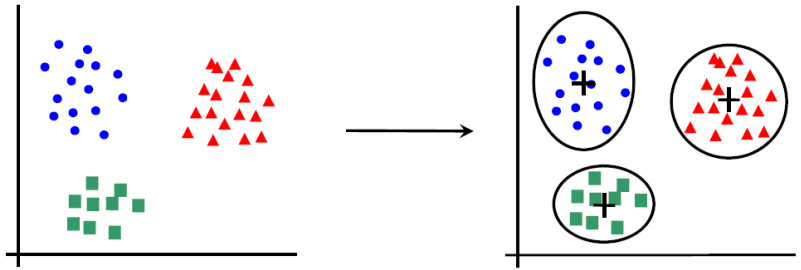
Fokus heute!

Trainingsdaten enthalten keine Labels, sondern finde "Struktur" in den Daten

(1) Dimensionalitätsreduktion



(2) Clustering



(3) Dichteschätzung: Generatives Modell der Daten → in VL8

Lernziele der heutigen Vorlesung

Nach dieser Vorlesung werden Sie in der Lage sein:

- Zu erklären, wozu **Dimensionalitätsreduktion** in verschiedenen Anwendungen genutzt wird, von der Visualisierung bis hin zur Beschleunigung von überwachten Lernalgorithmen.
- Die **Hauptkomponentenanalyse/Principal Component Analysis** und den **Autoencoder** als lineare bzw. nichtlineare Algorithmen zu kontrastieren und einzeln in ihren Grundschritten und Grundideen zu beschreiben.
- Die Grundideen hinter **Clustering** und die einzelnen Schritte des **K-Means Algorithmus** darzustellen.
- Die jeweiligen Stärken und Schwächen der Algorithmen zu bewerten.

Überblick

Hauptkomponentenanalyse:

- Lineare Dimensionalitätsreduktion
- Lineare orthogonale Projektionen
- Reproduktionsfehler

Autoencoder:

- Reconstruction Loss
- Nichtlineare neuronale Netzwerke
- Selbstüberwachtes Lernen

Clustering:

- Problemstellung
- K-Means-Clustering

Die Folien basieren teilweise auf Material von Jan Peters

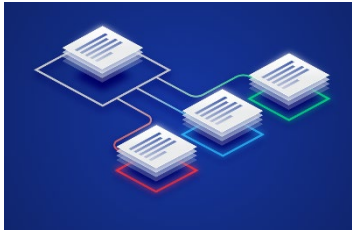
Dimensionalitätsreduktion
+
Hauptkomponentenanalyse

Große, hochdimensionale Daten

Hochdimensionale Daten = viele Merkmale/Features

Dokumentenklassifikation

- Merkmale pro Dokument = Tausende von Wörtern/Unigrammen
- Millionen von Bigrammen, kontextbezogene Informationen



Hochdimensionale Bilddaten

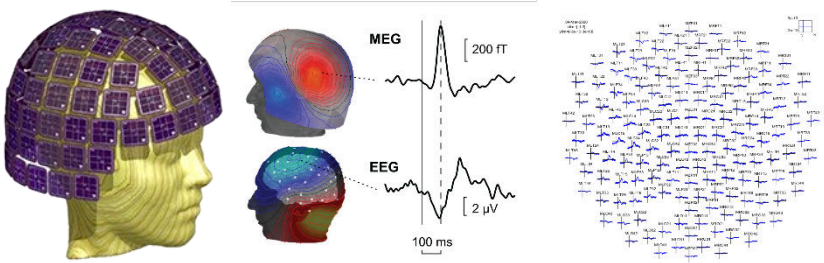
- Viele hochaufgelöste Pixel

Große, hochdimensionale Daten

cerebral-computing.com

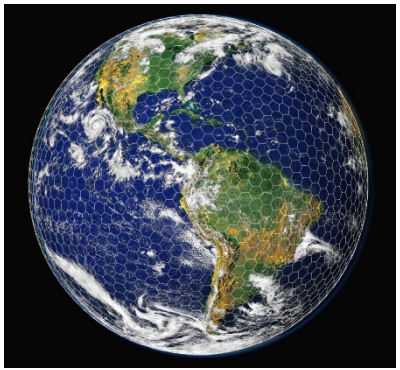
MEG-Gehirnbildgebung

- 120 Messorte × 500 Zeitpunkte × 20 Objekte

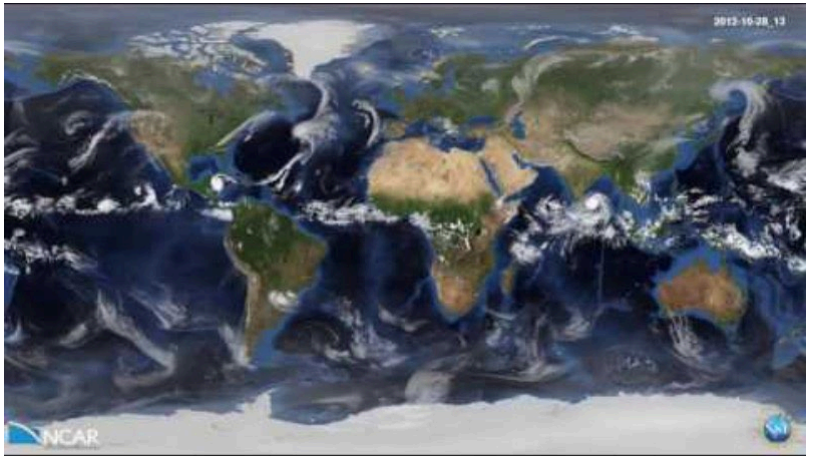


40 Jahre von täglichen Wetterdaten

- 14610 Zeitpunkte × 73 × 96 Gitterpunkte



NCAR



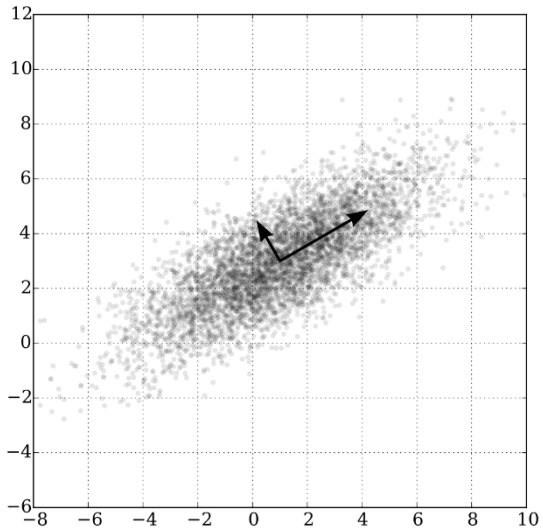
Dimensionalitätsreduktion

Nutze unüberwachtes Lernen, um Struktur in den Daten zu finden

- Damit: Reduziere die Dimensionalität der Daten

Mögliche Anwendungen

- Visualisierung der Daten
- Vorverarbeitung für Lernalgorithmen (schneller, weniger Rauschen, RAM, …)
- Geringere Datenmengen zur Speicherung
- …



Beispiel Hauptkomponentenanalyse

Lineare Dimensionalitätsreduktion

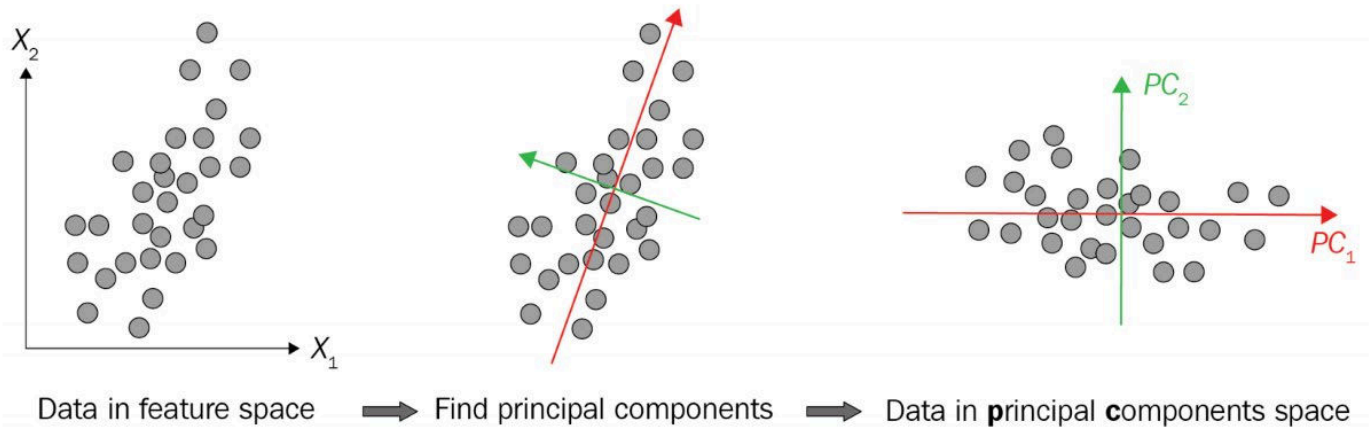
Problemstellung:

- Ursprünglicher Datenpunkt i: $\boldsymbol{x}_i \in \mathbb{R}^D$
- Niederdimensionale Darstellung des Datenpunkts i: $\boldsymbol{z}_i \in \mathbb{R}^M$ mit $D >> M$
- **Ziel:** eine lineare Abbildung finden $\boldsymbol{x}_i \rightarrow \boldsymbol{z}_i$

$$\boldsymbol{z}_i = \boldsymbol{W}\boldsymbol{x}_i, \text{ with } \boldsymbol{W} \in \mathbb{R}^{M \times D}$$

Algorithmus: Hauptkomponentenanalyse / Principal Component Analysis (PCA)

Hauptkomponentenanalyse / Principal Component Analysis (PCA)



Finden von Hauptkomponenten (PCs) der Daten

- Orthogonale Richtungen, in denen die Daten am stärksten gestreut sind (höchste Varianz)
- PC1 erklärt den größten Teil der Varianz in den Daten, gefolgt von PC2, PC3 usw.
- PCA ist primär eine Technik zur Dimensionsreduktion, aber auch nützlich für die Visualisierung
- Eine gute Trennung von unähnlichen Samples wird ermöglicht
- Die globale Datenstruktur bleibt erhalten

Orthonormale Basisvektoren (gebraucht zur Formalisierung von PCA)

Wir können einen Vektor immer in Form eines orthonormalen Basiskoordinatensystems schreiben

$$\boldsymbol{x} = \sum_{i=1}^{D} z_i \boldsymbol{u}_i, \text{ where } \boldsymbol{u}_i^T \boldsymbol{u}_j = \delta_{ij} \text{ and } \delta_{ij} = 1 \text{ if } i = j, \ 0 \text{ otherwise}$$

Orthonormalitätsbedingung: Das Produkt von zwei verschiedenen Basisvektoren ist 0. Die Norm jedes Basisvektors ist 1.

Beispiel: (Einheitsvektoren = einfachstes orthonormales Basissystem)

$$\begin{bmatrix} 3 \\ 7 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 7 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Projektionen mit einer orthonormalen Basis

Die Koeffizienten z_i erhält man durch Projektion von $\boldsymbol{x}$ auf den Basisvektor $\boldsymbol{u}_i$

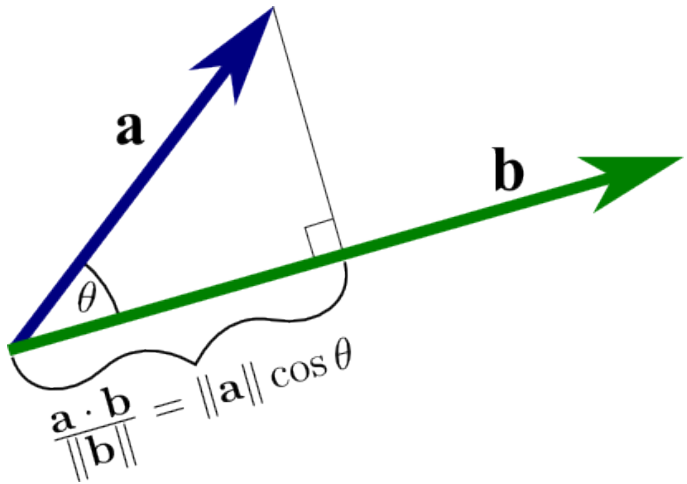
$$\underbrace{z_i}_{\text{scalar coefficient}} = \underbrace{\boldsymbol{u}_i^T \boldsymbol{x}}_{\text{projection}}$$

Beispiel:

$$\boldsymbol{x} = z_1 \boldsymbol{u}_1 + z_2 \boldsymbol{u}_2$$

$$\boldsymbol{u}_1^T \boldsymbol{x} = z_1 \underbrace{\boldsymbol{u}_1^T \boldsymbol{u}_1}_{=1} + z_2 \underbrace{\boldsymbol{u}_1^T \boldsymbol{u}_2}_{=0} = z_1$$

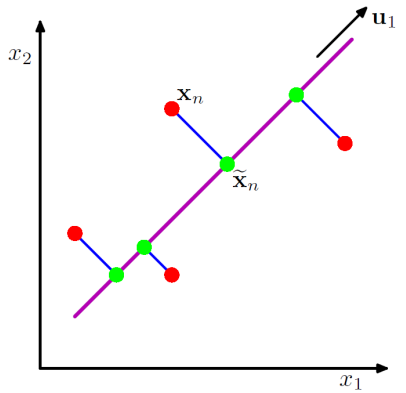
Projektion von 2 Vektoren



Zerlegung der Reproduktion – nur M Basisvektoren behalten

Verwenden Sie $M << D$ Basisvektoren: (maximal lineare Varianz erhaltend)

$$\boldsymbol{x} = \underbrace{\sum_{i=1}^{M} z_i \boldsymbol{u}_i}_{\tilde{\boldsymbol{x}} \approx \boldsymbol{x}} + \underbrace{\sum_{j=M+1}^{D} z_j \boldsymbol{u}_j}_{\text{skip}}$$



Bishop (2006)

Für PCA suchen wir die M Basisvektoren $\boldsymbol{u}_i$, die den mittleren quadratischen Reproduktionsfehler minimieren:

$$\arg\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M} E(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M) = \arg\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M} \sum_{i=1}^{N} ||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2$$

Dies ist das zu lösende Optimierungsproblem!

Minimierung des Fehlers in der Hauptkomponentenanalyse

Geht man zur Anschaulichkeit von **einem einzigen Basisvektor** aus, so lässt sich der Fehler wie folgt schreiben:

$$\tilde{\boldsymbol{x}}_i = \sum_{k=1}^{M} z_{ik}\boldsymbol{u}_k = \sum_{k=1}^{M} (\boldsymbol{u}_k^T \boldsymbol{x}_i)\boldsymbol{u}_k$$

$$E(\boldsymbol{u}_1) = \sum_{i=1}^{N} ||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2 = \sum_{i=1}^{N} ||\boldsymbol{x}_i - \underbrace{(\boldsymbol{u}_1^T \boldsymbol{x}_i)}_{z_{i1}} \boldsymbol{u}_1||^2$$

Skalar

$$= \sum_{i=1}^{N} \boldsymbol{x}_i^T \boldsymbol{x}_i - 2(\boldsymbol{u}_1^T \boldsymbol{x}_i)^2 + (\boldsymbol{u}_1^T \boldsymbol{x}_i)^2 \underbrace{\boldsymbol{u}_1^T \boldsymbol{u}_1}_{=1} = \sum_{i=1}^{N} \boldsymbol{x}_i^T \boldsymbol{x}_i - (\boldsymbol{u}_1^T \boldsymbol{x}_i)^2$$

$$= \sum_{i=1}^{N} \boldsymbol{x}_i^T \boldsymbol{x}_i - z_{i1}^2$$

Nur dieser Teil hängt von $\boldsymbol{u}_1$ ab, wonach wir optimieren wollen…

(1) $(a-b)^2 = a^2 - 2ab + b^2$

(2) $\boldsymbol{x}_i^T \boldsymbol{u}_1 = \boldsymbol{u}_1^T \boldsymbol{x}_i$

Minimierung des Fehlers in der Hauptkomponentanalyse

Also, für einen Basisvektor kann der Fehler geschrieben werden als

$$E(\boldsymbol{u}_1) = \sum_{i=1}^{N} \boldsymbol{x}_i^T \boldsymbol{x}_i - z_{i1}^2$$

$$\Rightarrow \arg\min_{\boldsymbol{u}_1} E(\boldsymbol{u}_1) = \arg\max_{\boldsymbol{u}_1} \sum_{i=1}^{N} z_{i1}^2 = \arg\max_{\boldsymbol{u}_1} \sum_{i=1}^{N} (\boldsymbol{u}_1^T \boldsymbol{x}_i)^2$$

- Die Minimierung des Reproduktionsfehlers ist gleichbedeutend mit der Maximierung der Varianz der Projektion im niedriger-dimensionalen latenten Raum (unter der Annahme einer Projektion mit Mittelwert Null)
- Wir können eine Projektion auf den Mittelwert Null sicherstellen, indem wir den Mittelwert von den Daten für jede Dimension abziehen

$$\bar{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{\mu}$$

Hauptkomponentenanalyse - Orthonormalität

Die erste Hauptrichtung $\boldsymbol{u}_1$ ist die Richtung, in der die Varianz der projizierten Daten maximal ist

$$\boldsymbol{u}_1 = \arg\max_{\boldsymbol{u}} \frac{1}{N} \sum_{i=1}^{N} \left(\boldsymbol{u}^T \underbrace{(\boldsymbol{x}_i - \boldsymbol{\mu})}_{\bar{\boldsymbol{x}}_i} \right)^2 \quad \text{s.t. } \boldsymbol{u}^T \boldsymbol{u} = 1$$

Annahme: Richtungen haben alle eine Einheitsnorm

Herleitung des PCA-Algorithmus in Matrixschreibweise

Zu maximieren:

$$\frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{u}^T(\boldsymbol{x}_i-\boldsymbol{\mu})\right)^2$$

$$=\frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{u}^T(\boldsymbol{x}_i-\boldsymbol{\mu})(\boldsymbol{x}_i-\boldsymbol{\mu})^T\boldsymbol{u}\right)$$

$$=\boldsymbol{u}^T\underbrace{\left(\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i-\boldsymbol{\mu})(\boldsymbol{x}_i-\boldsymbol{\mu})^T\right)}_{\text{covariance }\boldsymbol{\Sigma}}\boldsymbol{u}=\boldsymbol{u}^T\boldsymbol{\Sigma}\boldsymbol{u} \qquad \text{s.t. } \boldsymbol{u}^T\boldsymbol{u}=1$$

Das Optimierungsziel kann in Form der Kovarianzmatrix Σ geschrieben werden!

Herleitung des PCA-Algorithmus in Matrixschreibweise

Damit können wir das Optimierungsproblem mit Nebenbedingungen schreiben als:

$$\boldsymbol{u}_1 = \arg\max_{\boldsymbol{u}} \; \boldsymbol{u}^T \boldsymbol{\Sigma} \boldsymbol{u} \quad \text{s.t. } \boldsymbol{u}^T \boldsymbol{u} = 1$$

Dies kann mit Lagrange-Optimierung gelöst werden (nicht in dieser VL; Kapitel 12, Seite 578 and Appendix E in Bishop 2006)…

Ergebnis: Optimalitätsbedingung für $\boldsymbol{u}$:

$$\boldsymbol{\Sigma} \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1 \qquad \lambda \ldots \text{Lagrange Multiplier}$$

Dies ist ein Eigenwertproblem! $\boldsymbol{u}_1$ ist ein Eigenvektor der Kovarianzmatrix und die Varianz der projizierten Daten ist $\mathbf{u}_1^{\mathrm{T}} \boldsymbol{\Sigma} \boldsymbol{u}_1 = \lambda_1$. Der Eigenvektor ist PC1.

Die Varianz wird maximal wenn $\boldsymbol{u}_1$ der Eigenvektor mit dem größten Eigenwert λ_1 ist.

Recap: Eigenwerte und Eigenvektoren

See also Jason Brownlee's blog post

Eigenvektoren $\boldsymbol{u_k}$ $(k \leq D)$ und Eigenwerte λ_k von Matrix **C** sind definiert als

$$\boldsymbol{C}\boldsymbol{u}_k = \lambda_k \boldsymbol{u}_k \quad \text{with } \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D$$

Geordnete Liste der Eigenwerte

In Form einer Matrix:

$$\boldsymbol{C}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Lambda} \quad \text{with } \boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_D) \text{ and } \boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_D]$$

Da **U** orthonormal ist (Eigenvektoren haben eine Einheitsnorm), wissen wir, dass

$$\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$$

Dies bedeutet, dass wir jede (positive semi-definite) Matrix **C** zerlegen können als

$$(\boldsymbol{C}\boldsymbol{U})\boldsymbol{U}^T = (\boldsymbol{U}\boldsymbol{\Lambda})\boldsymbol{U}^T \Rightarrow \boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$$

Eigenwerte und Eigenvektoren

Jede positiv semi-definite symmetrische Matrix kann in ihre Eigendekomposition zerlegt werden

$$\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T = \underbrace{\begin{bmatrix} \boldsymbol{u}_1 & \dots & \boldsymbol{u}_D \end{bmatrix}}_{\text{Eigenvectors}} \underbrace{\begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_D \end{bmatrix}}_{\text{Eigenvalues}} \begin{bmatrix} \boldsymbol{u}_1^T \\ \vdots \\ \boldsymbol{u}_D^T \end{bmatrix}$$

Damit können wir die Basis-/Eigenvektoren und Eigenwerte unserer Hauptkomponentenanalyse erhalten.

In der Praxis wird hierfür oft die Singulärwertzerlegung (Singular Value Decomposition) als besonders effiziente und dabei numerisch stabile Methode verwendet.

Zurück zur Hauptkomponentenanalyse - Vorgehen

- Zentriere die Daten um den Mittelwert (für jede Dimension).
- Berechne und zerlege die Kovarianzmatrix $\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, wähle die M größten Eigenwerte und die entsprechenden Eigenvektoren.
- Dies ist analog dazu zusätzliche PCs inkrementell zu $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots$ zur Eigenbasis hinzuzufügen und dabei jede neue Richtung so zu wählen, dass sie die projizierte Varianz jeweils maximiert, orthogonal zu denen die bereits berücksichtigt sind.
- Es ist auch üblich, die Varianz der einzelnen Dimensionen vor der PCA zu normalisieren (d. h. die Varianz zu vereinheitlichen).

So erhalten wir eine (Eigen-)Basis für die Darstellung der Daten mit $\boldsymbol{W} = \begin{bmatrix} \boldsymbol{u}_1 & \ldots & \boldsymbol{u}_M \end{bmatrix}$

- Projektion auf niedriger-dimensional: $\boldsymbol{z}_i = \boldsymbol{W}^T(\boldsymbol{x}_i - \boldsymbol{\mu})$

- Reprojektion auf hoch-dimensional: $\tilde{\boldsymbol{x}}_i = \boldsymbol{\mu} + \boldsymbol{W}\boldsymbol{z}_i$

Hauptkomponentenanalyse – Wie M wählen?

- **Dimensionalität basierend auf λ_k** wählen: wenn $\lambda_k \approx 0$ für $k > M$ für $M << D$, dann können wir die Teilmenge der ersten M Eigenvektoren verwenden.

$$\boldsymbol{x}_i - \boldsymbol{\mu} = \underbrace{\sum_{j=1}^{M} z_{ij}\boldsymbol{u}_j}_{\tilde{\boldsymbol{x}}} + \underbrace{\sum_{j=M+1}^{D} z_{ij}\boldsymbol{u}_j}_{\text{close to } 0} \Rightarrow \boldsymbol{x}_i \approx \boldsymbol{\mu} + \sum_{j=1}^{M} z_{ij}\boldsymbol{u}_j$$

- Diese Darstellung hat den geringsten mittleren quadratischen Fehler (MSE), auch in der Rekonstruktion, aller linearen Darstellungen der Dimension M

$$\arg\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M} E(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M) = \arg\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M} \sum_{i=1}^{N} ||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2$$

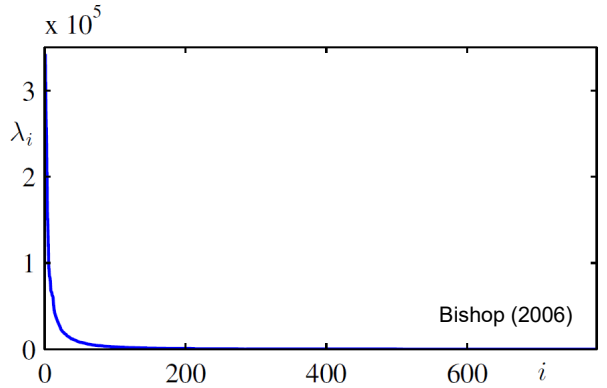
Hauptkomponentenanalyse – Wie M wählen?

- Ein größeres M führt zu einer besseren Approximation.
  Im Grenzfall, wenn $M = D$, bleiben wir in den ursprünglichen Datendimensionen.

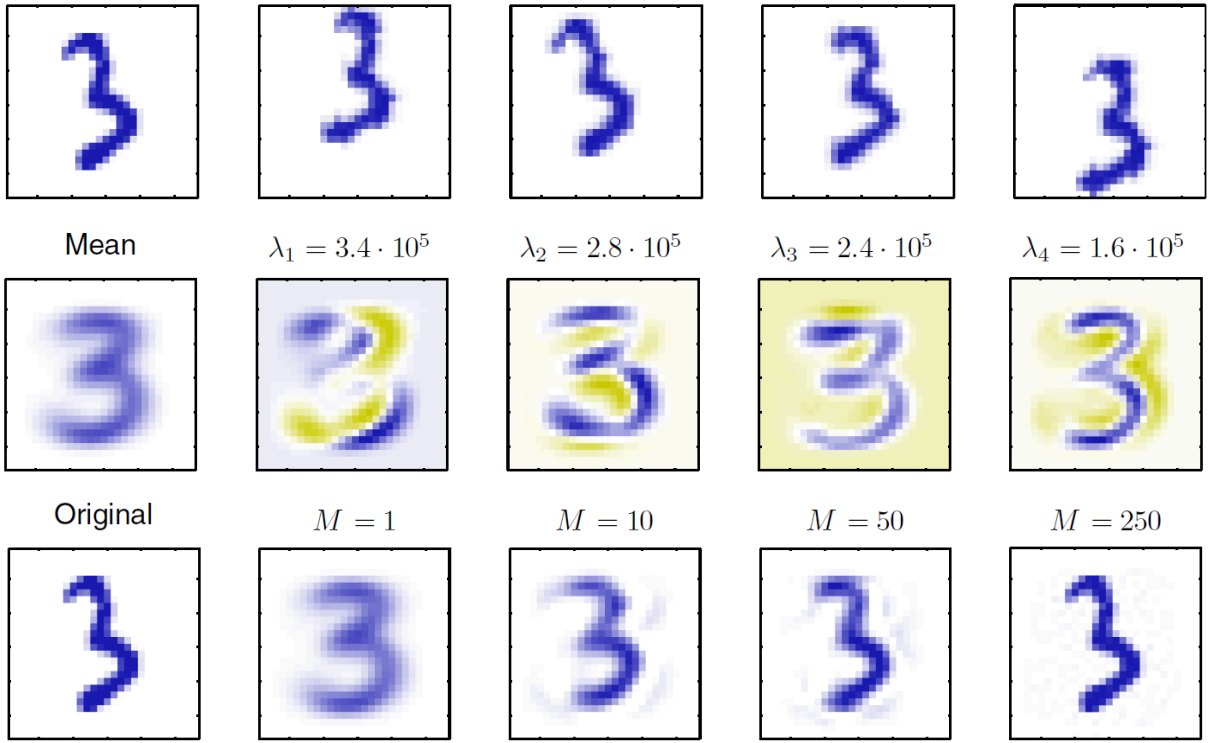
- In der Praxis: **zwei gute Möglichkeiten für die Wahl von M**
  - ✓ Wählen Sie M auf der Grundlage der Anwendung, d. h. wählen Sie das kleinste M, mit dem die Anwendung gut genug funktioniert.
  - ✓ Wählen Sie M so, dass die Eigenbasis einen Teil der Varianz erfasst (zum Beispiel η = 0,9). Der Eigenwert λ_i beschreibt die marginale Varianz, die durch $\boldsymbol{u}_i$ erfasst wird

$$\text{Choose } M \text{ s.t. } \sum_{i=1}^{M} \lambda_i = \eta \underbrace{\sum_{i=1}^{D} \lambda_i}$$

Gesamtvarianz der Daten



Anwendungsbeispiel 1: Bilder (28×28) handgeschriebener Zahlen

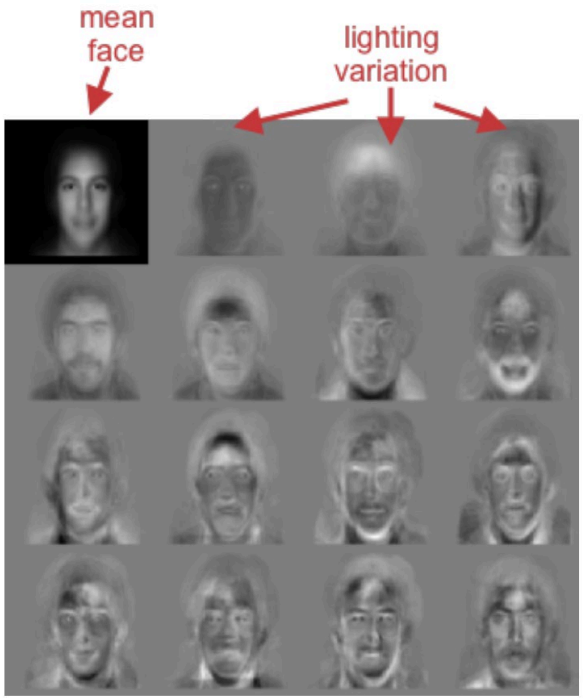


Bishop (2006)

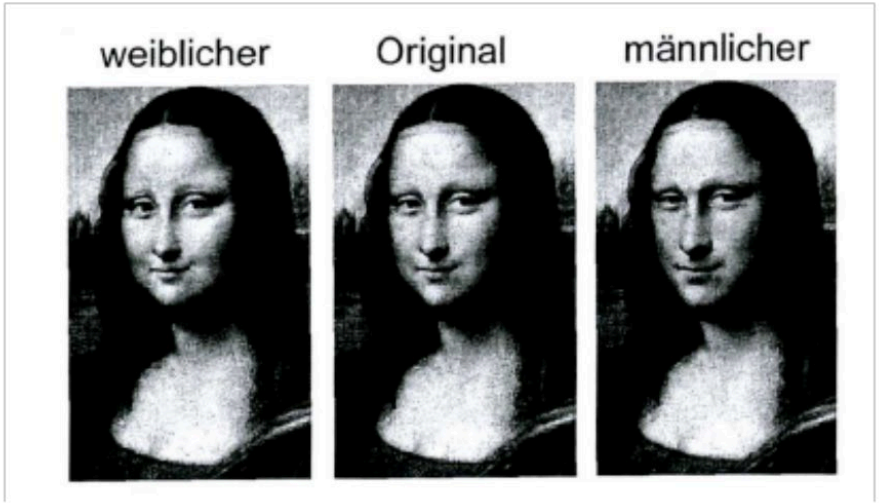
- Jedes Pixel ist eine Dimension/Merkmal.
- Erste Reihe: Beispiele von Original-Daten.
- Zweite Reihe: Durchschnittsbild $\bar{x}$ und die ersten vier Eigenvektoren.
- Dritte Reihe: Rekonstruktion eines Originals basierend auf M Komponenten.

Anwendungsbeispiel 2 - Gesichtserkennung

- Eine populäre Anwendung der PCA zur Objekterkennung war die Erkennung von Gesichtern [Turk und Pentland, 1991].
- Sammeln von Gesichtsbildern
- Normalisieren für Kontrast, Maßstab und Ausrichtung
- Hintergründe entfernen
- PCA anwenden und die ersten M Eigenbilder auswählen, die den größten Teil der Varianz der Daten ausmachen

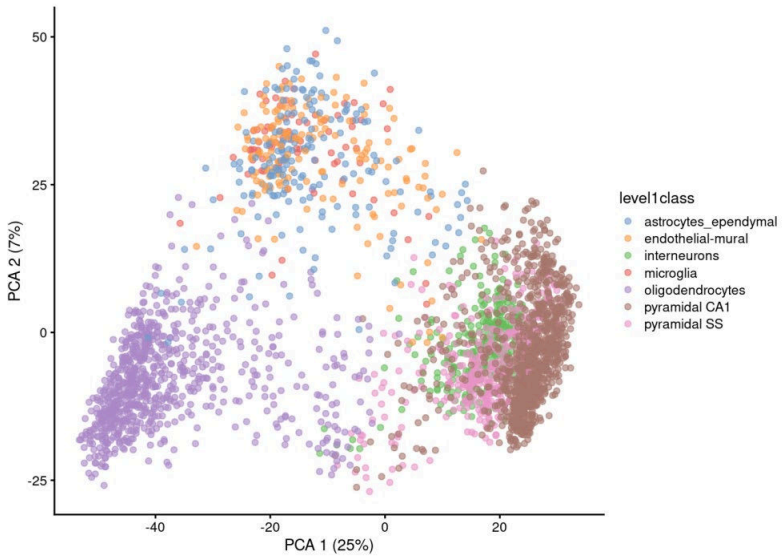


Bildmorphing mit PCA



Anwendungsbeispiel 3: Visualisierung – Genome expression

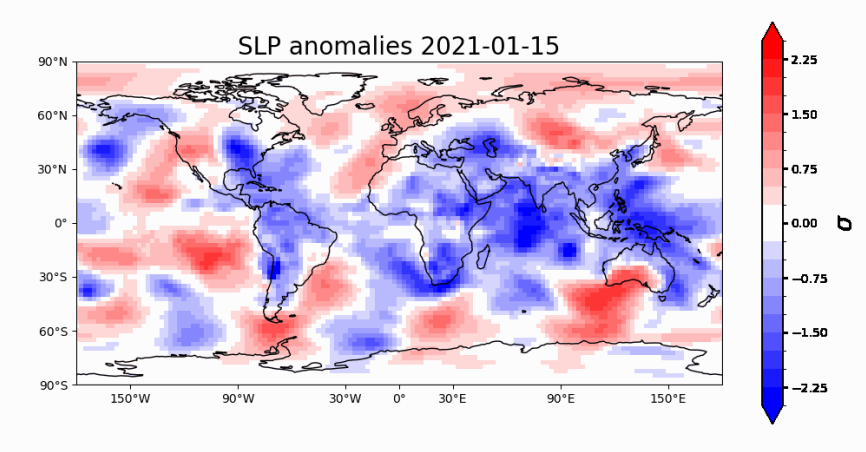
- Da PC1 und PC2 den größten Teil der Varianz der Daten erfassen, ist es üblich, die Daten in diesen beiden neuen Dimensionen zu visualisieren.
- Genexpressionsmuster werden von den Hauptkomponenten (PCs) erfasst
  → PCA kann Zelltypen trennen.
- Zellen werden verglichen basierend auf „gene expression values“, welche korreliert sein können wenn sie durch die gleichen biologischen Prozesse beeinflusst werden.
- Müssen nicht die zellspezifische Expression jedes Gens einzeln betrachten, stattdessen zusammengefasst als „Eigengene“.



Bioconductor.org

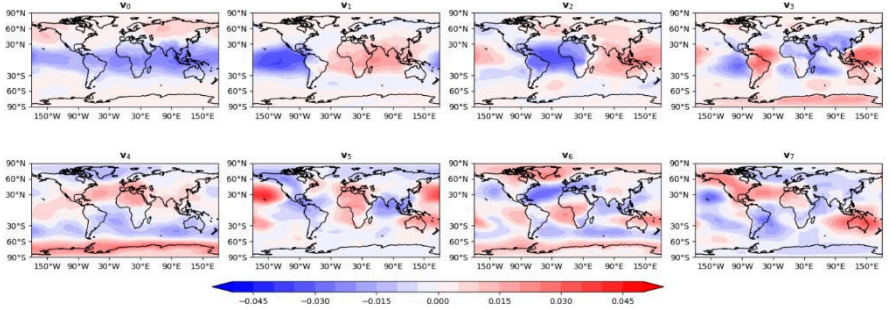
Basierend auf Zeisel et al. (2015)

Anwendungsbeispiel 4 – Wetter/Klimadaten

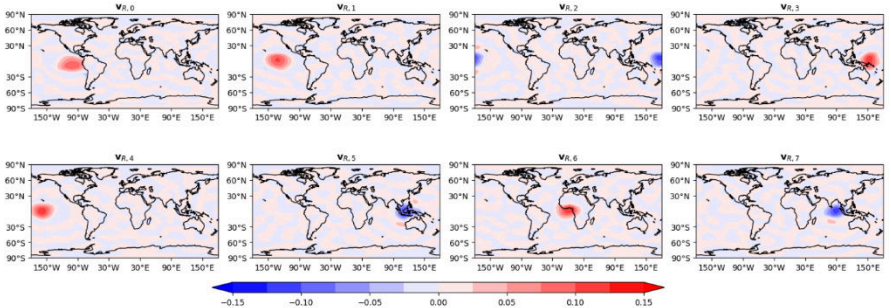


- 70 Jahre von täglichen Druckanomalien, 2.5°×2.5° räumliche Auflösung.
- Jeder Gitterpunkt ist ein Merkmal.
- Gibt es Kovarianzen über die Gitterpunkte hinweg, die sich mit den PCs visualisieren lassen?

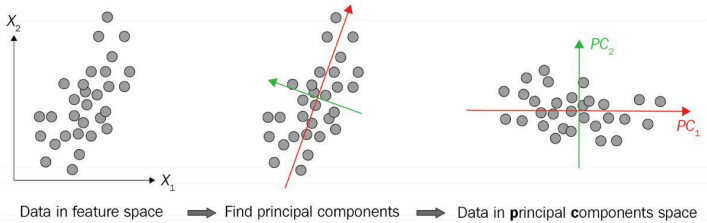
Die ersten 8 Eigenvektoren (36.5% der Varianz)



Zusätzliche Varimax Rotation der Eigenvektoren (Master-VL…)



Zusammenfassung: Hauptkomponentenanalyse/PCA



- PCA projiziert die Daten in einen linearen Unterraum
- PCA maximiert die Varianz der Projektion / minimiert den Fehler der Rekonstruktion
- PCA ist die einfachste lineare Dimensionalitätsreduktionstechnik

  - Es gibt viele ausgefeiltere Techniken

  - Autoencoder, Kernel PCA, t-SNE, Faktorisierung nicht-negativer Matrizen (interessant, aber keine Zeit, diese zu behandeln...)

Anwendungen:

- Die PCA ermöglicht es uns, einen hochdimensionalen Eingaberaum in einen niedrigdimensionalen Eigenraum umzuwandeln und dabei das Wesentliche der Daten zu erfassen
- PCA ist ein sehr gebräuchlicher Vorverarbeitungsschritt für hochdimensionale Eingangsdaten

Fragen?

You cannot vote anymore

Was trifft auf die Hauptkomponentenanalyse zu? (wählen Sie alle Aussagen aus die zutreffen)

1 Sie ist linear. 76% 67 ✓

2 Sie ist nichtlinear. 11% 10

3 Ihre Eigenvektoren spannen eine orthogonale Basis auf. 97% 85 ✓

4 Sie ist eine Form des überwachten Lernens. 2% 2

5 Wir können im Prinzip 100% der Varianz erhalten. 63% 55 ✓

6 Datenpunkte die im Featureraum nahe beieinander liegen, liegen auch im dimensionsreduzierten Raum nahe beieinander. 76% 67 ✓

7 Sie ist eine Art von Neuronalem Netzwerk. 2% 2

Click on the projected screen to start the question

wooclap

70%

85% correct 88 / 136

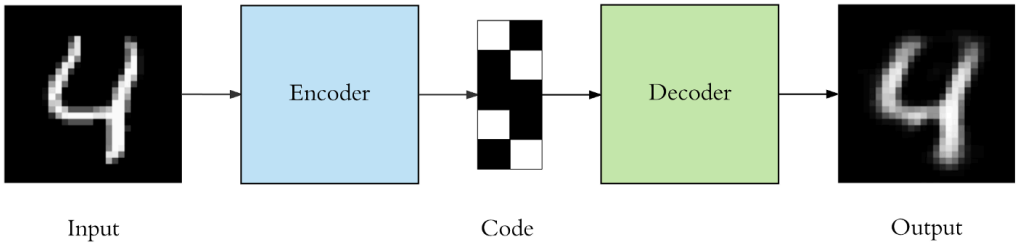
Nichtlineare Dimensionalitätsreduktion

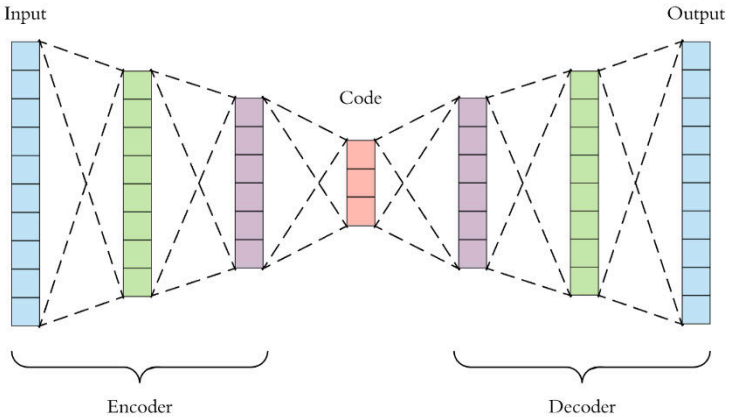
Autoencoder

Nichtlineare unüberwachte Modelle - Autoencoder

Autoencoder sind tiefe neuronale Netzwerke, die darauf ausgelegt sind, ihre Eingabe, insbesondere Bilder, zu rekonstruieren.

- Entscheidend ist, dass das Input aus einer gelernten Kodierung reproduziert wird.
- Ähnlich wie PCA: Encoder performt Dimensionsreduktion in den latenten Raum/Kodierung; Decoder rekonstruiert Input mit einem Rekonstruktionsfehler.
- Trainiert durch Backpropagation.





Autoencoder vs. PCA

PCA

- Die PCA reduziert eine Menge von Eingangsvektoren (z.B. Bilder) auf -- üblicherweise weniger -- neue Basisvektoren, die die ursprünglichen Daten durch lineare Kombination möglichst genau approximieren.
- Sehr effizient für bestimmte Anwendungen.
- Leicht zu berechnen.

Autoencoder (AE) – eine Form der tiefen neuronalen Netzwerke

- Kann nichtlineare Abhängigkeiten lernen.
- Kann Faltungsschichten (convolutions) verwenden (auf Bilder zugeschnittene Schichten - behandelt in Vorlesung 9).
- Wenn wir nur eine einzige lineare Schicht für Encoder und Decoder verwenden, sind AEs ähnlich zu PCA.

Aufbau eines Autoencoders

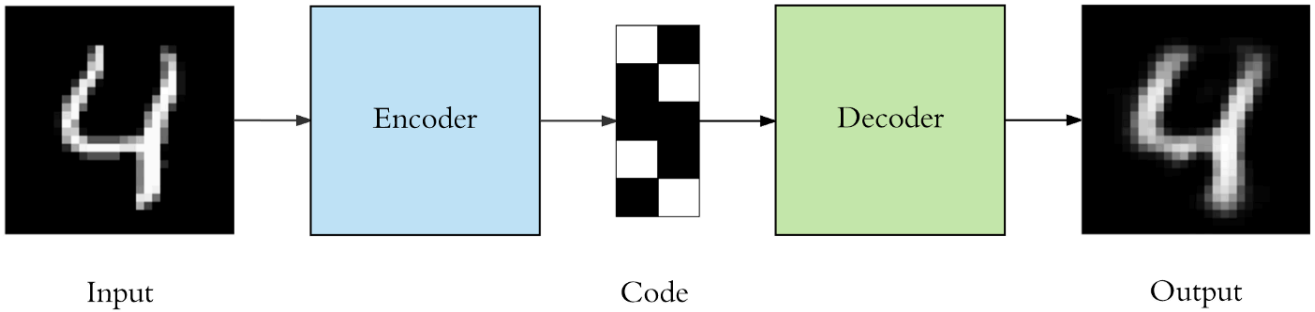
- **Encoder:** Komprimierung der Eingabe in einen latenten Raum von (in der Regel) kleinerer Dimension.

$$\boldsymbol{z} = h(\boldsymbol{x})$$

- **Decoder:** Rekonstruktion der Eingabe aus dem latenten Raum z.

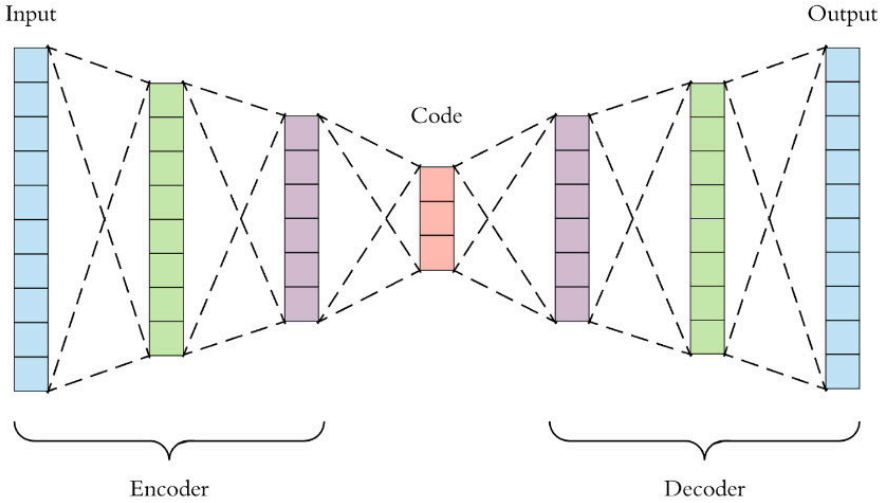
$$\tilde{\boldsymbol{x}} = g(\boldsymbol{z})$$

  - mit $\tilde{x}$ einer möglichst großen Nähe (z.B. quadratischer Fehler) zu x



Bottleneck-Schicht

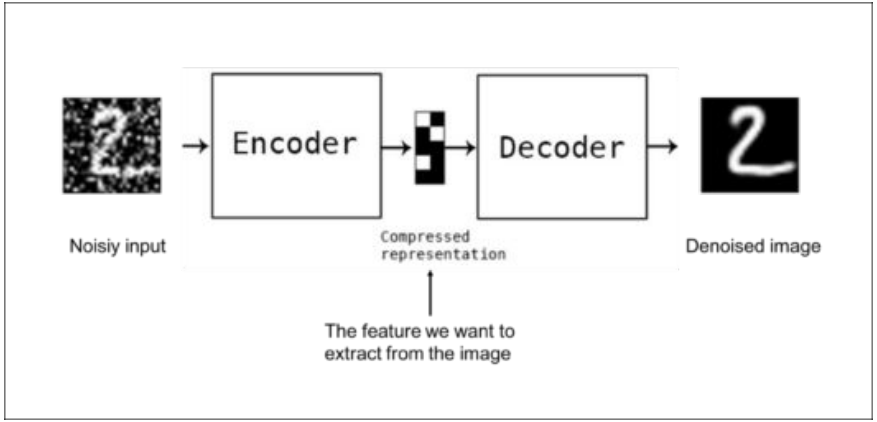
- Eingabebilder sind *n x n* und der latente Raum ist *m* < *n x n*.
- Dann reicht der latente Raum nicht aus, um alle Bilder exakt wiederzugeben.
- Es muss eine Kodierung erlernt werden, die die wichtigsten Merkmale in den Trainingsdaten erfasst und für eine ungefähre Rekonstruktion ausreicht.

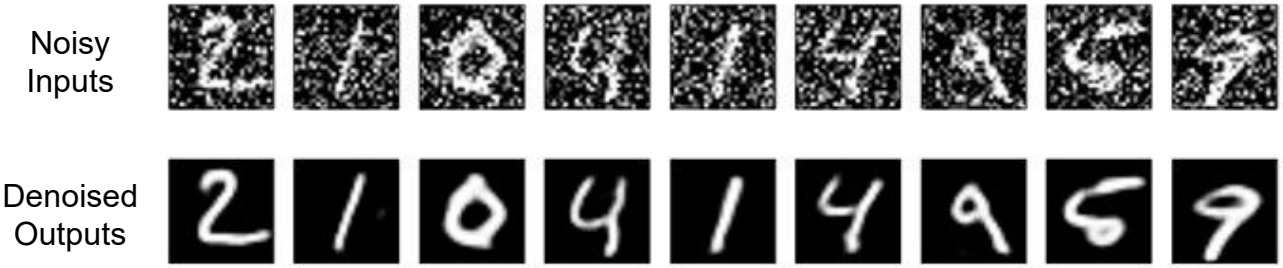


- **Beispiel:** Verwendung eines 32-dimensionalen latenten Raums zur Kodierung eines 28x28 MNIST-Bildes
  - Der typische Schreibstil kann gut dargestellt werden...
  - während die atypischen verzerrt werden.

Denoising Autoencoder

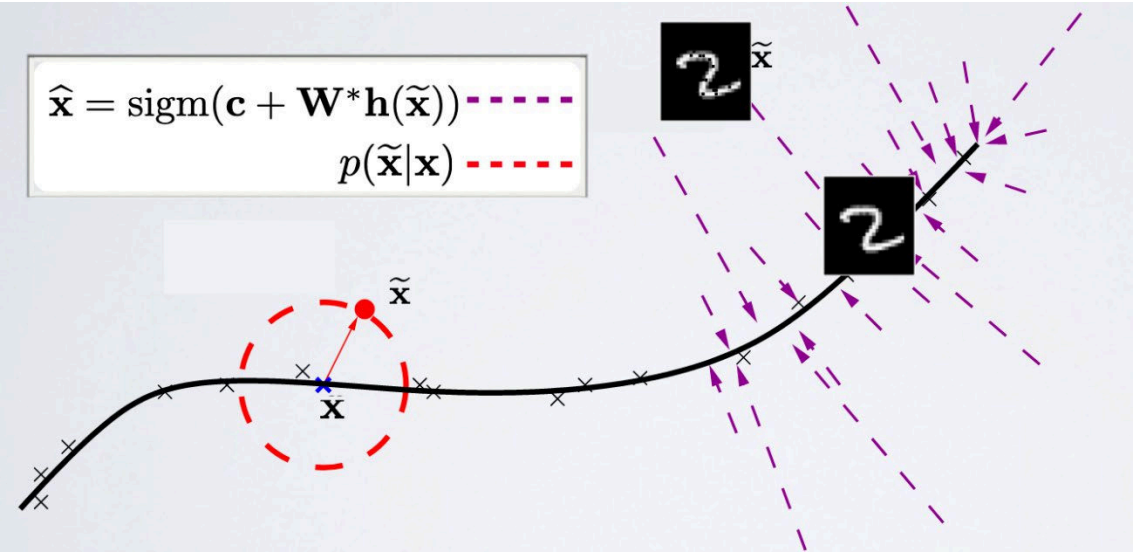
- **Rauschunterdrückung:** Eingabe eines Bildes + Rauschen. Training zur Reproduktion des rauschfreien Bildes
  - Ein Beispiel für "selbstüberwachtes" Lernen → Generierung von überwachten Lernaufgaben aus unüberwachten Daten
- **Grundlage für Diffusionsmodelle:** beste heute bekannte generative Modelle (z.B. Dall-E 2, nicht behandelt, siehe Master-VL)





Denoising Autoencoder

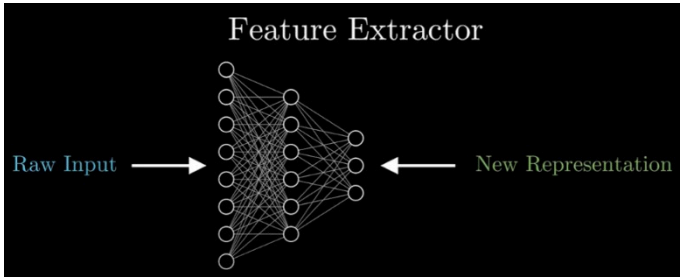
- Denoising-Autoencoder können sich nicht einfach die Relation zwischen Eingabe und Ausgabe “merken”.
- Intuitiv lernt ein Denoising-Autoencoder eine regularisierende Projektion aus einer Nachbarschaft unserer Trainingsdaten zurück auf die Trainingsdaten.



Autoencoder - Anwendungen

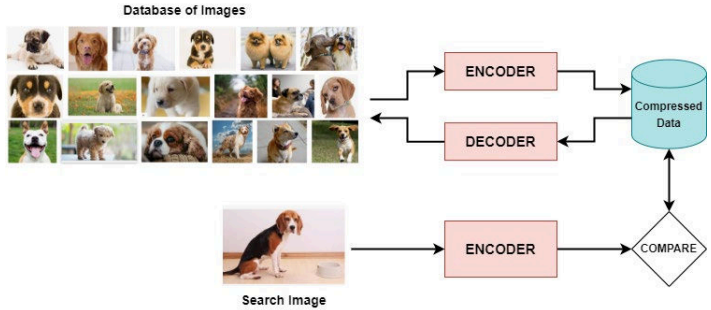
Merkmalsextraktion (Feature extraction):

- Merkmale als Eingabe für überwachten Algorithmus verwenden
- Nützlich, wenn nur ein kleiner überwachter Datensatz verfügbar ist



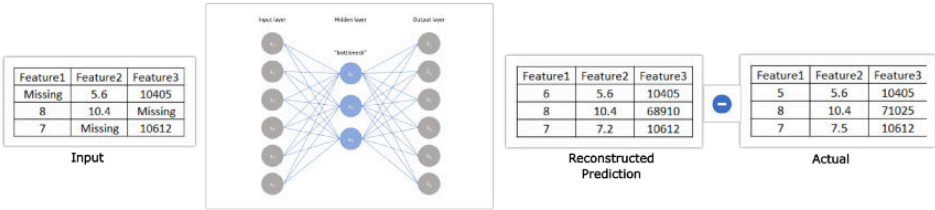
Bildsuche

- Das komprimierte Embedding kann mit einer kodierten Version des Suchbildes verglichen oder durchsucht werden.



Imputation fehlender Werte

- Fehlende Werte im Datensatz
- Trainieren fehlende Werte zu rekonstruieren



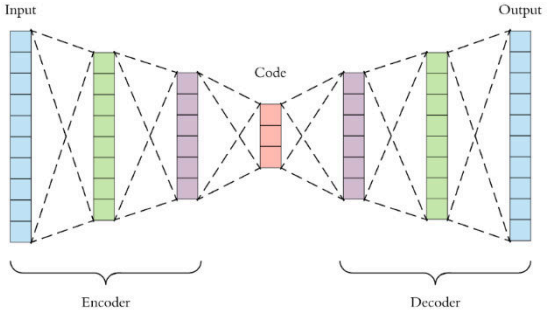
Autoencoder - Anwendungen

- **Bildeinfärbung:** Eingabe von Schwarzweißbildern und Training zur Erzeugung von Farbbildern

- **Entfernung von Wasserzeichen**

- **Erkennung von Anomalien:** Wenn der Rekonstruktionsfehler groß ist, handelt es sich um ein “ungewöhnliches” (out-of-distribution) Beispiel.

Zusammenfassung - Autoencoder



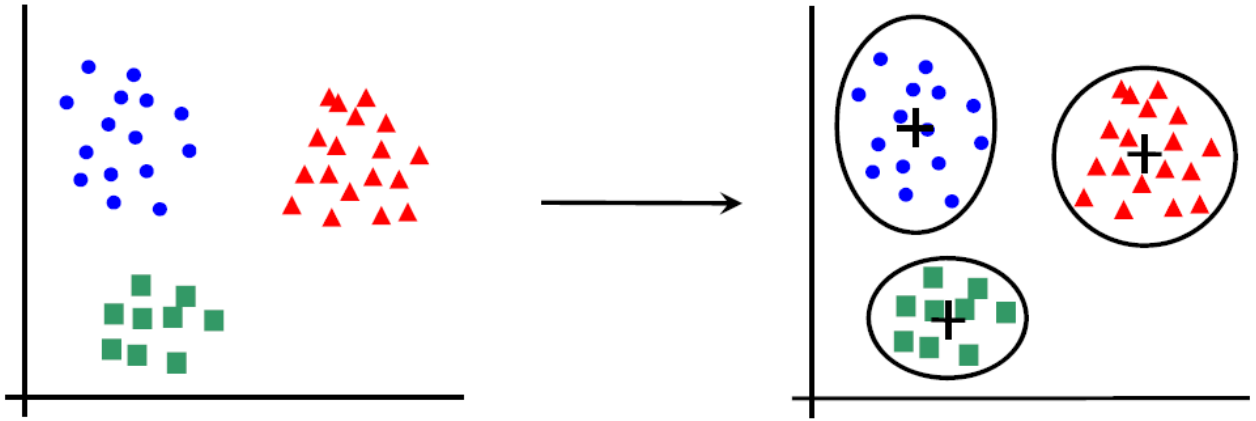
- Autoencoder (AE) projizieren Daten mit Hilfe von tiefen neuronalen Netzwerken in einen **nichtlinearen Unterraum.**
  - **Encoder:** Erzeugung eines latenten Codes (mehrere NN-Schichten)
  - **Decoder:** Rekonstruktion des Inputs (wiederum mehrere NN-Schichten)
- AE verwenden **Bottleneck-Schichten**, um einen "repräsentativen Code" zu erlernen.
- AE können für **selbstüberwachtes Lernen** eingesetzt werden: Rauschunterdrückung, Imputation, Einfärbung, Wasserzeichenentfernung, …

Fragen?

K-Means Clustering

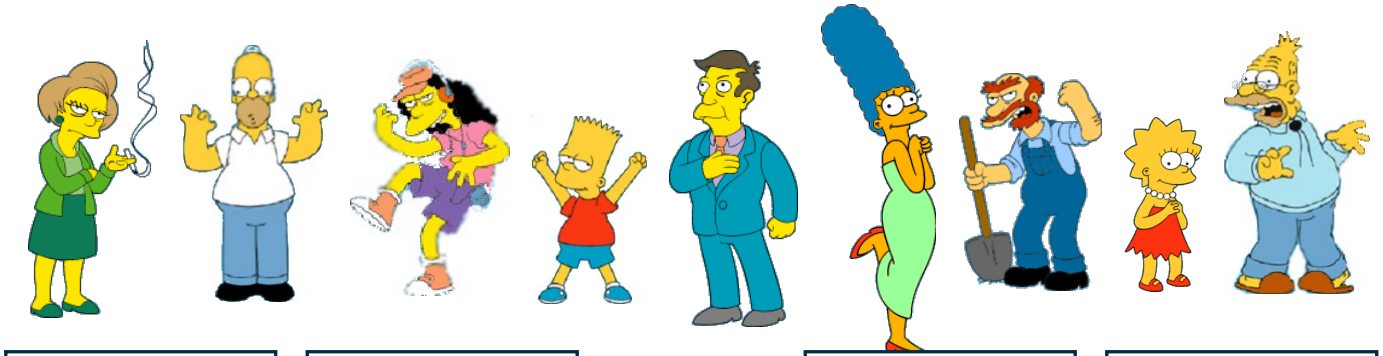
Clustering

Bei der Clusteranalyse oder dem **Clustering** geht es darum, eine Menge von Objekten so zu gruppieren, dass die Objekte in derselben Gruppe (**Cluster** genannt) einander (in gewissem Sinne) ähnlicher sind als die Objekte in anderen Gruppen.



Clustering ist subjektiv (wie Ähnlichkeit messen?)

Welche natürliche Gruppierung haben diese Objekte?



Simpsons Familie | Mitarbeiter der Schule | Frauen | Männer

Clustering ist subjektiv (wie Ähnlichkeit messen?)

Welche Merkmale wir wählen und wie wir Distanz messen ist zentral!

University of Illinois

Clustering: Auffinden von Strukturen in den Daten

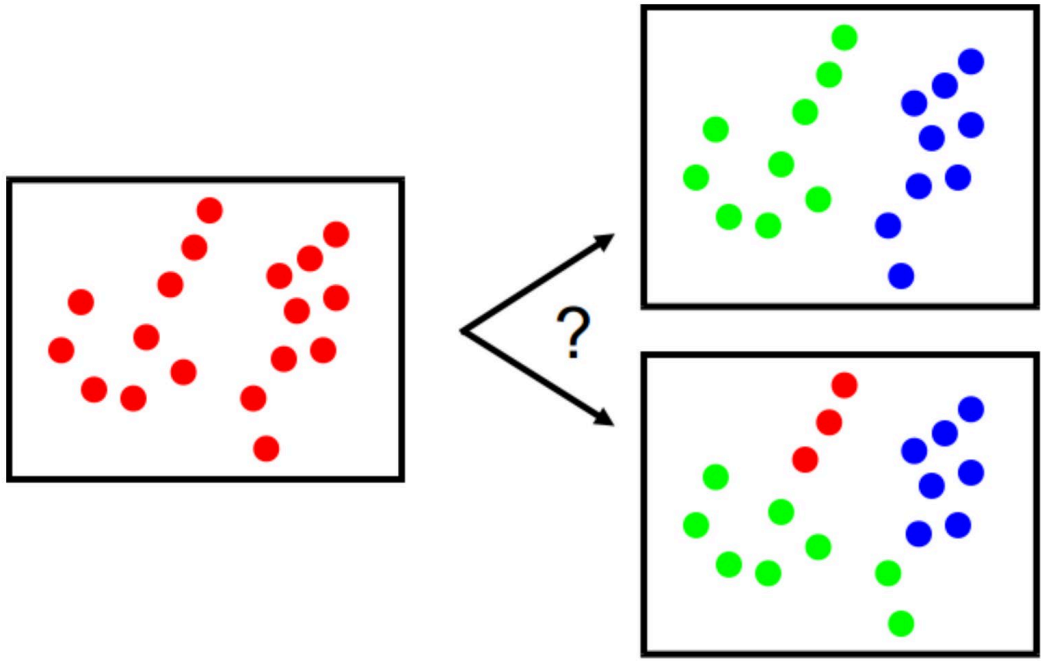
Welches sind die richtigen Cluster?

Groundtruth oft nicht verfügbar

Maß für die Ähnlichkeit

Clustering verlässt sich auf ein Ähnlichkeitsmaß

z.B. Position im Raum (euklidische vs. log-polare Koordinaten), Gewichtung verschiedener Dimensionen (Merkmale)...



Distanzmaße...was ist Ähnlichkeit?

Ähnlichkeit ist schwer zu definieren, aber...
"*Wir wissen es, wenn wir es sehen*"

Distanzmaße

Die gebräuchlichste Abstandsmetrik ist der **euklidische Abstand (ED)**:

$$d(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}|| = \sqrt{\left(\sum_{k=1}^{d} (\boldsymbol{x}_k - \boldsymbol{y}_k)^2\right)}$$

- ED ist sinnvoll, wenn verschiedene Merkmale die gleiche Einheit haben
- Wenn die Einheiten unterschiedlich sind, zum Beispiel Länge und Gewicht, müssen die Daten normalisiert/standardisiert werden:

$$\tilde{\boldsymbol{x}} = (\boldsymbol{x} - \boldsymbol{\mu}) \oslash \boldsymbol{\sigma}$$

Mittelwert μ , Standardabweichung σ, elementweise Division $\oslash$

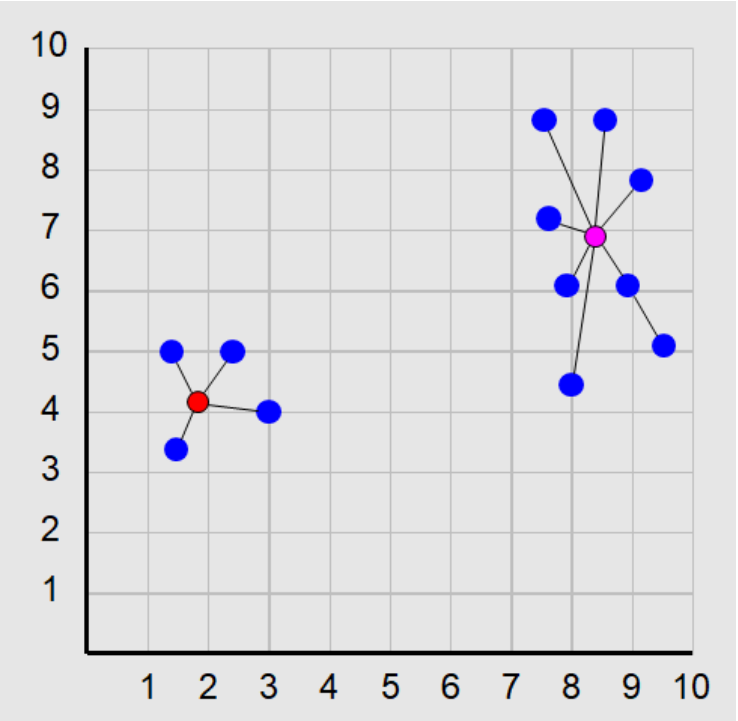
D.h. die resultierenden Eingangsdimensionen haben einen Mittelwert von Null und eine Einheitsvarianz

K-Means Algorithmus – ein einfacher Clustering-Algorithmus

Ziel: Minimierung des Quantisierungsfehlers!

- Gegeben: Daten $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$
- Suche nach Clusterzentren/Centroids $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_k\}$
- Bezeichne c(**x**) den Centroid $\boldsymbol{c} \in C$, der $\boldsymbol{x}$ am nächsten liegt.
- Die Summe der quadrierten Abstände (SSD) bezeichnet den Quantisierungsfehler:

$$\mathrm{SSD}(C; \mathcal{D}) = \sum_{i=1}^{n} d(\boldsymbol{x}_i, c(\boldsymbol{x}_i))^2$$



K-Means Algorithmus (Lloyd's Algorithmus)

Iteratives Verfahren

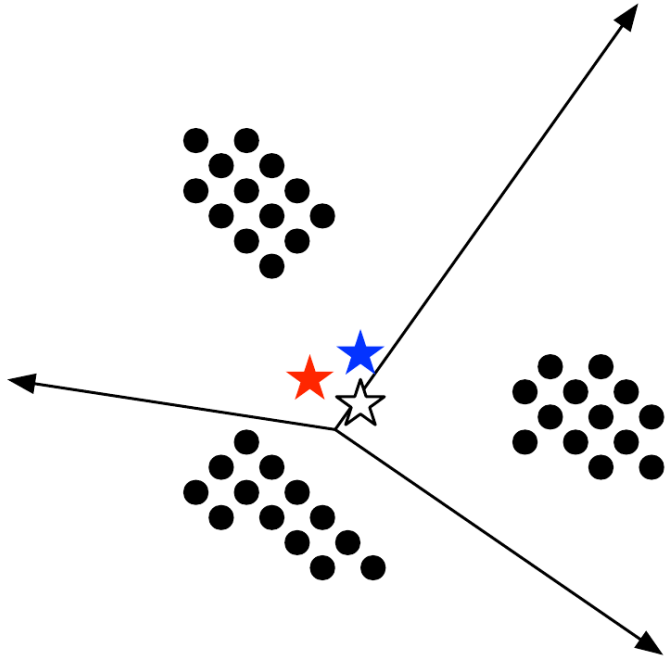
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt 1: Die Sterne sind Zentren von Clustern, zunächst zufällig zugewiesen

K-Means Algorithmus

Iteratives Verfahren

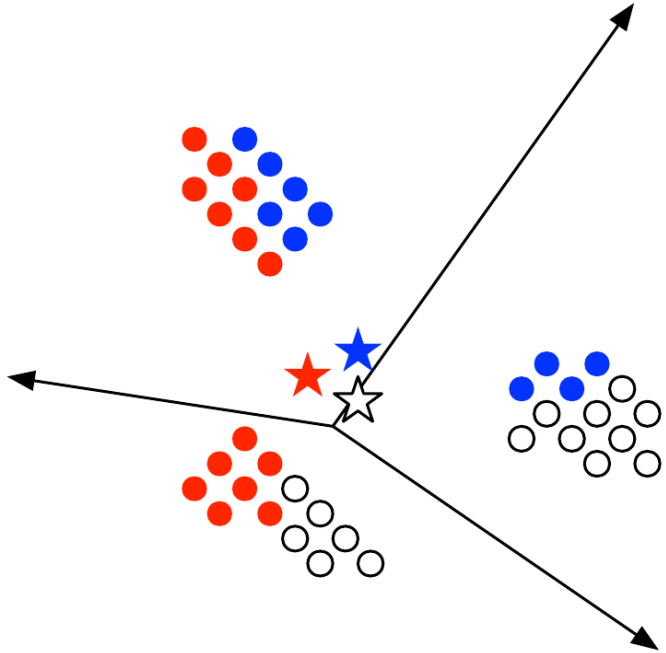
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt 2: Ordne jedes Beispiel dem nächstgelegenen Zentroid zu

K-Means Algorithmus

Iteratives Verfahren

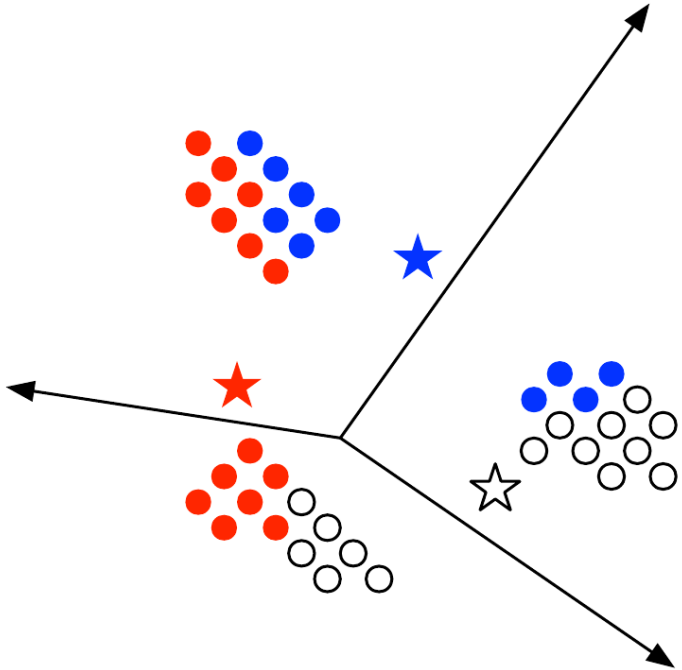
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt 3: Anpassung der Zentroide an die Mittelwerte der ihnen zugeordneten Beispiele

K-Means Algorithmus

Iteratives Verfahren

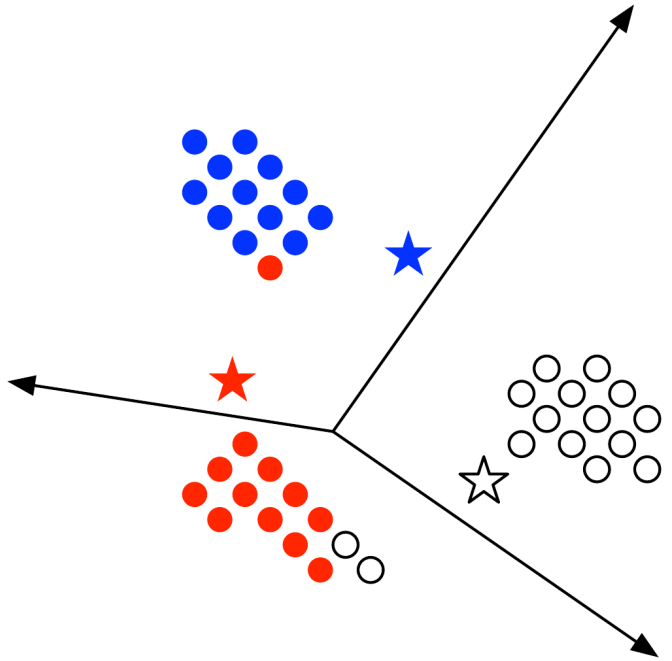
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt 4: Ordne jedes Beispiel dem nächstgelegenen Zentroid zu (Wiederholung Schritt 2)

K-Means Algorithmus

Iteratives Verfahren

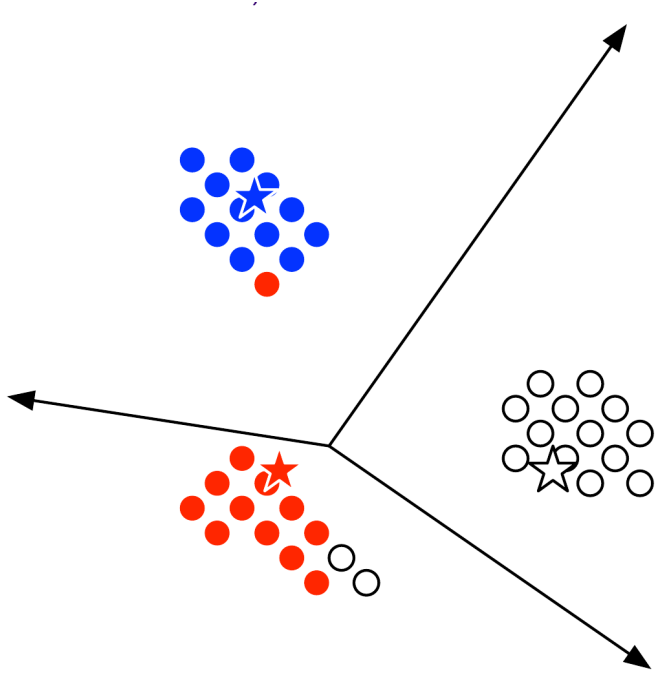
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt 5: Anpassung der Zentroide an die Mittelwerte der ihnen zugeordneten Beispiele (Wiederholung Schritt 3)

K-Means Algorithmus

Iteratives Verfahren

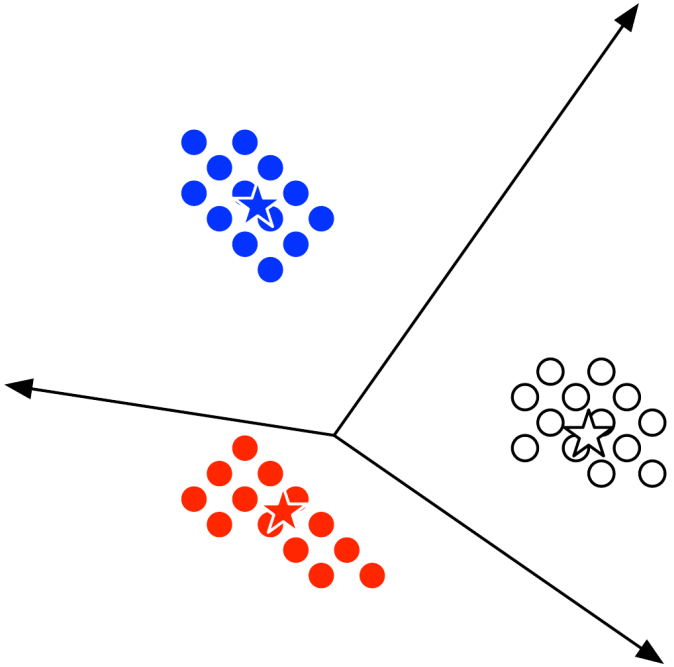
1. Wähle K beliebige Clusterzentren
2. Jedes Beispiel dem nächstgelegenen Centroid zuordnen

$$z_n = \arg\min_k ||\boldsymbol{c}_k - \boldsymbol{x}_n||^2$$

3. Anpassung der Centroids an die Mittelwerte der zugewiesenen Beispiele

$$\boldsymbol{c}_k = \frac{1}{|X_k|} \sum_{\boldsymbol{x}_i \in X_k} \boldsymbol{x}_i, \quad X_k = \{\boldsymbol{x}_n | z_n = k\}$$

4. Schritte 2,3 wiederholen, bis keine Änderung mehr erfolgt



Schritt N: Konvergenz…

You cannot vote anymore

Welche der folgenden Aussagen, denken Sie, trifft auf den K-Means Algorithmus zu? (wählen Sie alle korrekten Aussagen)

1	K-Means mit euklidischem Distanzmaß wird immer konvergieren.	34%	27 ✓
2	Wir finden immer ein globales Optimum.	6%	5
3	"Random Initialization" ist der beste Weg um Konvergenz zu garantieren.	21%	17
4	Keine der drei Aussagen oben ist korrekt.	55%	44

Konvergiert K-Means?

Um die Konvergenz zu analysieren, schreiben wir den SSD in Form von Zuweisungen z_k

$$\mathrm{SSD}(C;\mathcal{D}) = \sum_{i=1}^{n} d(\boldsymbol{x}_i, c(\boldsymbol{x}_i))^2 = \sum_{i=1}^{n}\sum_{k} q_{ik} d(\boldsymbol{x}_i, \boldsymbol{c}_k)^2,$$

wobei $q_{ik} = \mathbb{I}(z_i = k)$ den Wert 1 annimmt, wenn das i-te Beispiel dem k-ten Cluster zugeordnet ist, und andernfalls 0 (1-Hot-Coding).

- **Zuweisungsschritt:** Minimiert SSD in Bezug auf z_i = q_{ik}
  - Setzt q_{ik} für den nächstgelegenen Cluster auf 1, alle anderen Werte sind 0

- **Anpassungsschritt:** Minimiert SSD in Bezug auf die Mittelpunkte $\boldsymbol{c_k}$

$$\boldsymbol{c}_k = \frac{1}{\sum_j q_{jk}} \sum_{i=1}^{n} q_{ik}\boldsymbol{x}_i$$

  - Der Durchschnittsvektor ist der Vektor mit dem geringsten quadratischen Abstand zu allen zugeordneten Beispielen

K-Means Analyse

Konvergiert K-Means? Ja, es minimiert (lokal) die SSD!

- Es gibt nur eine endliche Anzahl von möglichen Werten für die Centroids
- Jeder Zuweisungs- oder Anpassungsschritt reduziert die SSD (oder sie bleibt konstant)

Konvergiert K-Means zur globalen Lösung mit minimalen Kosten? Nein!

- Das Ziel ist ein NP-schweres Problem, nicht möglich ohne im Wesentlichen (fast) alle Zuweisungen zu überprüfen.
- Es hängt stark von der Initialisierung der Zentren ab.

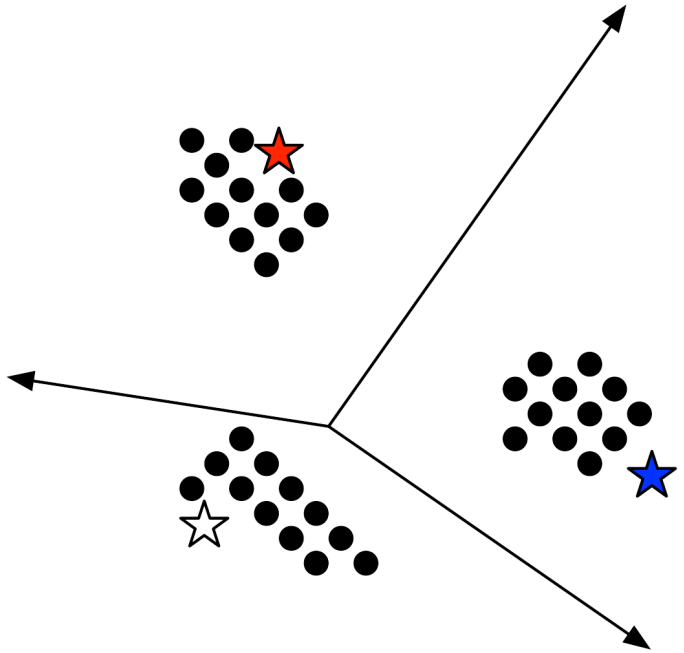
K-means++

Furthest Point Initialisierung:

- Wählen Sie einen zufälligen Datenpunkt als ersten Centroid
- **für k ∈ {2, . . . , K} do**
  - Finde den Datenpunkt $\boldsymbol{x}_i$ der am weitesten von allen zuvor ausgewählten Centroids entfernt ist

$$\text{let } n = \arg \max_{i \in \{1,\ldots,n\}} \left(\min_{k' \in \{1,\ldots k-1\}} ||\boldsymbol{x}_i - \boldsymbol{c}'_k||^2 \right)$$

  - Centroid zuweisen: $\boldsymbol{c}_k = \boldsymbol{x}_i$



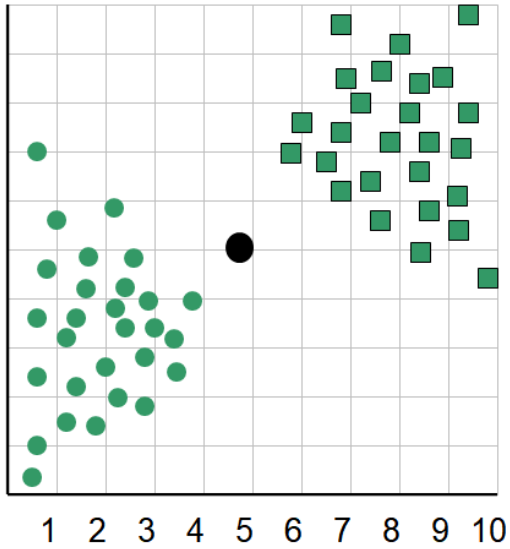
Die Furthest Point Initialization in Aktion...
Konvergiert in diesem Fall nach einer Anpassung

Anzahl von Clustern

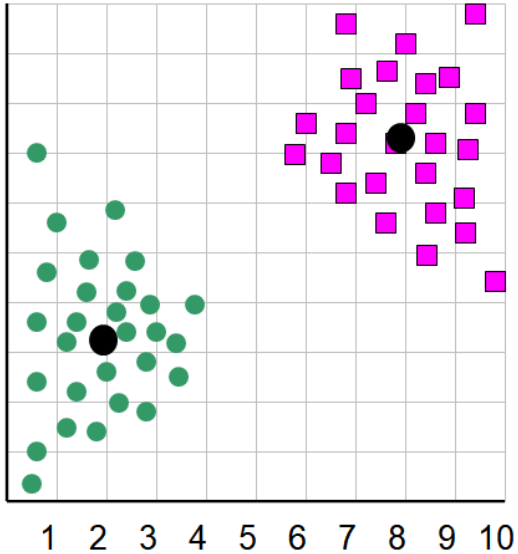
Wie wähle ich K?

- Verwendung eines "Holdout"-Set / Cross-Validation (gut, aber teuer)
- Oder: “Elbow Method" (ähnlich der PCA, heuristisch, aber billig)

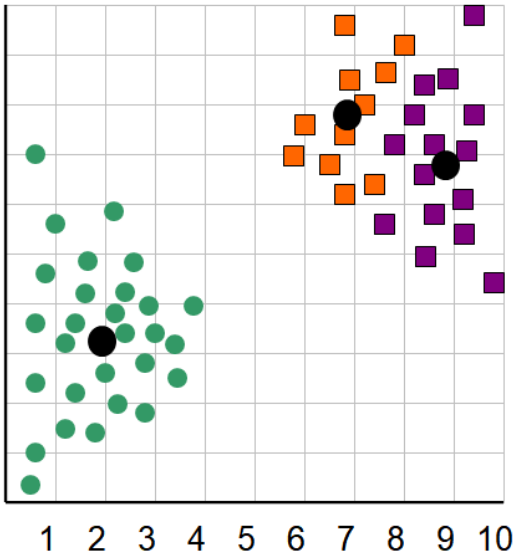
When k = 1, the objective function is 873.0



When k = 2, the objective function is 173.1

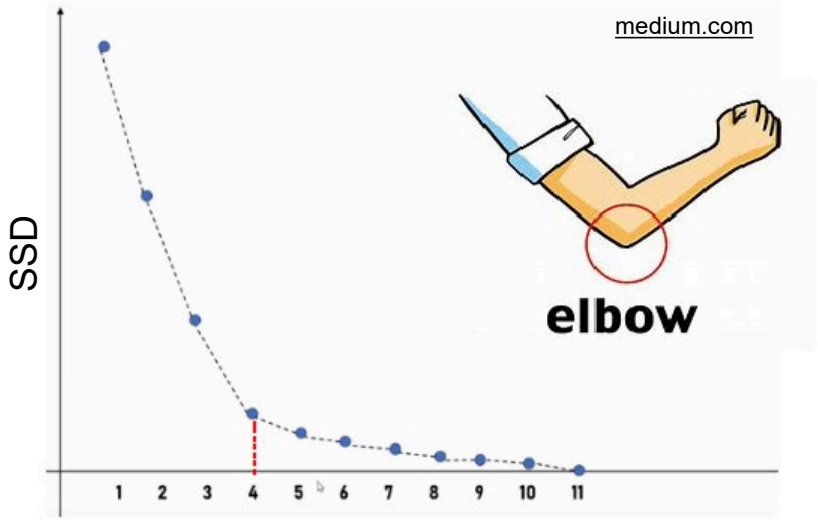


When k = 3, the objective function is 133.6

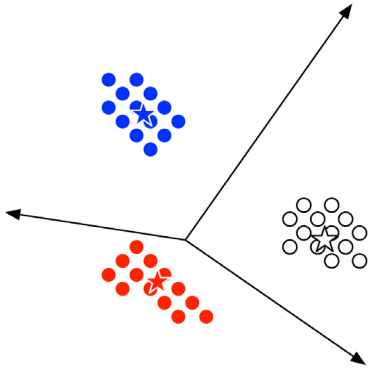


Elbow method

- Werte der Zielfunktion (SSD) abhängig von k plotten.
- SSD sinkt mit höherem k (im Durchschnitt)
- Rechts: der abrupte Wechsel bei $k = 4$ deutet auf vier große Cluster in den Daten hin.
- Es gibt alternative Maße, die noch besser die Struktur der Cluster relativ zueinander abbilden, z.B. das “Silhouette Score” (hier nicht behandelt).



Zusammenfassung K-means



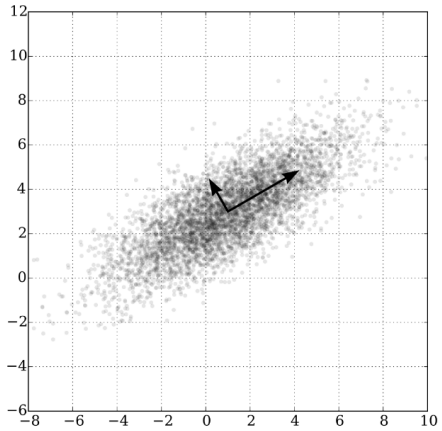
Stärken:

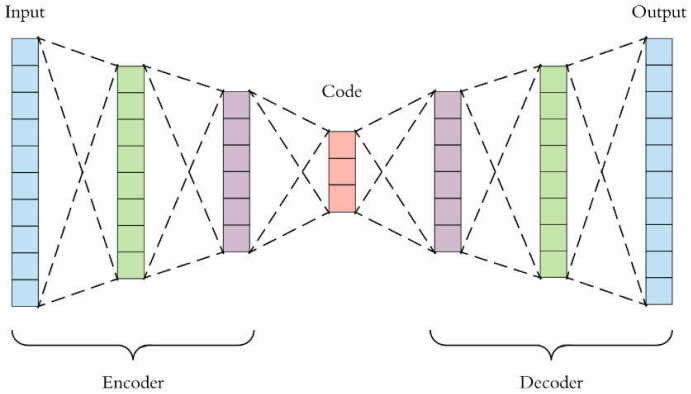
- K-means konvergiert in der Praxis in der Regel sehr schnell.
- K-means++ noch besser, aber findet noch immer nicht garantiert das globale Optimum
  - In der Praxis können wir in lokalen Optima stecken bleiben.
  - Arbeiten mit mehrere Initialisierungen (verwenden Sie ein wenig Zufälligkeit in K-means++ und lassen Sie den Algorithmus mehrmals laufen).

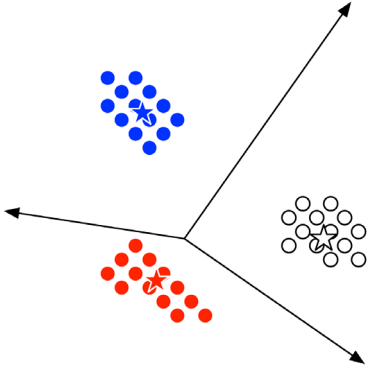
Limitierungen:

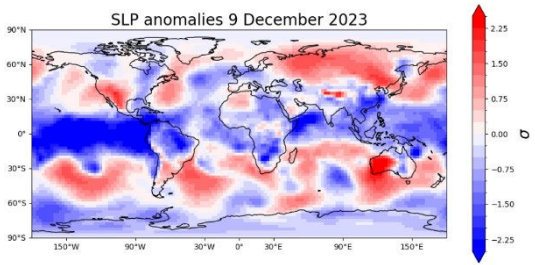
- Gilt nur, wenn der Mittelwert definiert ist, was ist dann mit kategorischen Daten?
- Nicht gut geeignet für verrauschte Daten und Ausreißer (Euklidische Distanz)
- K-means kann keine Formen von Clustern lernen; Clusterzugehörigkeit rein durch den minimalen Abstand zum Mittelpunkt bestimmt.

Heute haben wir gelernt...









1) Dimensionalitätsreduktion
2) PCA
3) Autoencoder
4) Selbstüberwachtes Lernen
5) K-means Clustering

Fragen?