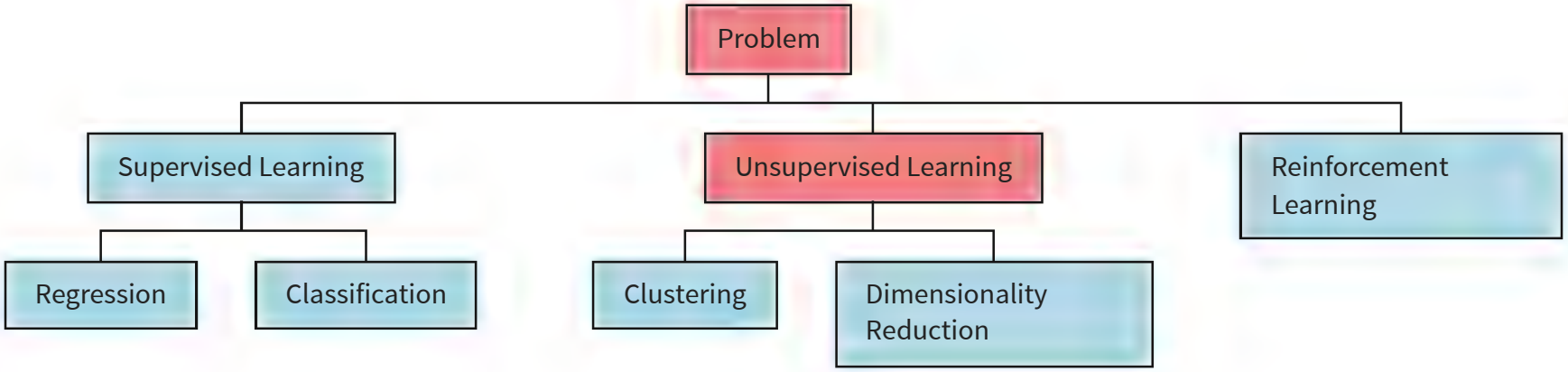
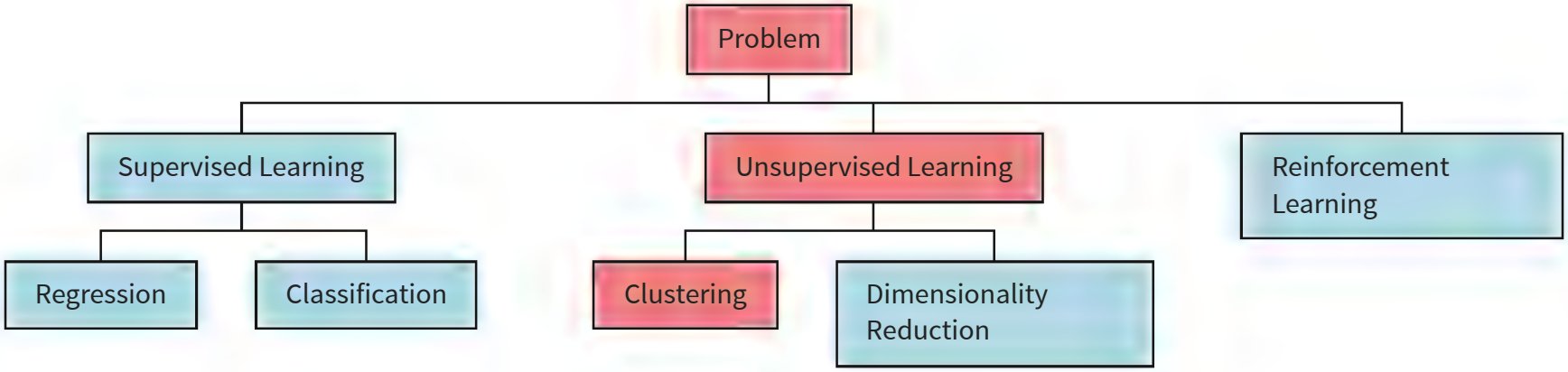
TAG 3

Unsupervised Learning, Clustering, Dimensionality Reduction

UNSUPERVISED LEARNING

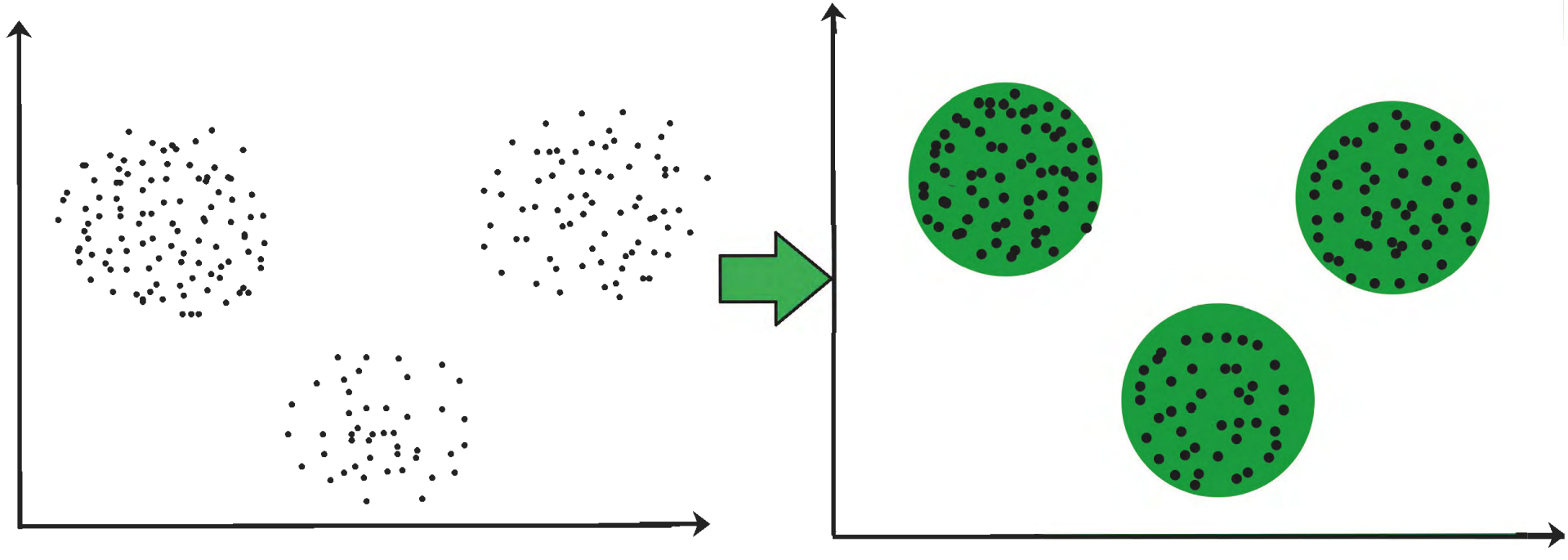


CLUSTERING

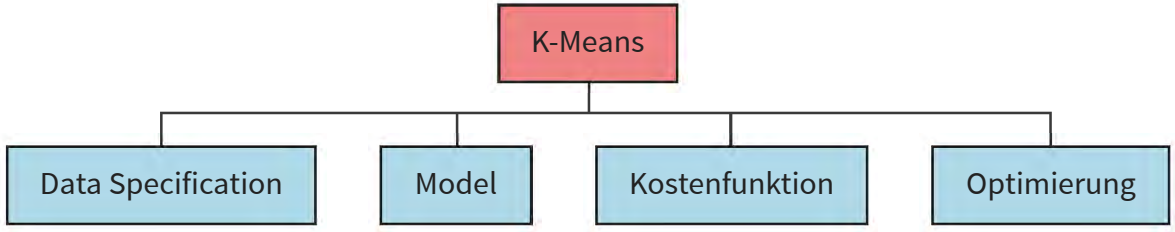

Problem
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Regression
Classification
Clustering
Dimensionality Reduction

CLUSTERING

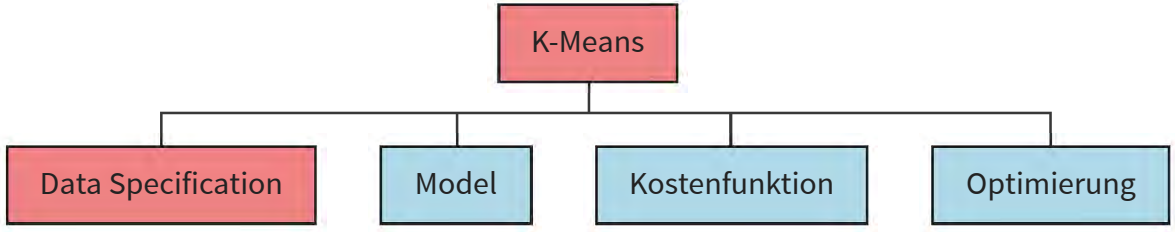
Ziel: Finde zusammenhängende Gruppen in Datenwolke.



K-MEANS


K-Means
Data Specification
Model
Kostenfunktion
Optimierung

K-MEANS

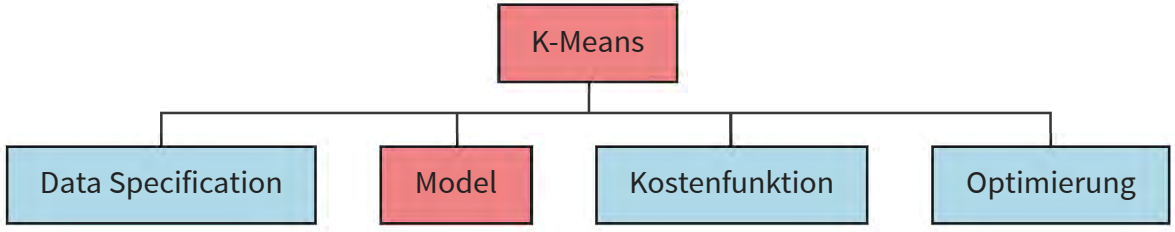

K-Means
Data Specification
Model
Kostenfunktion
Optimierung

K-MEANS - DATA SPECIFICATION

- Welche Features wählen wir.
- Kategorische Features müssen encoded werden.
- Numerische Features müssen standardisiert werden.
- Wie viele Clusters suchen wir.

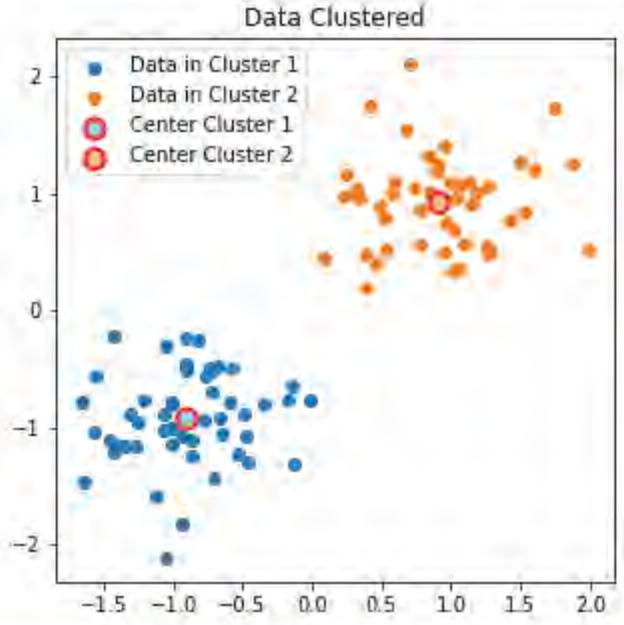
Beachte: Wir haben keine Zielvariable. Wir sind im Unsupervised Learning

K-MEANS


K-Means
Data Specification
Model
Kostenfunktion
Optimierung

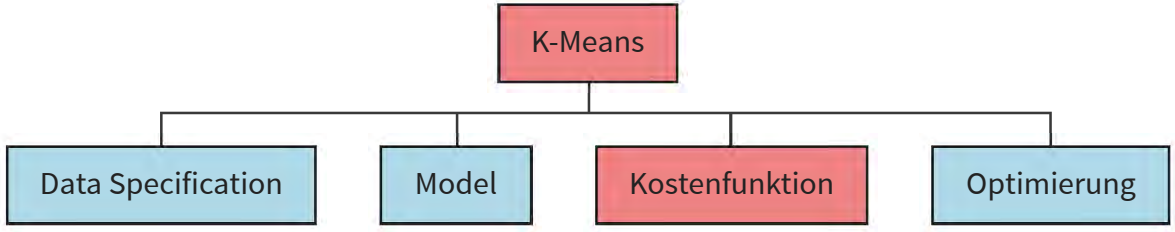
K-MEANS - INTUITION

Finde k (hier 2) Cluster-Schwerpunkte. Daten werden dem nächsten Cluster-Schwerpunkt zugewiesen.



Punkte haben kein (bekanntes) Klassen-Label.

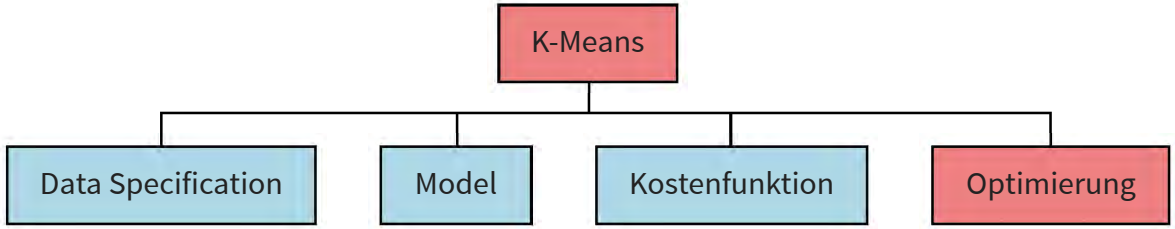
K-MEANS


K-Means
Data Specification
Model
Kostenfunktion
Optimierung

K-MEANS - KOSTENFUNKTION

$$
\begin{aligned}
J(C) &= \frac{1}{2} \sum_{k=1}^{K} \sum_{x^{(i)}, x^{(j)} \in C_k} \| x^{(i)} - x^{(j)} \|^2 \\
&= \frac{1}{2} \sum_{k=1}^{K} N_k \sum_{x^{(i)} \in C_k} \| x^{(i)} - \mu_k \|^2 \\
&\sim \sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \| x^{(i)} - \mu_k \|^2
\end{aligned}
$$

K-MEANS


K-Means
Data Specification
Model
Kostenfunktion
Optimierung

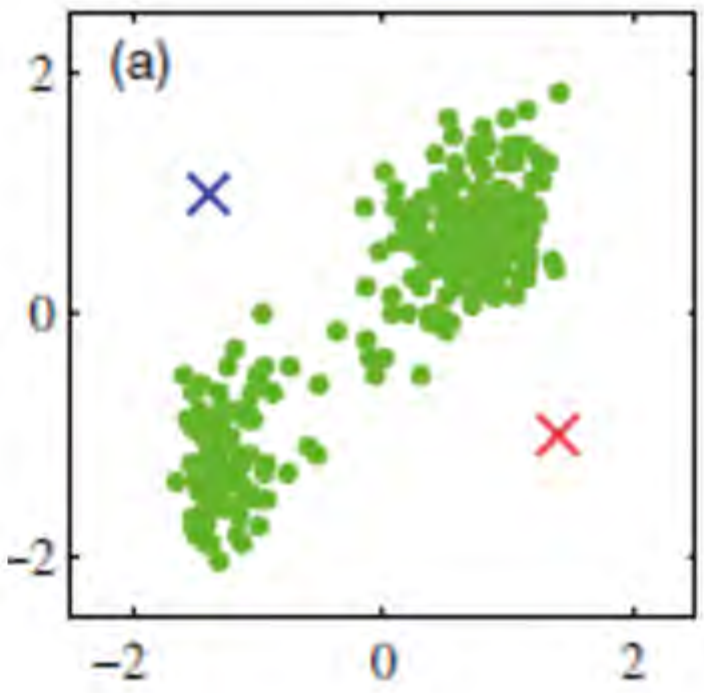
K-MEANS - OPTIMIERUNG

Der Algorithmus funktioniert wie folgt:

1. Initialisiere Schwerpunkte mit zufälligen Werten.
2. Weise Datenpunkte nächstem Schwerpunkt zu.
   Minimiert Kostenfunktion betreffend momentaner Cluster Zugehörigkeit
3. Aktualisiere Schwerpunkte anhand der zugewiesenen Datenpunkte.
   Minimiert Kostenfunktion betreffend momentanem Cluster Schwerpunkt
4. Wiederhole 2. und 3., bis keine Veränderung mehr statt findet.

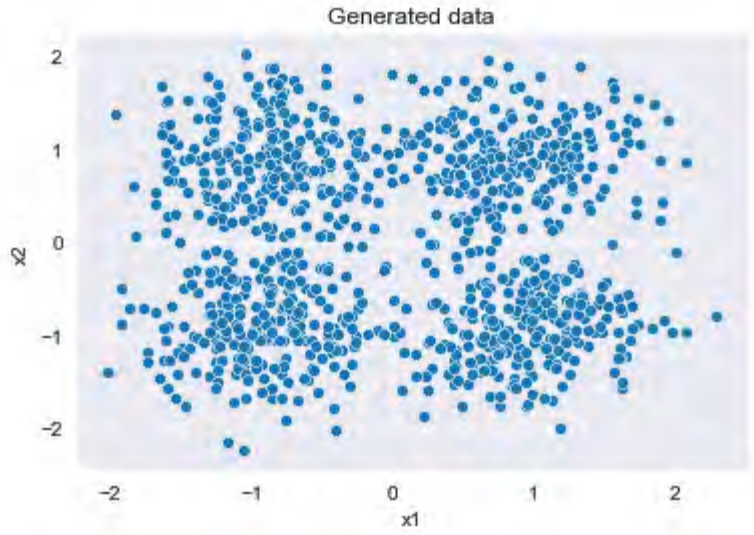
Findet lokales Minimum! Ganzen Algorithmus mehrmals wiederholen mit anderen Initialwerten.

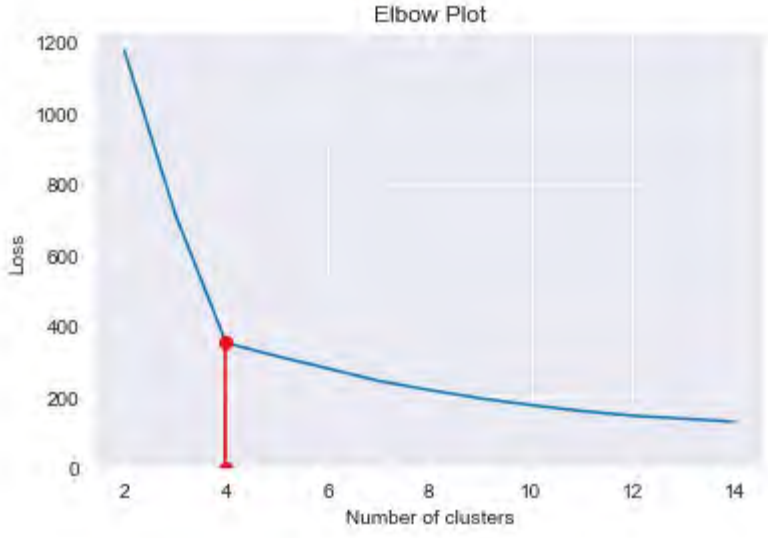
K-MEANS - OPTIMIERUNG - INTUITION



K-MEANS - ELBOW PLOT

num_cluster gibt an wie viele Cluster wir suchen.



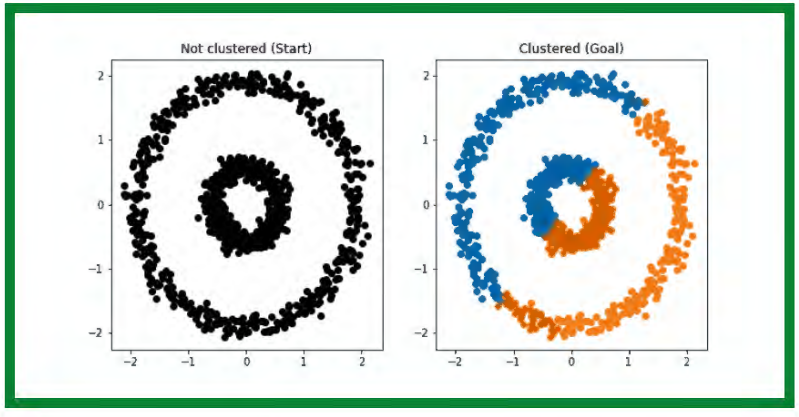


K-MEANS - CODE

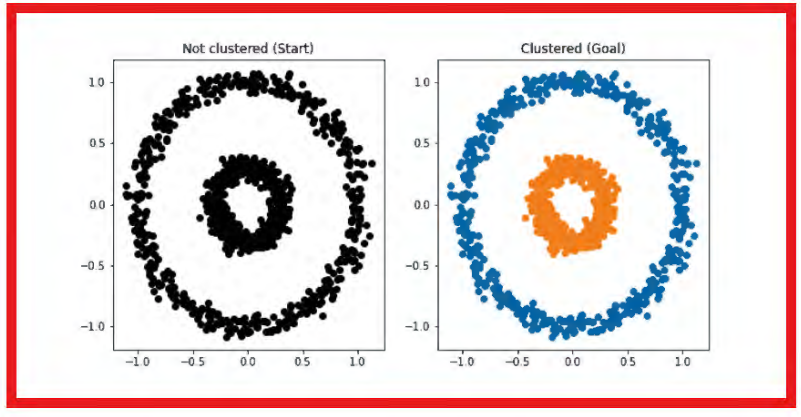
`k_means.ipynb`

K-MEANS - LIMITS

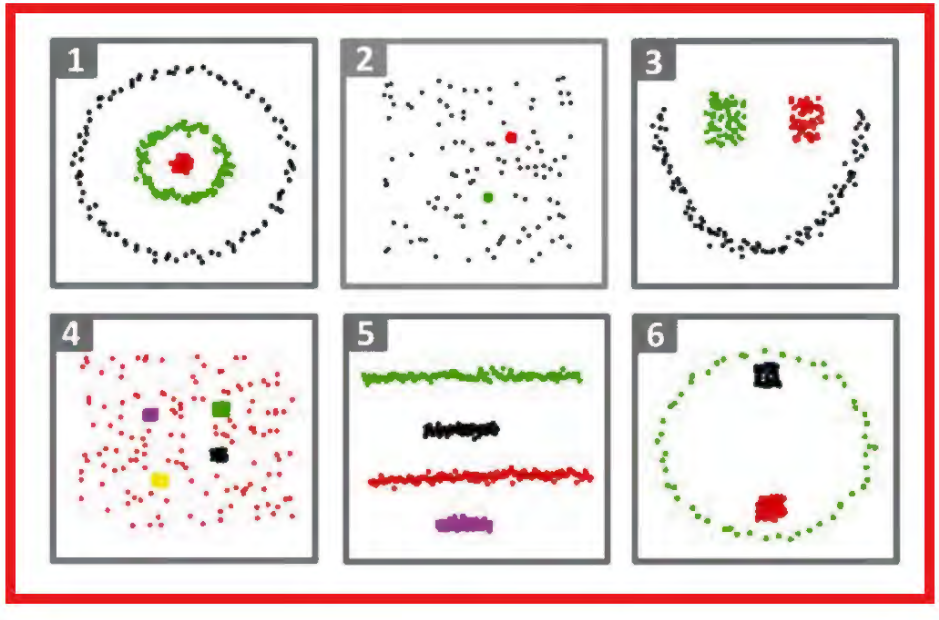
POSSIBLE



IMPOSSIBLE (IN FEATURE SPACE)

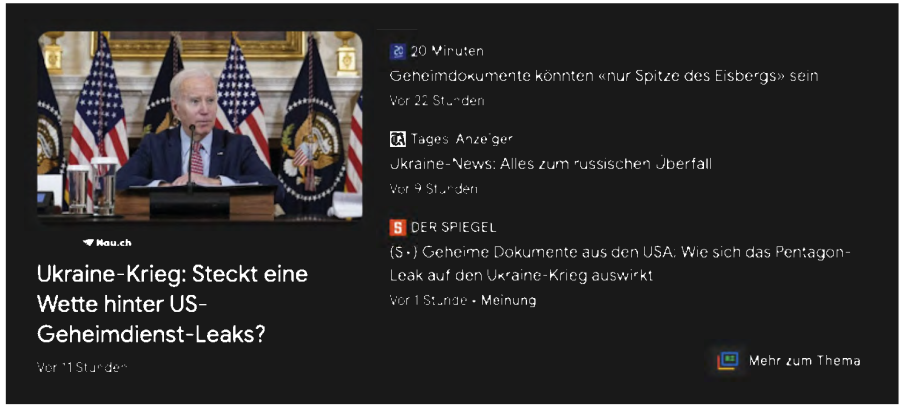


ANDERE "IMPOSSIBLE" BEISPIELE

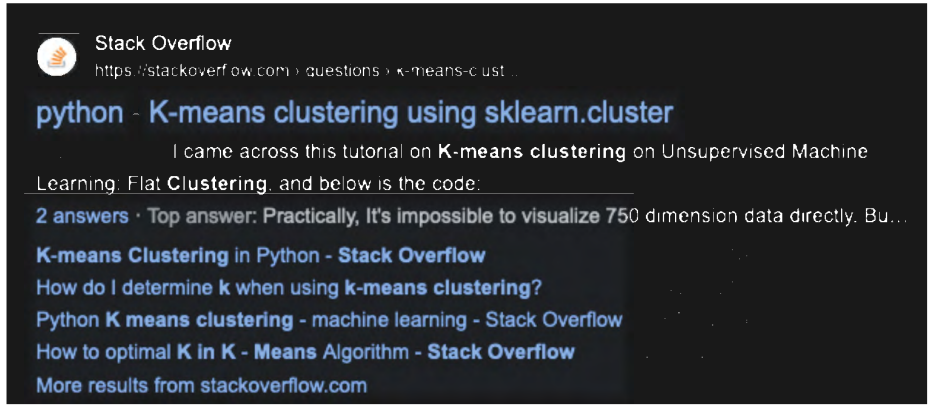


CLUSTERING - REAL WORLD EXAMPLES

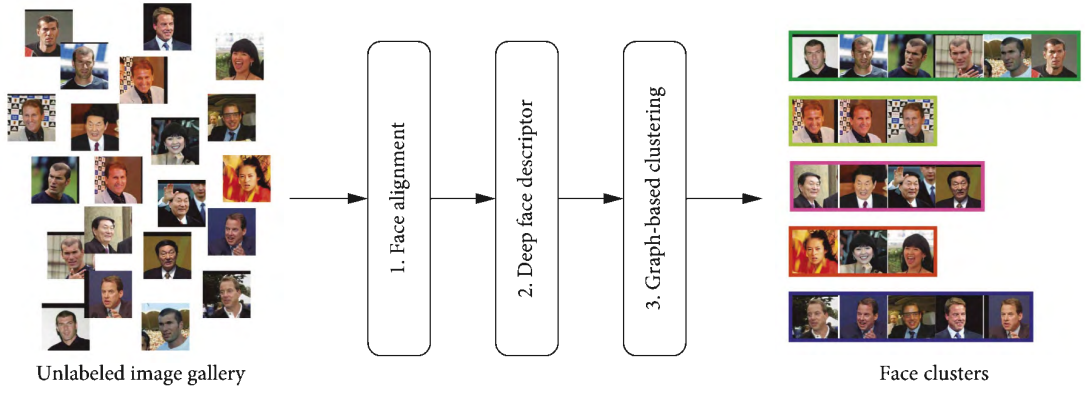
GRUPPIEREN VON NEWS STORIES



GRUPPIEREN VON SUCHERGEBNISSEN



GRUPPIEREN VON GESICHTERN



Source: Effective and Generalizable Graph-Based Clustering for Faces in the Wild

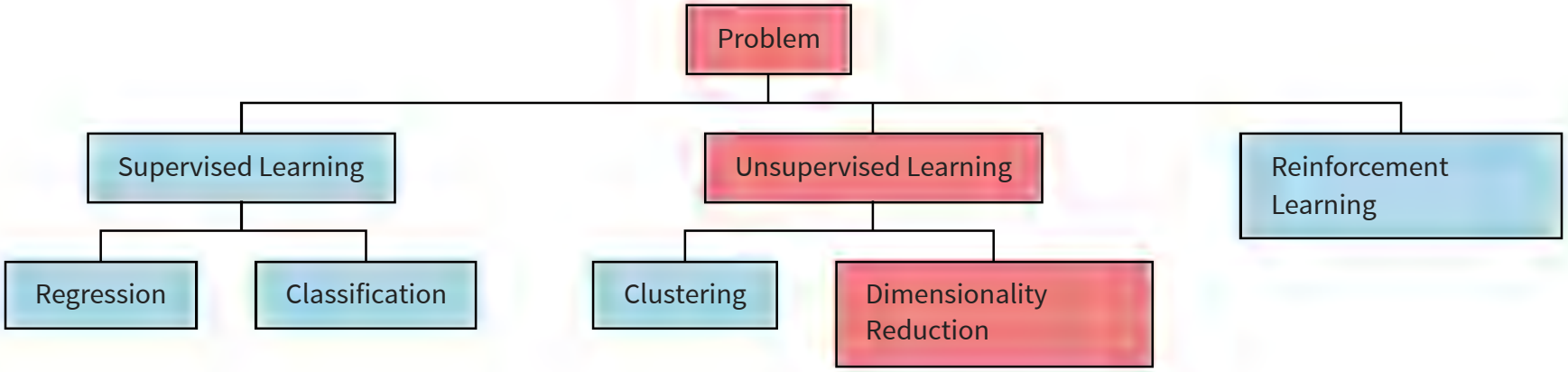
ALTERNATIVE & ADDITIONAL RESOURCES

- [Alternative] K-Means
  - sklearn User Guide
  - "Praxiseinstieg Machine Learning" - Kapitel 9

[Additional] Clustering

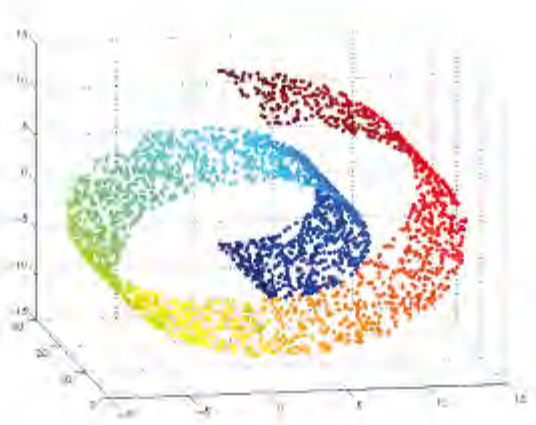
- Survey Paper: A Comprehensive Survey of Clustering Algorithms
- Survey Paper: A Short Review on Different Clustering Techniques and Their Applications

DIMENSIONALITY REDUCTION



DIMENSIONALITY REDUCTION - MANIFOLD

3 FEATURES

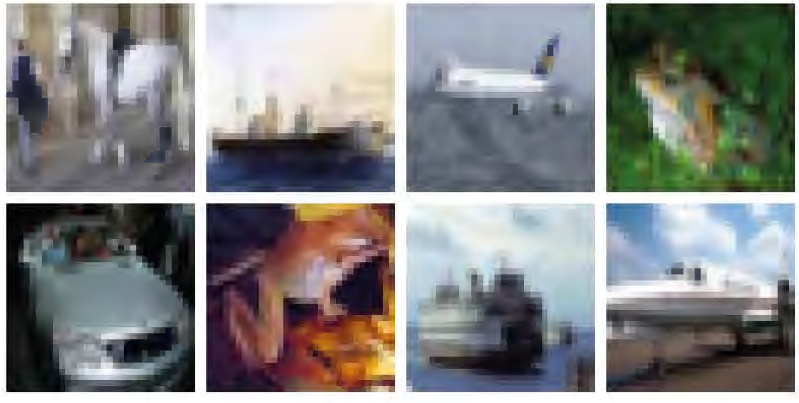


ABER EIGENTLICH 2 FEATURES

- 2D-Manifold im 3D-Input-Space
- Manifold-Annahme: Hochdimensionale Daten liegen auf einem tiefer-dimensionalen Manifold

DIMENSIONALITY REDUCTION - MANIFOLD - BEISPIEL

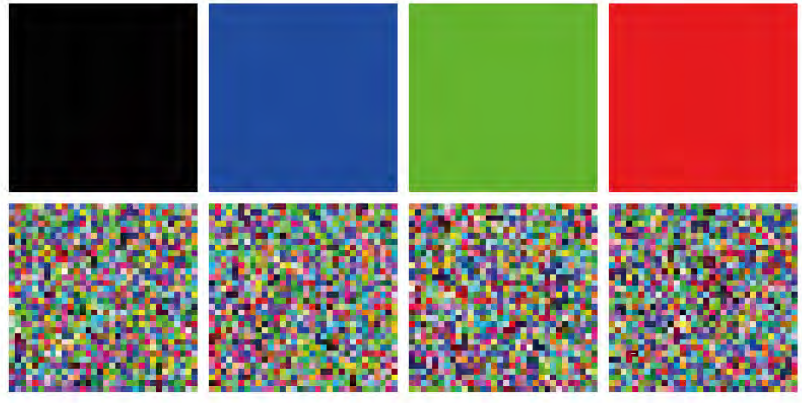
CIFAR-SPACE



32 Pixel * 32 Pixel * 3

Channels = 3072 Features

IMAGE-SPACE (3072 FEATURES = 32 PIXEL * 32 PIXEL * 3 CHANNELS)



32 Pixel * 32 Pixel * 3

Channels = 3072 Features

Annahme: CIFAR-Space ist ein Manifold im Image-Space

Dimensionsreduktion: Finde neue Features (z.B. 200), die die Daten präziser beschreiben können.

DIMENSIONALITY REDUCTION - BEISPIELE

INPUT SPACE
3072 PIXELWERTE (FEATURES)

Encoder

z.B. Neural Network

Backbone, PCA Encoder

LATENT SPACE
200 FEATURES

```
[ 167, 235, 207,
76, 97, 119, 50,
8, 239, 168,
215, ... ]
```

INPUT SPACE
1'000'000 WORDS (1-HOT-FEATURES)

```
[0, ..., 0, 1,
0, ..., 0]
```

Encoder / Embedder

z.B. Word2Vec

LATENT SPACE
300 FEATURES

```
[ -1.7, 0.97,
1.9, 0.73, ...
-0.53 ]
```

Ziel ist die Reduktion von Features (Dimensionen).

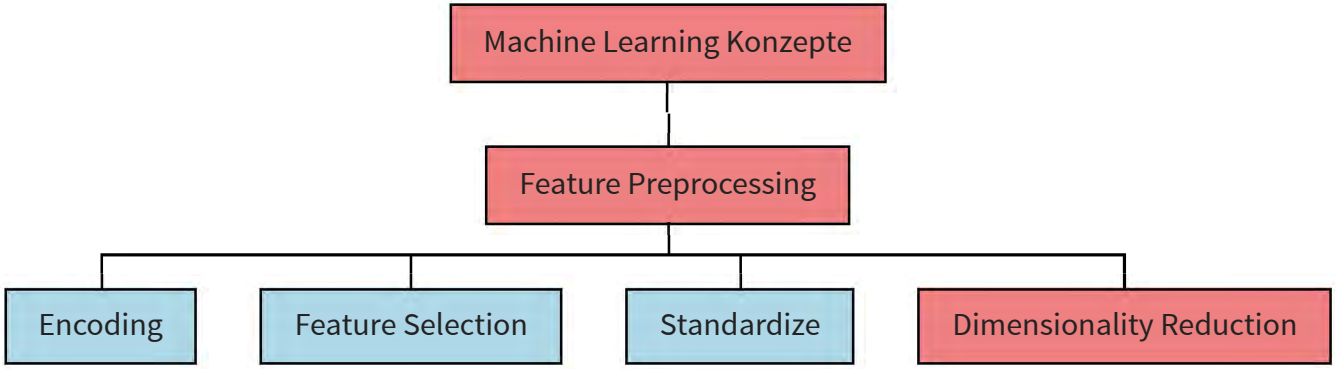
DIMENSIONALITY REDUCTION - LEARNING

- Eine Dimensionality Reduction wird gelernt indem ein ganzer Datensatz (z.B. 40'000 Bilder) auf Gemeinsamkeiten analysiert wird.
- Es wird die zugrundeliegende Struktur (z.B. Manifold) des Datensatzes gelernt.
- Die gelernte Reduktion kann dann auf neue Daten (neue Bilder) angewandt werden.
- Die gelernte Reduktion kann auch auf die Daten selbst angewandt werden.

DIMENSIONALITY REDUCTION - EINSATZGEBIET

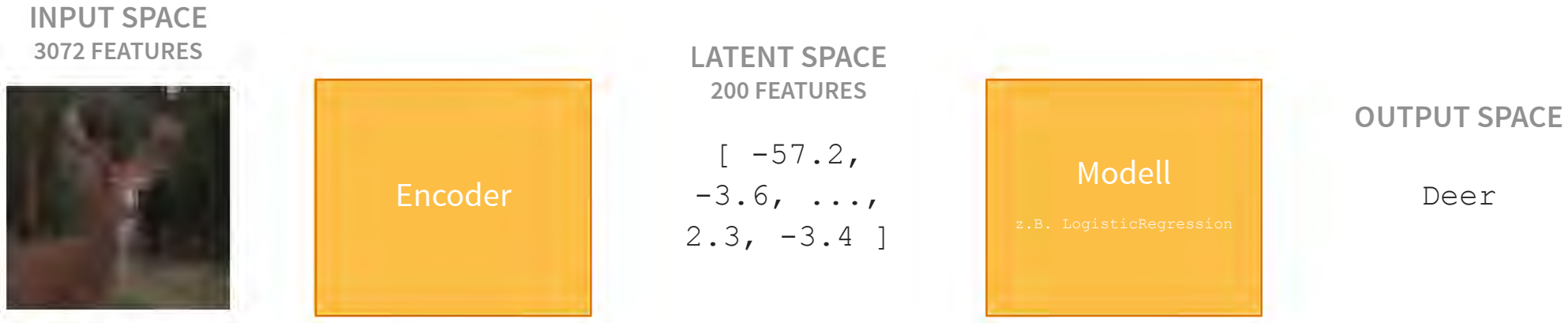
- Feature Preprocessing
- (Interpretation der Daten)
  - Visualisierung
  - Faktor-Analyse

DIMENSIONALITY REDUCTION



DIMENSIONALITY REDUCTION - FEATURE PREPROCESSING

1. Dimensionality Reduction (Feature Preprocessing)
2. Modell (e.g. Supervised Learning) im Latent Space



- Overfitting
- Performanz Optimierung:
  - Speed-up (CPU)
  - Less memory (RAM/disk)

DIMENSIONALITY REDUCTION - OVERFITTING

INPUT SPACE
3072 FEATURES

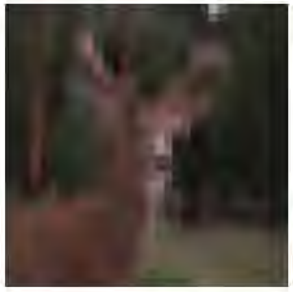
Modell
z.B. LogisticRegression

OUTPUT SPACE

Frog

Mit wenig Daten overfitted das Modell

INPUT SPACE
3072 FEATURES



Encoder

LATENT SPACE
200 FEATURES

[-57.2,
-3.6, ...,
2.3, -3.4]

Modell
z.B. LogisticRegression

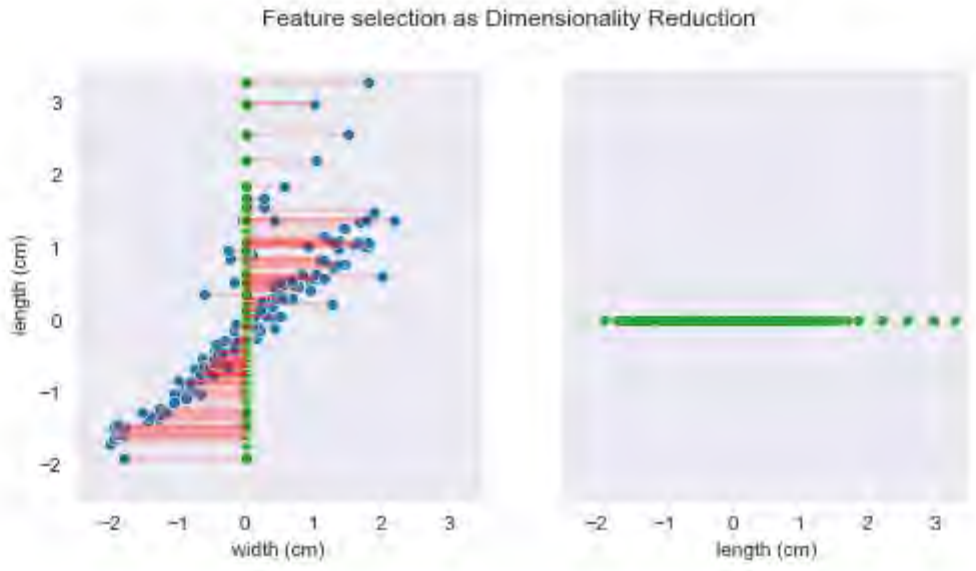
OUTPUT SPACE

Deer

Präzisere Features => Bessere Lern-Ausgangslage

FEATURE SELECTION IST DIMENSIONALITY REDUCTION?

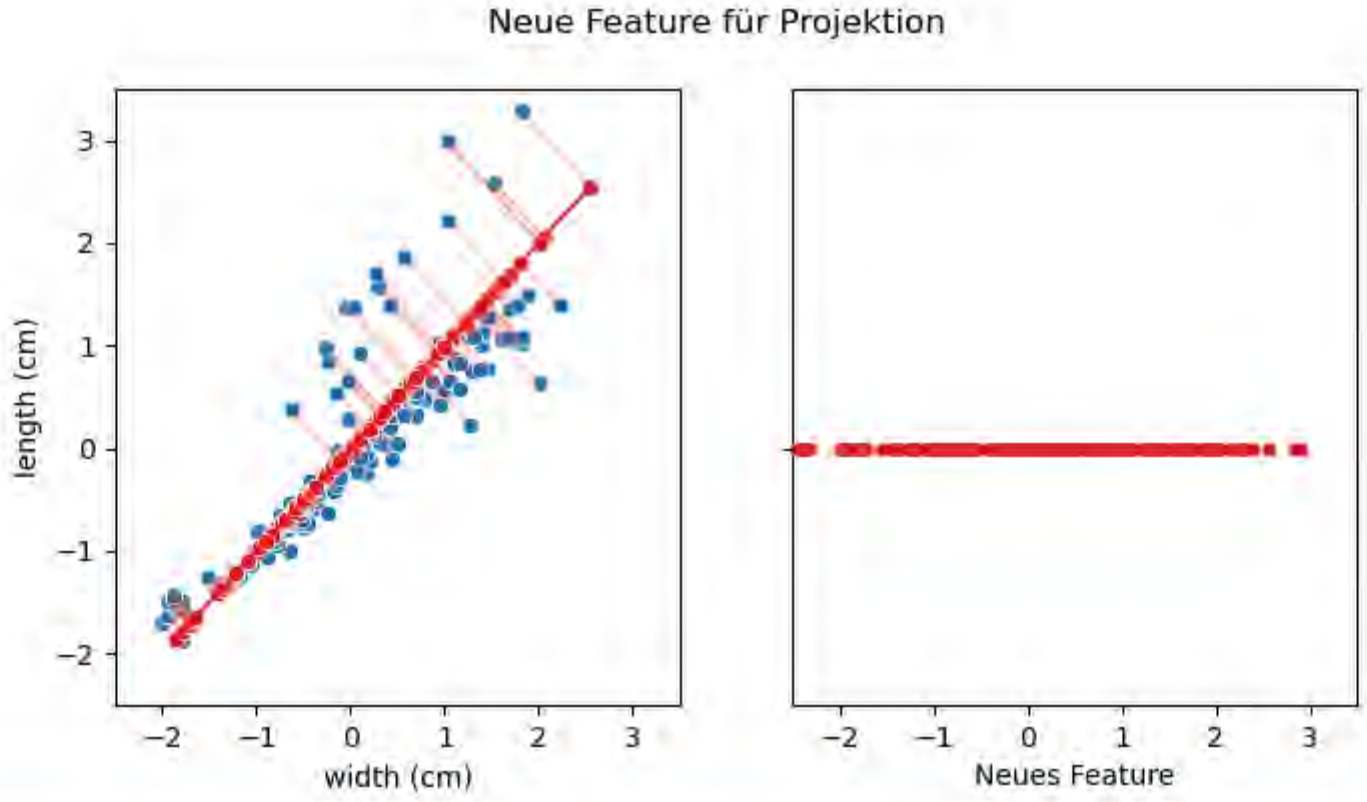
- Idee: Wir reduzieren die Anzahl der Features (Dimensionen) indem wir Features weglassen.



Problem: Grosser Informationsverlust

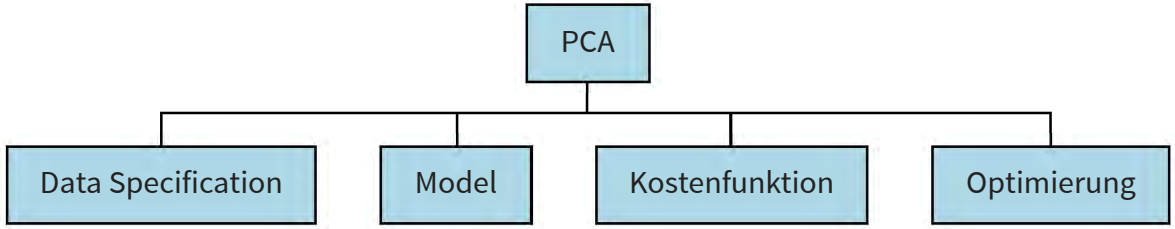
BESSERE DIMENSIONALITY REDUCTION?

- Idee: Wir reduzieren die Anzahl der Features indem wir weniger neue Features erstellen.

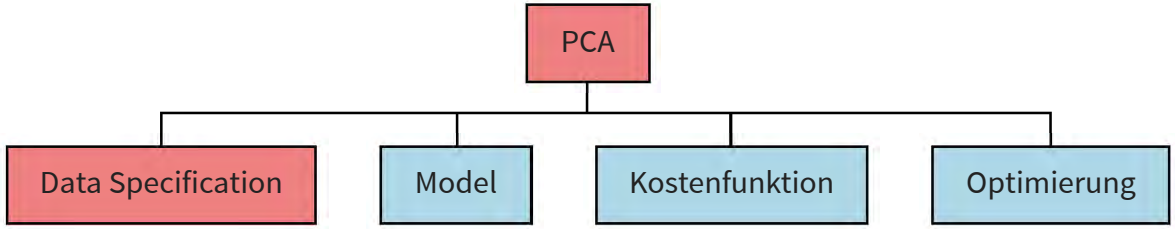


Weniger Informationsverlust

PCA


PCA
Data Specification
Model
Kostenfunktion
Optimierung

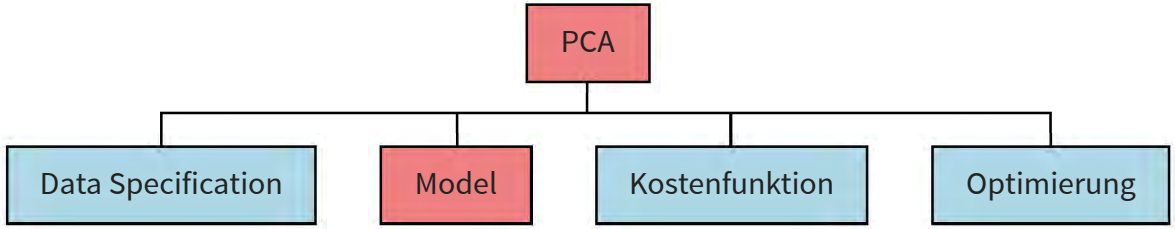
PCA


PCA
Data Specification
Model
Kostenfunktion
Optimierung

PCA - DATA SPECIFICATION

1. Welche Features haben wir, z.B. `pixel`
2. Kategorische Features müssen encoded werden.
3. Numerische Features müssen standardisiert werden.

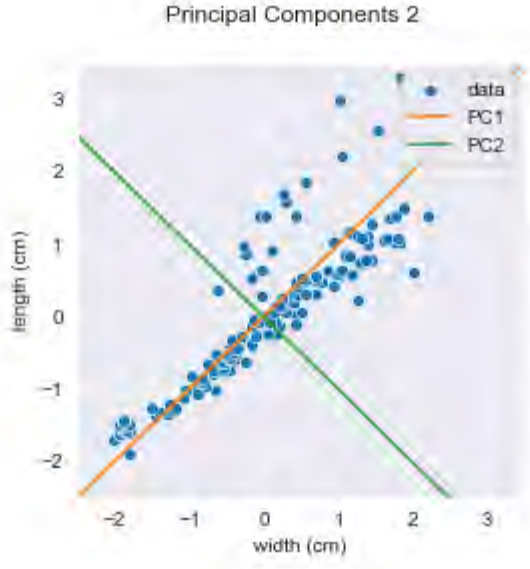
PCA


PCA
Data Specification
Model
Kostenfunktion
Optimierung

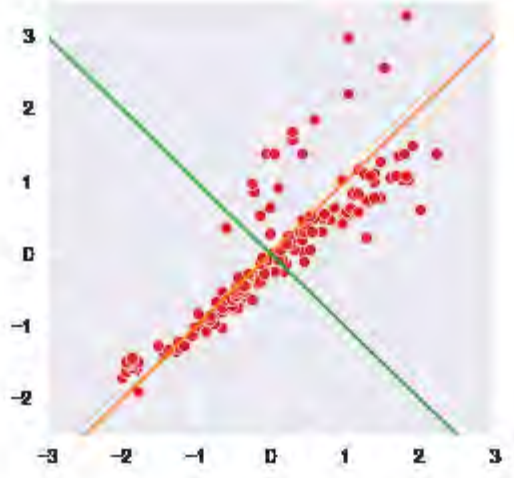
PCA - INTUITION

- Wir suchen die Principal Components (Hauptkomponenten) des Datensatzes
- Die Hauptkomponenten werden zu unserem neuen Koordinatensystem (neuen Features)

HAUPTKOMPONENTEN IM INPUT SPACE

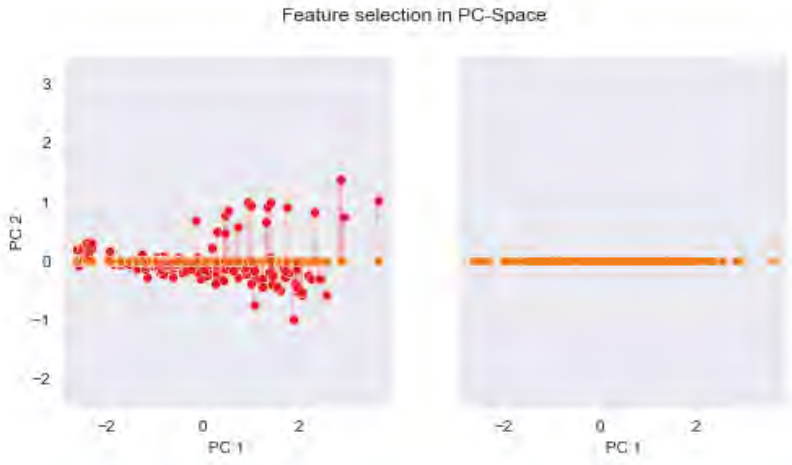


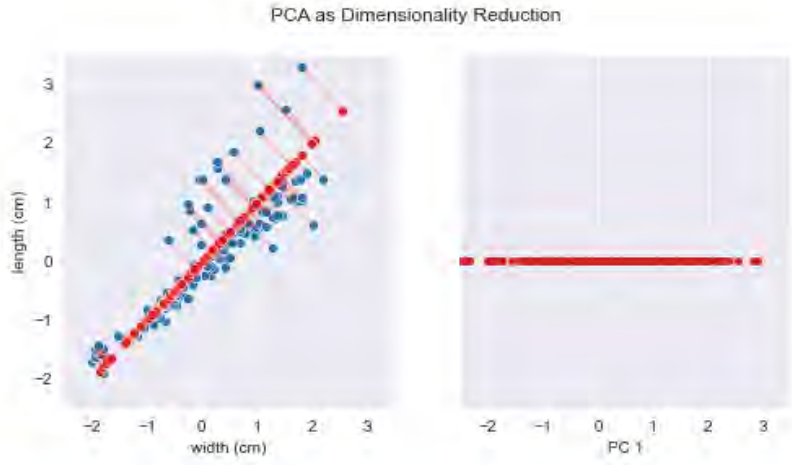
DATEN IM PC-SPACE



PCA - PC SELECTION

Im PC-Space machen wir dann "Feature Selection", sprich wir behalten nur die wichtigsten Hauptkomponenten (neuen "Features").





Weniger Informationsverlust, aber es gehen weiterhin Informationen verloren!

PCA - MODELL

$$L = xW$$

INPUT SPACE
3072 FEATURES

$L = xW$
PCA.transform

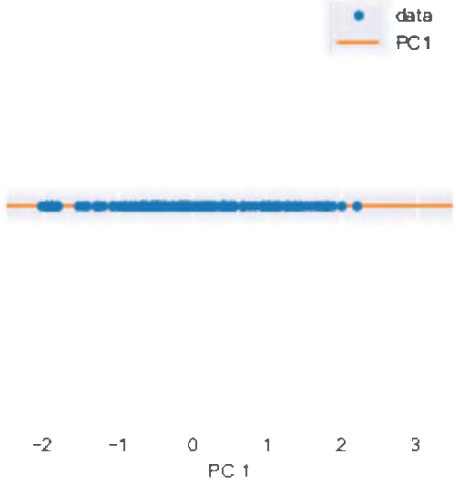
LATENT SPACE
200 FEATURES

```
[ -57.2, -3.6,
..., 2.3, -3.4 ]
```

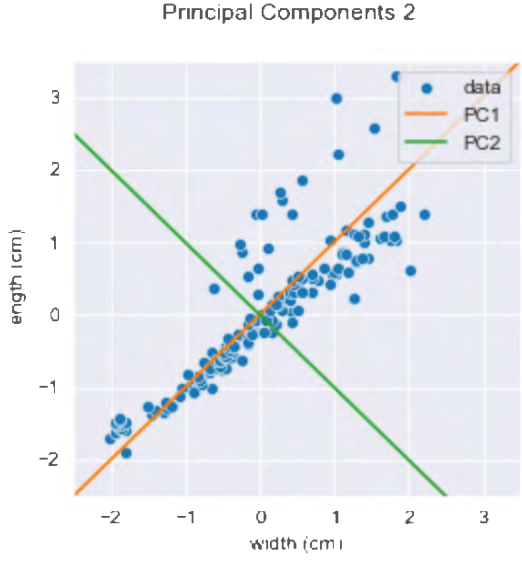
Das Modell ist eine Lineare Abbildung

PCA - MEHRERE FEATURES

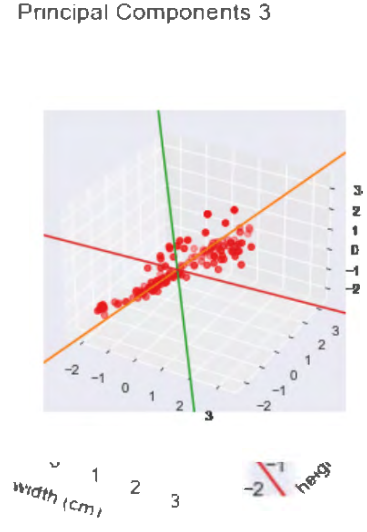
1 FEATURE



2 FEATURE



3 FEATURE



PCA - KOMMT "GRATIS" MIT EINEM DECODER

INPUT SPACE
3072 FEATURES

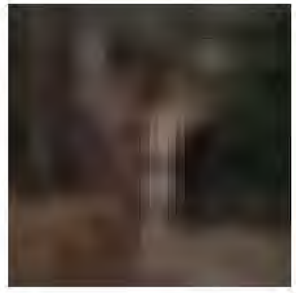
$$L = xW$$
PCA.transform

LATENT SPACE
200 FEATURES

[-57.2,
-3.6, ...,
2.3, -3.4]

$$\hat{x} = LW^T$$
PCA.inverse_transform

RECONSTRUCTED
INPUT SPACE
3072 FEATURES



INPUT SPACE
p FEATURES

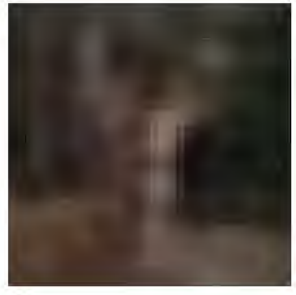
$$L_{1\times l} = x_{1\times p} W_{p\times l}$$
PCA.transform

LATENT SPACE
l FEATURES

[-57.2,
-3.6, ...,
2.3, -3.4]

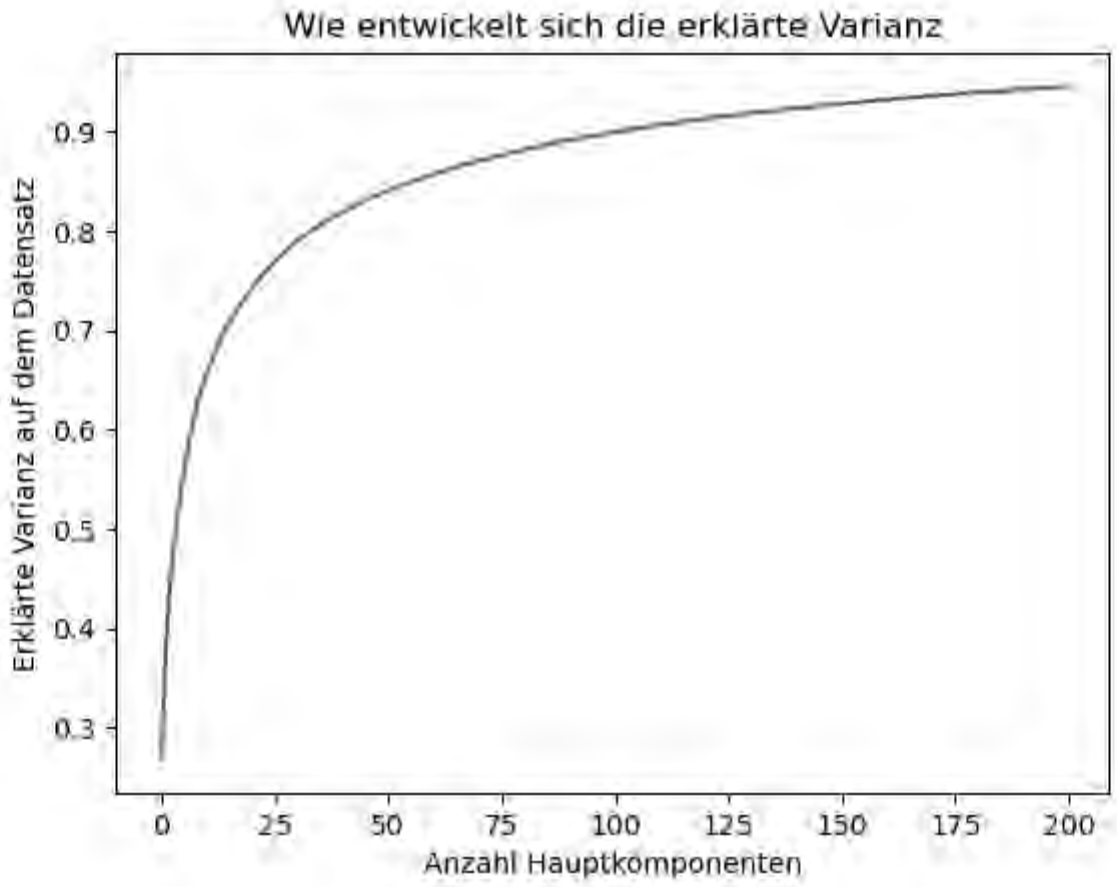
$$\hat{x}_{1\times p} = L_{1\times l} W^T_{p\times l}$$
PCA.inverse_transform

RECONSTRUCTED
INPUT SPACE
p FEATURES



$$\hat{x} = xWW^T$$

WIE WÄHLEN WIR DIE ANZAHL DIMENSIONEN



WIE WÄHLEN WIR DIE ANZAHL DIMENSIONEN

EXAMPLE IMAGE

RECONSTRUCTION MIT 1 FEATURE, 2 FEATURE, ..., 200 FEATURE

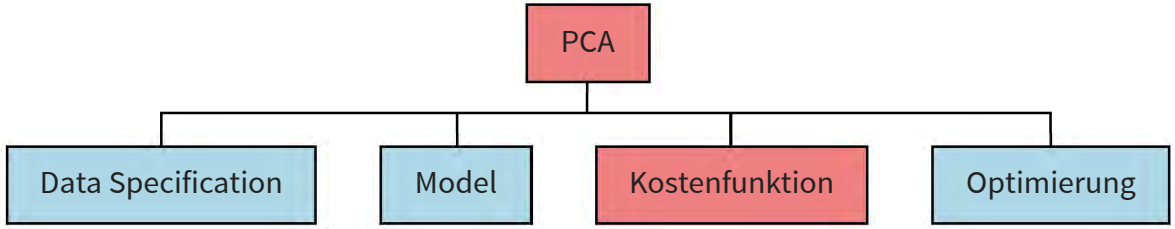
PC SELECTION VS. FEATURE SELECTION - INTUITION

ORIGINAL IMAGE (100%)

FEATURE SELECTION: PIXELS OF 200 FEATURES (9.3%)

PC-SELECTION: RECONSTRUCTION 200 FEATURES (PCA) (94.4%)

PCA


PCA
Data Specification
Model
Kostenfunktion
Optimierung

PCA - KOSTENFUNKTION

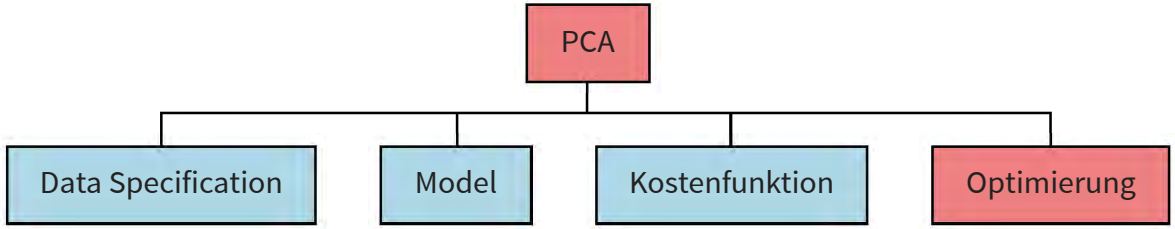
$$J(W) = \| X - \hat{X} \|_2$$

$$\| X - XWW^T \|_2$$

L2-Distanz der Rekonstruktion.

Encoder und Decoder sind einfach eine Matrix, beziehungsweise sind linear.

PCA

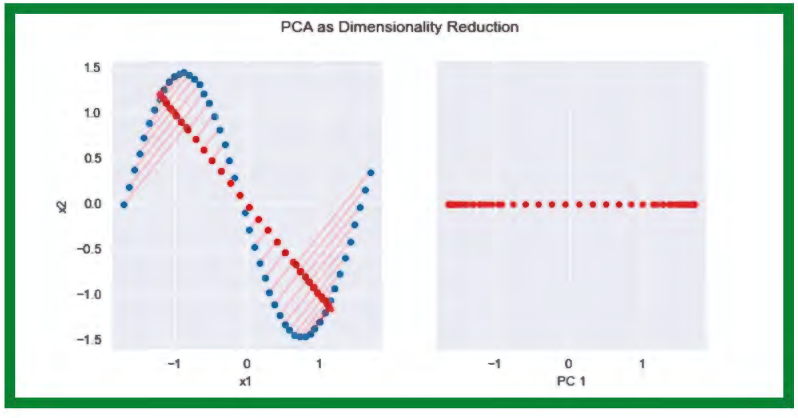
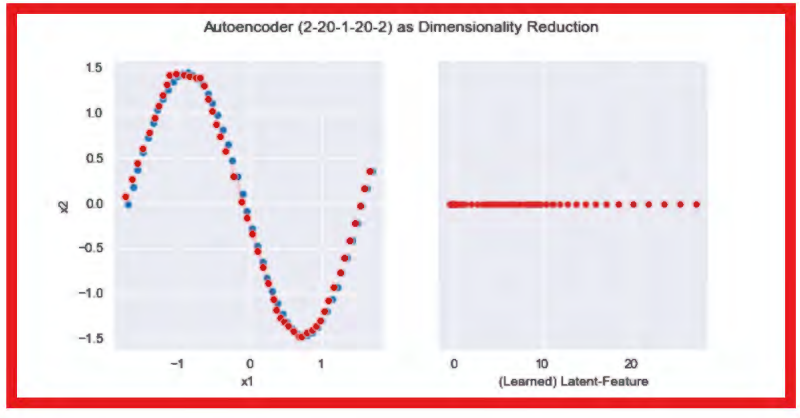
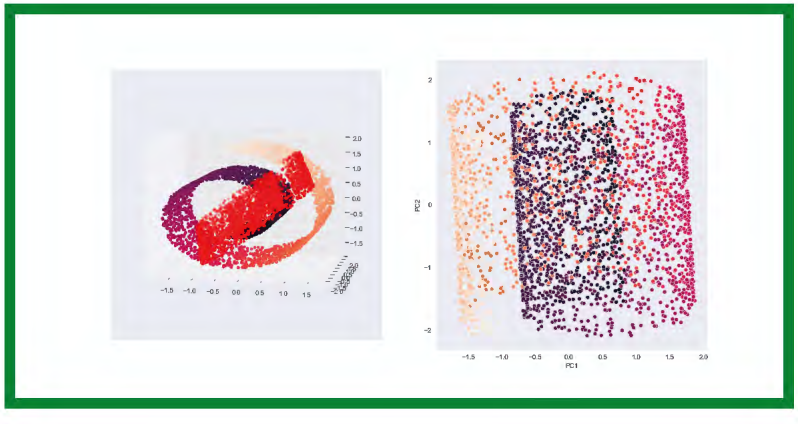
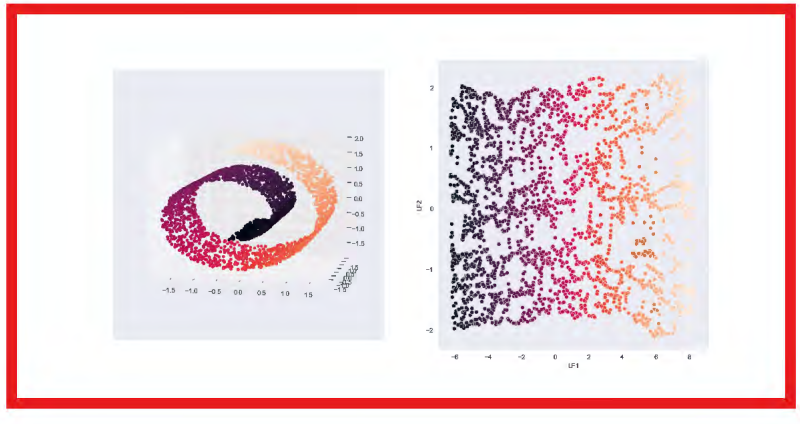

PCA
Data Specification
Model
Kostenfunktion
Optimierung

PCA - OPTIMIERUNG

- "Einfach" Lineare Algebra
- Schauen wir nicht im Detail an.
- Nutzt Singular Value Decomposition (SVD)

- `U, S, V = numpy.linalg.svd(X.T)`
- `U` sind unsere Principal Components (Achsen)
- `S` ist die Wichtigkeit der Achsen.
- `W = U[:, 0:l]`

PCA - LIMITS

	POSSIBLE	IMPOSSIBLE (IN FEATURE SPACE)
2 FEATURES ZU 1		
3 FEATURES ZU 2		

PCA hat die starke Annahme, dass das gesuchtes Manifold im Feature-Space linear ist!

PCA ALS PREPROCESSING

INPUT SPACE
3072 FEATURES

Encoder
PCA.transform

LATENT SPACE
200 FEATURES

```
[ -57.2,
-3.6, ...,
2.3, -3.4 ]
```

Modell
z.B. SVC

OUTPUT SPACE

```
Deer
```

WANN?

PCA sollte für Performanz Optimierung eingesetz werden (Speed-up (CPU), Less memory (RAM/disk))

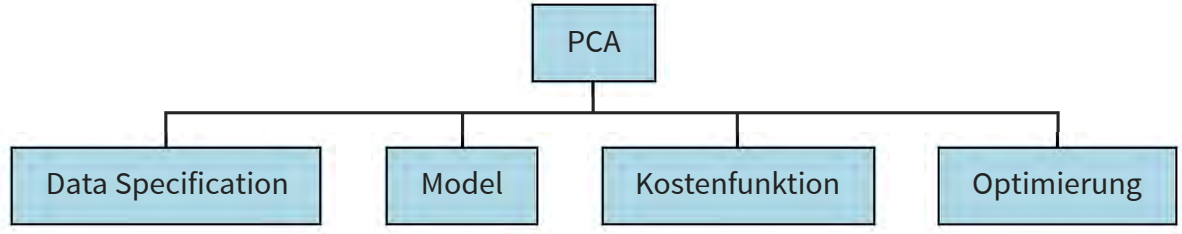
WANN NICHT?

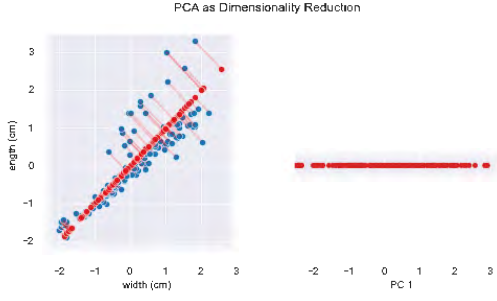
PCA (sollte man) nicht verwenden für Overfitting. Bei Overfitting ist Regularisierung meistens die bessere Wahl.

ALTERNATIVE & ADDITIONAL RESOURCES

- [Alternative] Dimensionality Reduction and PCA
  - Video: "Lecture 14.1 bis Lecture 14.7"
- [Additional] SVD & PCA
  - Einführung in SVD Video: "What is the Singular Value Decomposition?"
  - SVD und PCA in mehr Detail:
    - Lecture: The Singular Value Decomposition (SVD)
    - Lecture: Principal Componenet Analysis (PCA)
- [Additional] Weitere Dimensionality Reductions
  - Factor analysis
  - Independent component analysis (ICA)
  - Isomap
  - ...

PCA

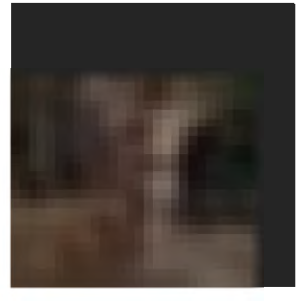




$$L = xW$$

$$J(W) = \|X - \hat{X}\|_2$$

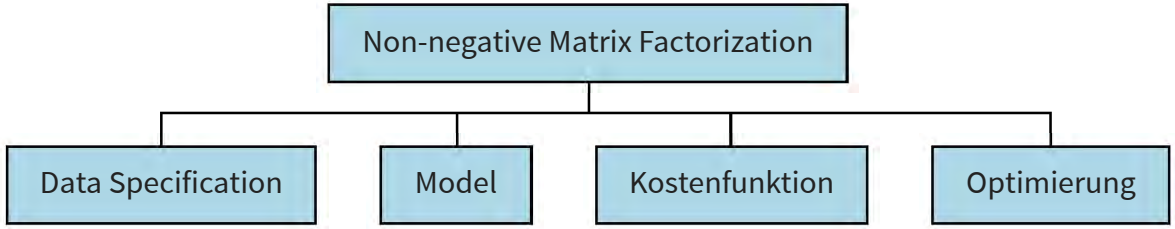
$$\|X - XWW^T\|_2$$



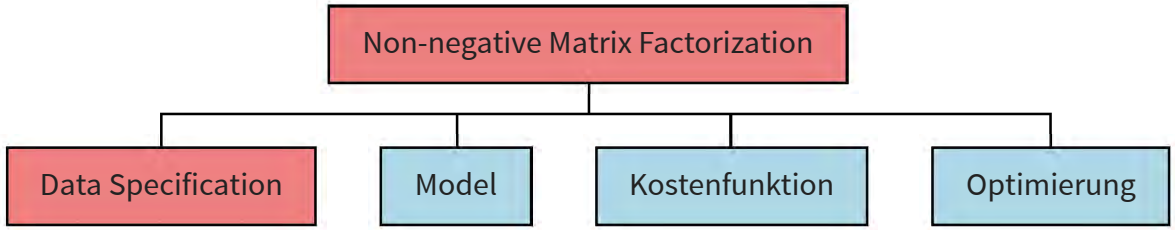
PCA - CODE

pca.ipynb

NON-NEGATIVE MATRIX FACTORIZATION



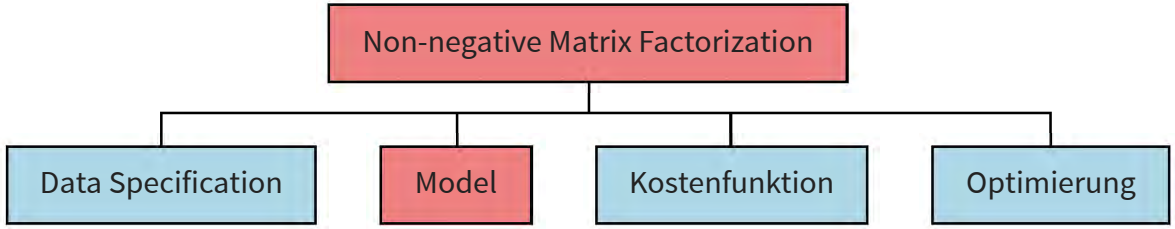
NON-NEGATIVE MATRIX FACTORIZATION



NMF - DATA SPECIFICATION

1. Welche Features haben wir, z.B. `pixel`
2. Features müssen positive Werte (grösser gleich 0) haben
3. Features müssen einheitliche Grösse haben

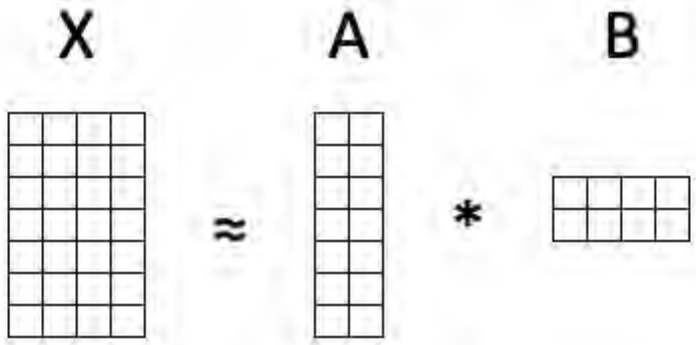
NON-NEGATIVE MATRIX FACTORIZATION



NMF - MODEL

Ziel $X \approx AB$

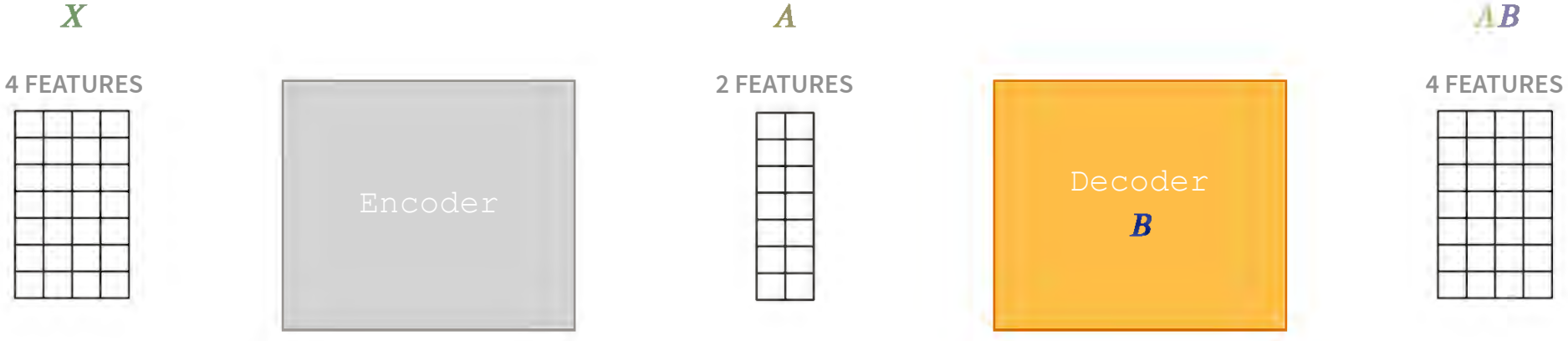
Bedingung $A \geq 0, B \geq 0$



$$X_{n*m} \approx A_{n*k} B_{k*m}$$

$$k \leq min(n, m)$$

NMF - INTUITION

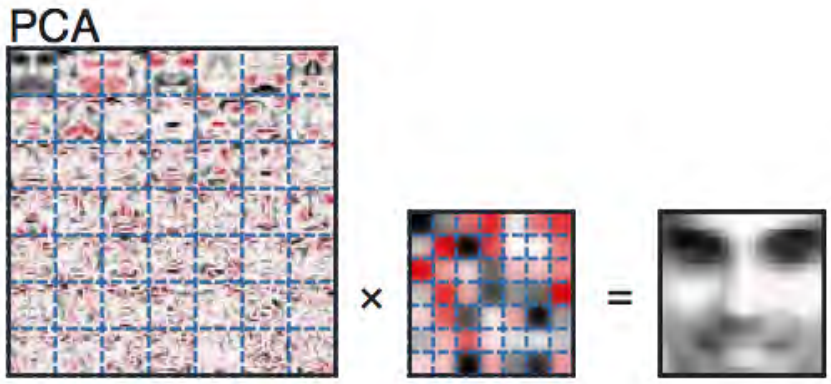


Kein Encoder => Nicht für Preprocessing für Supervised Learning Task geeignet.

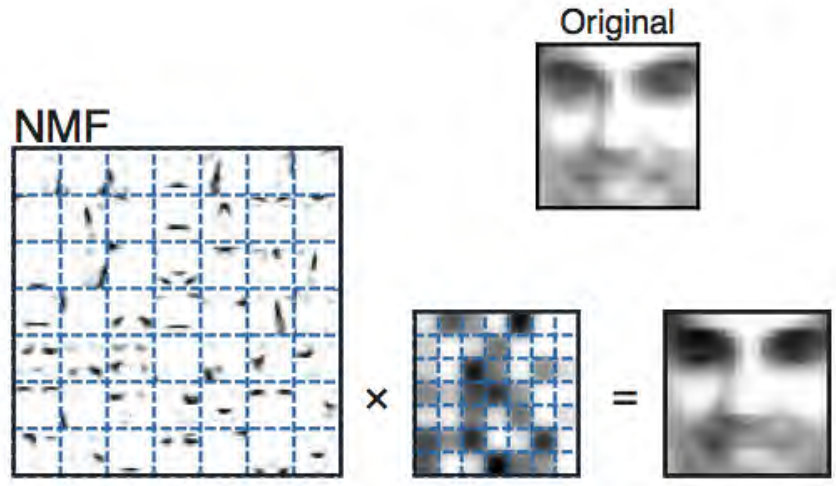
NMF - MOTIVATION

Motivation: Addition von non-negative Konzepten (hidden Features) macht Konzepte verständlicher.

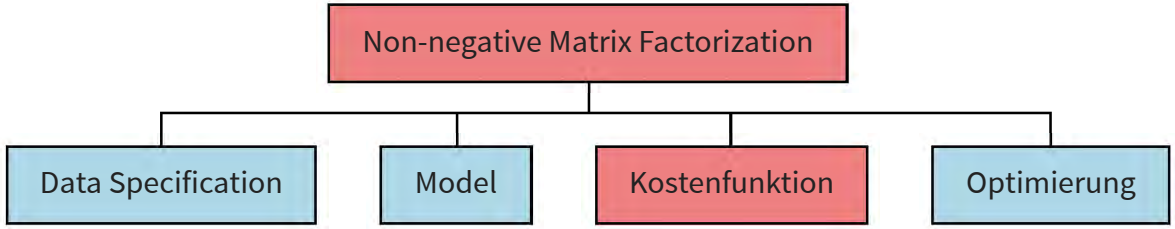
PCA



NMF



NON-NEGATIVE MATRIX FACTORIZATION

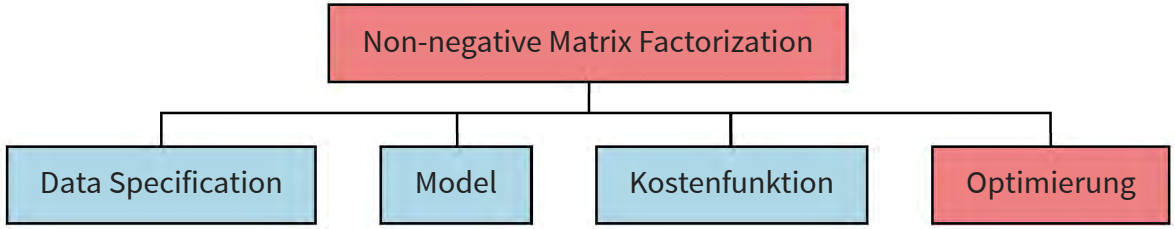


NMF - KOSTENFUNKTION

$$J(A, B) = \|X - \hat{X}\|_2$$
$$\|X - AB\|_2$$

Gleiche Kostenfunktion wie bei PCA (nur unter anderem Model).

NON-NEGATIVE MATRIX FACTORIZATION



NMF - OPTIMIERUNG

Der Algorithmus funktioniert wie folgt:

1. Initialisiere A, B mit zufälligen Werten.
2. Optimiere A nach Kostenfunktion, B bleibt fix
   Optimiere Repräsentation gegeben Decoder B
3. Optimiere B nach Kostenfunktion, A bleibt fix
   Optimiere Decoder gegeben Repräsentation A
4. Wiederhole 2. und 3., bis nur noch wenig Veränderung statt findet.

Findet lokales Minimum! Ganzen Algorithmus mehrmals wiederholen mit anderen Initialwerten.

QUESTIONS

QUESTIONS

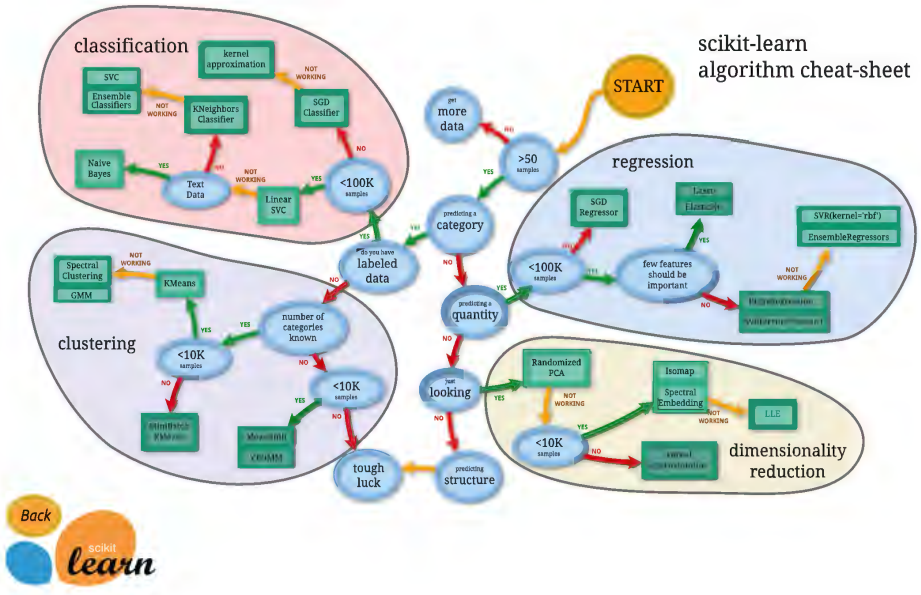
1. Was ist Dimensions Reduktion?
2. Was ist PCA?
3. Was ist der Unterschied von NMF und PCA?

DATA SCIENCE PITFALL - MODEL-WAHL

"Wie ein Problem angehen" Punkte sind erledigt vor Model-Wahl
Für die Model-Wahl spielt folgendes eine Rolle:

- Problem-Komplexität (z.B. Katze-Hund Bildererkennung vs. Tesla Auto Pilot)
- Problem-Domäne (z.B. strukturierte Daten vs. unstrukturierte Daten)
- Anzahl Datenpunkte (10 Daten, 100 Daten, 1000 Daten, ... , 1000000 Daten, ...)
- Anzahl Features (5 Features, 10 Features, 100 Features, ... , 10000 Features, ...)
- Modell Interpretierbarkeit
- (Verfügbare Rechenzeit)
- ...

SKLEAN USER GUIDE



Source: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

MEINE PERSÖNLICHES VORGEHEN

- Konsultiere Literatur und Best Practices in Problem-Domäne
- Unstrukturierte Daten, dann Deep Learning
  - Bilder, dann Convolutional Neural Network
  - Text, dann RNN oder Transformer
  - Zu wenig Daten? Dann siehe Genug Daten
- Strukturierte Daten, dann "klassisches" Machine Learning
  - Wenig Daten (< 500), dann Lineare Modelle mit vielen Annahmen (Feature Selection, kreatives Feature Engineering, Monotonic Constraints)
  - Mehr Daten (> 500), dann Nicht-Lineare Modelle mit weniger Annahmen (kreatives Feature Engineering)
  - Extra Fokus auf Datenqualität, siehe Garbage-in-Garbage-Out

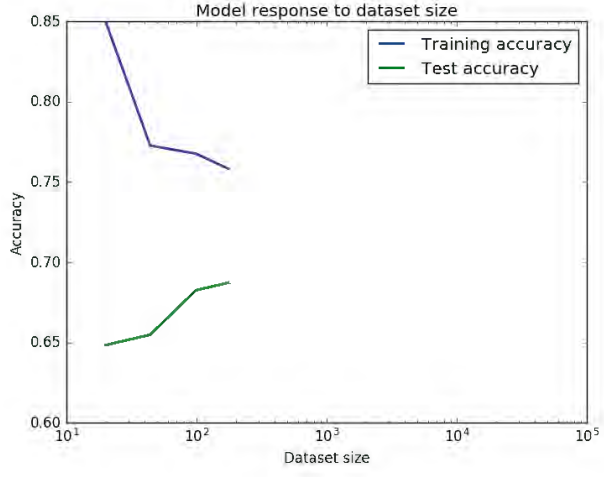
DATA SCIENCE PITFALL - GENUG DATEN

- Zu Beginn: Haben wir genug Daten, um das gewünschte Problem zu lösen?
- Möglichkeiten dies fest zu stellen:
  - Erfahrung, Austausch mit anderen Data-Science-Experten
  - Wie viel Daten brauchten ähnliche Probleme
  - Für spezifisches Modell: Learning Curve visualisieren (nächste Slide)
- Was kann man gegen zu wenig Daten tun?
  - Mehr Daten sammeln (z.B. Tesla Autopilot)
  - Problem vereinfachen (z.B. nur bestimmte Städte befahren)
  - Projekt abbrechen oder verschieben
  - Data Augmentation: Vorhandene Daten künstlich variieren
  - Transfer Learning: Daten eines ähnlichen Problems ausnutzen
  - Modell vereinfachen, z.B. mehr sinnvolle Annahmen (mit Domänen Experten) treffen, wie kreatives Feature Engineering oder Feature Encoding (z.B. durchschnittlicher Quadratmeterpreis im Umkreis)

[Alternative Quelle]: Section 2.3 in Paper "How to avoid machine learning pitfalls: a guide for academic researchers"

GENUG DATEN - LEARNING CURVE

Die Learning Curve ist für ein spezifischen Modell (z.B. eine bestimmte Neural Network Architektur)
Die Learning Curve zeigt uns, wie gut das Modell mit unterschiedlich vielen Trainings-Daten ist
Die Learning Curve zeigt uns, wann und ob das Modell "gesättigt" ist



Das spezifische Modell würde (wahrscheinlich) von mehr Daten profitieren.
Kann Projekt-Entscheid steuern, z.B. Modelentwicklung pausieren und lieber mehr Daten sammeln

Aber: Eine andere Model-Wahl könnte bereits jetzt besser funktionieren...

DATA SCIENCE PITFALL - RESULTATE INTERPRETIEREN

- Performanz richtig einordnen, Resultate nicht Generalisieren auf andere Probleme
  - Gegeben der Metrik
  - Gegeben dem Problem
  - Gegeben den Daten, z.B. dem Selection Bias (Datenquelle(n), Datenfilter)
- Nicht überbewerten, Grenzen bewusst sein, Klar kommunizieren
- Cross-Validation nutzen für den Vergleich zwischen Modellen
- Kleine Verbesserung für grosse Komplexität im Modell ist meistens unerwünscht
  - Einfachere Modelle einfache maintenance
  - Occam's Razor

RESULTATE INTERPRETIEREN - BEISPIELE

- ML Lab Tag 1: 20% Genauigkeit, was bedeutet das?
  - Vorhersage ist in den Daten im Durchschnitt +-20% genau
  - Angebotspreis nicht Verkaufspreis wird vorhergesagt
  - Nur für Objekte aus dem Inserats-Jahr 2018
  - Nur für bestimmte Nutzungen, maximale Grösse
  - Nur für Objekte von gecrawlter Internet-Platform

- ML Lab Tag 2: 50% Genauigkeit, was bedeutet das?
  - 50% der Bilder wurden in den Daten richtig erkannt
  - Nur 10 Kategorien können erkennt werden
  - Nur 32x32x3 Pixel Bilder
  - Nur Bilder, wo Objekt zentral abgebildet

Resultate nicht über die Daten hinaus generalisieren

ÜBUNGSZEIT (WEITERE 60 MINUTEN)

`exercise/dimensionality_reduction.ipynb`