Dr. rer. nat. Valentin Khaydarov
Professur für Prozessleittechnik & Arbeitsgruppe Systemverfahrenstechnik

Machine Learning II

Vorlesung 3, Lehrveranstaltung Simulation und Optimierung, WS 2021/2022

Gliederung – Vorlesung II

- Wiederholung des Materials der letzten Vorlesung
- Einzelne Schritte des Vorgehensmodells
- MLOps

Wiederholung der letzten Vorlesung

Zusammenfassung

A computer program is said to learn from **experience E** with respect to **some task T** and **some performance measure P**, if its performance on T, as measured by P, improves with experience E.

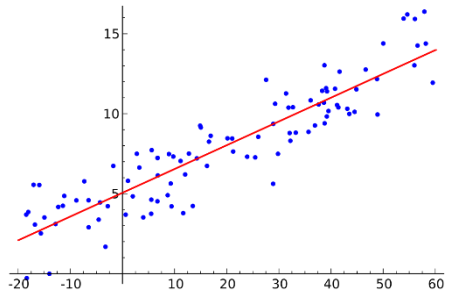
Je nach Typ der Zielvariable: **Regressionsaufgabe** (Kardinalskala), **Klassifikation** oder **Clustering** (Ordinal- oder Nominalskala)

Aufgaben:

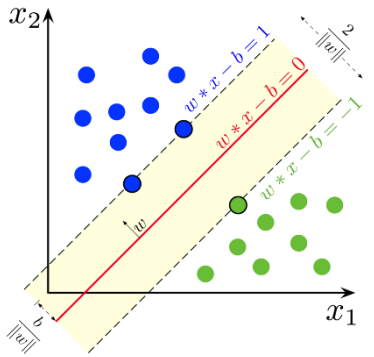
	Erfahrung	Aufgabe
Überwachtes Lernen	Input-Variablen und Ziel-Variablen für alle Datenpunkte	Prädiktion Ziel-Variablen
Unüberwachtes Lernen	Nur Input-Variablen	Erkennung Muster in Input-Variablen
Semiüberwachtes Lernen	Input-Variablen und Ziel-Variablen für begrenzte Anzahl Datenpunkte	Prädiktion Ziel-Variablen für Datenpunkte ohne Labels im Datensatz
Bestärkendes Lernen	Agent und Umgebung	Entwicklung optimaler Strategie für Agent zum Umgang mit Umgebung

Zusammenfassung

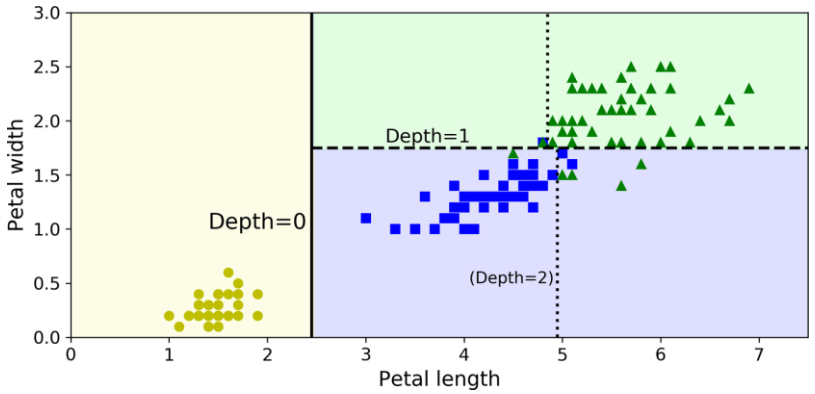
- Regression



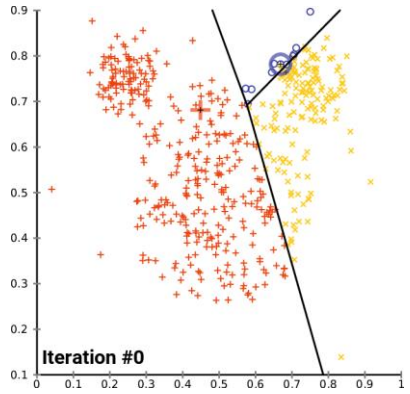
- Support-Vector-Maschine(und Regression)



- Entscheidungsbäume

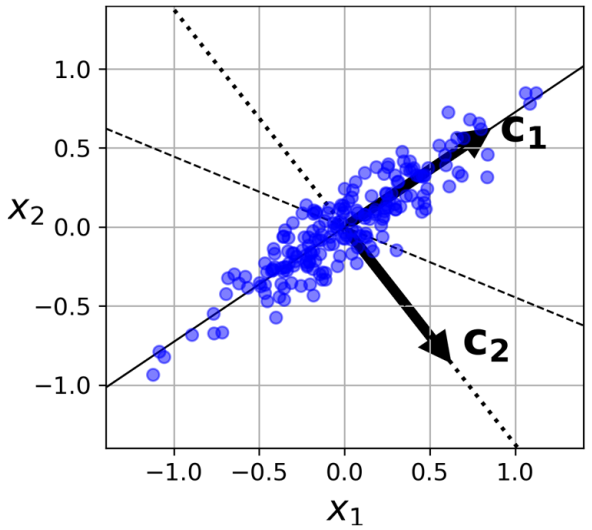


- K-Means-Clustering

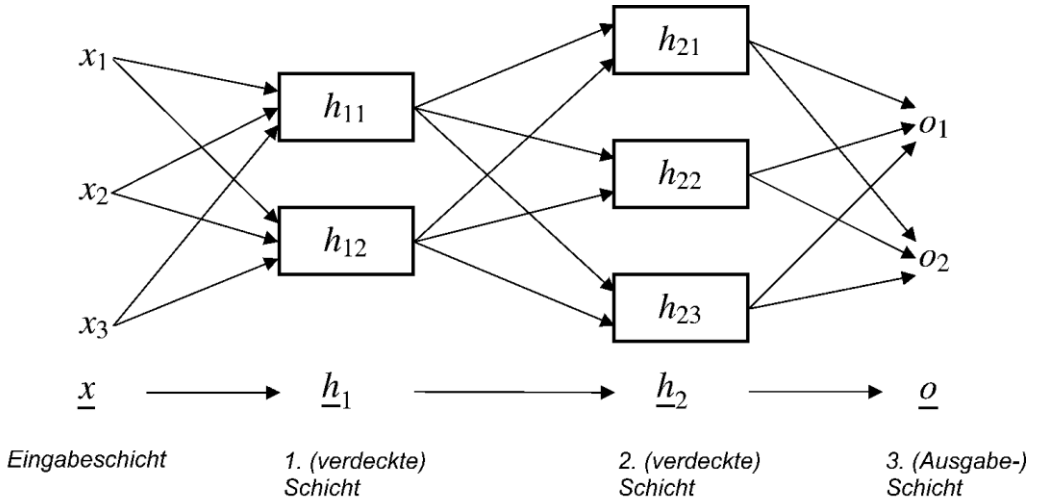


Zusammenfassung

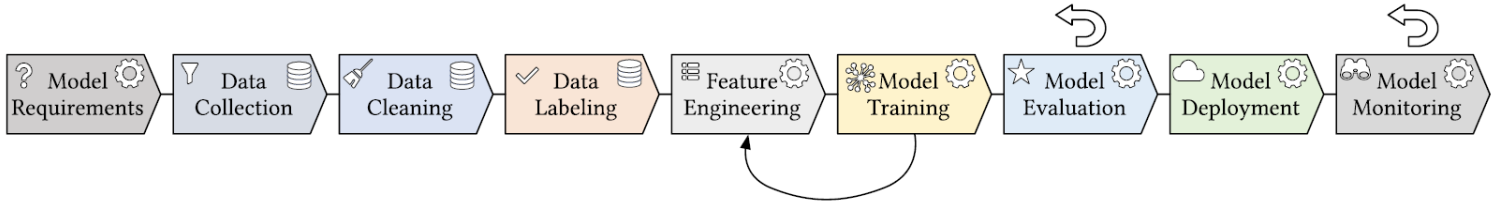
- Hauptkomponentenanalyse (PCA)



- Neuronale Netze



Vorgehensmodell nach Amershi



- Data collection – Bereitstellung des Datensatzes: Datenimport oder –gewinnung
- Data cleaning – Aufbereitung des Datensatzes
- Data labelling – Markierung von Daten (Überwachstes Lernen)
- Feature engineering – Auswahl von Features und deren Aufbereitung für das Training
- Model training – Trainieren, Optimierung von Modell- und Training-Hyperparametern
- Model evaluation – Testen des Modells mit einem Testdatensatz, Berechnung von Metriken, Auswahl eines Modells für den Einsatz in der Produktion
- Model deployment – Aufbau Runtime-Umgebung für Modell-Inference, Einsetzen des Modells
- Model monitoring – Evaluation des Modells im Betrieb, Sammlung Daten für Verbesserung
- Model maintenance – Aktualisierung des Modells (z.B. nach Erweiterung/Anpassung des Training-Datensatzes)

M. Haakman, L. Cruz, H. Huijgens, and A. van Deursen, "AI lifecycle models need to be revised: An exploratory study in Fintech," *Empir. Softw. Eng.*, vol. 26, no. 5, pp. 1–30, 2021, doi: 10.1007/s10664-021-09993-1.

Schritt 1:
Model requirements (inkl. Business understanding)

Formulierung der Problemstellung

Nach diesem Schritt sollen Antworten auf folgende Fragen gegeben werden:

- **Zielstellung** der zu entwickelnden Anwendung
- **Verfügbarkeit von Daten? Was genau enthalten die Daten? Wie wurden sie aufgenommen?**
  - **Daten verstehen**
- **Anforderungen** und relevante **Metriken**: Performance, Robustheit, Skalierbarkeit, Erweiterbarkeit, Echtzeitfähigkeit, Sicherheit, Ressourcenaufwand usw.
- **Minimale Anforderungen an die ML-Anwendung**
- Falls sie existieren: Wie funktionieren aktuell bestehende Lösungen? (**Baseline**)
- **Annahmen** und deren Validierung (falls möglich)
- **Wie** wird die Anwendung eingesetzt (online, offline)?
- Ist eine Modelldegradation zu erwarten und deshalb eine **Modellwartung** erforderlich?

Formulierung der Problemstellung

Das Modell ist in der Regel nur ein Teil der Anwendung und dessen Performance ist nicht immer entscheidend

Für die Modullauswahl relevante Anforderungen:

- Gehören Eingangsvariablen Kardinal-, Ordinal- oder Nominalskala? Sind die Daten eine Sequenz (Zeitreihe) oder unabhängige Messungen?
- Zielvariablen: Vorhanden? Kardinal-, Ordinal- oder Nominalskala? Vollständig und zuverlässig annotiert?
- Umfang vorhandener Daten
- Ist Expertenwissen vorhanden?
- Modellgüte bzw. Zielfunktion sowie weitere relevanten Metriken

Output: **vorläufiger ML-Versuchsplan**

Typische Problemstellungen in der Prozessindustrie

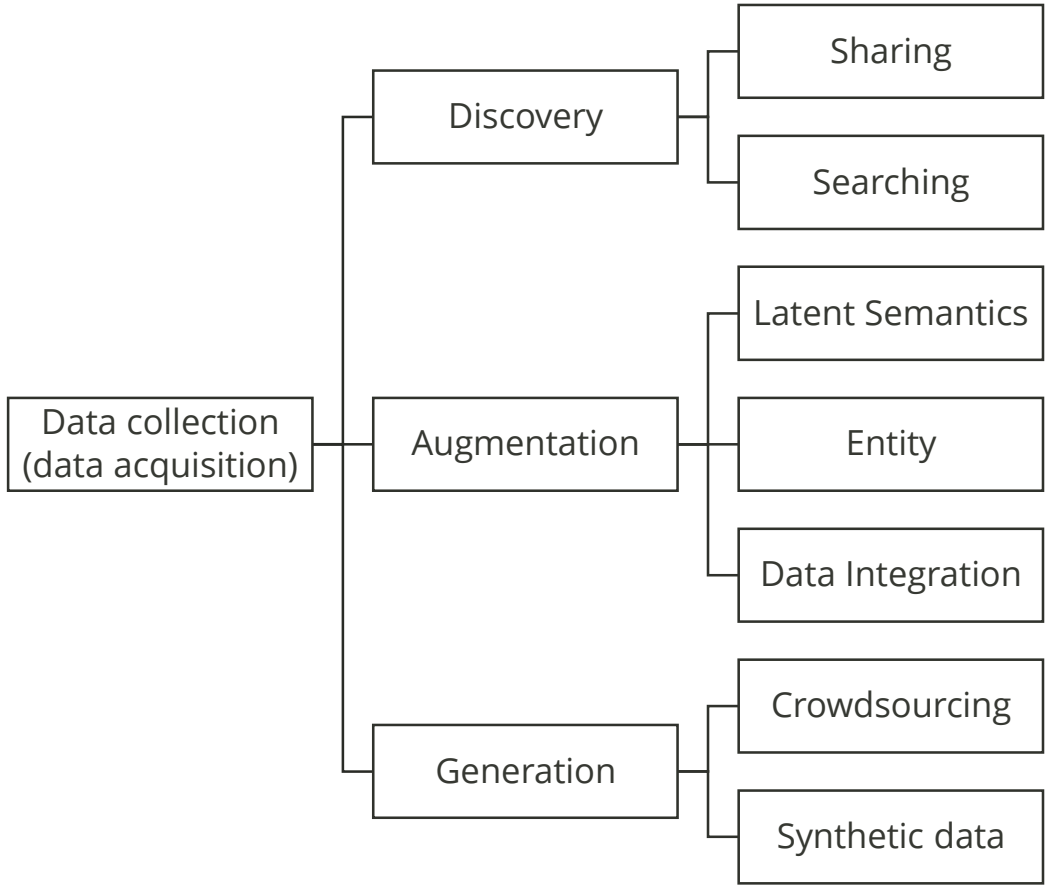
- Bewertung einer Prozessvariablen, die sich nicht direkt messen lässt → Softsensor
  - Regression oder Klassifikation
- Prädiktion des weiteren Zeitverlaufs → Prädiktive Wartung
  - Regression
- Erkennung eines Prozesszustandes oder einer bestimmten Störung
  - Klassifikation
- Anomalie-Erkennung
  - Clustering, Klassifikation
- Prozessregelung
  - Reinforcement-Learning

Schritt 2: Data collection

Data collection

Datengewinnung umfasst eine Vielfalt an Methoden zur **Beschaffung** von Daten, die weiterhin bei dem **Modelltraining** verwendet werden.

- Bis zu 90% aller Zeitaufwände bei der Entwicklung von ML-Anwendungen ist die Arbeit mit Daten
- Modellqualität hängt unmittelbar von der Datenqualität ab
- Neue Modellierungsansätze zur Abbildung komplexen Verhaltens (z.B. DeepNN) erfordern proportional mehr Daten für das Training



Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2019, doi: 10.1109/TKDE.2019.2946162.

Industrielle Daten

Datentypen:

- **Prozessdaten von Sensoren sowie Signalgebern**, die primär für Prozessüberwachung, - optimierung und Regelung eingesetzt werden
  - Temperatur, Druck, Füllstand, Stoffeigenschaften, Sollwerte usw. in der Regel Daten der Kardinal- oder Ordinalskala (analoge und binäre Signale) als Zeitreihen
  - Verfügbar über das Prozessleitsystem und archiviert in einem Prozesshistorian oder offline als Protokoll
- **Diskrete Ereignisse** wie Prozessphase oder Rezeptschritt, Operatoreingriffe, Alarme der Nominalskala
  - In der Regel unstrukturiert und teilweise manuell aufgenommen
- **Engineeringdaten zur Anlage**
  - Meistens statische und **unstrukturierte** Daten
  - In der Regel in der Form von Dokumentation zur Produktionsanlage wie R&I-Fließbild, Konstruktionsdokumente, aktuelle Anlagekonfiguration, Regelungskreise, Rezepte usw.
  - Liefert sehr nützliche Kontextinformation für besseres Verständnis von Daten
- **Rohdaten von Analysegeräte (z.B. Spektroskopie, Chromatographie, Laserbeugung), Bild- und Videodaten**
  - Integriert mit relativ komplexen Preprocessing- und Datenauswertung-Algorithmen
  - Gekapselt in von dem SPS entkoppelten Systemen (z.B. Edge-Node-Konzept)

Besonderheiten und Herausforderungen industrieller Daten

Datenvarianz je nach Betriebsweise der Anlage:

- Batch-Betrieb: Varianz zwischen Chargen durch Änderungen im Startbedingungen und Wiederholungen
- Konti-Betrieb: Varianz ist in der Regel nur bedingt aussagekräftig → Data-rich-but-information-poor (DRIP)-Daten
- Daten sind in der Regel nicht nach Klassen bilanziert

Fluch der Dimensionalität:

- Enorme Anzahl diverser und möglicher Datenquellen: Dutzende oder Hunderte Sensoren und Signalgeber

Zeit:

- Je nach der Prozessskala können Messwerte fein ausgelöst sein (Millisekunden-Bereich)
- Datenaufnahme verschiedener Sensoren in der Regel nicht synchronisiert (s. OPC UA)
- Time-Sampling kann im Laufe der Zeit variieren (ms-Bereich im Betrieb und Min-Bereich im Prozesshistorian)

Besonderheiten und Herausforderungen industrieller Daten

Big-Data:

- Prozesse können bis zu vielen Tagen dauern:
  - ein Parameter, eine Messung pro Sekunde, eine Woche lang: 60x60x24x7 = 604800 Messpunkte

Einbindung von Kontextinformation:

- Normen schreiben Anlagenbauern und –betreibern die Durchführung einer sehr umfangreichen Anlagendokumentation vor
- Verwendung dieser Information zur ausführlichen Meta-Beschreibung von Datensätzen

Weitere spezifische datenbezogene Herausforderungen werden weiter in den Abschnitten „Data cleaning“ und „Data labeling“ betrachtet

Data discovery

Methoden für das **Teilen** und die **Wiederverwendung** von bestehenden Datensätzen.

Allererster Schritt - Suche nach relevanten Datensätzen (lokal und global).

Voraussetzung ist die Erfüllung der FAIR-Prinizipen:

- **Findable**: Eindeutiger und dauerhafter Identifikator vorhanden und Metadaten erlauben die Findung
- **Accessible**: Standardisierte Protokolle für Datenzugriff
- **Interoperable**: Daten sind in einer (Computer-) lesbaren und interpretierbaren Form
- **Reusable**: Daten sind zusammen mit ausführlicher Beschreibung und Lizenzbedingungen zur Verfügung gestellt

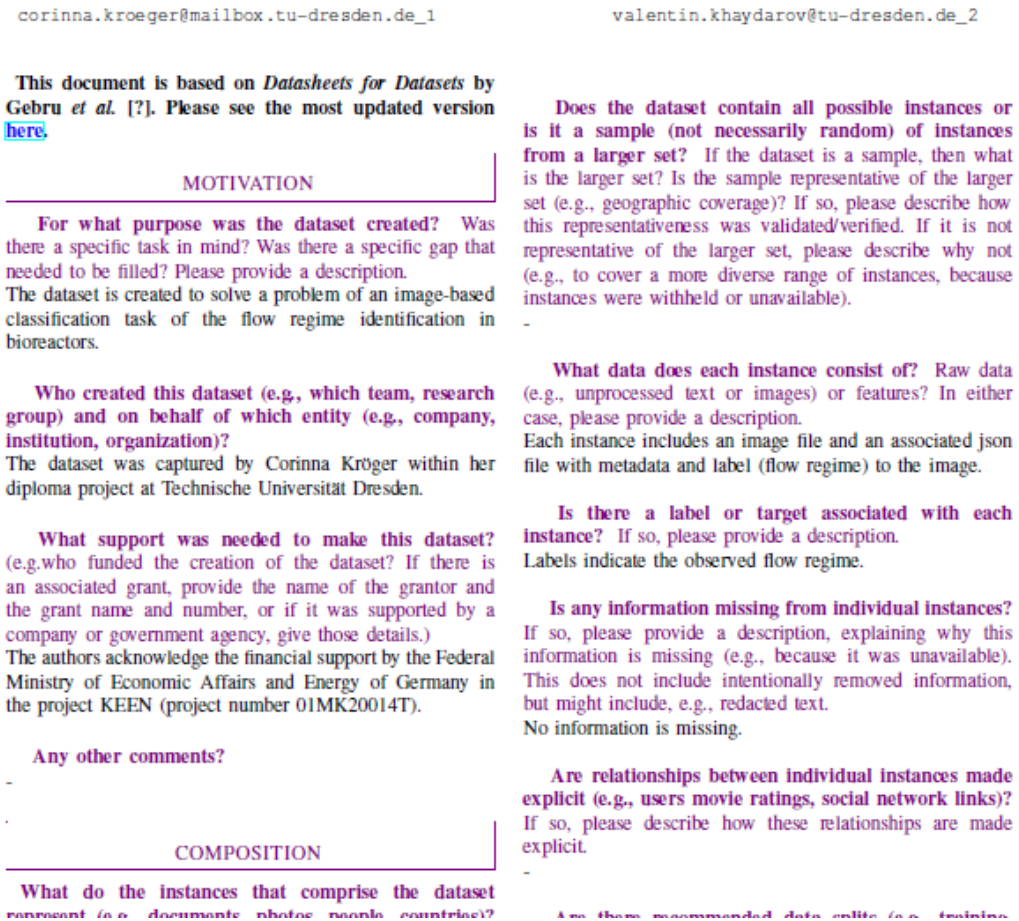
Mehr auf https://www.go-fair.org/fair-principles/

Data discovery: Datasheet for Dataset

Abschnitte:

- Motivation
- Composition
- Collection
- Preprocessing/Cleaning/Labeling
- Uses
- Distribution
- Maintenance

Quelle: Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Iii, H. D., & Crawford, K. (2018). Datasheets for datasets. ArXiv.



corinna.kroeger@mailbox.tu-dresden.de_1 valentin.khaydarov@tu-dresden.de_2

This document is based on *Datasheets for Datasets* by Gebru *et al.* [?]. Please see the most updated version here.

MOTIVATION

For what purpose was the dataset created? Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.
The dataset is created to solve a problem of an image-based classification task of the flow regime identification in bioreactors.

Who created this dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?
The dataset was captured by Corinna Kröger within her diploma project at Technische Universität Dresden.

What support was needed to make this dataset? (e.g.who funded the creation of the dataset? If there is an associated grant, provide the name of the grantor and the grant name and number, or if it was supported by a company or government agency, give those details.)
The authors acknowledge the financial support by the Federal Ministry of Economic Affairs and Energy of Germany in the project KEEN (project number 01MK20014T).

Any other comments?
-

COMPOSITION

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)?

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).
-

What data does each instance consist of? Raw data (e.g., unprocessed text or images) or features? In either case, please provide a description.
Each instance includes an image file and an associated json file with metadata and label (flow regime) to the image.

Is there a label or target associated with each instance? If so, please provide a description.
Labels indicate the observed flow regime.

Is any information missing from individual instances? If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.
No information is missing.

Are relationships between individual instances made explicit (e.g., users movie ratings, social network links)? If so, please describe how these relationships are made explicit.
-

Are there recommended data splits (e.g., training

Data discovery: Datenrepositories

Datenrepositories (Daten intern gespeichert):

Kaggle.com, Amazons opendata.aws, UCI ML Repository, Microsofts msropendata.com, EU Open Data Portal, OpenML

Die meistverwendeten sind direkt in Tools integriert: Matlab (s. Sample Data Sets), sklearn (s. Toy datasets) und tensorflow (s. Tensorflow-Datasets)

Suchportale (Nur Verweise auf externe Quellen):

Google Datasets Search Engine

Listen von Datenrepositories:

https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research

https://github.com/awesomedata/awesome-public-datasets

In der Regel nur Daten für allgemeine Zwecke (viel Bilddaten). **Prozessdaten sind kaum öffentlich verfügbar.**

Prominente industrielle Datensätze

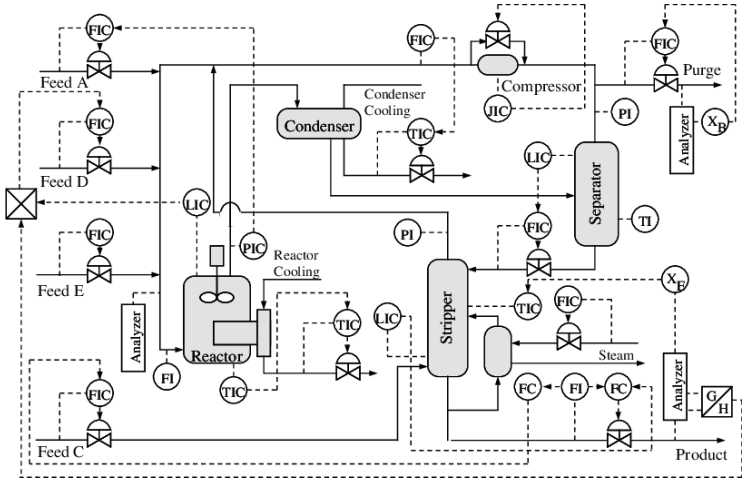
Tennessee Eastman Process (TEP)

Down und Vogel, 1993

Open-Loop-Prozess mit 12 Regelventilen und 41 Messstellen

Als Challenge für Störungserkennung und dynamische Optimierung

Simulationsmodell und reale Datensätze auffindbar



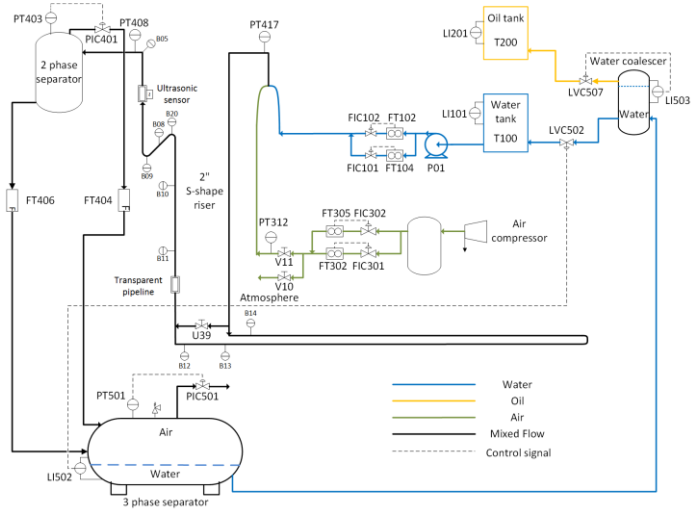
Kano u. a., „Contribution Plots for Fault Identification Based on the Dissimilarity of Process Data“.

Multiphase flow facility (PRONTO)

Stief, Tan, Cao und Ottewill, 2019

Heterogene Datenquellen (Prozesswerte, Alarme, Logs, Hochfrequente Ultraschall- und Druckmessungen und Videodaten)

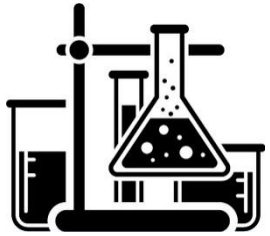
Multimodale und Multirate Störungserkennung und Anlagediagnose



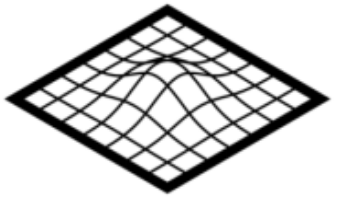
https://doi.org/10.5281/zenodo.1341583

Data generation: Datenquellen

Produktion
- Vollständig repräsentativ
- Große Menge
- Kleine Varianz
- Datenkosten niedrig
- Nur intern verfügbar



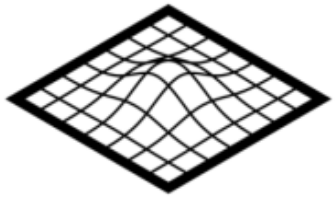
Test-Umgebung
- Wenig repräsentativ
- Kleinere Datenmenge
- Höhere Varianz
- Datenkosten höher
- Dürfen veröffentlicht werden



Simulationsmodelle
- Bilden Modell ab
- So viel wie man will
- Beliebige Varianz
- Niedrige Kosten
- Grundsätzlich frei teilbar

Transfer-Learning ist der Einsatz Erfahrung auf ein ähnliches aber anderes Problem zu übertragen. Sehr beliebt und effektiv in Computer-Vision-Anwendungen.

Data generation und Transfer learning



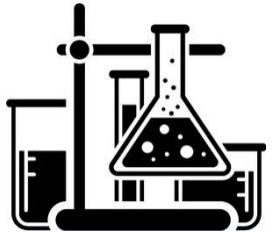
Synthetische Daten
(Simulation)

Realle Daten

Transfer-
Learning

Pre-trainiertes Modell (Source)

Modell (Target)



Daten
(Test-Umgebung)

Realle Daten

Transfer-
Learning

Pre-trainiertes Modell (Source)

Modell (Target)

Data exploration

Datenexploration ist ein Verfahren zur vorläufigen Analyse von Daten.

Datenexploration ist der allererste Schritt nach der Erwerbung von Daten.

Teilziele:

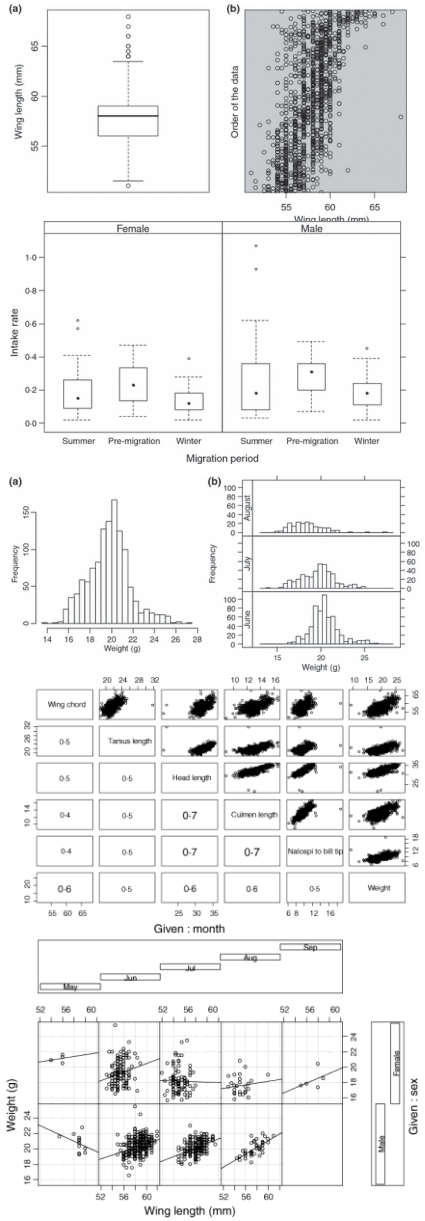
- **Formalisierung** Information über Variablen: Beschriftung, Typ, Einheit, Grenzen/Mögliche Werte
- Verschaffung eines **Überblicks** über und **Verständnis** von Daten
- **Initiale Analyse** von Mustern und Charakteristika der Daten
- Formulierung **initialer Hypothesen** für Modellierung
- **Vorplanung nächster Schritte** für Datenbereinigung, Datenaufbereitung, Featureauswahl sowie Vorauswahl geeigneter Modelltypen

Protokoll für Datenexploration (Zuur, 2010)

1. Ermittlung von Ausreißern in X und Y: Box-Plot, Cleveland Dotplot, Multivariate Analysis
2. Annahme der Varianzhomogenität Y: Box-Plot für Gruppen
3. Normalverteilung von Y: Histogramm, Q-Q-Plot
4. Anteil von Nullen in Y: Frequenzhistogramm
5. Multikolinearität von X: VIF, Corellogram, PCA
6. Zusammenhänge zwischen X und Y: Scatterplots, Box-Plot für Gruppen
7. Interaktionseffekt: Coplot
8. Unabhängigkeit von Y: ACF, Variogramm, Plot Y vs. Zeit/Ort

- Bei Anwendung von Modellen sollen Annahmen recherchiert werden.
- Die Reihenfolge lässt sich je nach Datensatz variieren.

Tools: Jupyter Notebook, Matplotlib, Plotly, Seaborn; Matlab Livescript



A. F. Zuur, E. N. Ieno, and C. S. Elphick, "A protocol for data exploration to avoid common statistical problems," *Methods Ecol. Evol.*, vol. 1, no. 1, pp. 3–14, Mar. 2010, doi: 10.1111/j.2041-210X.2009.00001.x.

Schritt 3: Data cleaning

Typische Probleme industrieller Daten

Allgemeines Vorgehen:

1. Identifikation möglicher Probleme
2. Suche nach Instanzen von Problemen
3. Korrektur von Fehlern
4. Dokumentierung von Fehlerpunkten
5. Anpassung der Routine zur Datenexploration

Typische Probleme:

- **Fehlende Werte**
  - Ausgefallen Geräte, Unterbrochene Verbindung
  - ➢ Datenimputation
- **Ausreißer**
  - Falsche Einstellung, zufälliger Sensorausfall
  - ➢ Korrektur/Entfernung
- **Datendrift**
  - Systematische Sensordegradation
  - ➢ Drift-Identifikation und Korrektur
- **Multikolinearität**
  - Redundanz, Regelkreise
  - ➢ Manuelle Feature-Selection, PCA, PLS
- **Abtastrate und Verzögerungen**
  - ➢ Multi-rate-Modelle, Resampling

Schritt 4: Data labeling

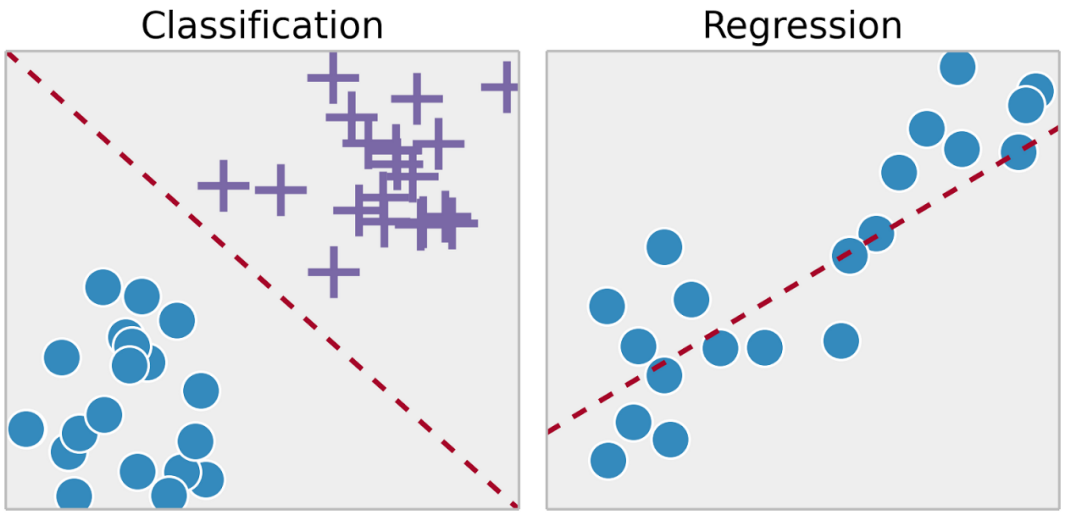
Labels

Labels sind im Trainingsdatensatz erhaltene **Zielvariablen**, die dem Trainieren des Modells dienen.

Relevant nur für **Überwachtes Lernen**.

Labelling ist ein Verfahren in dem Labels für Datensatz **manuell**, **semi-automatisch** oder **automatisch** erhoben werden.

Labelling setzt **Wissensexpertise** oder **automatische Gewinnung** von bereits annotierten Daten voraus.

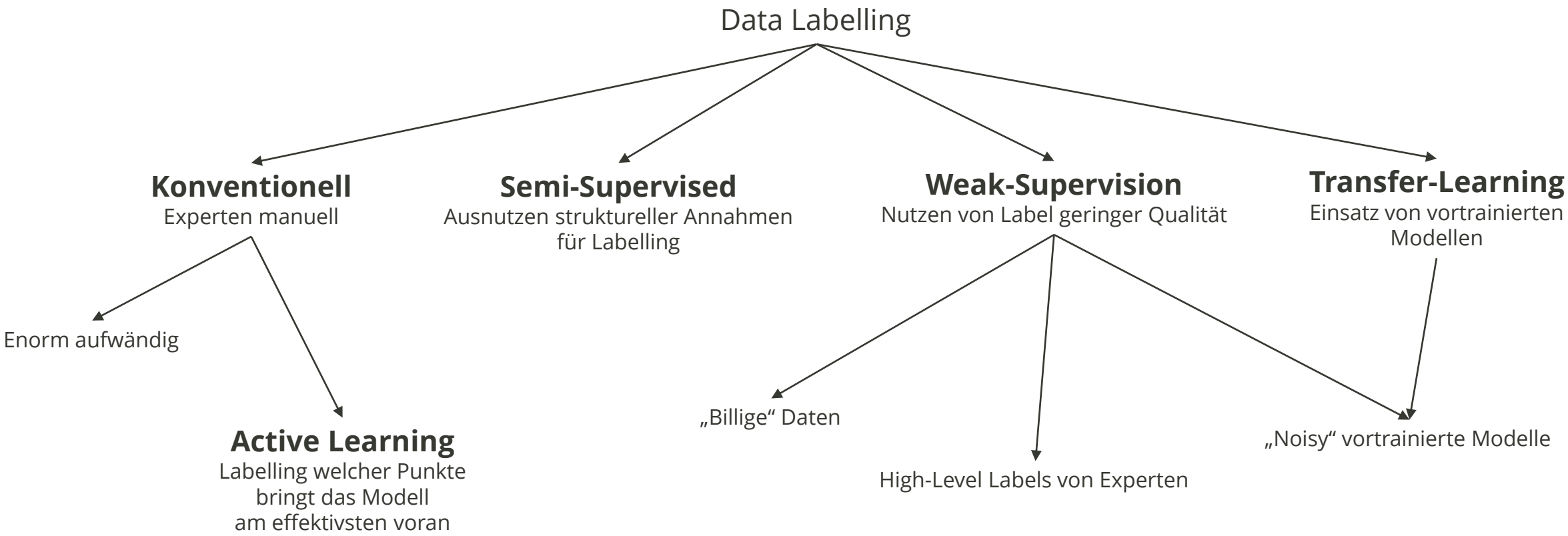


Labels: Klassen

Labels: Werte der Zielvariablen

Bild: https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d

Ansätze für Labelling



http://ai.stanford.edu/blog/weak-supervision/

Semi-supervised learning

Motivation:

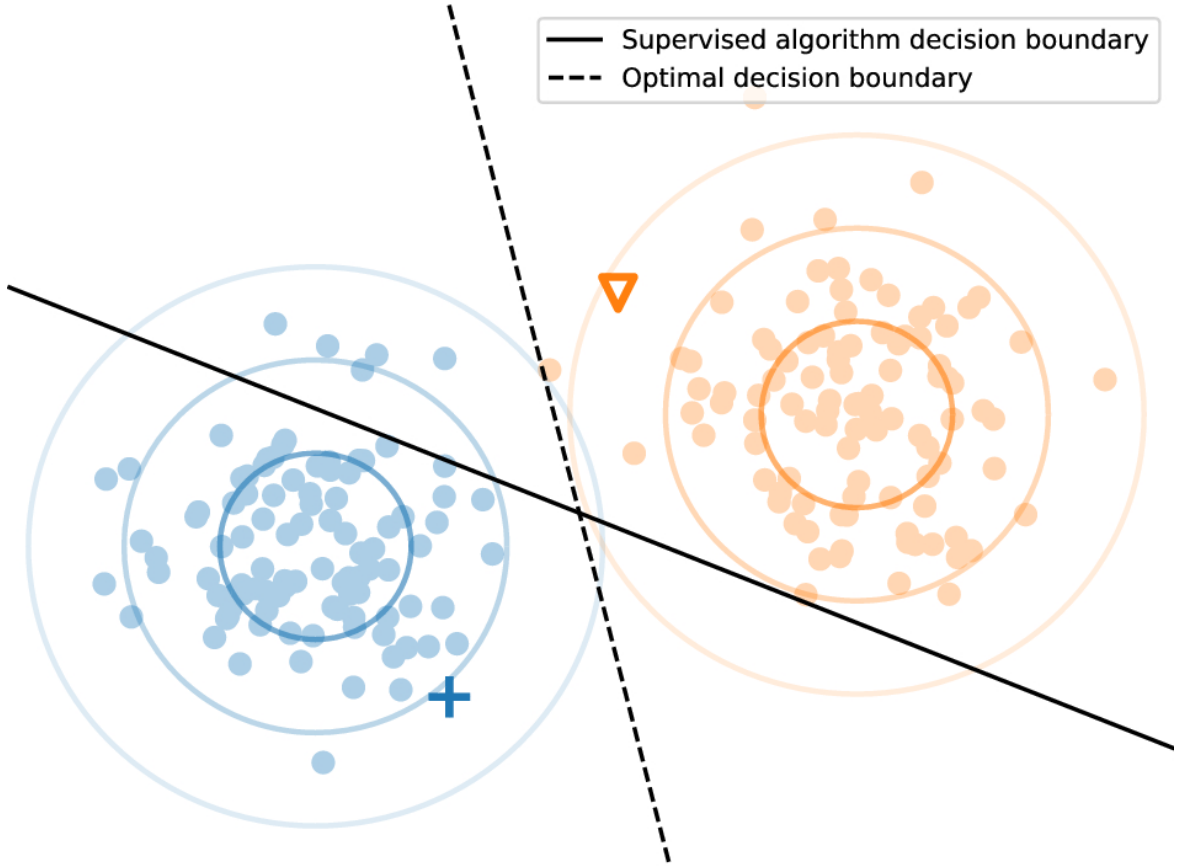
- Gewinnung von ungelabelten Daten ist mehrfach einfacher
- Labelling ist aufwändig

Grundidee:

- Nicht annotierte Daten liefern weitere nützliche Information für das Trainieren des Modells

Annahmen:

- **Glattheit**: wenn x_1 und x_2 nahe stehen, dann y_1 und y_2 sollen auch
- Entscheidungsgrenze liegt in der Region mit geringer Dichte
- Manifold-Annahme



J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, vol. 109, no. 2, pp. 373–440, 2020, doi: 10.1007/s10994-019-05855-6.

Schritt 5: Feature engineering

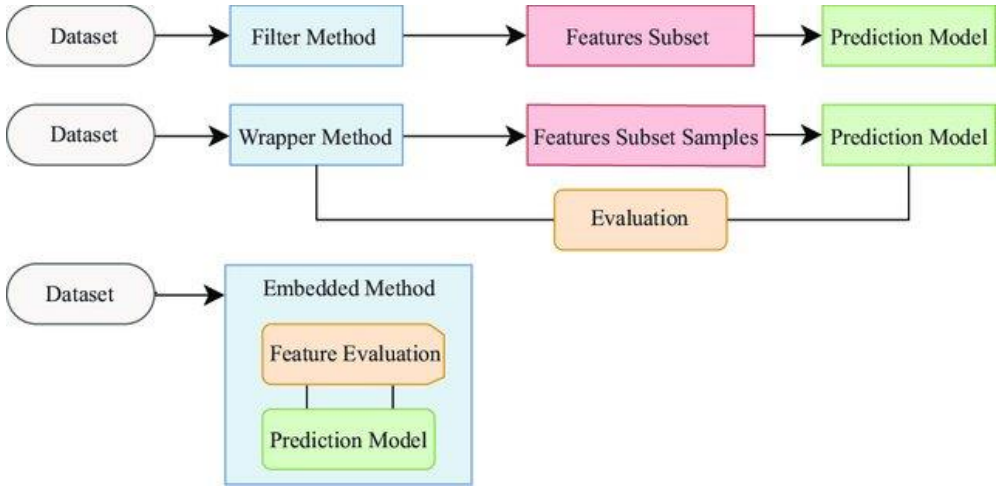
Feature Engineering

Feature-Engineering ist ein Schritt im Rahmen der Vorbereitung eines Trainingsdatensatzes aus Rohdaten, in dem Eingangsvariablen **extrahiert**, **ausgewählt** und **transformiert** werden.

Mehrwert von effektiv ausgewählten Features – effektiveres Modell:

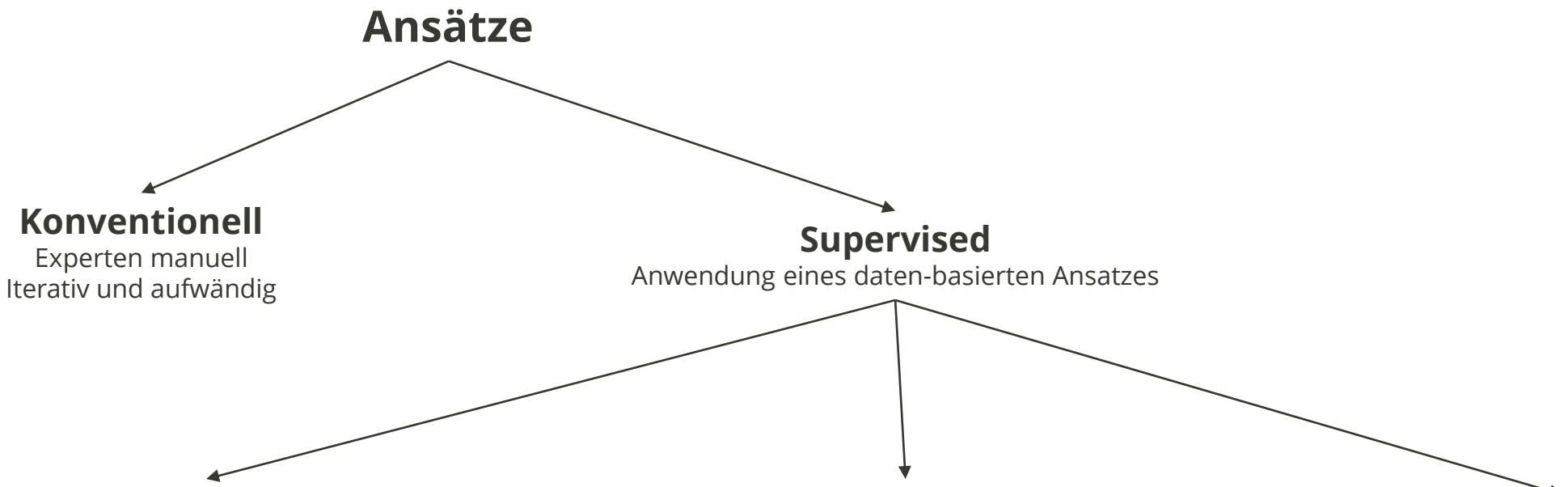
- Geringe Modellkomplexität
- Höhere Generalisierbarkeit und Reduzierung des Overfitting-Effekts
- Bessere Interprätierbarkeit
- Schnelles Modelltraining

Feature Selection



M. BABIKER, E. KARAARSLAN, and Y. HOŞCAN, "A hybrid feature-selection approach for finding the digital evidence of webapplication attacks," *TURKISH J. Electr. Eng. Comput. Sci.*, vol. 27, no. 6, pp. 4102–4117, Nov. 2019, doi: 10.3906/elk-1812-18.

Feature Selection



Filter

Auswahl von Features basiert auf **statistischen Metriken**:

- Varianz-Threshold
- Information-Gain
- Chi-Square-Test

Modellunabhängig

Wrapper

Numerische Optimierung, wo **das Zielkriterium dem Modellgüte** entspricht:

- Forward selection
- Backward elimination
- Stepwise selection

Overfitting-Gefahr, da Modell-spezifisch; Zeitaufwändig

Embedded

Feature-Selection ist **ein Anteil des Modells**:

- Regularization z.B. L1, L2
- Feature-Importance

Modellunabhängig

http://ai.stanford.edu/blog/weak-supervision/

Feature Transformation und Skalierung

Feature Transformation - zielgerichtete Wandlung von Daten ohne Verlust von Information.

Mögliche Ansätze: Logarithmieren, Multiplikation von Inputvariablen und Darstellung als einzelnes Feature, Differenzieren, Wechsel des Koordinatensystems

Skalierung: MinMax, Standard-Scaler, MaxAbsScaler, Robust-Scaler, L1-, und L2-Normalizer

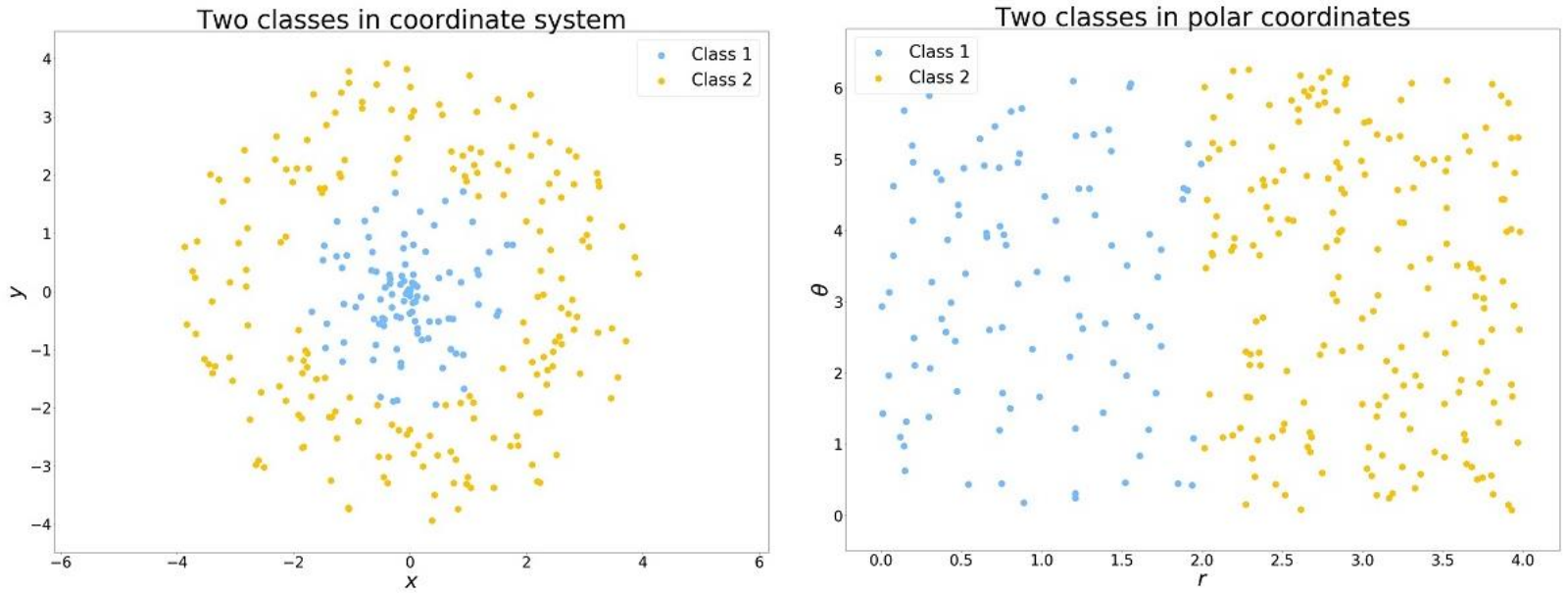


Bild: https://www.kdnuggets.com/2018/12/feature-engineering-explained.html

Schritt 6: Model training

Allgemeine Aspekte

Modell-Training ist das Lösen eines Optimierungsproblems zur Findung eines optimalen Sets von Modellparametern. Die Zielfunktion hängt dabei von dem Problemtyp ab.

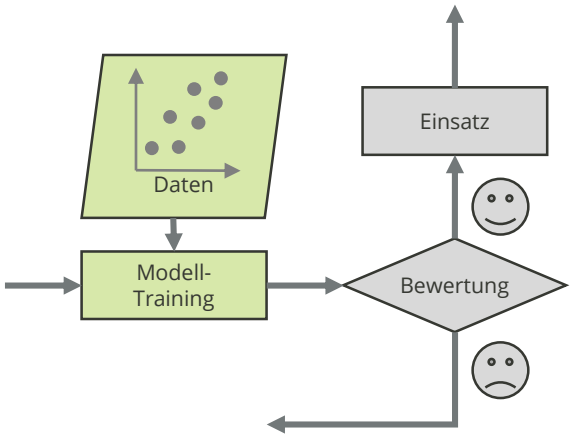
- Numerisch und iterativ (**Abstiegsverfahren** – sehr verbreitet)
- Zielfunktion ist Aufgabe- und Modelltypabhängig
- In der Regel Findung eines lokalen Optimums ausreichend

Frameworks:

- Scikit-Learn – Vielfalt an Modellen (Regression, Tree-based), aber nicht für DeepNN
- Tensorflow mit Keras – Fokus auf Modelle mit Schichte (kNN, CNN usw.)
- PyTorch (wenig populäre Alternative von Facebook)
- Matlab

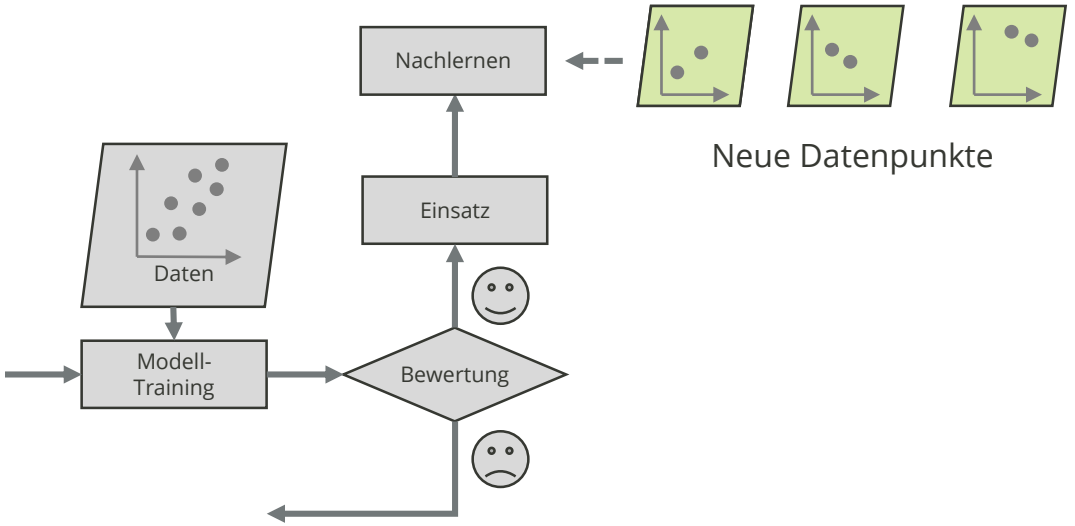
Arten von Modelltraining

Batch-Learning



Für initiales Training

Online-Learning



Für echtzeitige Aktualisierung des Modells im Betrieb

Adaptiert aus A. Géron, Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, 2019.

Model Training

Abstiegsverfahren ist die Basis.

Das Modelltraining erfolgt iterativ, wobei Modellparameter durch das Lernen von Daten schrittweise in **Batches** aktualisiert werden.

Epoch enthält alle Datenpunkte des Datensatzes aber nur **einmal**.

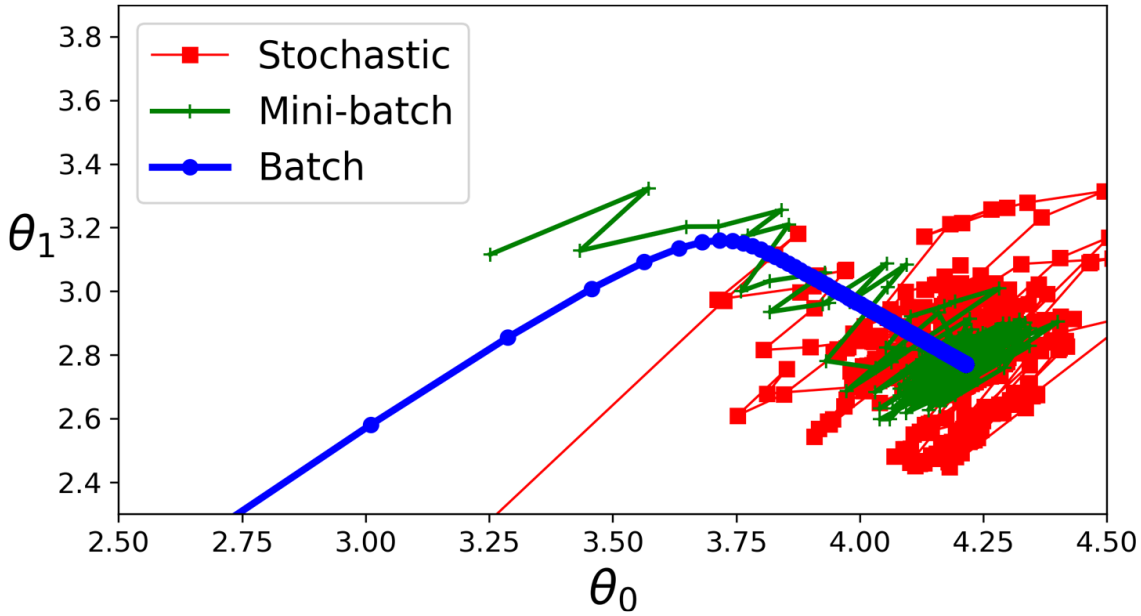
Zwischen Epochen sollen Batches neu zufällig erstellt.

Batches:

- **Kompletter Datensatz** (= Epoch) – Gradient-Descent
- **Ein** zufällig ausgewählter Punkt – Stochastic GD
- **Mehrere** zufällig ausgewählte Punkte – Mini-batch GD

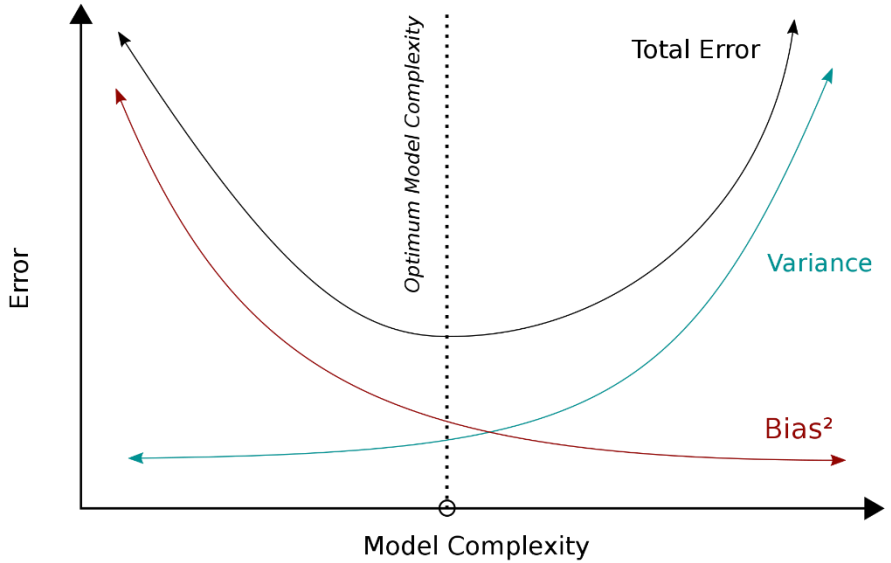
Training											
Epoch 1				Epoch 2				Epoch 3			
B1	B2	B3	B4	B1	B2	B3	B4	B1	B2	B3	B4

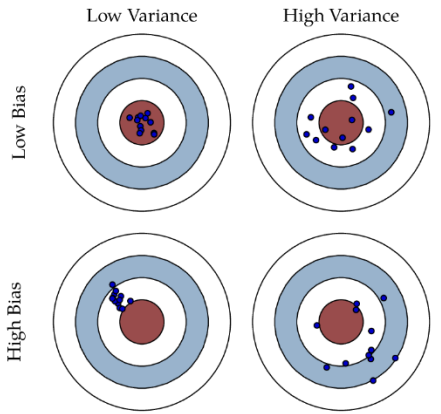
B1: Punkte 1, 4, 7, 11	B1: Punkte 1, 5, 12, 15	B1: Punkte ....
B2: Punkte 2, 5, 10, 14	B2: Punkte 2, 4, 10, 14	B2: Punkte ....
B3: Punkte 6, 9, 12, 16	B3: Punkte 3, 7, 13, 16	B3: Punkte ....
B4: Punkte 3, 8, 13, 15	B4: Punkte 6, 8, 9, 11	B4: Punkte ....

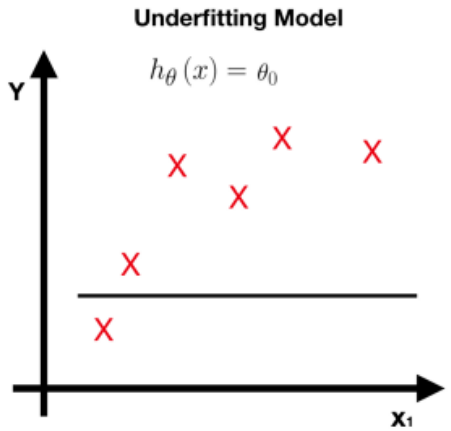


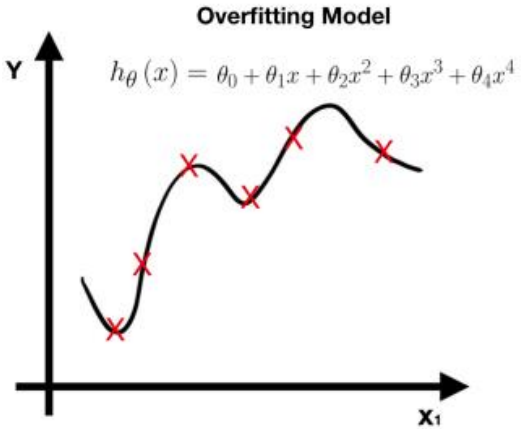
A. Géron, Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, 2019.

Bias, Variance und Overfitting





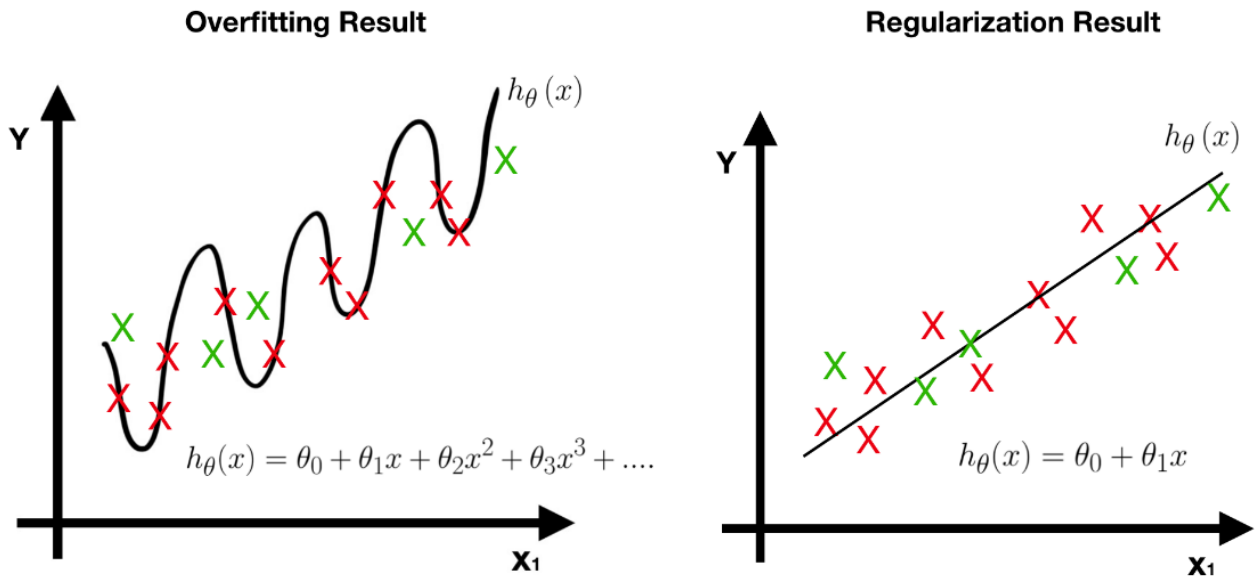




https://de.wikipedia.org/wiki/Verzerrung-Varianz-Dilemma#/media/Datei:Bias_and_variance_contributing_to_total_error.svg

https://medium.com/@qempsil0914/courseras-machine-learning-notes-week3-overfitting-and-regularization-partii-3e3f3f36a287

Regularization



Regularization ist ein Ansatz zur künstlichen Einschränkung von wenig relevanten Modellelementen (z.B. Input-Features oder Neuronen), z.B. L2-Regularization

Regularization Term

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j{}^2$$

start at θ_1

Regularization Parameter

Bilder: https://medium.com/@qempsil0914/courseras-machine-learning-notes-week3-overfitting-and-regularization-partii-3e3f3f36a287

Optimierung von Hyperparametern

Nachdem der Typ bzw. die Architektur des Modells definiert wurde, sollen **Modell- und Trainingsparameter** optimiert (andere Ebene der Optimierung als beim Modell-Training) werden.

Exemplarische Parameter:

- Ordnung des Regressors
- Art des Kernels im SVM
- Anzahl von Schichten und Neuronen in NN

- Beiwerte und Art der Regularisierung
- Learning-Rate des Abstiegsverfahrens

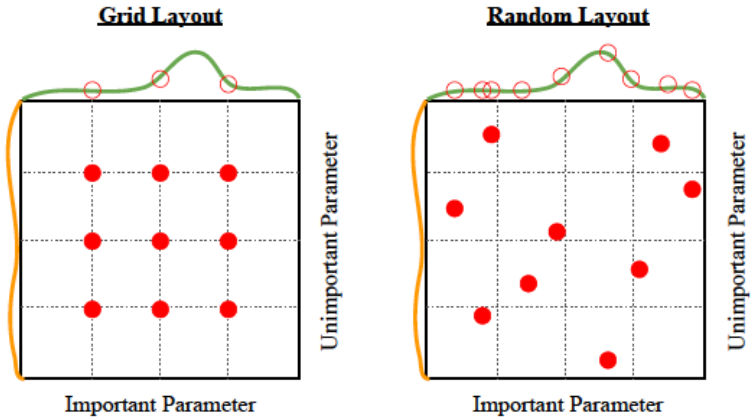
Direkte Verfahrens (Parallelisierung):

- **Grid search**

Alle mögliche Kombinationen von Parametern werden evaluiert

- **Random-Search**

Zufällig ausgewählte Kombination von Parametern



Iterative: **Bayesian, Graident descent** usw.

Bild: https://srdas.github.io/DLBook/DL_images/HPO1.png

Schritt 7: Model evaluation

Model testing

Modelltests dienen der Bewertung der Modellgüte hinsichtlich der Qualitätsmetriken (**nicht unbedingt dasselbe wie die Zielfunktion**).

Qualitätsmetriken – Problemtypabhängig:

- Regression: MSE, MAE
- Klassifikation: Accuracy, Confusion-Matrix, ROC, AUC

Datenaufteilung:

- Training (70-80%) – Optimierung des Modells
- Validation (15-20%) – **unabhängige Bewertung** der Hyperparameteroptimierung
- Test (15%) – **unabhängige Bewertung** des resultierenden Modells (inkl. Hyperparameter-Tuning)

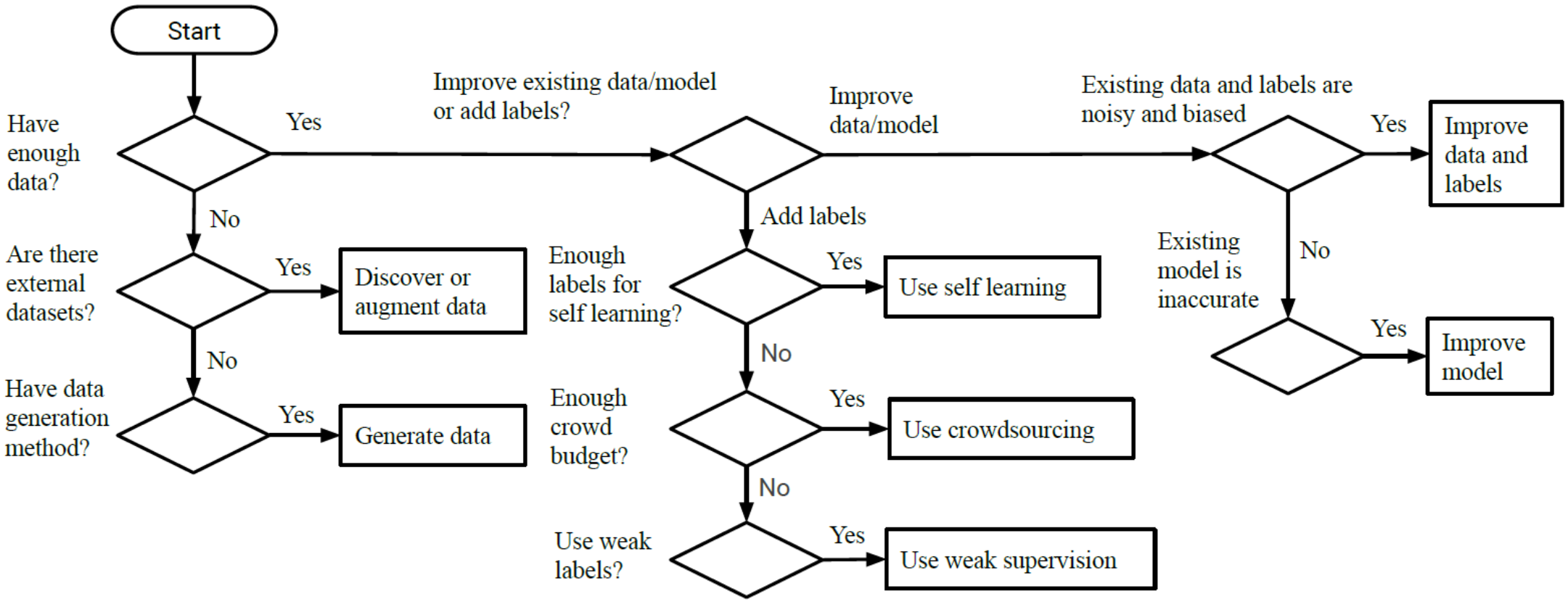
k-fold Cross Validation:

- Aufteilung des Datensatzes auf k Anteilen, die für Evaluation des Modells verwendet werden

Vorteil: kein separater Validerungsdatensatz erforderlich

https://de.wikipedia.org/wiki/Kreuzvalidierungsverfahren#/media/Datei:KfoldCV.gif

Allgemeiner Workflow

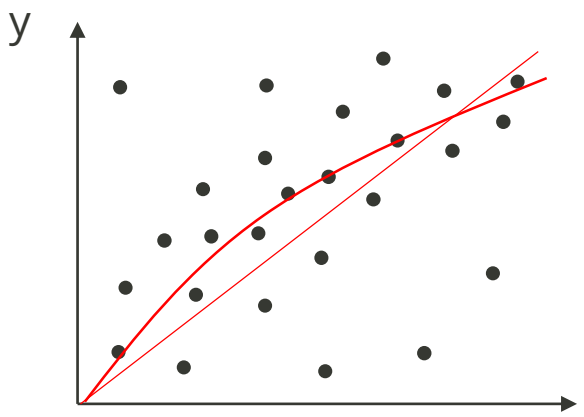


Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2019, doi: 10.1109/TKDE.2019.2946162.

Datenzentrierte Paradigma in ML

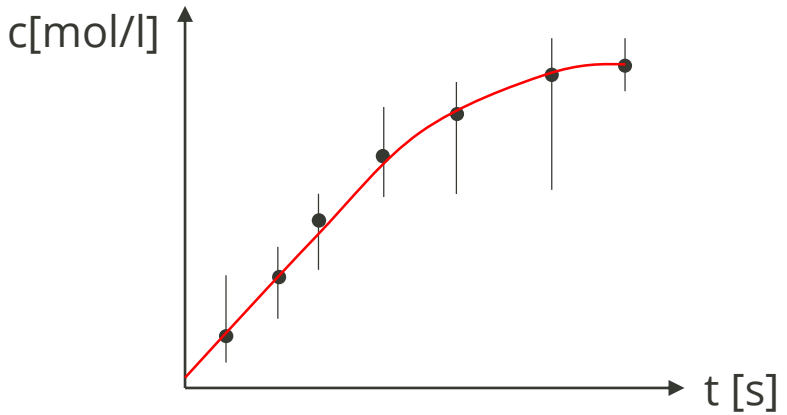
Modellzentrierter Ansatz

- Extensive Datengewinnung (je größer der Datensatz desto besser)
- Iterative Optimierung des Modells für bessere Performance steht im Fokus
- Datensatz wird als eine vorgegebene und unveränderbare Komponente betrachtet



Datenzentrierter Ansatz

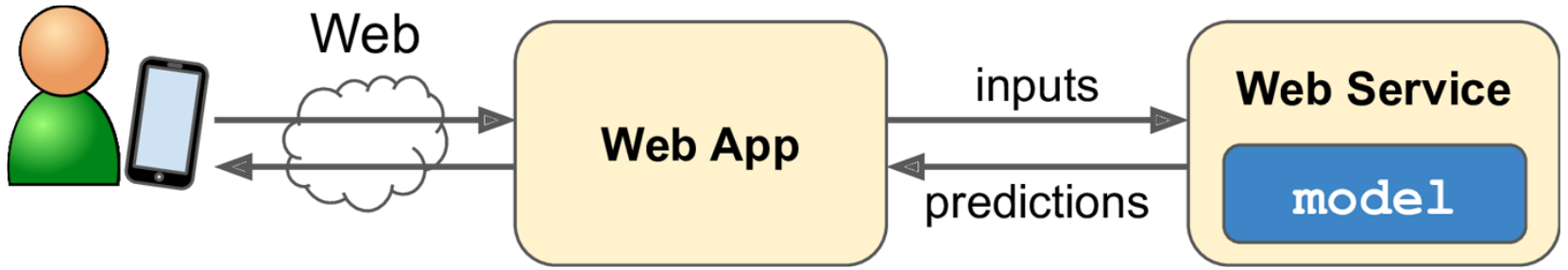
- Intensive Datengewinnung (Qualität statt Quantität)
- Datenkonsistenz und iterative Verbesserung der Datenqualität steht im Fokus
- Sobald gelungen, perfomen alle Modelle besser



Adaptiert von Andrew NG (2021), Link: https://youtu.be/06-AZXmwHjo

Schritte 8 und 9: Model deployment und monitoring

Model deployment

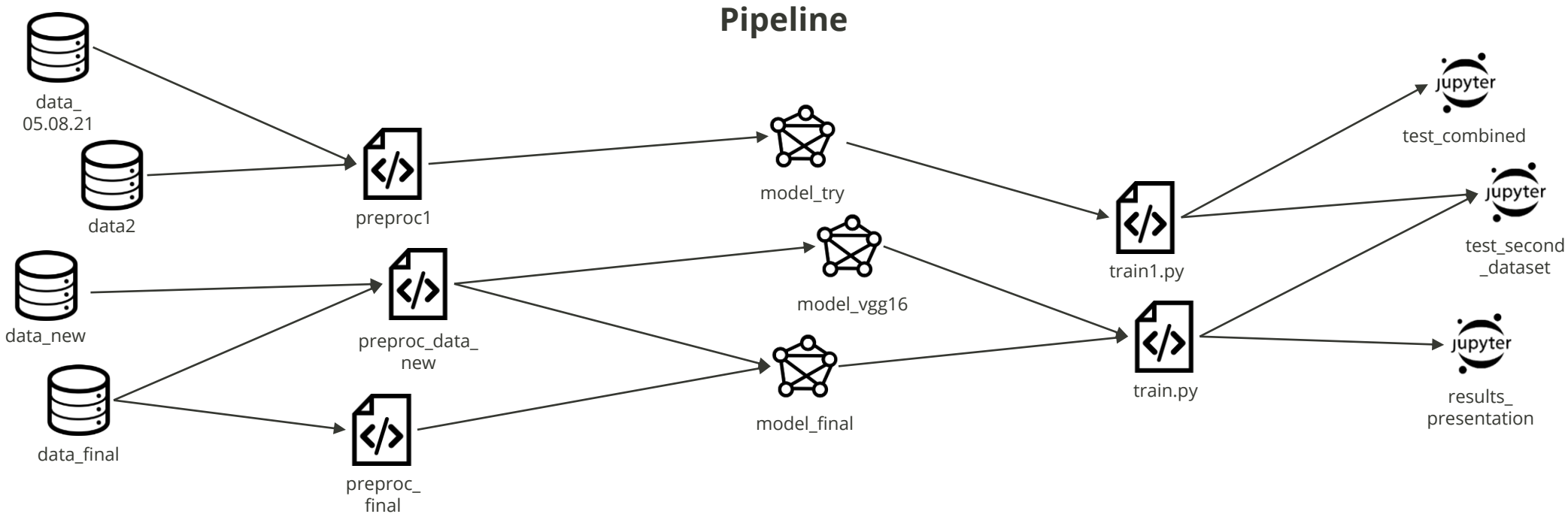


Technologien:
- RestAPI, GraphQL, MQTT
- OPC UA

Frameworks:
- Custom-Webserver mit Matlab, Python Flask oder Easy API
- Tensorflow Serving

MLOPs

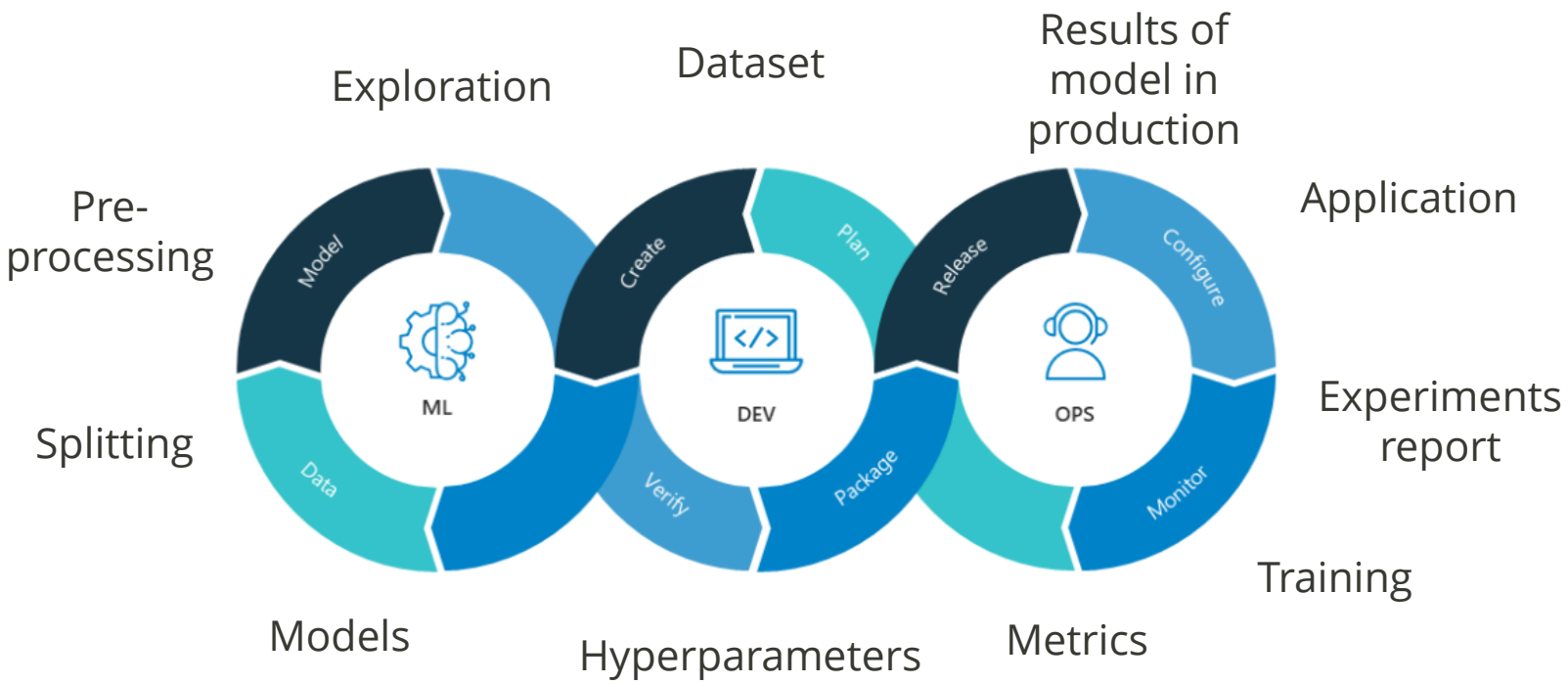
ML Experiments



Versionierung

Überischt

MLOps



Automation

Continuous X

Versioning

Experiments tracking

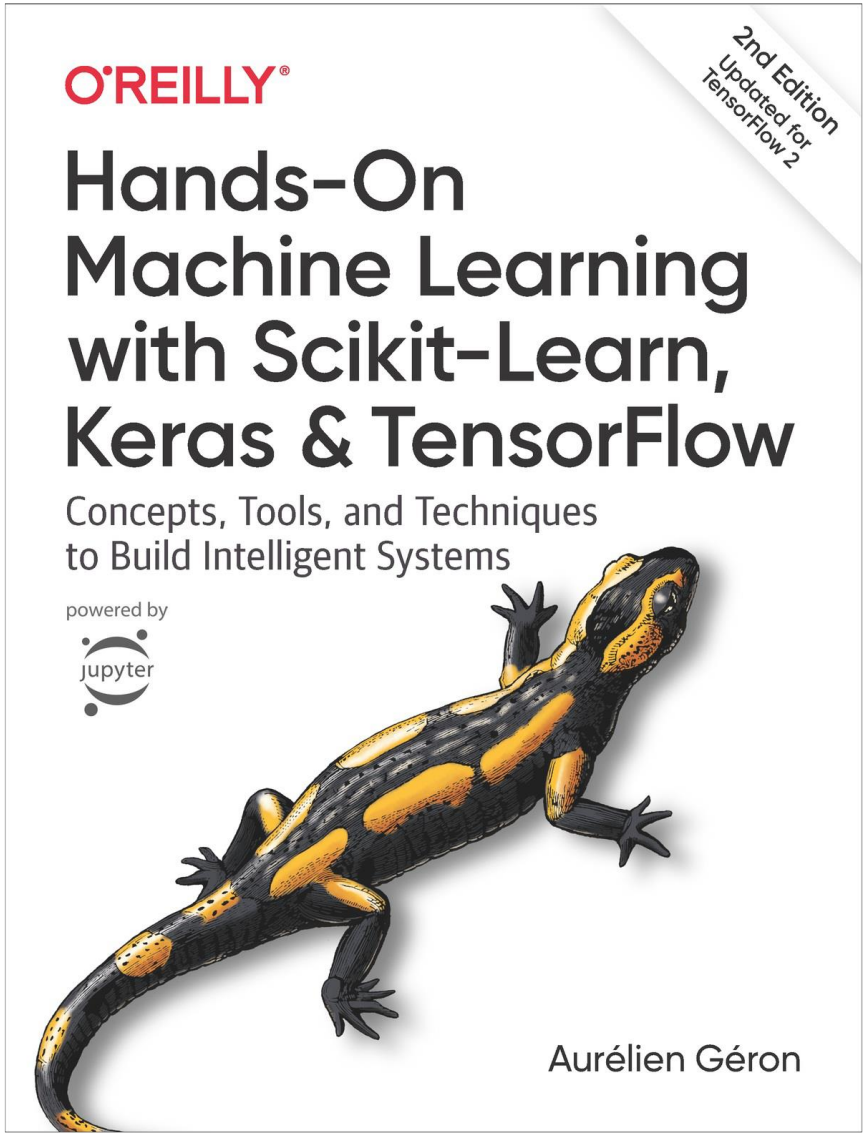
Testing

Monitoring

https://blogs.nvidia.com/blog/2020/09/03/what-is-mlops/

https://ml-ops.org/content/mlops-principles

Zusammenfassung


O'REILLY®
2nd Edition
Updated for TensorFlow 2
Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow
Concepts, Tools, and Techniques to Build Intelligent Systems
powered by
jupyter
Aurélien Géron

kaggle

PROCESS CONTROL SYSTEMS
PROCESS SYSTEMS ENGINEERING

Dr. rer. nat. Valentin Khaydarov
Email: valentin.khaydarov@tu-dresden.de
Telefon: 0351 463 33387

Vielen Dank für Ihre Aufmerksamkeit!