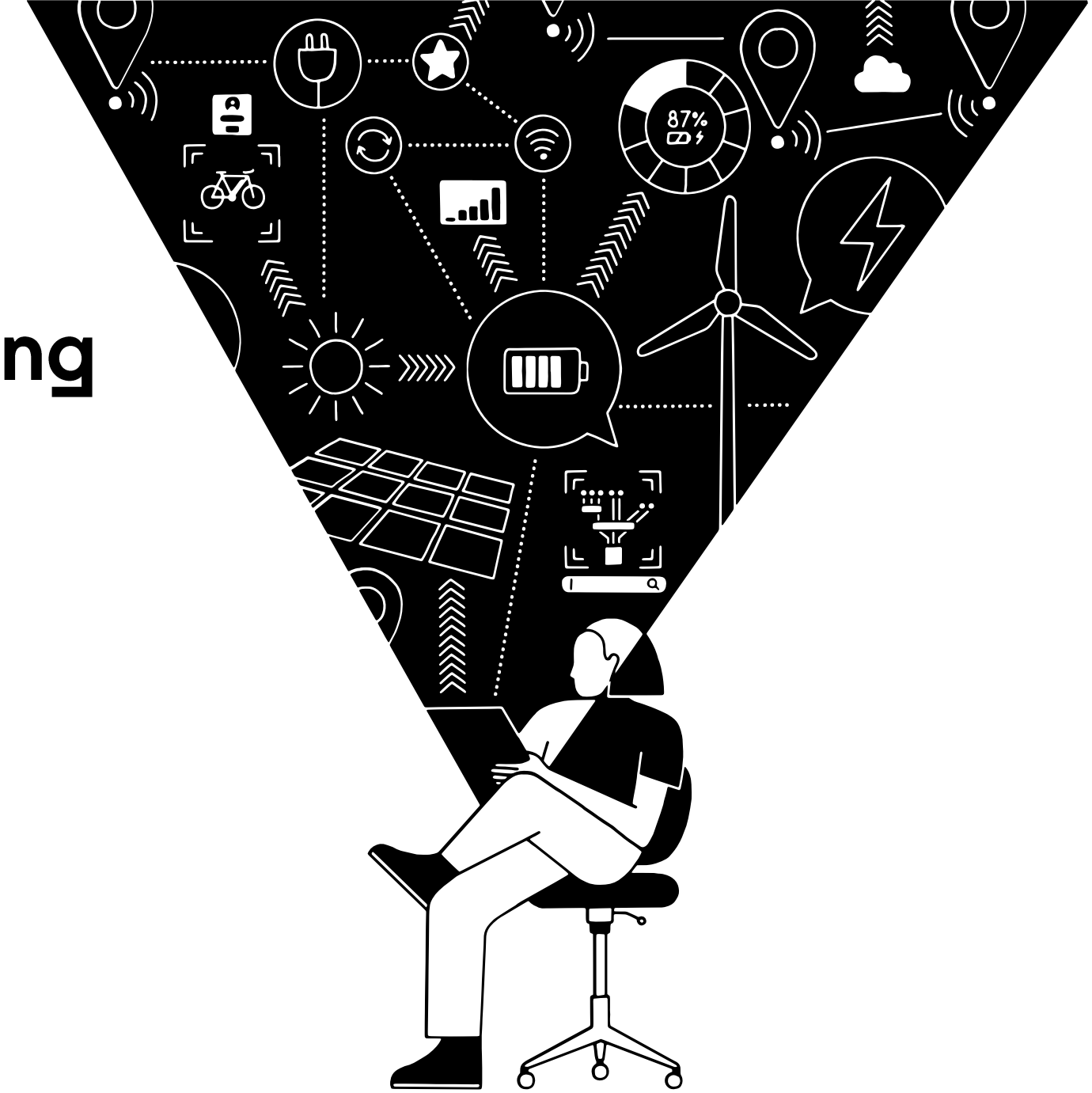
Grundlagen Supervised & Unsupervised Learning Tag 1

Mit Anwendungsbeispielen in TensorFlow Keras

GFU Cyrus AG

Tobias Krebs

exeta



TOBIAS KREBS

Associate Data Engineer

Tobias Krebs verfügt über mehr als 3 Jahre Erfahrung in der Entwicklung von Python Anwendungen. Insbesondere in der Softwareentwicklung und der Entwicklung von Machine Learning Applikationen konnte er diese sammeln. Darüber hinaus verfügt er über praktische Erfahrung im Bereich des Cloud Computing und gibt sein Wissen in Schulungen, insbesondere zu den Themen Data Science und Machine Learning, als Dozent weiter.

Biografie

- EXXETA
- M.Sc Economics Humboldt Universität zu Berlin
- Information Systems Chair Humboldt Universität zu Berlin
- Computer Science Fakultät Universität Kopenhagen

Beratungskompetenz

- Data Science, Machine Learning, Deep Learning, Data Engineering
- IT Expertise u.a. in Python (Pandas, scikit-learn, tensorflow, keras, pytorch, numpy)
- AWS Cloud: Elastic Computing 2 (EC2)

Sprachen

- Deutsch, Englisch

Auszug relevante Projekterfahrung

Data Engineer | Automatisierte Bereitstellung von GPU Cloud Umgebungen

- Auswahl geeigneter Umgebungen
- Automatisierte Erstellung von passenden Machine Images für Data Science Anwendungen
- Erstellung eines konfigurierbaren Skripts zum Starten benötigter Computing Umgebungen

Data Scientist | Masterarbeit: Predicting media bias of news articles using deep learning

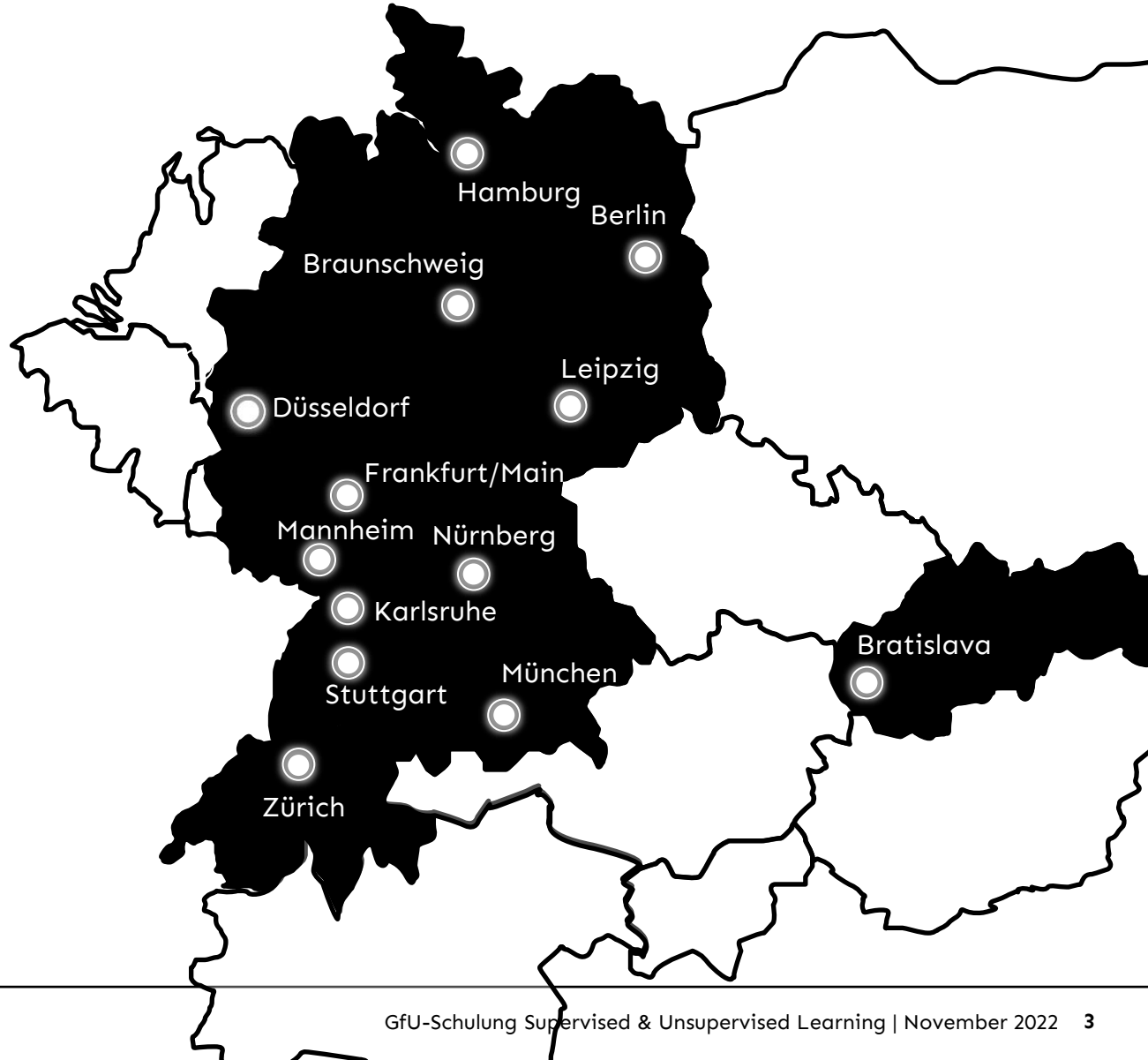
- Ausführliche Literatur Recherche zum Thema
- Aufbereiten der Daten für NLP Usecase
- Anwenden und vergleichen verschiedener Machine Learning Modelle (Random Forest, LSTM, BERT)

Data Scientist | Erstellen von Seminaraufgaben für Python und Machine Learning

- Erstellen einer Übersicht aller benötigten Aufgaben
- Schreiben der einzelnen Jupyter Notebooks mit Aufgaben und Lösungen

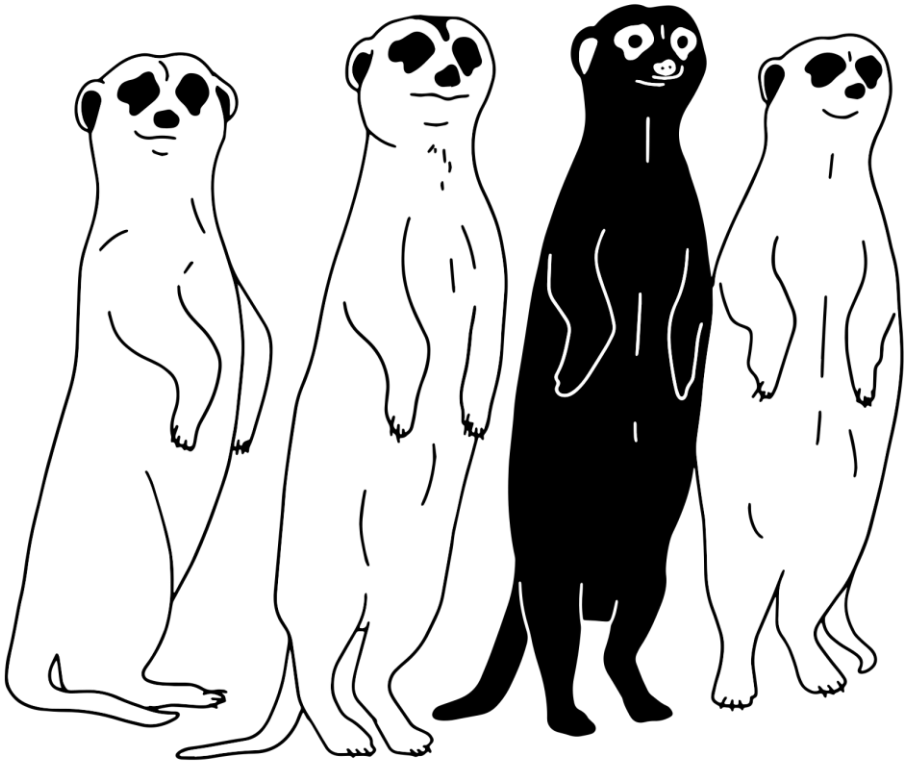
Exxeta in Zahlen

Gegründet	**2005**
Mitarbeitende	**> 1.100**
Umsatz	**> €100M**
Standorte	**13 (D, CH, SK)**



Vorstellungsrunde

- Hintergrund – Was mache ich in meinem Beruf/Tätigkeit?
- Habe ich bereits Erfahrungen mit Machine Learning gemacht?
- Warum besuche ich diese Schulung?
- Welche Vorkenntnisse habe ich? (Allgemein ML, Python, tensorflow/keras oder ähnliche Tools bspw.: PyTorch)
- Was erwarte ich mir von dieser Schulung?



Agenda - Tag 1

Vorstellungsrunde

Grundlagen & Überblick Machine Learning

Supervised Learning

Neural Networks (MLPs)

Deep Neural Networks (CNNs)

Agenda - Tag 2

Offene Themen von Tag 1

Klassisches Unsupervised Learning

Deep Neural Networks (RNNs + LSTMs)

Agenda - Tag 3

Offene Themen von Tag 2

Deep Learning Unsupervised Learning

Offene Themen & Fragen

Feedback

Lernziele

- Grundlegendes Verständnis über die Methoden und Konzepte des Supervised und Unsupervised Learning
- Überblick über Theorie und Praxis von neuronalen Netzen und deren Erweiterung: Deep Learning Methoden wie beispielsweise Convolutional Neural Networks (CNNs) oder Recurrent Neural Networks (RNNs), und deren Einsatzfelder
- Einführung in das Package Tensorflow/tensorflow.keras zum Trainieren der Modelle
- Ausblick auf weitere Methoden

Literatur

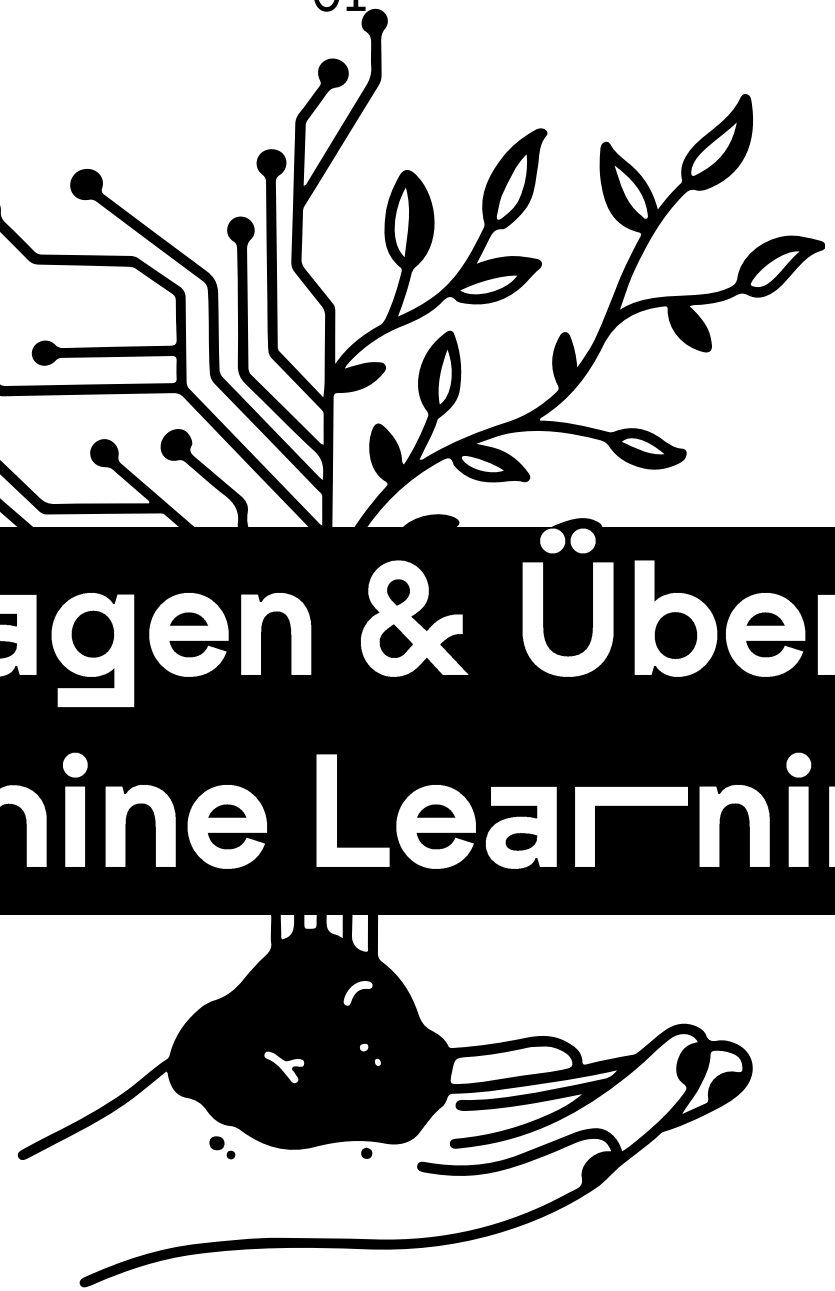
- Provost, Foster, and Tom Fawcett. Data Science für Unternehmen : Data Mining und datenanalytisches Denken praktisch anwenden, mitp, 2017. → Eher Business

- Géron, A., Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems, O'Reilly, 2019. → Eher technisch

- Raschka, S., Machine Learning mit Python: Das Praxis-Handbuch für Data Science, Predictive Analytics und Deep Learning, mitp, 2016 → Eher technisch

Unterlagen herunterladen

- Öffnen der bevorzugten Git Konsole (z.B Git Bash)
- Auswahl des Ordners in dem das Repository abgelegtwerden soll
- Klonen des Repositorys: **git clone https://gitfront.io/r/user-2025188/sv5fADUekGjL/supervised-unsupervised-learning-keras/**



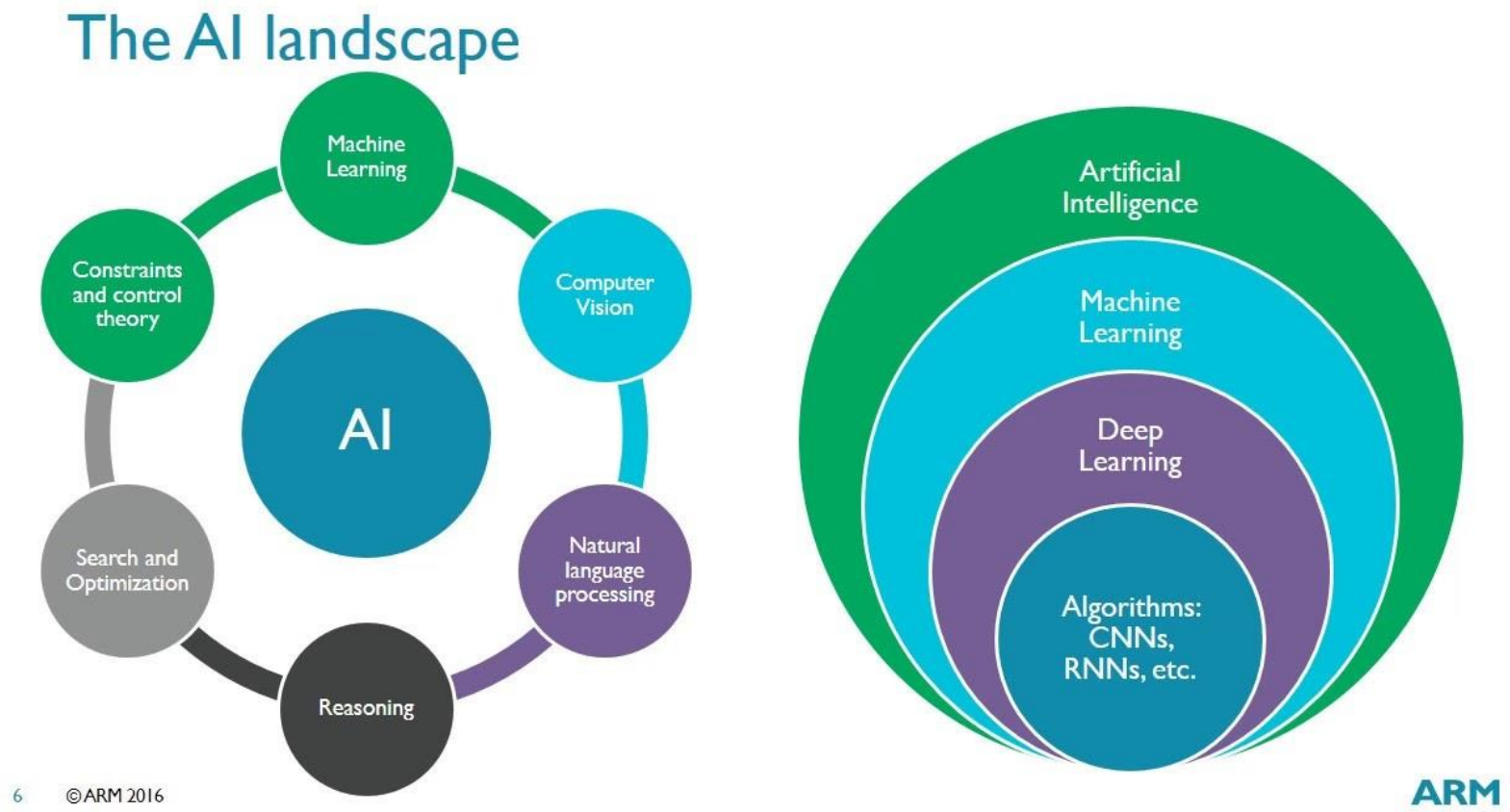
Grundlagen & Überblick Machine Learning

Was ist Machine Learning?

„Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed“

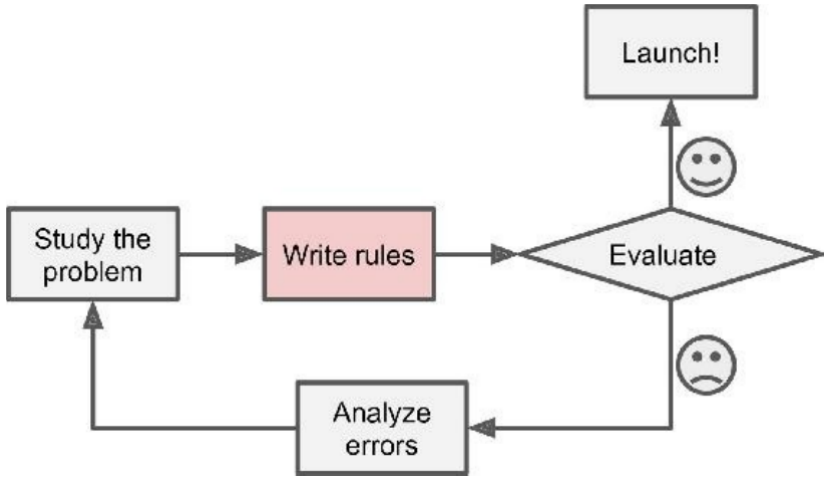
Arthur Samuel, 1959

Artificial Intelligence und Machine Learning

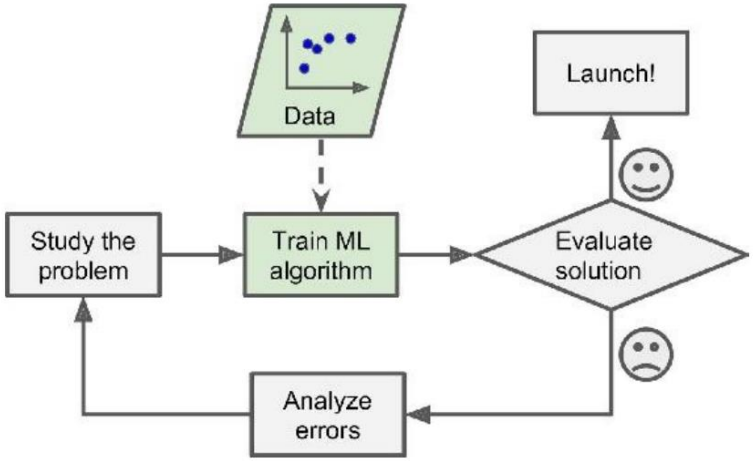


Warum Machine Learning?

Motivation – Spam Filter



Traditioneller Ansatz
- Komplex, hard-codiert
- Schwer zu warten



Machine learning Ansatz
- Automatisches lernen aus Daten
- Automatisches re-trainieren

Géron, A., Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow,2019, p. 3 f.

Beispiele aus dem Alltag

Sentiment-Analyse (NLP):

Hat ein bestimmter Post/Review eine positive oder negative Aussage → Automatische Erstellung von Tags für Nutzerreviews

Gaming-Bot:

Intelligente Systeme, welche mithilfe von Reinforcement Learning trainiert werden. Beispiel ist das berühmte AlphaGo, welche den World Champion besiegte.

Entdecken von Hirntumoren in MRT Scans:

Bestimmen der Größe und Position von Tumoren in MRT Scans mithilfe von Image Classification (CNNs)

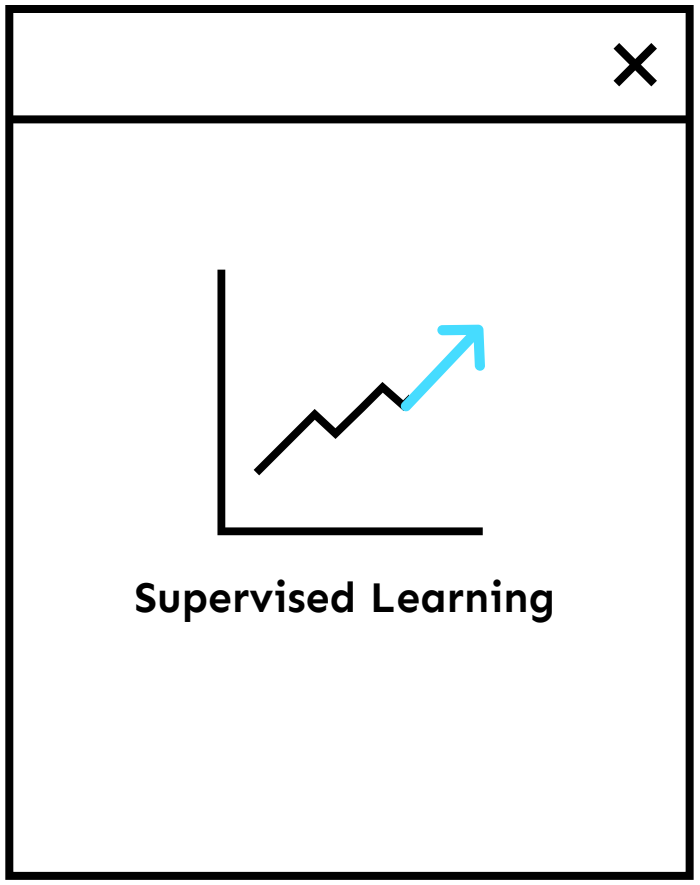
Vorhersage des nächsten Jahresumsatzes:

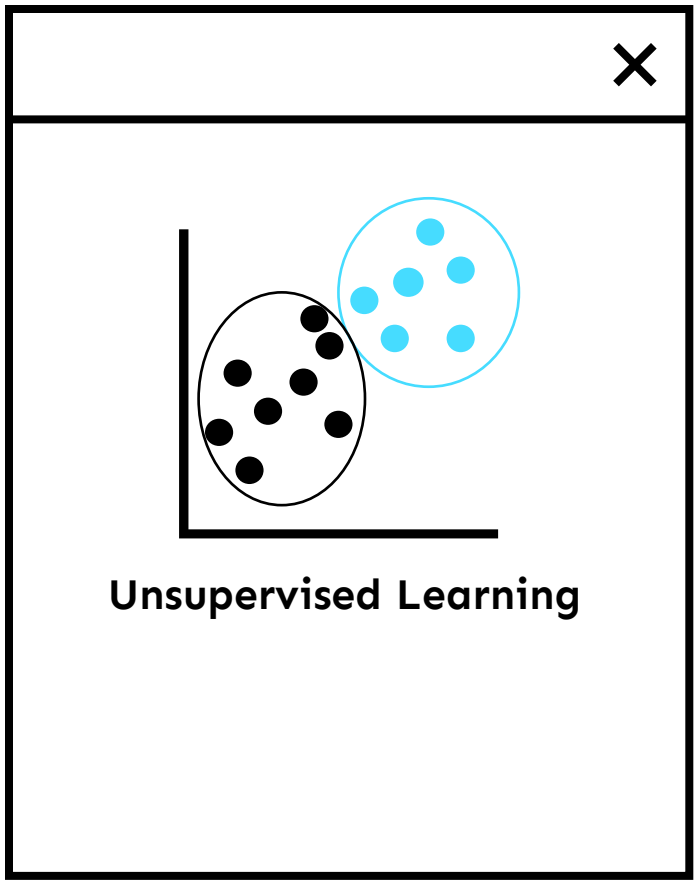
Regressionsmodelle, welche beispielsweise aus historischen Daten trainiert werden. Diese können beispielsweise neuronale Netze sein.

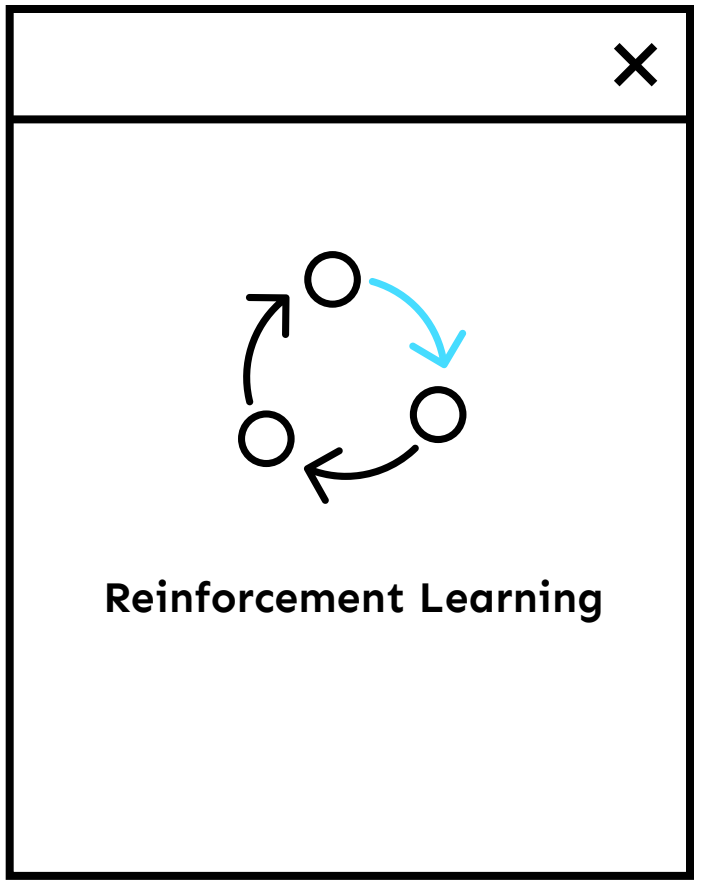
Fallen Ihnen weitere Beispiele ein?

Géron, A., Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow,2019, p. 5 f.

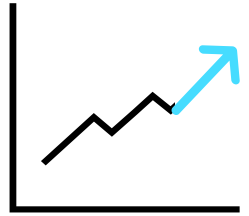
Supervised vs. Unsupervised vs. Reinforcement Learning







Supervised Learning



„Überwachtes Lernen“: In den historischen Daten ist das Ergebnis (**Label**), das in der Zukunft vorhergesagt werden soll, bekannt.

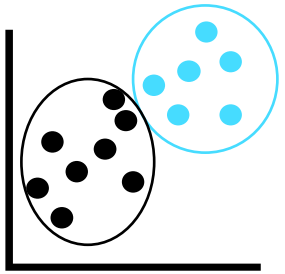
- Die historischen Daten nutzt der Machine-Learning-Algorithmus, um das Ergebnis in der Zukunft bestmöglich vorherzusagen.
- Das Modell wird zielgerichtet **trainiert,** d.h. das Muster, das den Zusammenhang zwischen Eingangsdaten und Ergebnis darstellt, wird erlernt.

Prediktive Fragestellungen (**Predictive Analytics**):

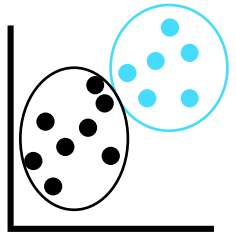
- **Klassifikation**: Das vorherzusagende Merkmal ist **kategorisch**
- **Regression**: Das vorherzusagende Merkmal ist **numerisch** mit nicht klar abgrenzbaren Werten

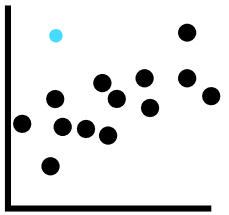
Die **Güte** des Modells ist durch die **Abweichung** zwischen Prognosewert und Label direkt messbar.

Unsupervised Learning



„Unüberwachtes Lernen“: Es gibt keine Vergleichswerte aus der Historie; allgemeine Muster in den Daten werden erkannt.





- **Clustering**: Das Modell sucht nach gemeinsamen Merkmalen z.B. Kunden werden in Gruppen mit ähnlichem Bestellverhalten geclustert
- **Anomaly Detection**: Das Modell sucht nach Auffälligkeiten oder Ausreißern im Datenset.

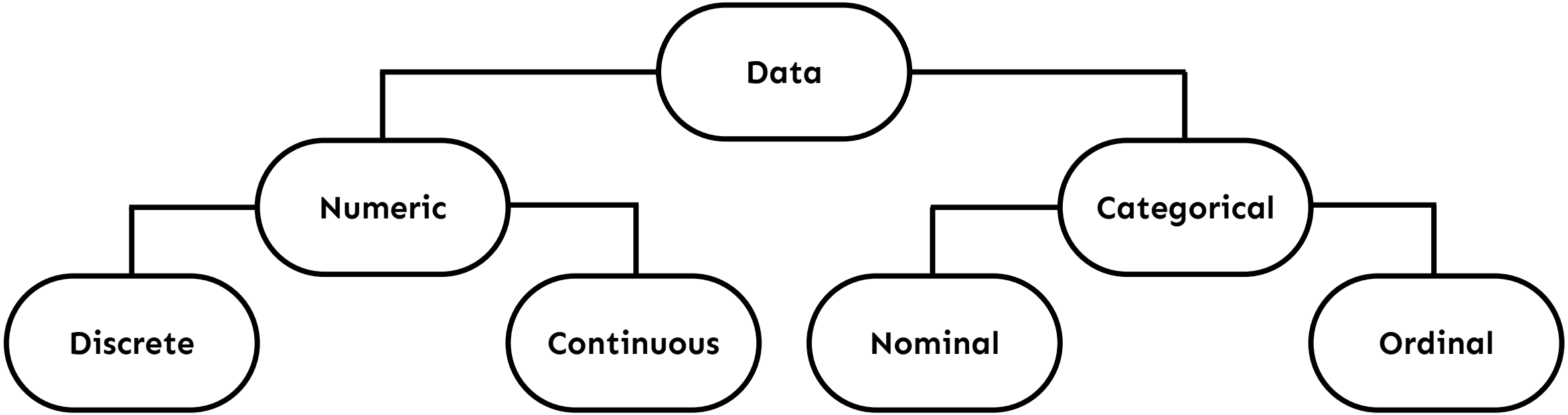
  z.B. Kunden mit unüblichem Bestellverhalten werden als auffällig gekennzeichnet

Die **Güte** des Modells ist nicht direkt messbar, vielmehr liegt diese im Auge des Betrachters

- Müssen die Cluster feingranularer gebildet werden? Können Cluster zusammengelegt werden? Hilft das Clustering für weiterführende Analyseschritte?
- Handelt es sich bei den Auffälligkeiten tatsächlich um problematische Anomalien? Oder handelt es sich nur um sehr seltene, aber dennoch nachvollziehbare Ausreißer?

Die **Fachlichkeit** spielt eine große Rolle. Das Ergebnis muss im fachlichen Kontext **interpretiert** und bewertet werden.

Datentypen Machine Learning



Werte sind Integer:
- Anzahl Studierende
- Alter

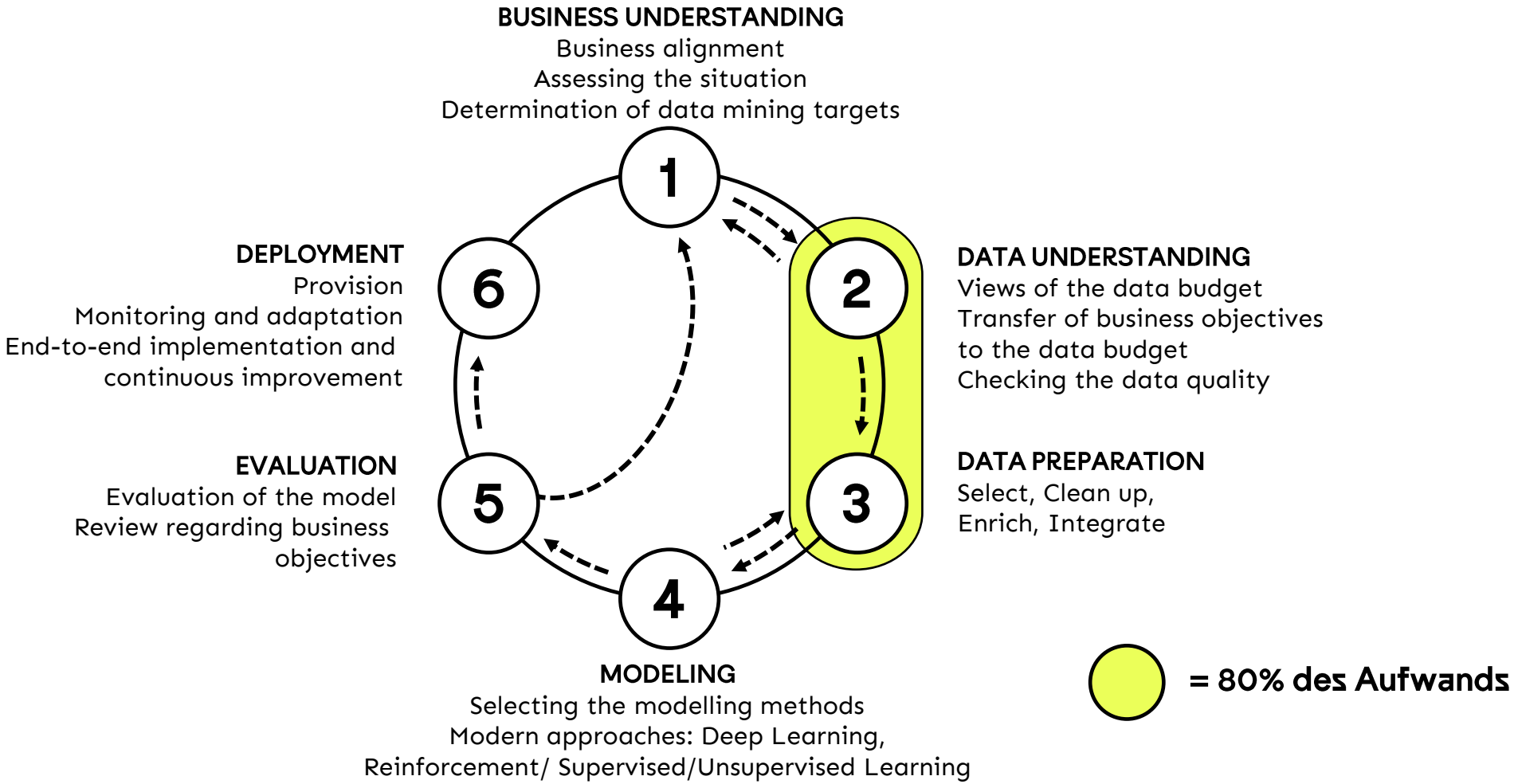
Werte können jeden Wert annehmen, üblicherweise innerhalb einer Range:
- Temperatur
- Alter

Keine natürliche Reihenfolge zwischen Kategorien:
- Geschlecht
- Länder
- Farbnamen

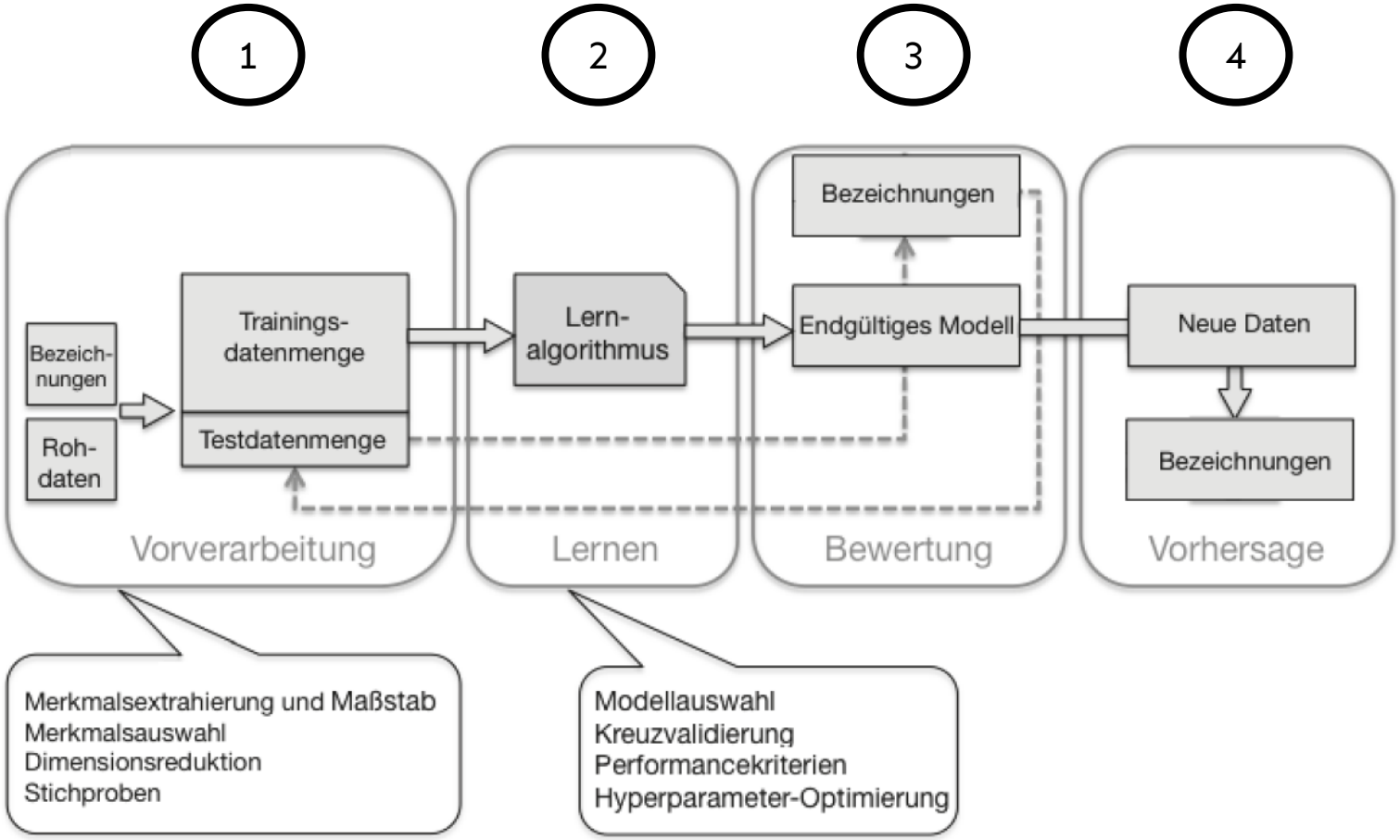
Eine Reihenfolge zwischen Kategorien:
- Tshirt Größen (S, M, L)
- Tageszeit (morgens, mittags, abends)

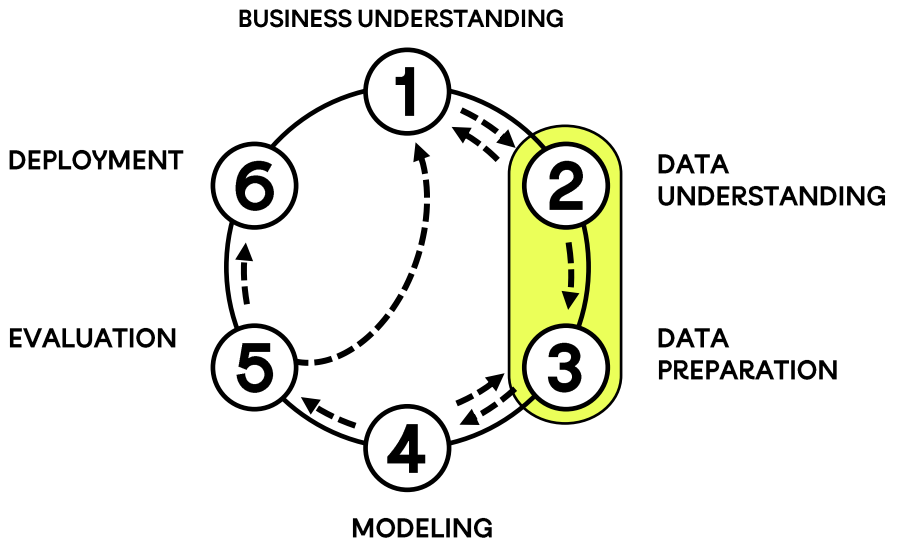
Cross Industry Standard Process of Data Mining (Crisp-DM)

Der Data Science / Machine Learning Workflow



Machine Learning Pipeline





Raschka, S., Machine Learning mit Python: Das Praxis Handbuch für Data Science, Predictive Analytics und Deep Learning, 2016, S.31 f.

Typen von Machine Learning Problemen in dieser Schulung

1. **Klassifikation und Wahrscheinlichkeitsabschätzung der Klassenzugehörigkeit**

- Unterteilung der Daten in Klassen z.B.: Betrug/kein Betrug, Spam/kein Spam
- Ziel ist es, jeden Datenpunkt einer Klasse zuzuordnen
- Fragestellung: Welche Kunden haben hohes Abwanderungspotenzial?

2. **Regression**

- Vorhersagen eines numerischen Wertes z.B.: Aktienkurs, Umsatz
- Fragestellung: Wieviel Umsatz werden wir im nächsten Jahr voraussichtlich machen?

3. **Clustering**

- Zusammenfassen von Datenpunkten zu Gruppen anhand ihrer Ähnlichkeit
- Es wird vorrangig kein richtiges „Ziel“ verfolgt
- Fragestellung: Finden wir ähnliche Gruppen in unseren Daten wieder?

Provost, Foster, Tom Fawcett, Data Science für Unternehmen, 2017, S. 46-49

Herausforderungen Machine Learning

Machine Learning bringt im Vergleich zu traditionellen Software Systemen **neue Herausforderungen** mit sich:

- Unzureichende Trainingsdaten
- Trainingsdaten sind nicht repräsentativ (Sampling Bias)
- Schlechte Datenqualität (Fehlwerte → oder Fehlwerte, die nicht als solche erkennbar sind Bsp.: als „unknown“ gekennzeichnet)
- Irrelevante oder unzureichende Features
- Overfitting → grundlegendes Problem ist Bias/Variance Trade-Off
- Generell gilt **„Garbage in, garbage out“**!
  → Besonders bei der Automatisierung von Geschäftsentscheidungen


Vorbereitung

Installation Miniconda

1. Miniconda herunterladen: **https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86_64.exe**

2. Und Installieren: **https://conda.io/projects/conda/en/stable/user-guide/install/windows.html**

Anlegen einer Virtuel Environment

1. Anaconda Prompt öffnen

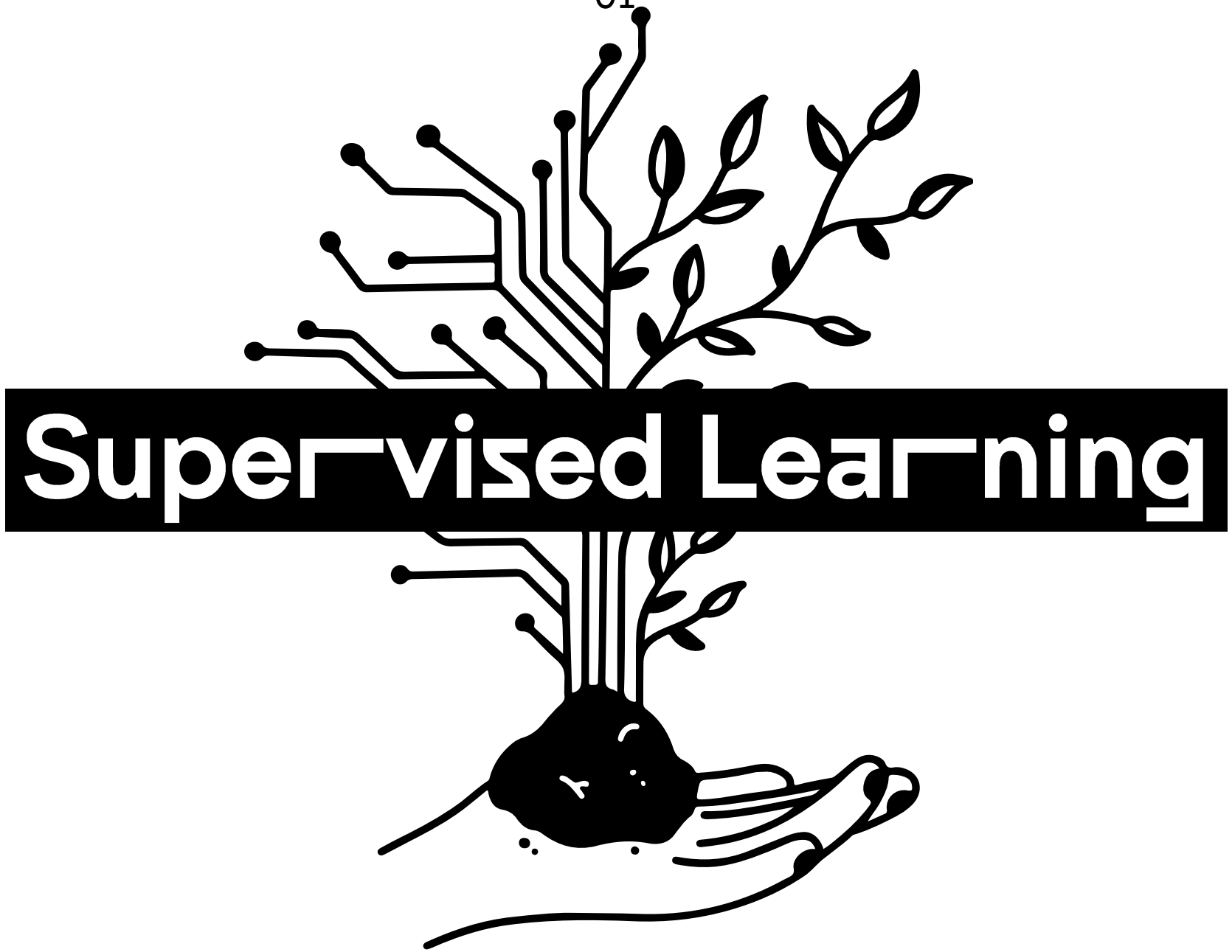
2. **conda create –n <myenv> python=3.10**

3. Entweder **pip install –r requirements.txt** oder **pip install package** (z.B. tensorflow, matplotlib, ...)

4. Aktivieren der VEnv mit : **conda activate <myenv>**

Installation VS Code

1. Setup exe herunterladen und installieren:
   https://code.visualstudio.com/
2. Python Extension installieren
3. Jupyter Notebook öffnen:
   - Ctrl+Shift+P
   - Create: New Jupyter Notebook
4. Wähle Python Interpreter deiner Conda Environment in der rechten oberen Ecke aus
5. **import tensorflow as tf** eingeben und ausführen


Supervised Learning

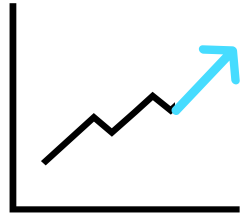
Supervised Learning

Fragestellung Supervised Learning:

z.B.: Gibt es Kunden, welche eine hohe Wahrscheinlichkeit besitzen ihren Vertrag zu kündigen?

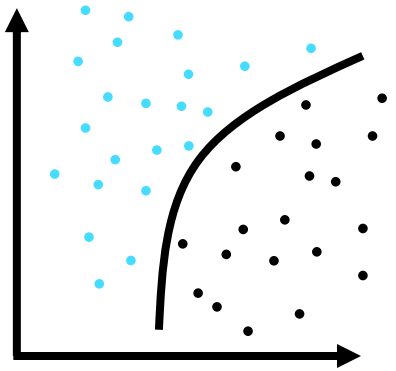
- Training erfolgt auf historischen Daten, welche alle gelabelt sind.
- Ziel ist es durch die vorliegenden Daten (Features), die Zielvariable so genau wie möglich vorhersagen zu können und ggf. eine Wahrscheinlichkeit der Klassenzugehörigkeit zu ermitteln.
- Vorhersage erfolgt mit Daten bei der die Klasse (das Target) fehlt.

Typen des Supervised Learnings



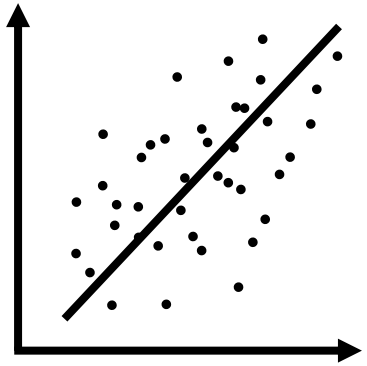
1. **Klassifikation und Wahrscheinlichkeitsabschätzung der Klassenzugehörigkeit**

- Unterteilung der Daten in Klassen z.B.: Betrug/kein Betrug, Spam/kein Spam, erkrankt/gesund
- Ziel ist es, jeden Datenpunkt einer Klasse zuzuordnen
- Zugehörigkeit zu mehreren Klassen meist ausgeschlossen
- Fragestellung: Welche Kunden haben hohes Abwanderungspotenzial?
- Vorhersagen durch Trennung des Entscheidungsraums



2. **Regression**

- Vorhersagen eines numerischen Wertes z.B.: Aktienkurs, Umsatz
- Fragestellung: Wieviel Umsatz werden wir im nächsten Jahr voraussichtlich machen?
- Aufstellung einer Funktion, welche eine Art Approximation darstellt



Provost, Foster, Tom Fawcett, Data Science für Unternehmen, 2017, S. 46-49


Bias Variance Trade-Off

Bias- Variance Tradeoff

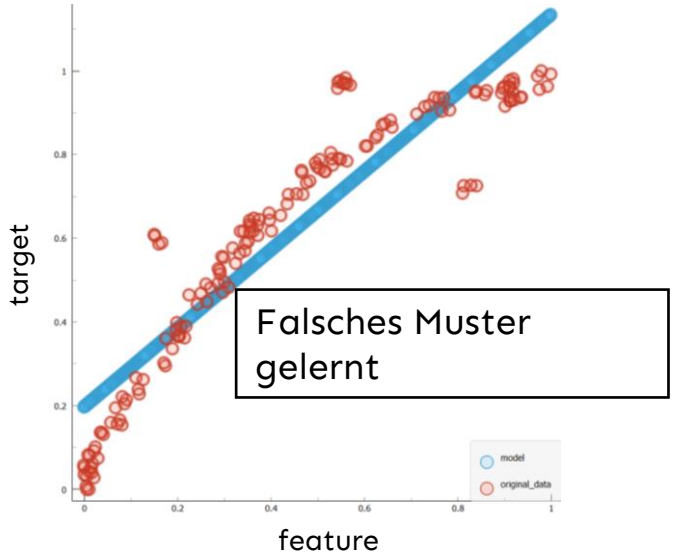
Over- und Underfitting

Grundlegende Herausforderung im Machine Learning:

Wie kann ein Modell
a. die richtigen Muster in den historischen Daten finden
b. gut performen auf neuen Daten (Generalisierung)

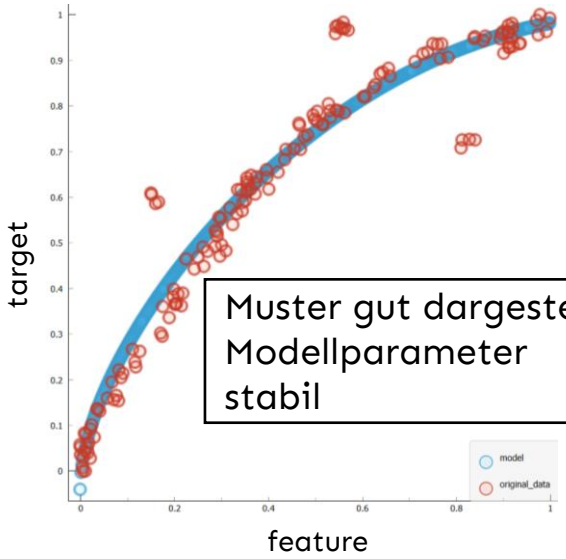
Bias-Variance Tradeoff

Linear model
(degree 1)



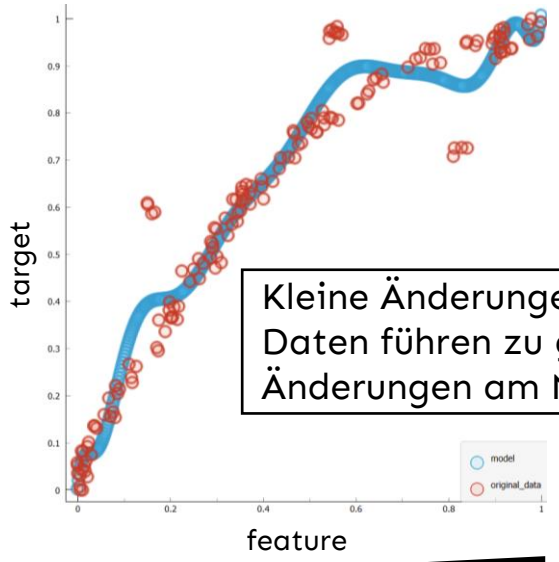
Falsches Muster gelernt

Linear model
(degree 2)



Muster gut dargestellt, Modellparameter stabil

Linear model
(degree 14)



Kleine Änderungen der Daten führen zu großen Änderungen am Modell

model complexity

UNDERFITTING = HIGH BIAS

Bias-Variance Tradeoff

OVERFITTING = HIGH VARIANCE

Prof. Dr. Küppers, MW330 – L2B – Bias Variance Tradeoff, S.15; eigene Übersetzung

Bias-Variance Tradeoff

Low Bias, Low Variance (so gut wie unmöglich):

- Modell hat informative/relevante Features
- Modellparameter sind stabil
- Modell performt gut auf unbekannten Daten (Generalisierung)

Low Bias, High Variance:

- Sprunghafte Modellparameter bei der Verwendung unterschiedlicher Trainingssets (inkonsistent)
- Modell lernt eventuell zu viel Noise oder Daten auswendig (keine Generalisierung möglich)
- Overfitting

High Bias, Low Variance:

- Vorhersagen sind konsistent, aber inakkurat im Durchschnitt
- Underfitting

High Bias, High Variance:

- Modell hat weder informative Features, noch ist es richtig dimensioniert.
- Vorhersagen sind inkonsistent und inakkurat im Durchschnitt

Source. https://medium.com/analytics-vidhya/difference-between-bias-and-variance-in-machine-learning ; eigene Übersetzung

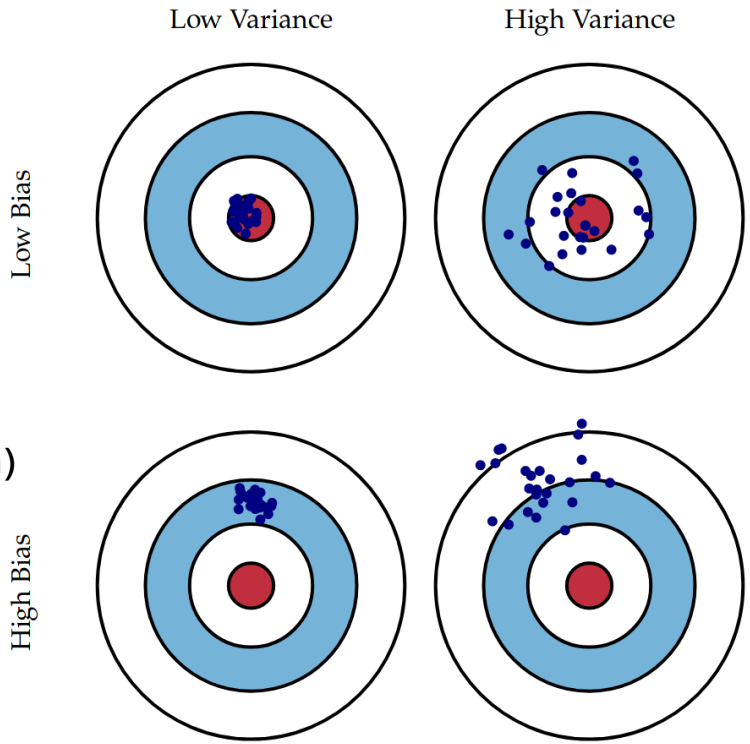


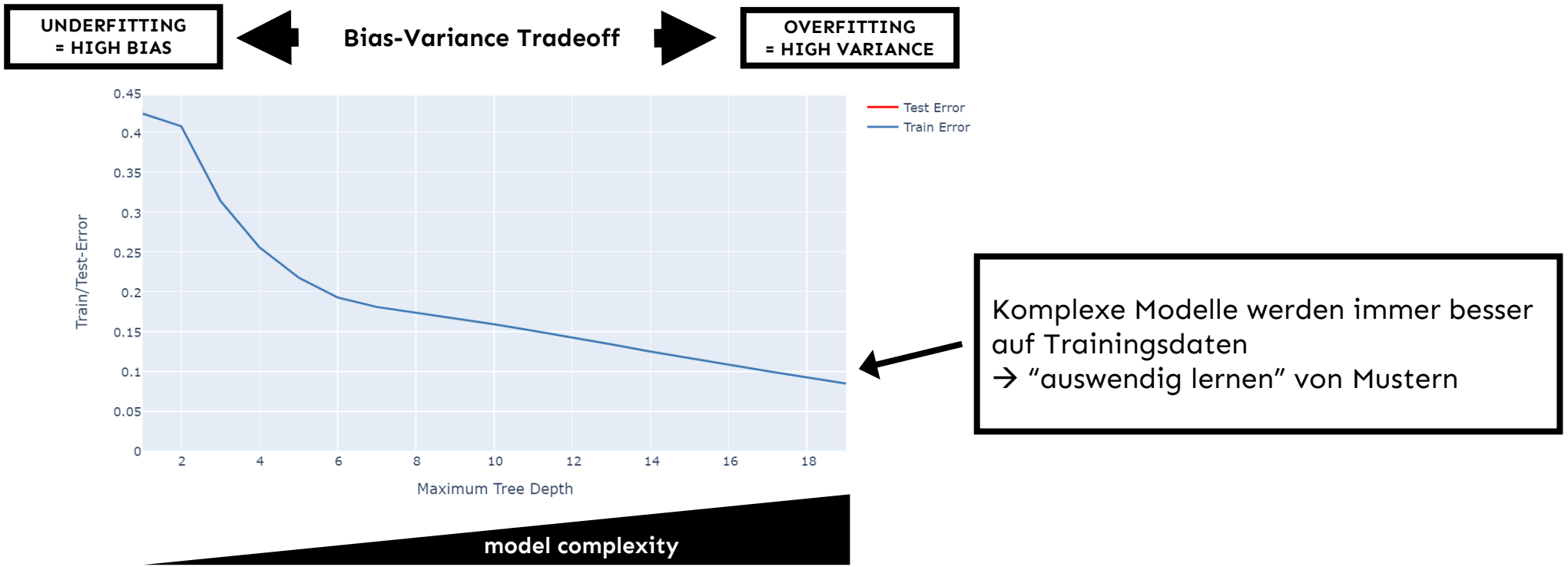
Fig. 1 Graphical illustration of bias and variance.

Sources: Fortmann-Roe, Scott. 2012. "Understanding the Bias-Variance Tradeoff."

Bias-Variance Tradeoff

Train/Test-Error

Eine numerische Betrachtung des Bias-Variance Tradeoffs - Variieren von Modellparametern & messen des Train-Test-Errors



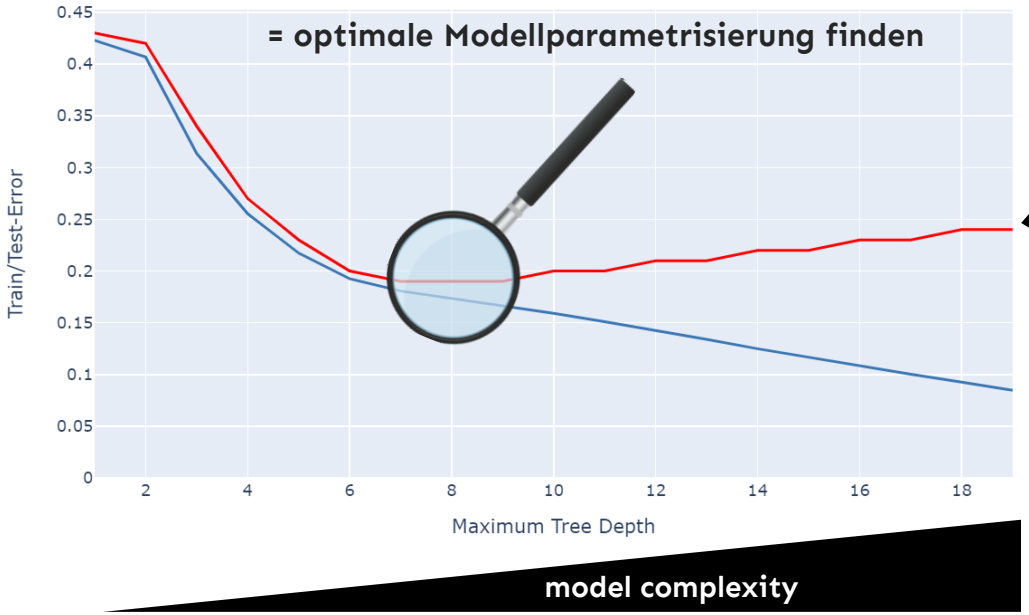
Prof. Dr. Küppers, MW330 – L2B – Bias Variance Tradeoff, S17 ; eigene Übersetzung

Bias-Variance Tradeoff

Train/Test-Error

Eine numerische Betrachtung des Bias-Variance Tradeoffs - Variieren von Modellparametern & messen des Train-Test-Errors

UNDERFITTING = HIGH BIAS ← **Bias-Variance Tradeoff** → **OVERFITTING = HIGH VARIANCE**



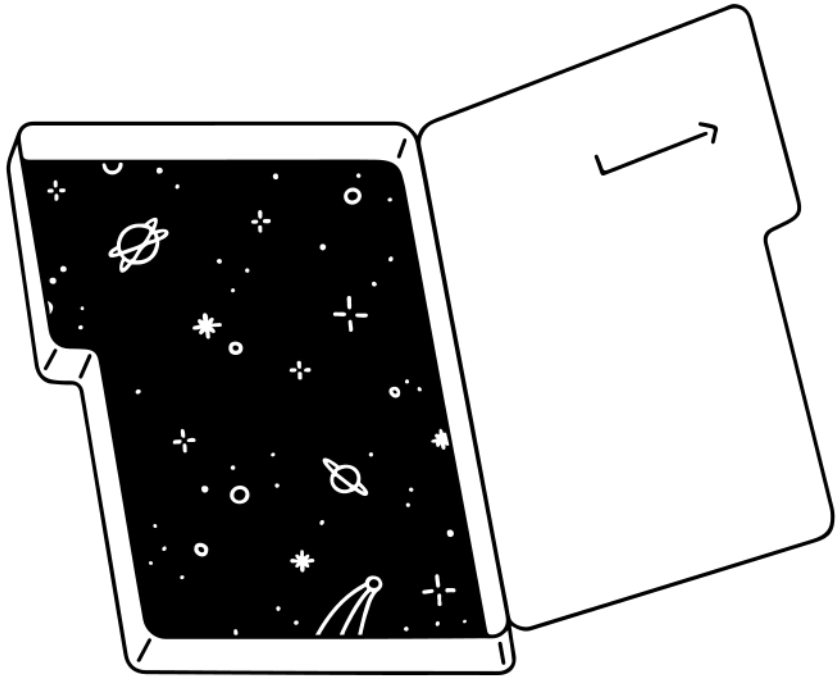
Zu komplexe Modelle performen schlechter auf Testdaten (neue Daten)

Prof. Dr. Küppers, MW330 – L2B – Bias Variance Tradeoff, S17; eigene Übersetzung

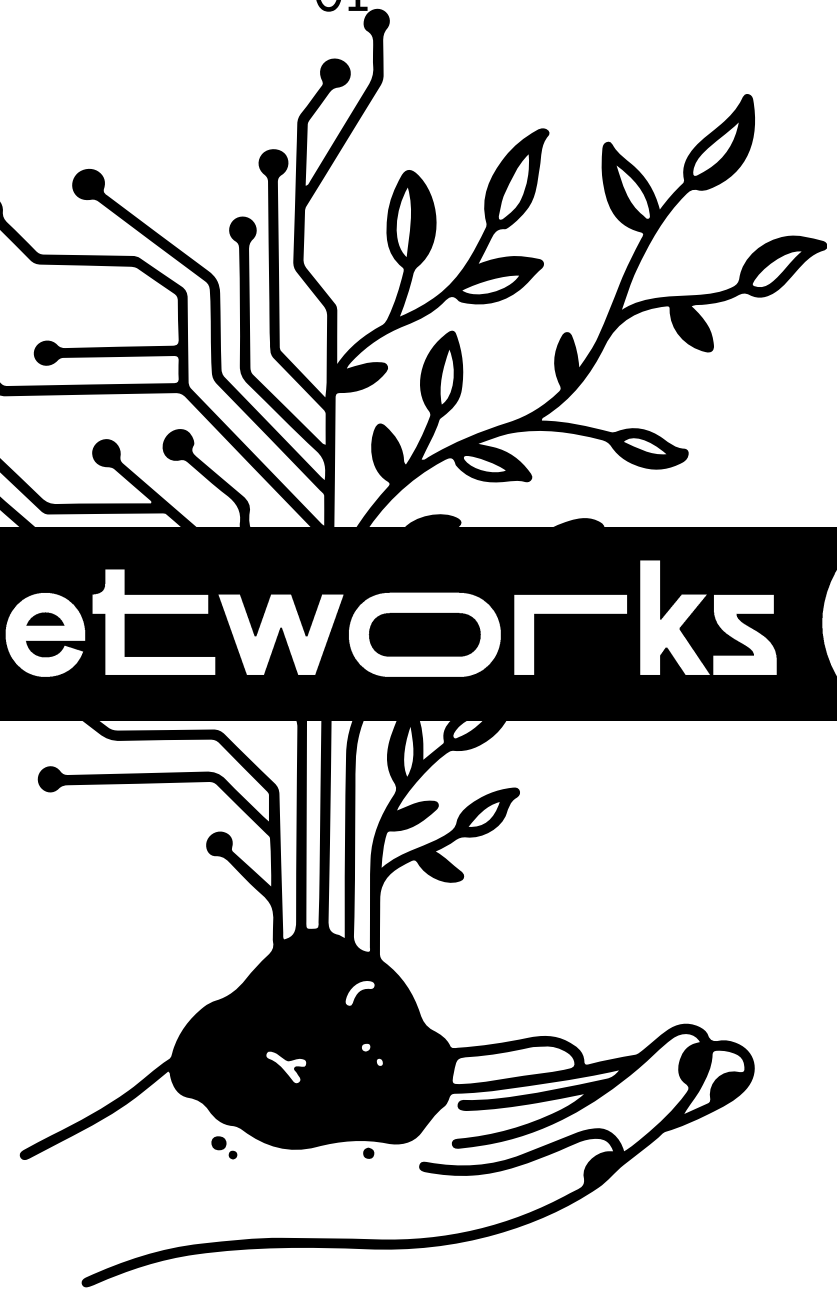
Python Basics

- Möglichkeit, Python Basiswissen aufzufrischen: **A Whirlwind Tour of Python** von Jake VanderPlas
- Wiederholung der wichtigsten Python Basics
- **Übung_00_Python_Auffrischung**

PRAXIS

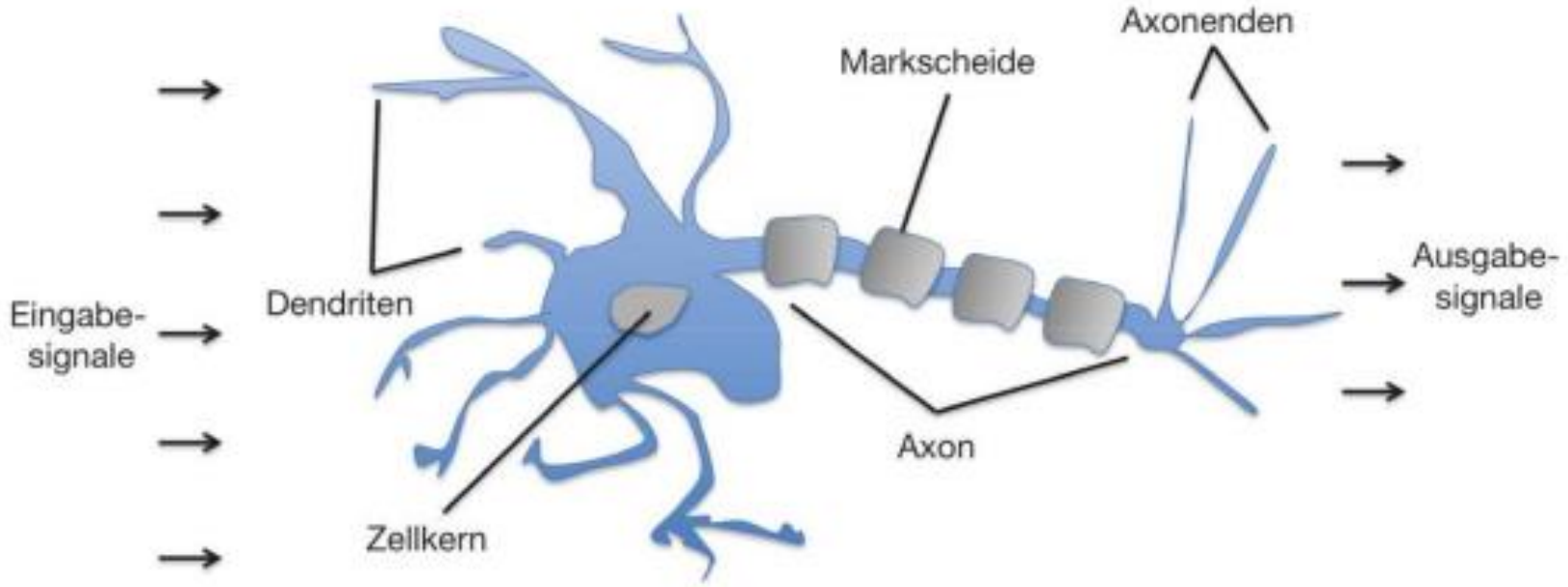


Neural Networks (MLPs)



Ursprung Neural Networks

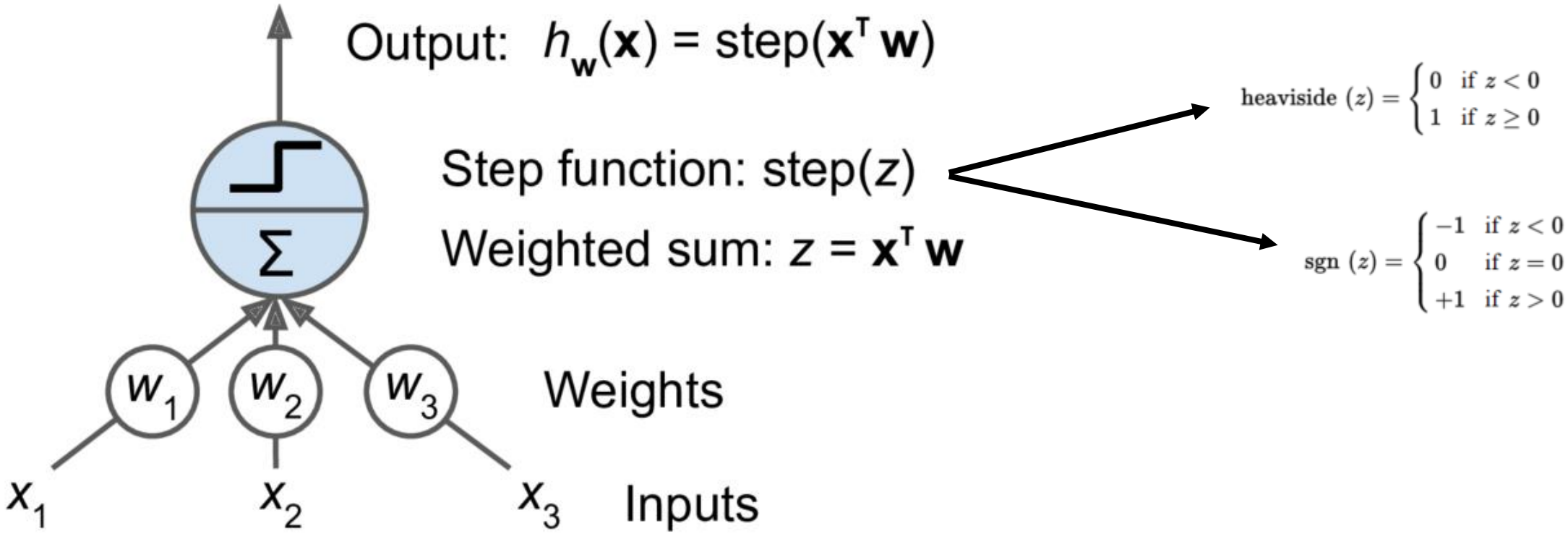
Biologisches Neuron



Cf. Raschka 2016, 38

Ursprung Neural Networks

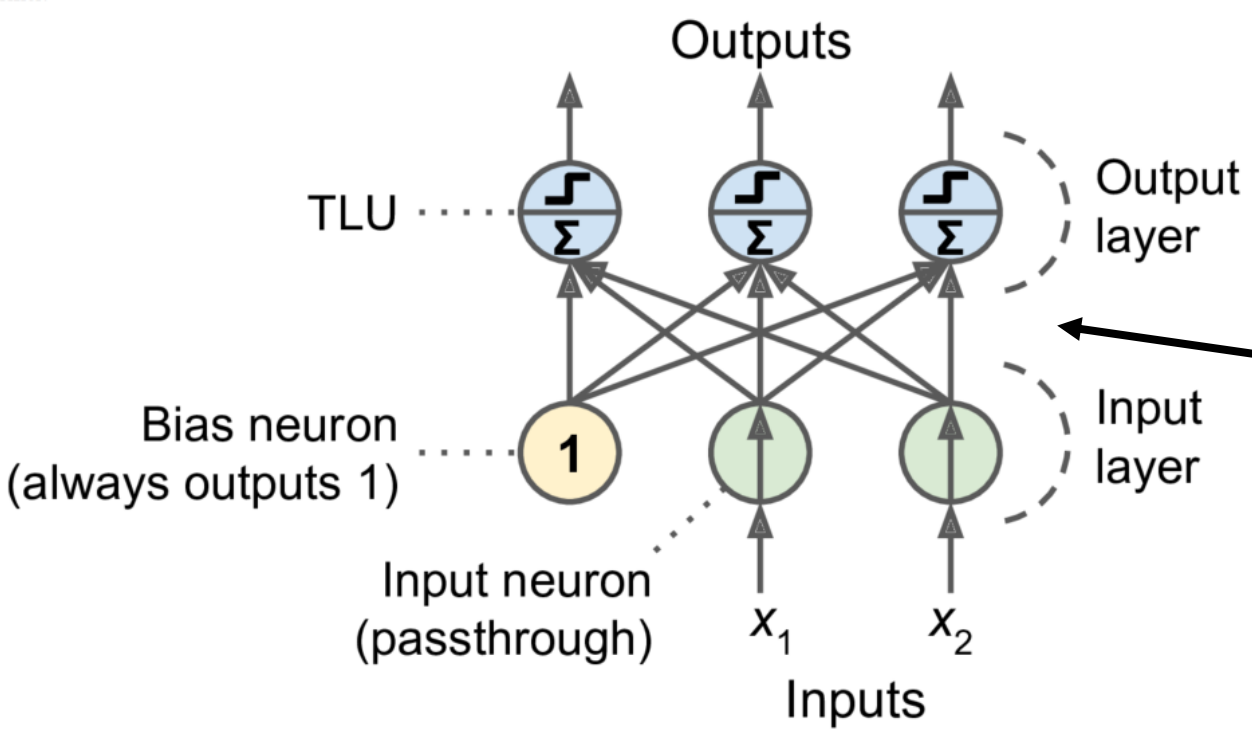
Perceptron (TLU → Threshold Logical Unit)



Géron, 2019, p.291 f.

Ursprung Neural Networks

Perceptron



Calculating the final output:

$$h_{\mathbf{W},\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{X}\mathbf{W} + \mathbf{b})$$

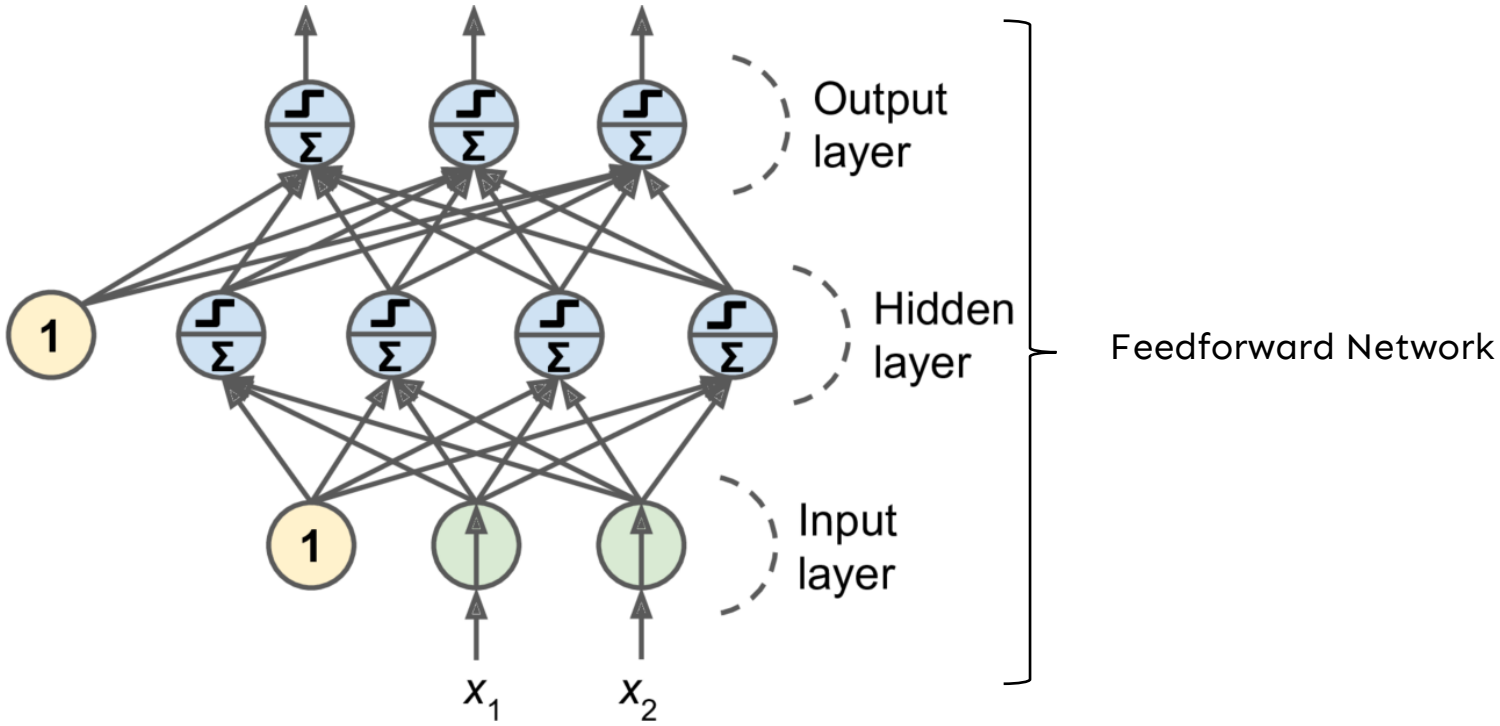
Fully Connected Layer (Dense-Layer)

Géron, 2019, p.291 f.

Neural Networks

Multi-Layer Perceptron (MLP's)

Normale Perceptrons können keine XOR Funktion abbilden, jedoch durch stapeln mehrerer Perceptrons, kann dies erreicht werden → deshalb Multi-Layer Perceptrons

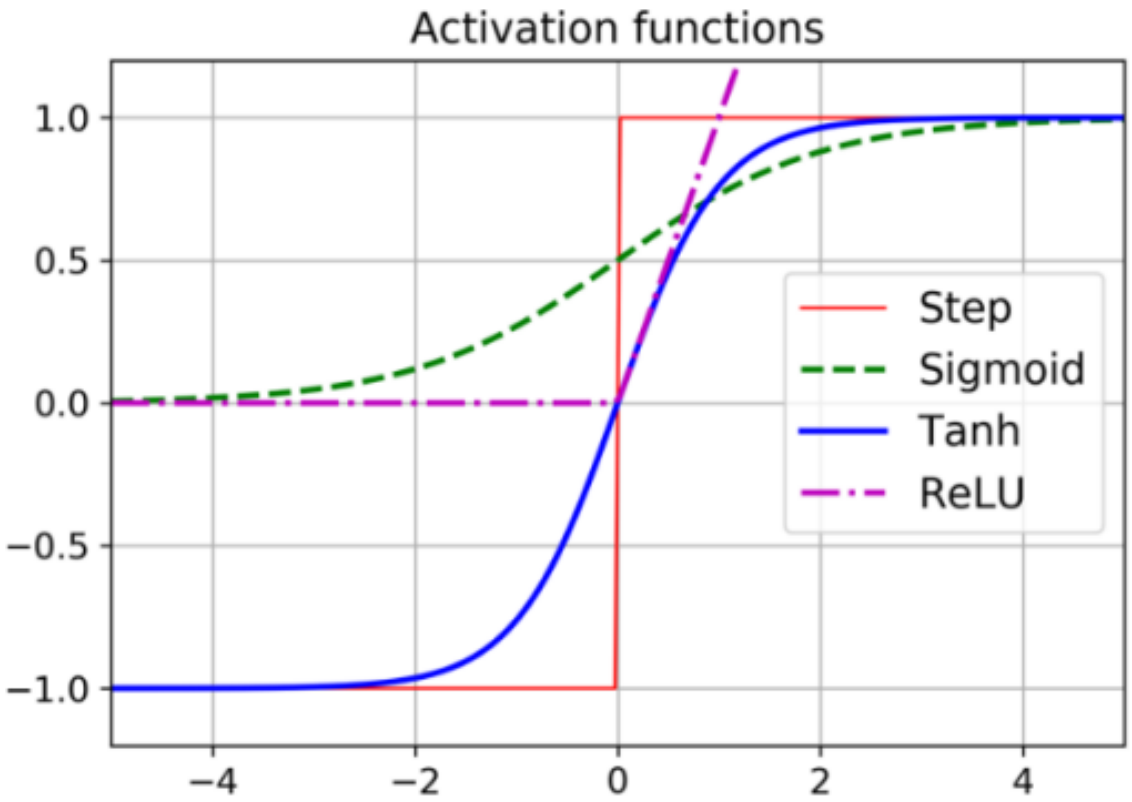


Géron, 2019, p.291 f.

Neural Networks

Aktivierungsfunktionen

- Tensorflow-Keras bietet verschiedene Arten von Aktivierungsfunktionen an, welche auf den Output von Neuronen angewandt werden
- Überblick: https://keras.io/api/layers/activations/#available-activations
- Wir werden vor allem ReLU benutzen (Rectified Linear Unit), die meistgenutzte Aktivierungsfunktion



Neural Networks

Backpropagation of Error – How neural networks learn

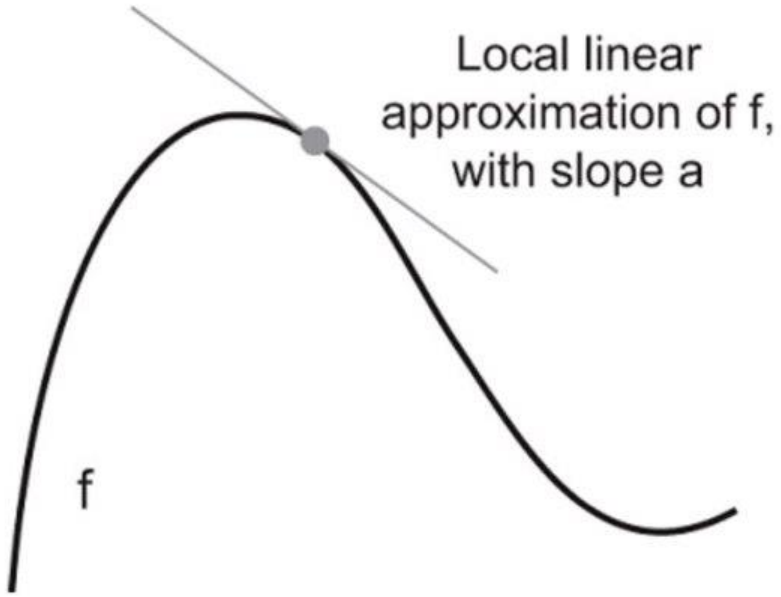
- Basierend auf Paper von Rumelhart, Hinton & Williams über backpropagation training algorithm 1986
- Benötigt zwei Schritte: forward pass & reverse pass
- Ist in der Lage den Gradient des „Network errors“ in Bezug auf jeden einzelnen Modellparameter zu berechnen → ermöglicht es herauszufinden wie die Weights & Biases angepasst werden müssen um den Error zu minimieren
- Sobald der Gradient berechnet ist, kann ein Gradient Descent Step durchgeführt werden

Gute Zusammenfassung von Géron:
„...for each training instance, the backpropagation algorithm first makes a prediction (forward pass) and measures the error, then goes through each layer in reverse to measure the error contribution from each connection (reverse pass), and finally tweaks the connection weights to reduce the error (Gradient Descent step).”

Géron, 2019, p.291 f.

Gradientenabstieg

Die Ableitung einer Funktion sagt uns in welche Richtung sie lokal kleinere Werte hat.
Im mehrdimensionalen nennt man die Ableitung Gradient.
Der Gradient zeigt in Richtung der größten Steigung.



Gradient: $\mathbf{g} = \nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$

Abstieg: $\mathbf{x}_{\text{new}} = \mathbf{x} - \epsilon \mathbf{g}$

ϵ : Learning rate (Parameter)

Goebel, M. GfU-Schulung Deep Learning, p.39

Gradientenabstieg

Learning Rate

Zu kleine Learning Rate

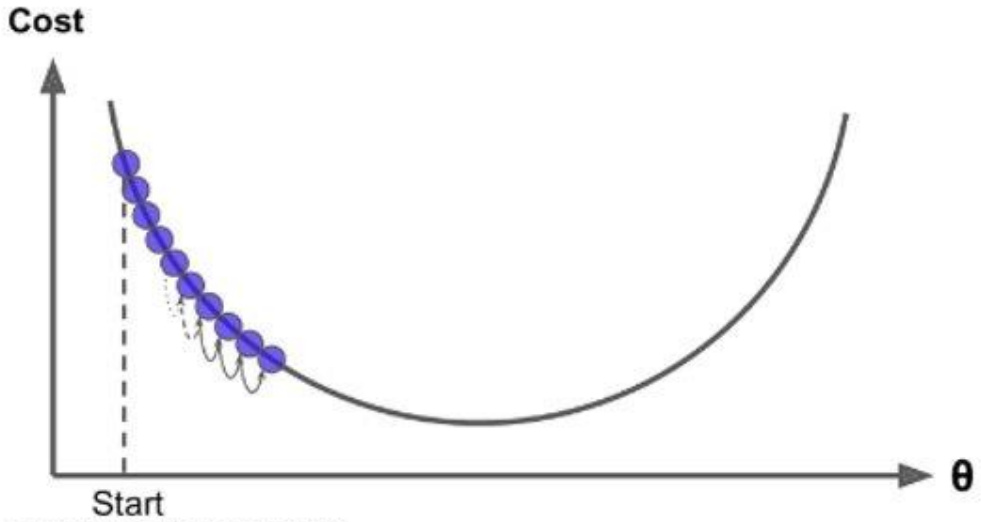


Figure 4-4. Learning rate too small

Zu große Learning Rate

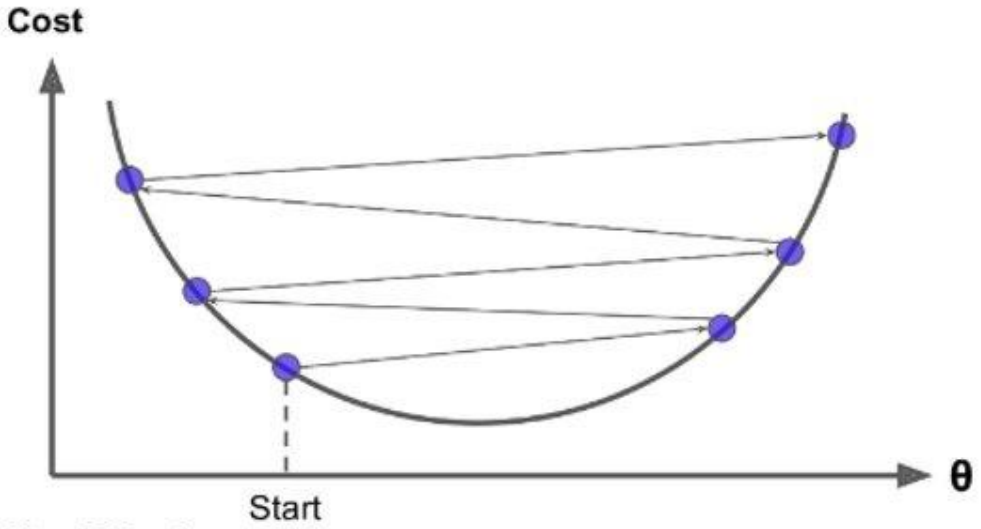


Figure 4-5. Learning rate too large

Goebel, M., GfU-Schulung Deep Learning, p.41

Tensorflow Playground

Experimentieren mit unterschiedlichen Neural Network Parametern

https://playground.tensorflow.org

Grundlagen TensorFlow

Was ist ein Tensor?

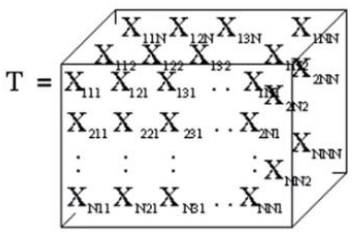
In diesem Kontext, ein multi-dimensionaler Array aus Zahlen

Rank 0: Einfaches Skalar (1,2,3,1/2, π)

Rank 1: Vektor → $\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$

Rank 2: Matrizen → $\begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nm} \end{bmatrix}$

Rank3: 3D-Array →



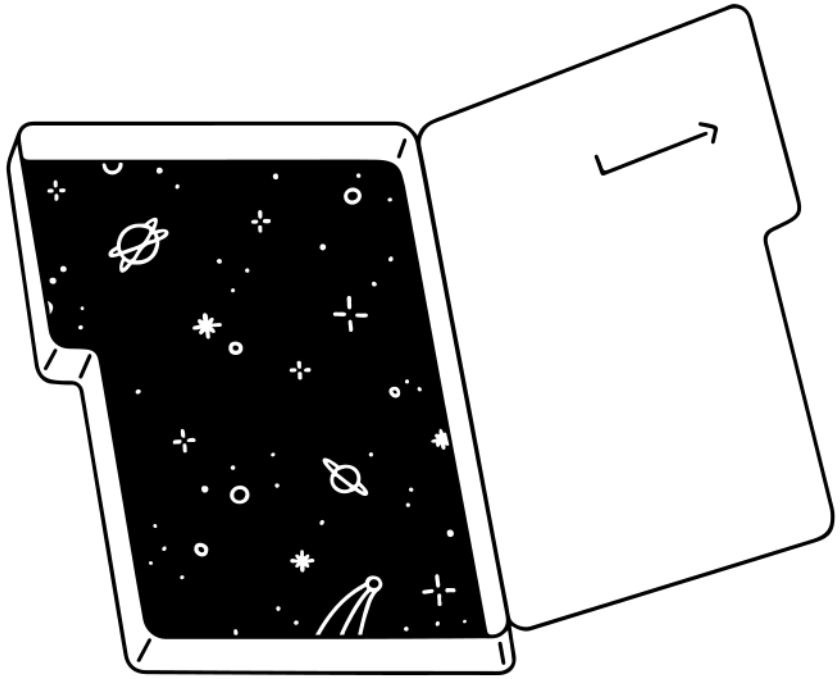
https://www.youtube.com/watch?v=TvxmkZmBa-k

Numpy/Tensorflow Basics

- Vorschau:
  01_Einführung_tensorflow_numpy

- **Übung_01_Einführung_tensorflow_ numpy**

PRAXIS



Hyperparameter - Regression

Output Neurons

- 1 Output Neuron falls lediglich 1 numerischer Wert vorhergesagt werden soll
- 2 Output Neuronen für bspw. Koordinaten (1 Neuron pro Output Dimension) → exakte Position in einem Bild
- 4 Output Neuronen für Bounding Box: exakte Position + Höhe und Breite der Box (Object Detection)

Géron, A., Hands-On Machine Learning, 2019, p. 292 ff.

Hyperparameter - Regression

Output Activation

- „linear“ → Wert wird unverändert weitergegeben
- „ReLU“ → Nur Werte >= 0 werden ausgegeben
- Logistic function → range 0 bis 1 (Min-Max Skalierung des Labels nötig)
- hyperbolic tangent → range -1 bis 1 (Labels müssen dementsprechend skaliert werden)

Géron, A., Hands-On Machine Learning, 2019, p. 292 ff.

Hyperparameter - Regression

Loss

$$\mathbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

$$\mathbf{MAE} = \frac{1}{n}\sum_{i=1}^{n}|x_i - \hat{x}_i|$$

Hyperparameter - Klassifikation

Output Neurons + Activation functions

- Binary Classification:
  - 1 Neuron für Binary Classification: Spam or no Spam
  - Logistic Activation Function

- Multilabel Binary Classification:
  - 1 Neuron per Label, bspw. 1 Neuron für Spam or no Spam und 1 Neuron für urgent or not urgent (Achtung nicht exklusiv → Prediction könnte Spam + urgent sein)
  - Logistic Activation Function

- Multiclass Prediction (exklusiv):
  - 1 Neuron per Class
  - Softmax Activation Function (Wahrscheinlichkeitsverteilung)
    → Werte normiert zwischen 0-1 und addieren sich zu 1 auf

Géron, A., Hands-On Machine Learning, 2019, p. 292 ff.

Hyperparameter - Klassifikation

Loss

- Categorical Crossentropy (One-Hot kodiert)

```
>>> y_true = [[0, 1, 0], [0, 0, 1]]
>>> y_pred = [[0.05, 0.95, 0], [0.1, 0.8, 0.1]]
```

- Sparse categorical crossentropy

```
>>> y_true = [1, 2]
>>> y_pred = [[0.05, 0.95, 0], [0.1, 0.8, 0.1]]
```

Géron, A., Hands-On Machine Learning, 2019, p. 292 ff.

Merktabellen

Regression

Hyperparameter	Typical value
# input neurons	One per input feature (e.g., 28 x 28 = 784 for MNIST)
# hidden layers	Depends on the problem, but typically 1 to 5
# neurons per hidden layer	Depends on the problem, but typically 10 to 100
# output neurons	1 per prediction dimension
Hidden activation	ReLU (or SELU, see Chapter 11)
Output activation	None, or ReLU/softplus (if positive outputs) or logistic/tanh (if bounded outputs)
Loss function	MSE or MAE/Huber (if outliers)

Géron, A., Hands-On Machine Learning, 2019, p. 293

Merktabellen

Klassifikation

Hyperparameter	Binary classification	Multilabel binary classification	Multiclass classification
Input and hidden layers	Same as regression	Same as regression	Same as regression
# output neurons	1	1 per label	1 per class
Output layer activation	Logistic	Logistic	Softmax
Loss function	Cross entropy	Cross entropy	Cross entropy

Géron, A., Hands-On Machine Learning, 2019, p. 294

Einführung tensorflow Keras

Sequential API

- Modellbildung anhand von Layer-Architekturen (keras.layers)
- Hinzufügen diverser Layer durch model.add() oder Sequential-Array
- Überblick über alle verfügbaren Layer: https://keras.io/api/layers/
- Outputs werden wie Name schon sagt sequentiell von Layer zu Layer weitergereicht
- Falls Individualisierung erforderlich Functional API nötig
  → Outputs und Inputs können individuell angepasst werden

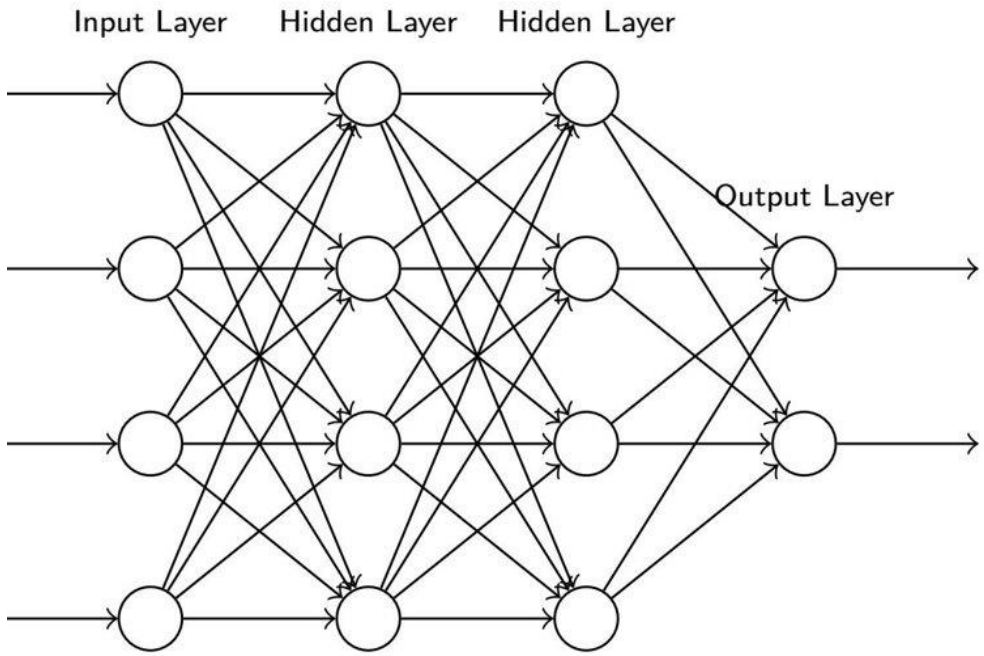
Einführung tensorflow Keras

Sequential API

Code

```
model = keras.models.Sequential([
    keras.layers.Flatten(input_shape=[4]),
    keras.layers.Dense(4, activation="relu"),
    keras.layers.Dense(4, activation="relu"),
    keras.layers.Dense(2, activation='softmax')
])
```

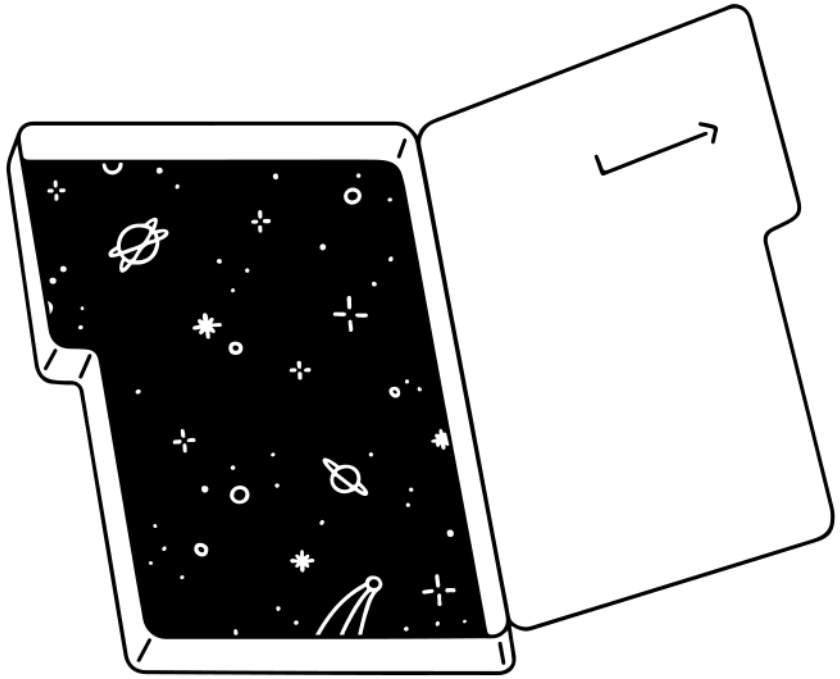
Repräsentation

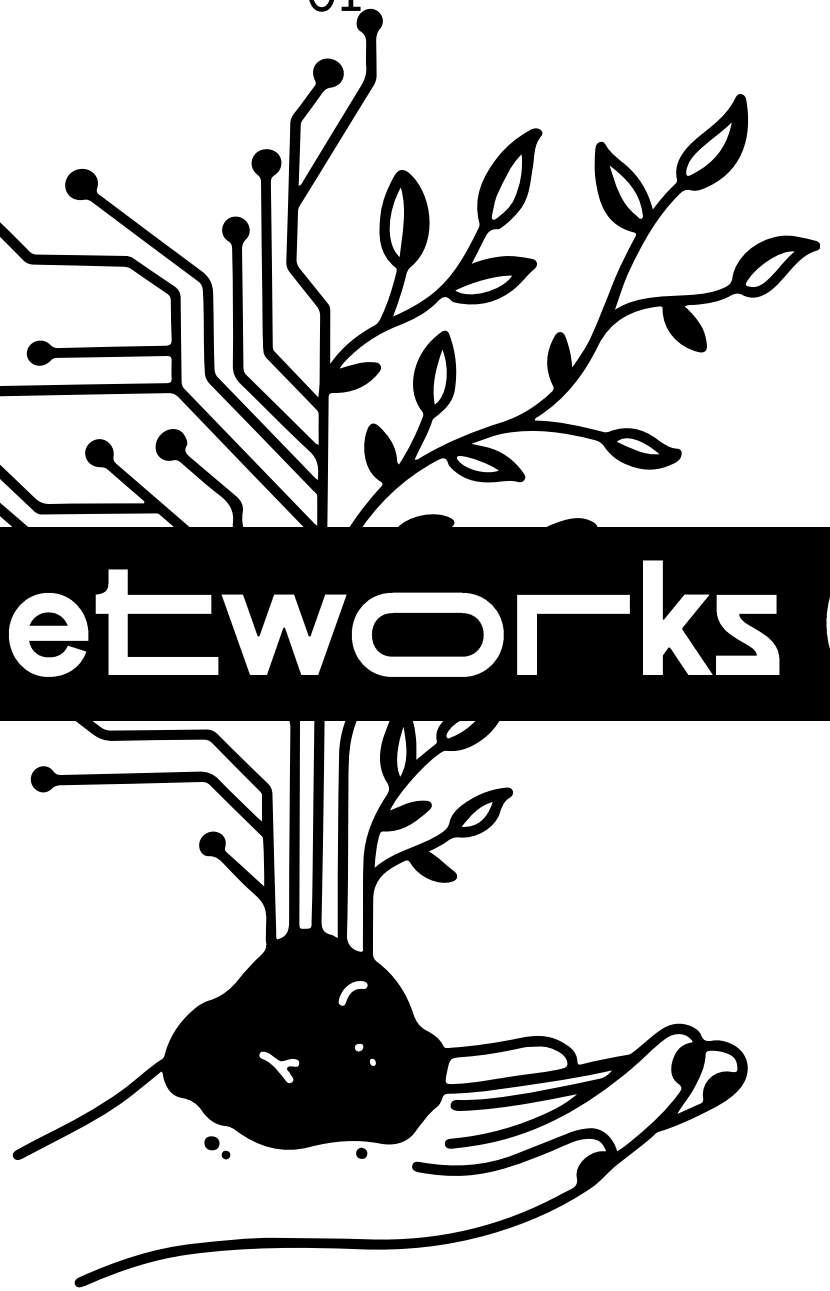


Regression & Klassifikation mit MLPs

- Vorschau:
  02_Supervised_Learning_Regression_Classification
- **Übung_02_Regression**
- **Übung_03_Klassifikation**

PRAXIS





Neural Networks (CNNs)

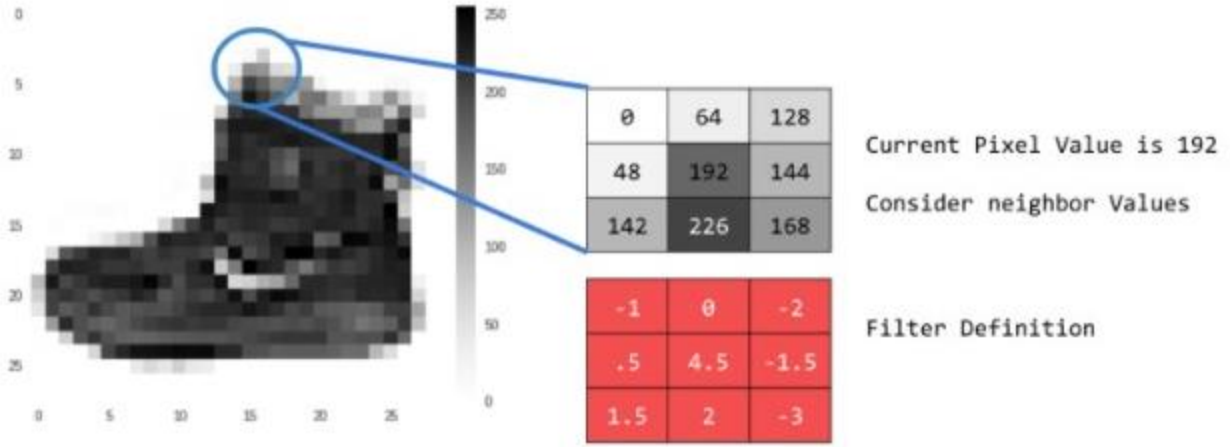
Convolutional Neural Networks (CNNs)

Grundidee

- Filter, welcher über Bild fährt und daraus wichtige Features extrahiert
  → Feature Mapping

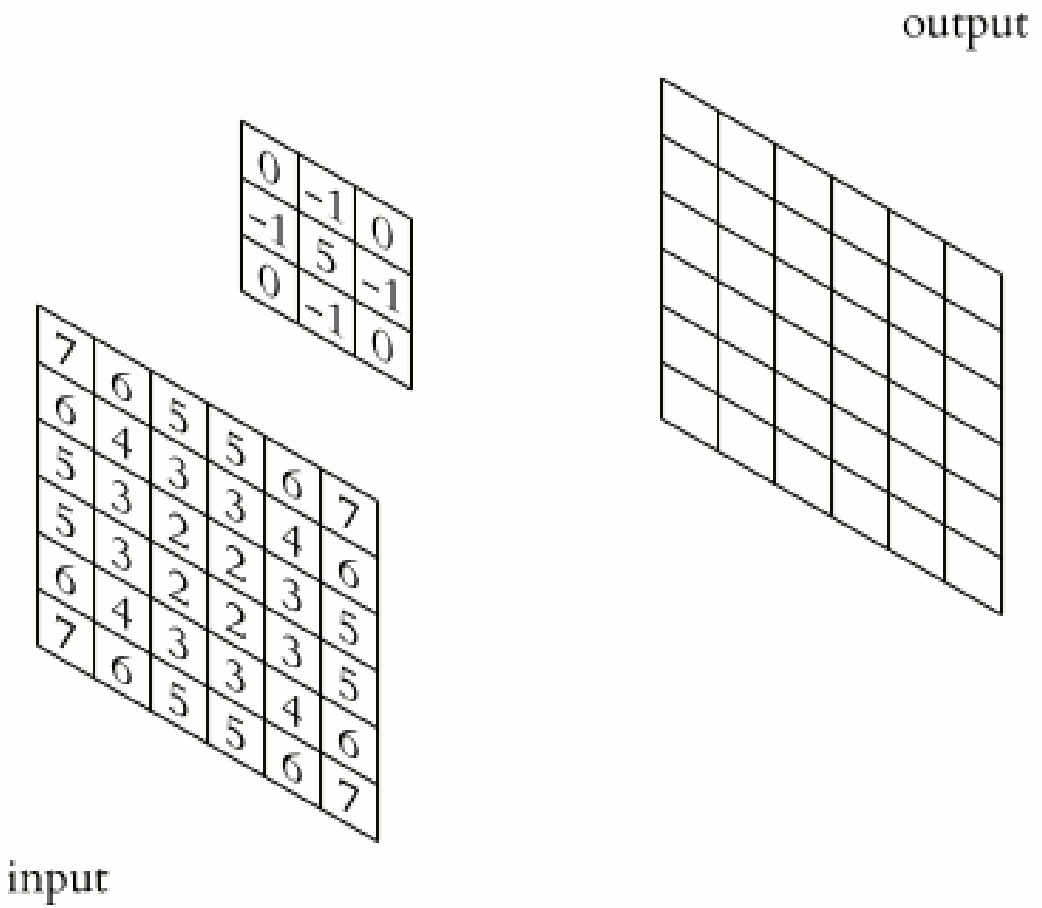
- Jeder Pixelwert in einem Bild wird gescannt und dabei auf dessen „Nachbarwerte“ geachtet

- Werte werden mit Filter/Kernel multipliziert und zu einem neuen Pixelwert aufsummiert (gewichtete Summe).



Convolutional Neural Networks (CNNs)

Convolutional-Operation



Convolutional Neural Networks (CNNs)

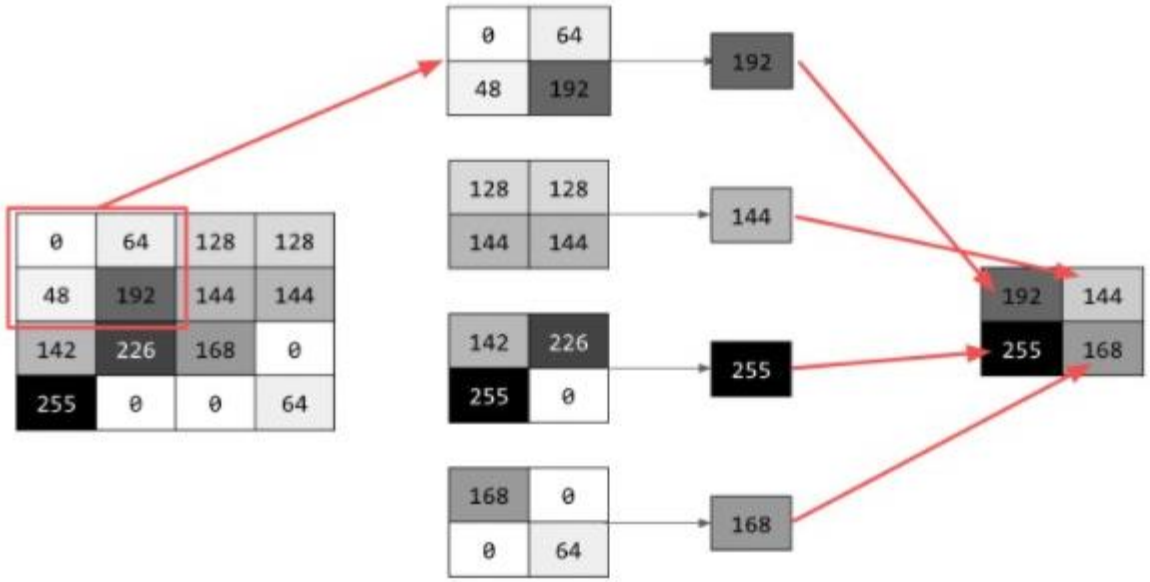
Pooling-Layer

- Nachdem alle essenziellen Features eines Bildes extrahiert wurden, wird Pooling durchgeführt
- Informationen des Bildes werden komprimiert und auf die „wichtigen“ Features reduziert
- Es gibt mehrere Arten von Pooling, besonders beliebt ist Maximum oder Average Pooling
- Arbeitet wie Convolutions auch mit einer Filter Size

Convolutional Neural Networks (CNNs)

Pooling-Layer

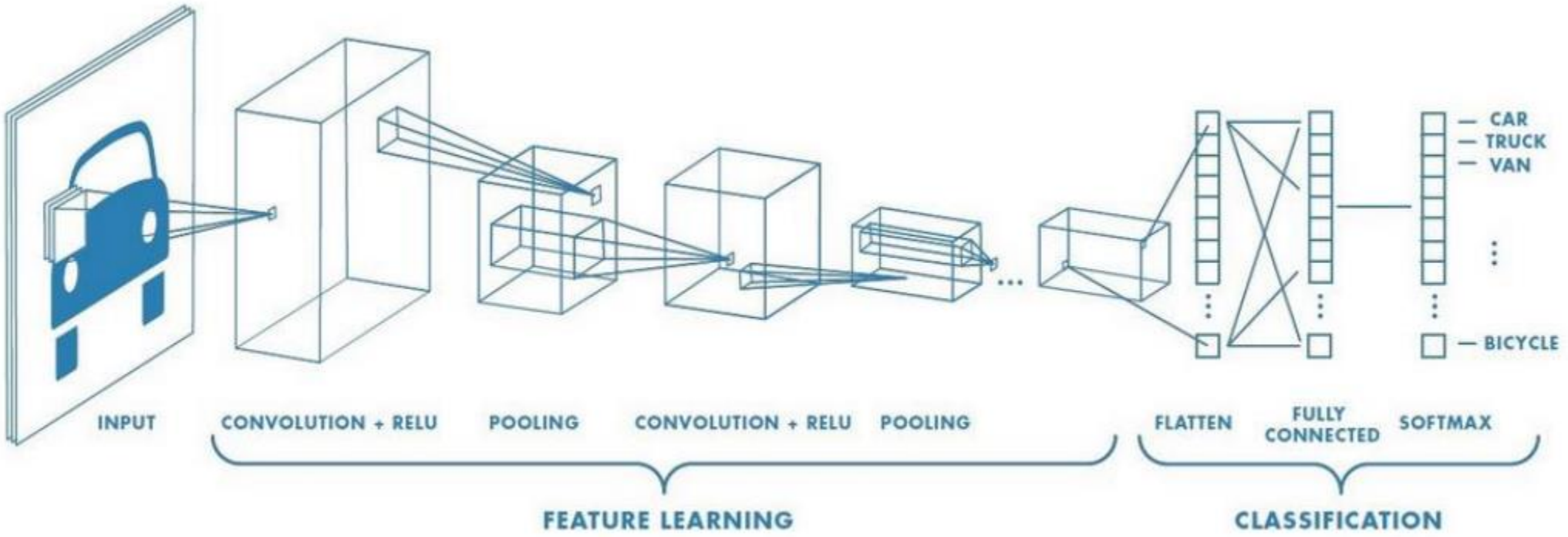
Max-Pooling mit einem (2,2) Filter:



$\frac{1}{4}$ x Size des ursprünglichen Bildes

Convolutional Neural Networks (CNNs)

Architektur



CNNs

- Vorschau: **03_MLP's vs CNNs**
- **Übung_04_CNNs**

PRAXIS