

› AI FOR BUSINESS APPLICATIONS

HILBERT, S., KRAUS, E., & LINDL, A. (2025). *Machine Learning: Eine Einführung für Psychologie, Geistes- und Sozialwissenschaften* (Quantitative Sozialforschung). Wiesbaden: Springer Fachmedien Wiesbaden.

KOCH, D., KOHNE, A., & BRECHBÜHLER, N. (2025). *Prompt Engineering im Unternehmen – eine Einführung*. Wiesbaden: Springer Gabler

Grafiken wurden teilweise mit LLMs erstellt.

- > Keine Open-Book-Klausur
- > Aber: Selbstgeschriebenes Cheat Sheet (1 Blatt, Vorder- + Rückseite)

1. Einleitung
2. Prompt Engineering
3. Maschinelles Lernen I (Supervised Learning)
4. Maschinelles Lernen II (Unsupervised Learning)
5. Maschinelles Lernen III (Reinforcement Learning)
6. Implementierung von KI-Lösungen im Unternehmen

› 1. EINLEITUNG

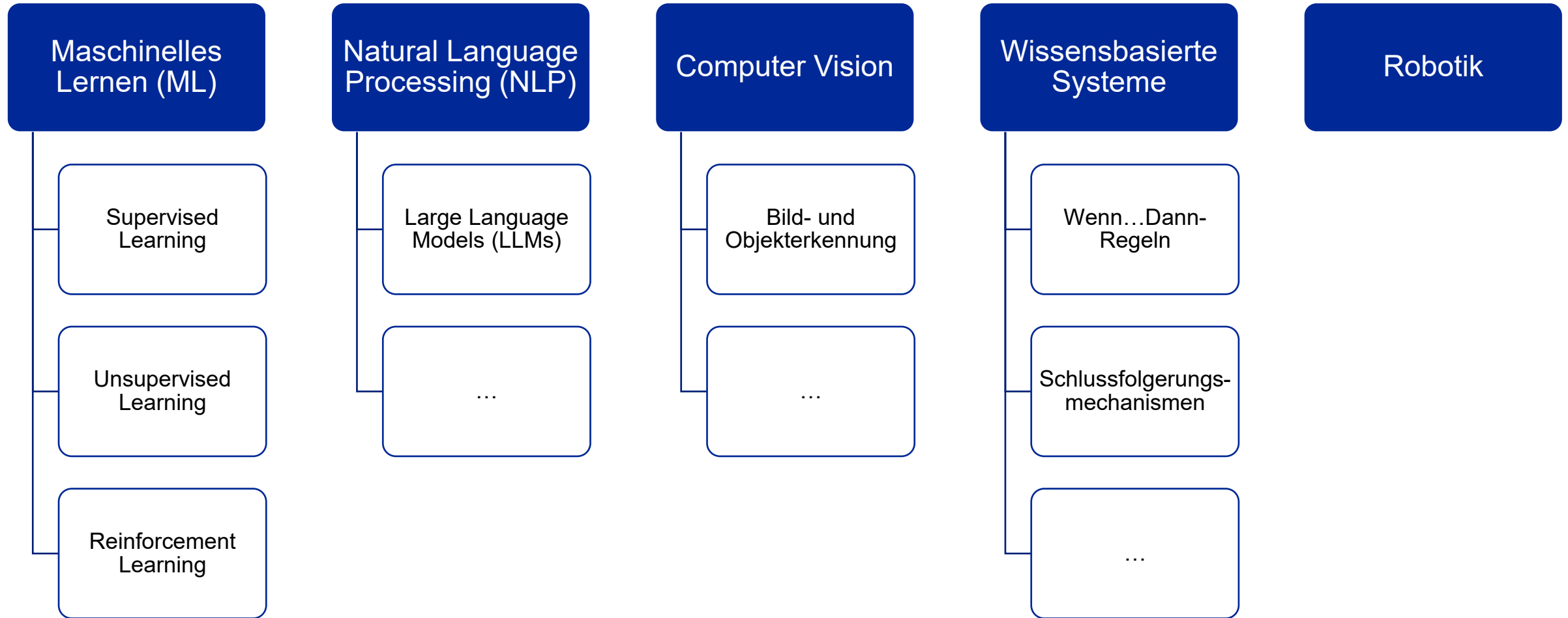
- > „Künstliche Intelligenz ist die Wissenschaft und Technik, intelligente Maschinen zu bauen, insbesondere intelligente Computerprogramme.“
(McCarthy, J., Minsky, M., Rochester, N., & Shannon, C. (1955). *A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence*)

- > „Künstliche Intelligenz ist die Fähigkeit einer Maschine, menschliche Fähigkeiten wie logisches Denken, Lernen, Planen und Kreativität zu imitieren.“
(<https://www.europarl.europa.eu/topics/de/article/20200827STO85804/was-ist-kunstliche-intelligenz-und-wie-wird-sie-genutzt>)

- > „KI ist die Entwicklung von Agenten, die ihre Umgebung wahrnehmen und Handlungen ausführen, um ihre Ziele zu erreichen.“
(Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach*.)

KÜNSTLICHE INTELLIGENZ

TEILBEREICHE



Installieren Sie Anaconda für die Nutzung von Python:

<https://www.anaconda.com/download>

› 2. PROMPT ENGINEERING

PROMPT ENGINEERING

LLMS: ABGRENZUNG VON GEN-AI

Generative Künstliche Intelligenz (GenAI):

- > lernt auf Basis von Trainingsdaten statistische Muster und
- > erzeugt daraus neue Inhalte

Beispiele:

- > DALL-E (text-to-image)
Link: <https://openai.com/de-DE/index/dall-e-3/>
- > ChatGPT (text-to-text)
Link: <https://chatgpt.com/>
- > MusicLM (text-to-music)
Link: <https://musiclm.com/>

PROMPT ENGINEERING

LLMS: ABGRENZUNG VON GEN-AI

Large Language Models (LLMs):

- > spezifische Unterkategorie der generativen KI
- > große neurale Netze mit Milliarden von Parametern (trainierbare Werte)
- > reagieren auf *Prompts* (= Eingabeaufforderungen)
- > Text fortsetzen, Fragen beantworten, Übersetzen, Zusammenfassen, Programmieren,...

PROMPT ENGINEERING

LLMS: ABGRENZUNG VON GEN-AI

Beispiele *Large Language Models* (LLMs):

> ChatGPT (OpenAI)

> Claude (Anthropic)

Link: <https://claude.ai/new>

> Gemini (Google)

Link: <https://gemini.google.com/app?hl=de>

> Mistral (Open Source)

Link: <https://mistral.ai/>

> LLaMA (Meta)

Link: <https://www.llama.com/>

Hauptkomponenten LLM:

- > Tokenisierung
- > Transformer-Architektur

Siehe:

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention Is All You Need*. Advances in Neural Information Processing Systems.

Link: https://papers.nips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Bevor ein Transformer Text verarbeiten kann, muss er den Text in kleine Einheiten zerlegen.

- > Diese Einheiten heißen *Tokens*.
- > Nicht unbedingt Wörter, sondern Wortteile, Silben oder sogar einzelne Zeichen
- > Beispiel (vereinfachtes Byte-Pair-Encoding-Verfahren):
 - Satz: „*Chatbots sind hilfreich.*“
 - Tokens: ["Chat", "bot", "s", "sind", "hilf", "reich", "."]

Tokenisierung = Übersetzung von Sprache in maschinenlesbare Bausteine.

1. Eingabe (Tokens → Vektoren)

- > Jedes Token wird in einen Vektor eingebettet (sogenanntes Embedding).
- > Zusätzlich: Positionskodierung → Modell weiß, in welcher Reihenfolge Wörter stehen.

2. Self-Attention (Aufmerksamkeitsmechanismus)

- > Bei jedem Wort Berücksichtigung alle anderen im Satz, um Relevanz zu bestimmen
- > Beispiel:
In „*Der Hund jagt die Katze, weil sie schnell ist*“ muss das Modell erkennen, dass „sie“ sich auf „Katze“ bezieht – dank Attention klappt das.

3. Feedforward-Netzwerk

- > Nach der Attention-Layer:

Vektoren durchlaufen kleine neuronale Netze, die Muster verstärken oder abschwächen.

4. Mehrere Schichten (Stacking)

- > Transformer bestehen aus zahlreichen solcher Attention- + Feedforward-Blöcken.

- > Je tiefer (mehr), desto „intelligenter“ die Repräsentation von Sprache.

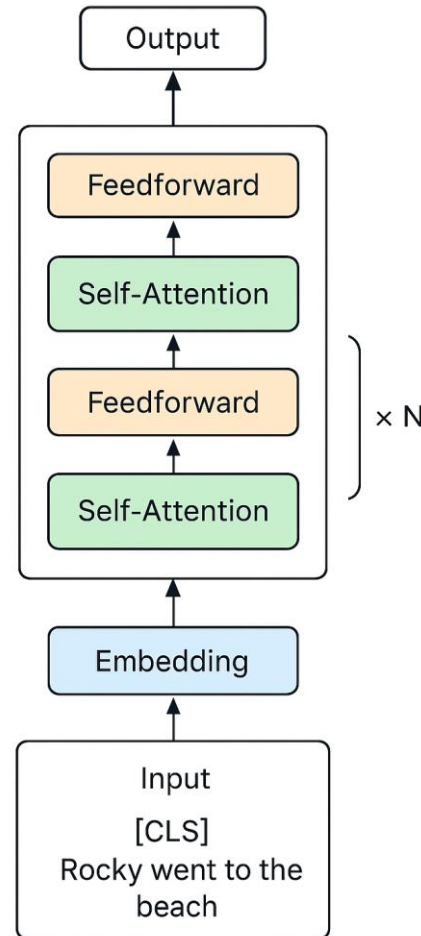
5. Ausgabe (Decoding)

- > Modell berechnet, welches Token am wahrscheinlichsten als Nächstes kommt.

- > Beispiel: Prompt „Die Sonne geht im ____ auf“ → Wahrscheinlichstes Token: „Osten“.

PROMPT ENGINEERING

LLMS: TRANSFORMER-ARCHITEKTUR



PROMPT ENGINEERING

LLMS: STÄRKEN & SCHWÄCHEN

✓ Stärken von LLMs

- > Breites Wissen und Vielseitigkeit
- > Sprachverständnis & Kontextverarbeitung
- > Kreativität & Ideenfindung
- > Anpassbarkeit
- > Produktivitätssteigerung

✗ Schwächen von LLMs

- > Halluzinationen (Faktenfehler)
- > Begrenztes Kontextfenster
- > Bias & Ethikprobleme
- > Keine echte Logik oder Weltmodell
- > Kosten & Ressourcen
- > Abhängigkeit von Datenqualität
- > Pfadabhängigkeit

1. Klarheit & Präzision
2. Kontextbereitstellung
3. Rollen- und Stilanweisung [optional]
4. Formatvorgabe
5. Iteratives Prompting
6. Begrenzung
7. Explizite Aufgabenstellung

Grundlegende Prompt-Techniken

1. Zero-Shot Prompting:
Frage oder Aufgabe, ohne Beispiel oder zusätzliche Anweisung.
2. One-Shot Prompting:
Beispiel als Orientierung, bevor die eigentliche Aufgabe gestellt wird
3. Few-Shot Prompting:
Mehrere Beispiele, um das gewünschte Muster, Format oder den Stil zu verdeutlichen.

Präzisierende Prompt-Techniken

4. Chain-of-Thought Prompting (CoT):
Modell wird aufgefordert, den Denkprozess Schritt für Schritt zu erklären.
5. Self-Consistency Prompting:
Modell soll mehrere Lösungswege generieren und konsistenteste Antwort wird gewählt.
6. ReAct Prompting (Reason + Act):
Kombination aus Schlussfolgern (Reasoning) und Handeln (z. B. Informationen nachschlagen, Tools nutzen).

Formatierungs- und Strukturtechniken

7. Instruction Prompting:
Sehr klare Anweisungen
8. Role Prompting:
Du setzt das Modell in eine Rolle
9. Persona / Style Prompting:
Vorgabe eines Stils oder Charakter
10. Output-Formatting:
Du definierst das gewünschte Format (z. B. JSON, Tabelle, Markdown).

Fortgeschrittene Techniken

11. Prompt Chaining:

Komplexe Aufgabe wird in mehrere Prompts zerlegt, die aufeinander aufbauen.

12. Meta-Prompting:

Du erklärst explizit wie das Modell vorgehen soll.

13. Reflexion Prompting:

Modell wird aufgefordert, seine Antwort zu überprüfen und ggf. zu verbessern

14. Multimodal Prompting:

Kombination aus Text und Bild/Audio (falls unterstützt).

PROMPT ENGINEERING

BEWERTUNG VON PROMPTS

1. Manuelle Evaluation
2. Definierte Metriken
3. A/B-Tests
4. LLM-as-a-Judge
5. User Feedback
6. Stress-Tests / Robustheitstests

› 3. MASCHINELLES LERNEN I: SUPERVISED LEARNING

SUPERVISED LEARNING: DEFINITION UND AUFGABEN

Supervised Learning (überwachtes Lernen) ist ein Teilbereich des maschinellen Lernens,

- > bei dem ein Modell anhand von Beispieldaten trainiert wird,
- > die sowohl *Vorhersagekriterien (Features)*
- > als auch die dazugehörigen *Ausgaben (Labels/Zielwerte)* enthalten.

Typische Aufgaben:

- > *Klassifikation* → Vorhersage einer Kategorie (z. B. Spam vs. Nicht-Spam)
- > *Regression* → Vorhersage eines Werts (z. B. Temperatur morgen, Aktienkurs)

SUPERVISED LEARNING: EVALUATION VON KLASSIFIKATIONSMODELLEN

Basis Wahrheitsmatrix (Confusion Matrix):

Confusion Matrix

Tatsächlich	0	wahr negativ	falsch positiv
	1	falsch negativ	wahr positiv
		0	1
		Vorhersage	

$$Accuracy = \frac{Wahr\ Positiv + Wahr\ Negativ}{Gesamtanzahl}$$

SUPERVISED LEARNING: EVALUATION VON KLASSIFIKATIONSMODELLEN

Weitere Metriken:

> Sensitivität / True Positive Rate / Recall_1:

$$\text{Sensitivität} = \frac{\text{Wahr Positiv}}{\text{Wahr Positiv} + \text{Falsch Negativ}}$$

Confusion Matrix

wahr negativ	falsch positiv
falsch negativ	wahr positiv

> Spezifizität/ True Negative Rate / Recall_0:

$$\text{Spezifizität} = \frac{\text{Wahr Negativ}}{\text{Wahr Negativ} + \text{Falsch Positiv}}$$

Confusion Matrix

wahr negativ	falsch positiv
falsch negativ	wahr positiv

SUPERVISED LEARNING: EVALUATION VON REGRESSIONSMODELLEN

Gegeben wahre Werte (y_i), vorhergesagte Werte (\hat{y}_i) und Stichprobengröße (n), dann

- > ist der Mean Squared Error

$$MSE = \frac{1}{n} [(y_1 - \hat{y}_1)^2 + \dots + (y_n - \hat{y}_n)^2]$$

- > ist der Root Mean Squared Error

$$RMSE = \sqrt{MSE}$$

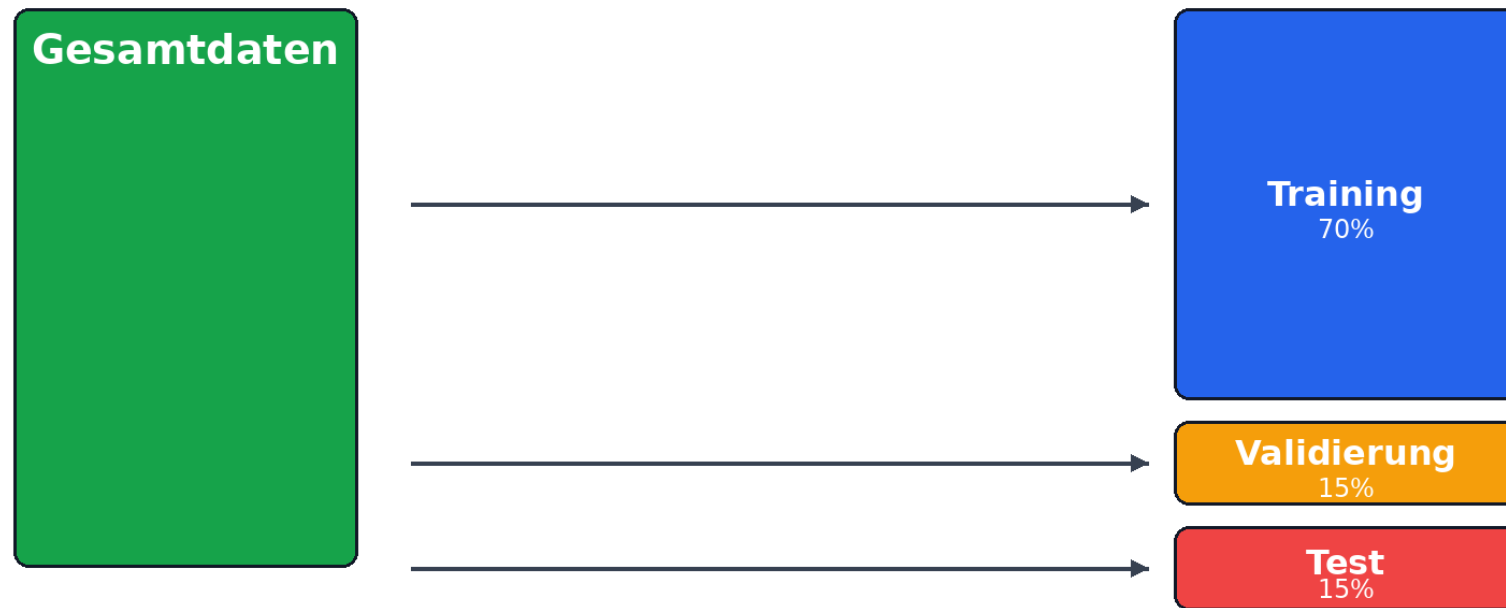
- > ist das Bestimmtheitsmaß R^2

$$R^2 = 1 - \frac{\text{Erklärte Varianz}}{\text{Totale Varianz}} = 1 - \frac{(y_1 - \hat{y}_1)^2 + \dots + (y_n - \hat{y}_n)^2}{(y_1 - \text{Mittelwert}_y)^2 + \dots + (y_n - \text{Mittelwert}_y)^2}$$

Daten → Aufbereiten → Modell wählen → Trainieren → Testen → Optimieren → Anwenden

SUPERVISED LEARNING: VORGEHENSWEISE

Train-Test-Split:



Train/Validation/Test Split

SUPERVISED LEARNING: STANDARD ALGORITHMEN

Lineare Modelle	Tree-basierte Modelle	Neuronale Netze	Sonstige
Lineare Regression Logistische Regression	Decision Trees Random Forest Gradient Boosting	Einfache KNN Deep Learning	Support Vector Machines k-Nearest Neighbors Naive Bayes

Decision Tree Algorithmus:

1. Wurzel: Startet mit allen Trainingsdaten
2. Splits: Wählt das Merkmal aus, das die Daten am besten trennt (gemessen durch Kriterien wie Gini-Impurity oder Entropie)
3. Rekursion: Wiederholt den Prozess für jeden Ast, bis ein Stopp-Kriterium erreicht ist
4. Blätter: Endknoten enthalten die finale Vorhersage (Klasse oder Wert)

Decision Tree Algorithmus: Gini-Impurity und Entropie

Gegeben Daten mit K Kategorien; p_i sei die relative Häufigkeit der Kategorie $i \in \{1, \dots, K\}$,
dann

> ist die *Gini-Impurity* definiert durch:

$$Gini = p_1(1 - p_1) + \dots + p_K(1 - p_K) = 1 - (p_1^2 + \dots + p_K^2)$$

> ist die *Shannon-Entropie* definiert durch:

$$Entropie = -(p_1 \log_2 p_1 + \dots + p_K \log_2 p_K)$$

SUPERVISED LEARNING: TREE-BASIERTE MODELLE

Decision Tree Algorithmus: Berechnungsbeispiel (Vorlesung)

Random Forest und Gradient Boosting

- > Ensemble-Methoden = Kombination mehrerer Modelle (hier: Trees)

Hauptarten von Ensemble-Methoden:

1. *Stacking / Voting*

- Mehrere unterschiedliche Modelle werden trainiert (z. B. SVM, Neuronales Netz).
- Stacking: Meta-Modell lernt, wie man deren Vorhersagen optimal kombiniert.
- Voting: Mehrheitsentscheidung (hard voting), Mittelung (soft voting)

Hauptarten von Ensemble-Methoden:

2. *Bagging (Bootstrap Aggregating)*

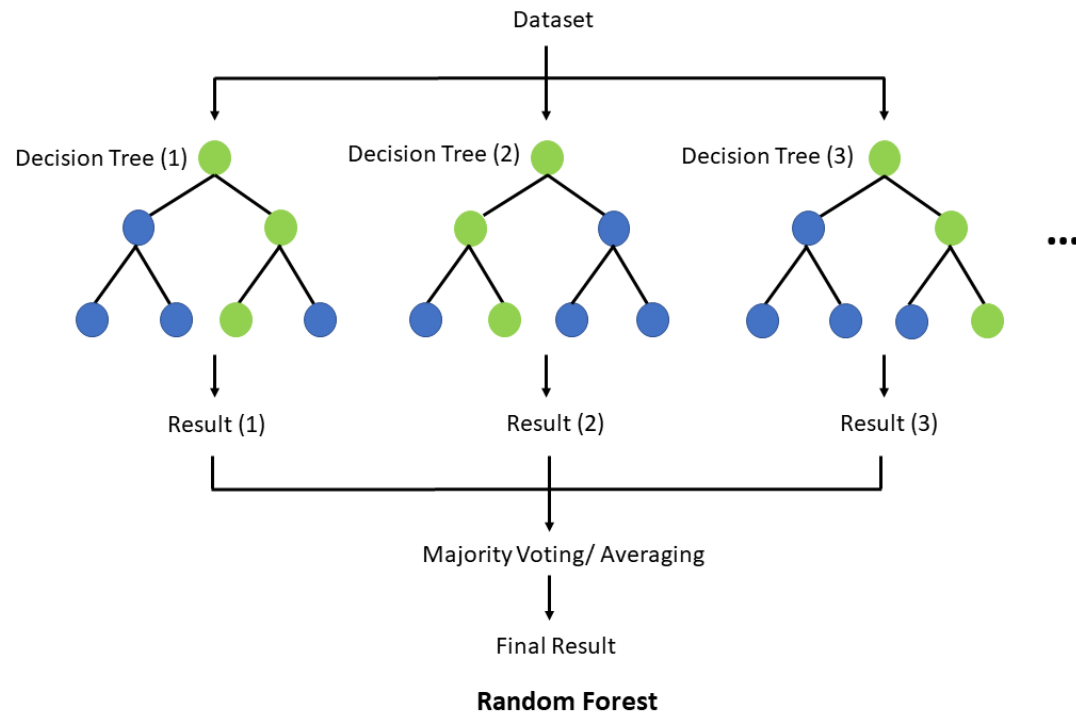
- Trainiert viele Modelle parallel auf zufälligen Teilmengen (Ziehen mit Zurücklegen).
- Ergebnisse werden gemittelt oder abgestimmt.
- Beispiel: **Random Forest**

3. *Boosting*

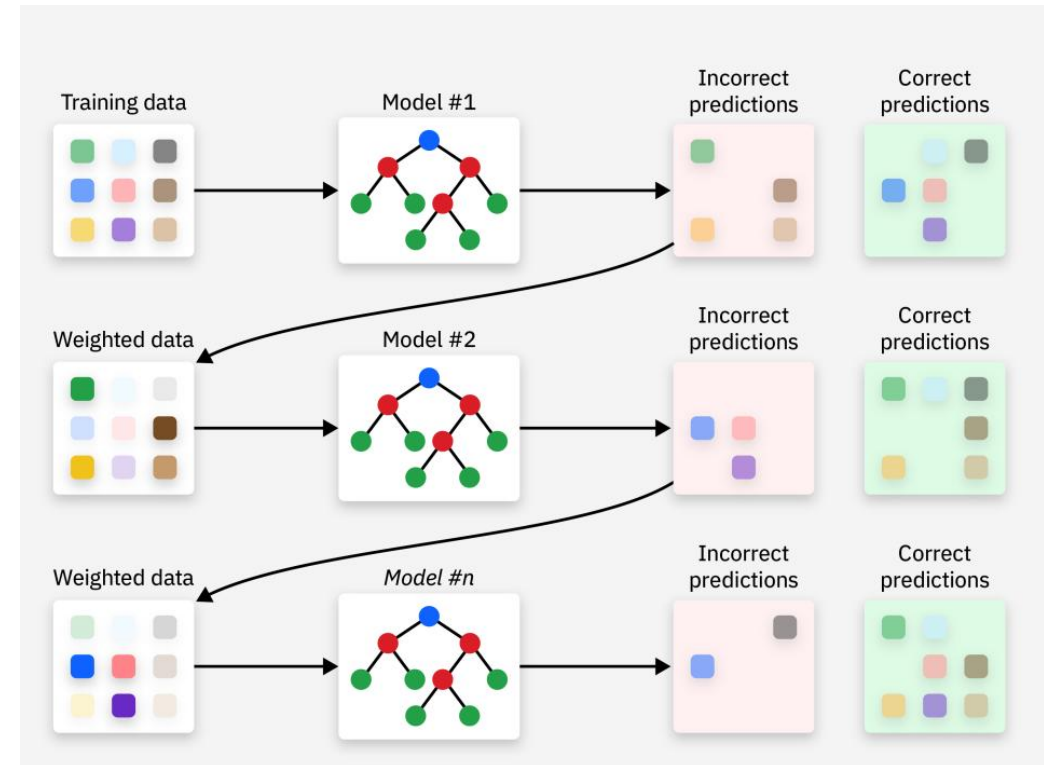
- Trainiert Modelle nacheinander. Neue Modelle reduzieren Fehler der vorherigen.
- Schwierig vorherzusagende Datenpunkte bekommen mehr Gewicht.
- Beispiel: **Gradient Boosting**

SUPERVISED LEARNING: TREE-BASIERTE MODELLE

Random Forest:



Gradient Boosting:



Quelle: https://de.wikipedia.org/wiki/Random_Forest

Quelle: <https://www.ibm.com/think/topics/gradient-boosting>

Gradient Boosting Algorithmus:

1. Initialisierung: Startet mit einer einfachen Vorhersage (oft der Mittelwert)
2. Iterativer Prozess:
 - Berechne Residuen (Fehler) der aktuellen Vorhersage
 - Trainiere einen neuen schwachen Lerner auf diese Residuen
 - Addiere die neue Vorhersage mit einem Gewichtungsfaktor (Learning Rate)
3. Wichtige Parameter
 - Learning Rate (α): Kontrolliert Schrittgröße (typisch 0.01-0.3)
 - Anzahl Bäume: Mehr Bäume = komplexeres Modell
 - Baumtiefe: Meist shallow trees (3-8 Ebenen)

Gradient Boosting Algorithmus: Odds und Log-Odds

Sei p die relative Häufigkeit eines Merkmals, dann

> sind die Odds (o) des Merkmals

$$o = \frac{p}{1 - p}$$

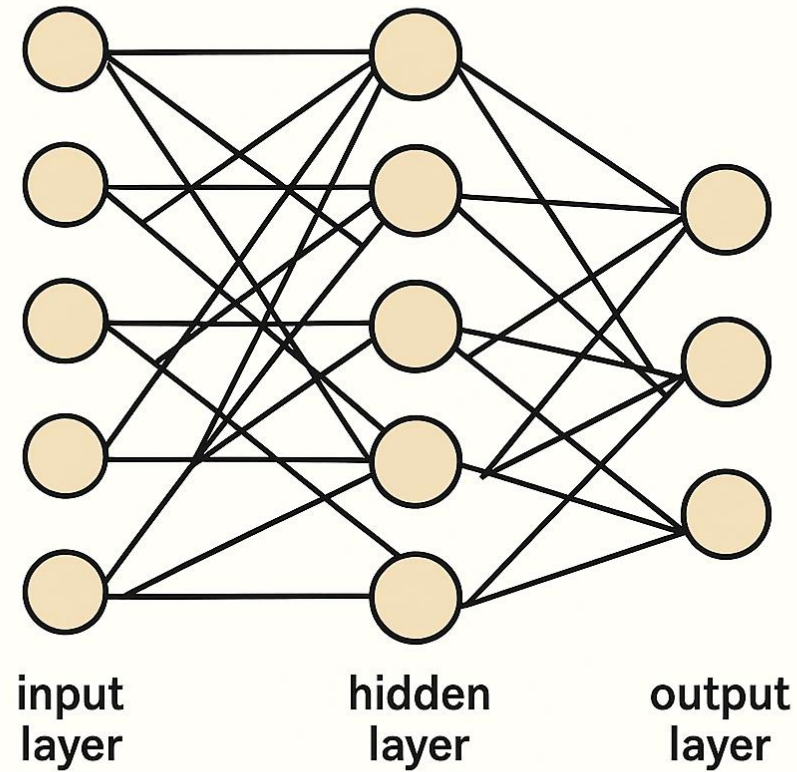
> und die Log-Odds sind:

$$\ln(o)$$

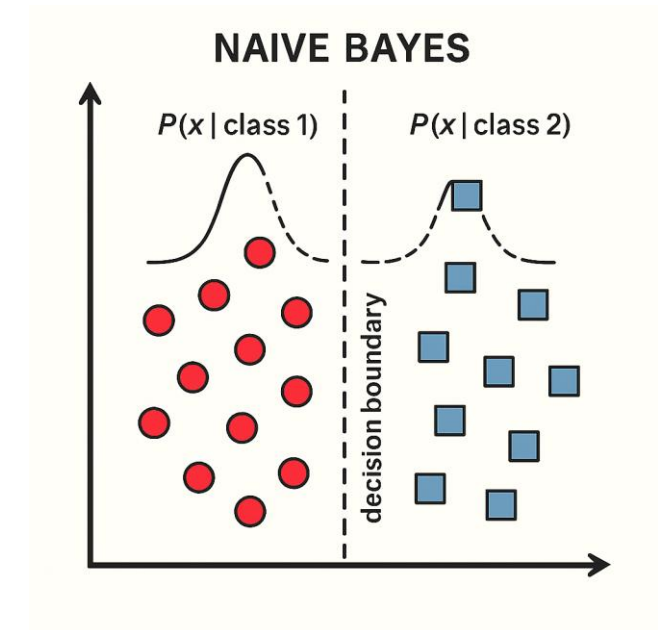
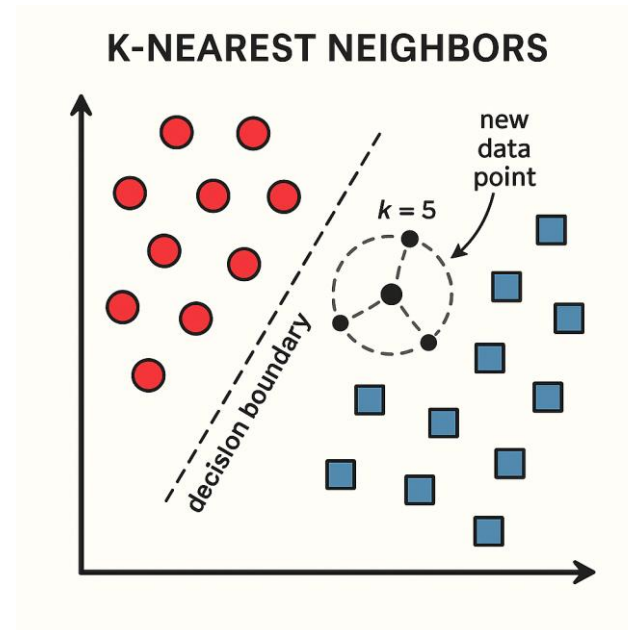
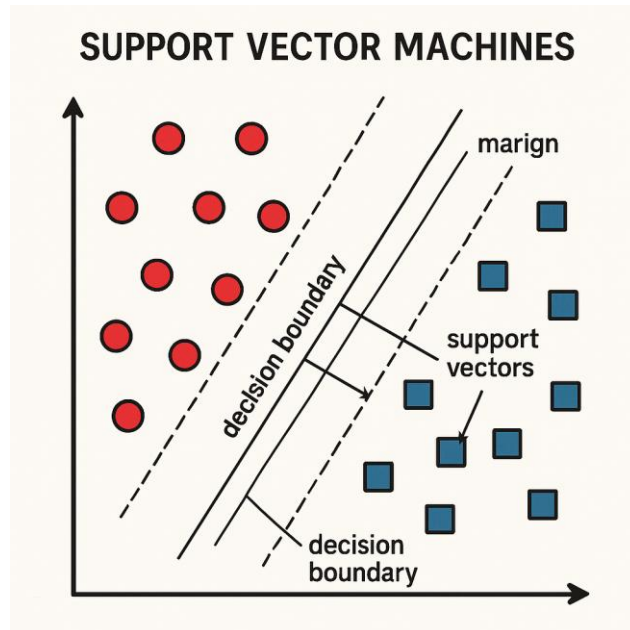
SUPERVISED LEARNING: TREE-BASIERTE MODELLE

Gradient Boosting Algorithmus: Berechnungsbeispiel (Vorlesung)

KÜNSTLICHES NEURONALES NETZ



SUPERVISED LEARNING: SONSTIGE MODELLE



› 4. MASCHINELLES LERNEN II: UNSUPERVISED LEARNING

UNSUPERVISED LEARNING: DEFINITION UND AUFGABEN

Unsupervised Learning (unüberwachtes Lernen) ist ein Bereich des maschinellen Lernens,

- > bei dem ein Modell anhand von Beispieldaten trainiert wird,
- > *ohne das Ausgaben (Labels/Zielwerte)* vorgegeben sind.
- > Aufgabe ist es selbstständig Muster und Strukturen in den Daten zu entdecken.

UNSUPERVISED LEARNING: STANDARD ALGORITHMEN

Clustering	Dimensionsreduktion	Anomalieerkennung
K-Means	Principal Component Analysis	Isolation Forest
Hierarchisches Clustering	t-distributed Stochastic Neighbor Embedding	One-Class SVM
DBSCAN (Density-Based Spatial Clustering of Applications with Noise)	Uniform Manifold Approximation and Projection	
Gaussian Mixture Models (GMMs)	Autoencoder	

UNSUPERVISED LEARNING: K-MEANS ALGORITHMUS

1. Wähle Anzahl an Clustern k
2. Initialisiere zufällig k Zentren (Centroids)
3. Weise Punkten den Clustern über minimale Distanz zu den Zentren zu
4. Aktualisiere Zentren: Für Cluster C mit Punkten x_1, \dots, x_n :

$$\mu_C = \frac{1}{n}(x_1 + \dots + x_n)$$

5. Wiederhole Schritt 3 und 4 bis sich die Cluster nicht mehr ändern

UNSUPERVISED LEARNING: K-MEANS ALGORITHMUS

Berechnungsbeispiel (Vorlesung)

› 5. MASCHINELLES LERNEN III: REINFORCEMENT LEARNING

REINFORCEMENT LEARNING: DEFINITION UND AUFGABEN

Reinforcement Learning (RL) ist ein Bereich des maschinellen Lernens,

- > bei dem ein Agent lernt, optimale Entscheidungen in einer Umgebung zu treffen,
- > indem er durch Versuch und Irrtum Erfahrungen sammelt.



- > Agent: Das lernende System, das Entscheidungen trifft
- > Umgebung: Die Welt, in der der Agent agiert
- > Zustand (s): Die aktuelle Situation der Umgebung
- > Aktion (a): Was der Agent tun kann
- > Belohnung (r): Das Feedback, das der Agent für seine Aktionen erhält
- > Policy (π): Die Strategie, die bestimmt, welche Aktion in welchem Zustand gewählt wird

REINFORCEMENT LEARNING: Q-LEARNING

1. Initialisierung:

- Lege Lernrate α , Diskontfaktor γ und Explorationsrate ϵ fest.
- Erstelle eine Q-Tabelle (Zustände \times Aktionen) mit zufälligen oder null Werten.

2. Wähle Aktion

- Mit Wahrscheinlichkeit ϵ : wähle zufällige Aktion (Exploration).
- Mit Wahrscheinlichkeit $1 - \epsilon$: wähle Aktion mit dem größten Q-Wert (Exploitation).

3. Führe Aktion aus (Übergang zum neuen Zustand s' , erhalte Belohnung r)

4. Aktualisiere Q-Werte:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

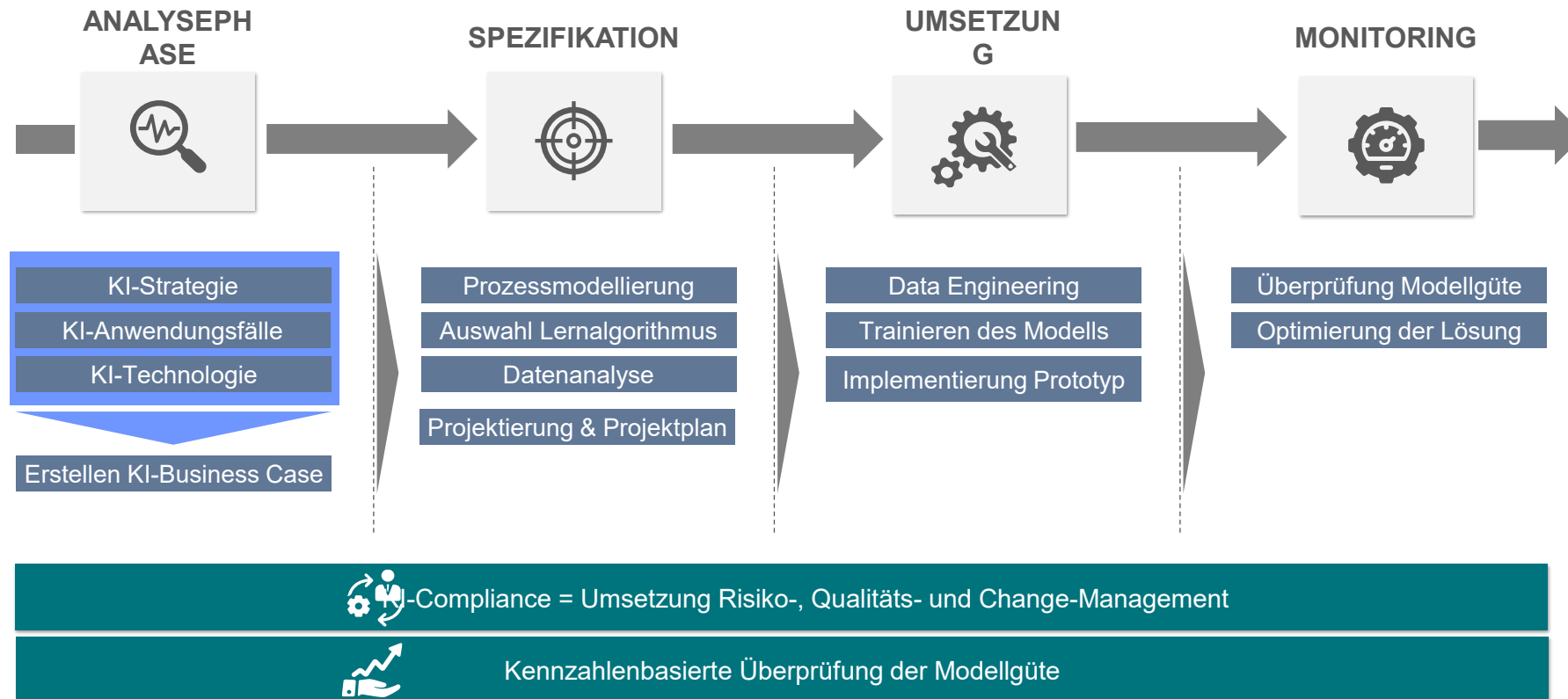
5. Wiederhole 2.-4. bis Q-Tabelle stabil ist oder maximale Anzahl Episoden erreicht

REINFORCEMENT LEARNING: Q-LEARNING

Berechnungsbeispiel (Vorlesung)

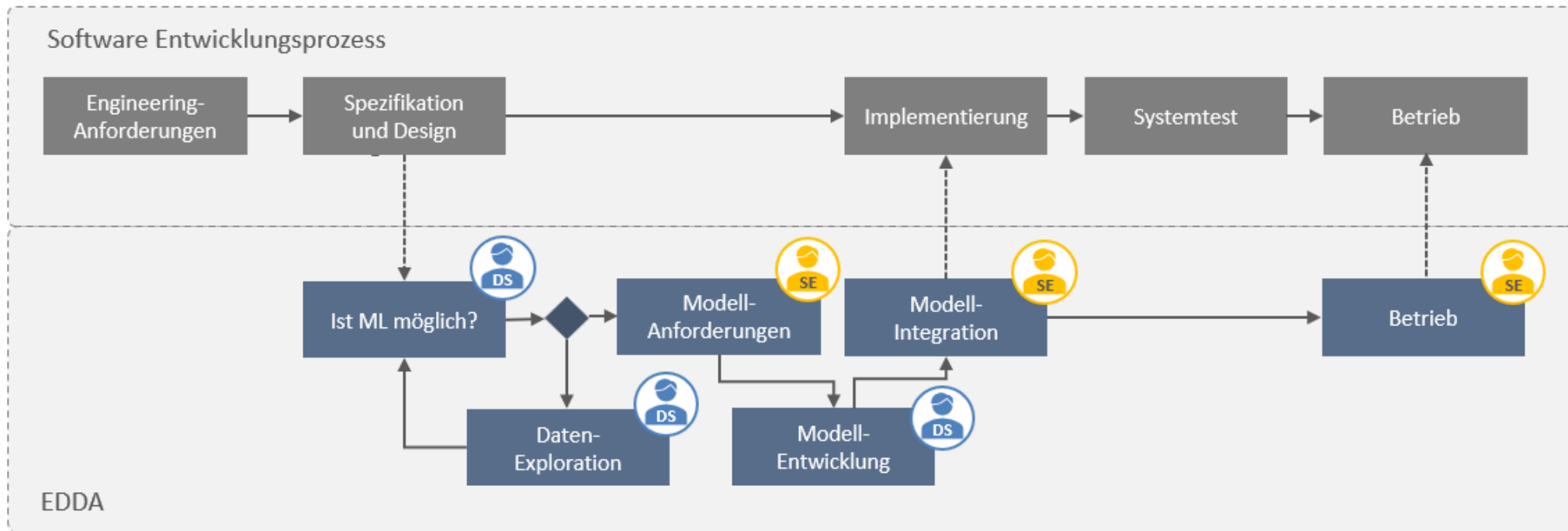
› 6. IMPLEMENTIERUNG VON KI- LÖSUNGEN

Ablauf KI-Projekt:



Quelle: DSC GmbH

Der Weg zur KI-Anwendung:



Quelle : EDDA – Engineering Data-Driven Applications, Universität Duisburg-Essen

Aufgabenverteilung KI-Projekt:



DOMAIN EXPERT

Kennt sich mit den Geschäftsprozessen des Unternehmens und der Branche aus



DATA SCIENTIST

Hat KI-Kenntnisse
Beherrscht Techniken der Datenanalyse
Hat grundlegende Kenntnisse in der Software-Entwicklung



DATA ENGINEER

Extrahiert Daten aus den unterschiedlichsten Systemen und stellt diese zur Verfügung

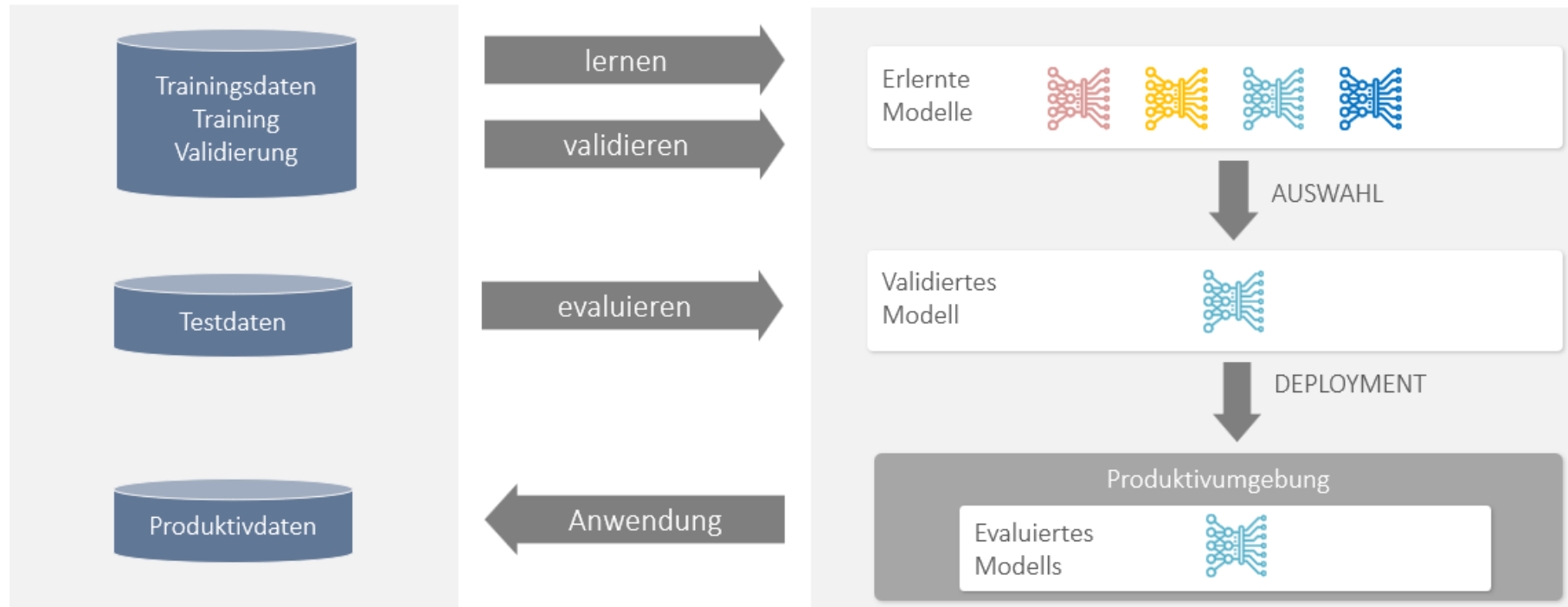


DATA ARCHITECT

Definiert und entwickelt technische Komponenten
Kennt technische Komplexität IT-Systeme

Quelle: DSC GmbH

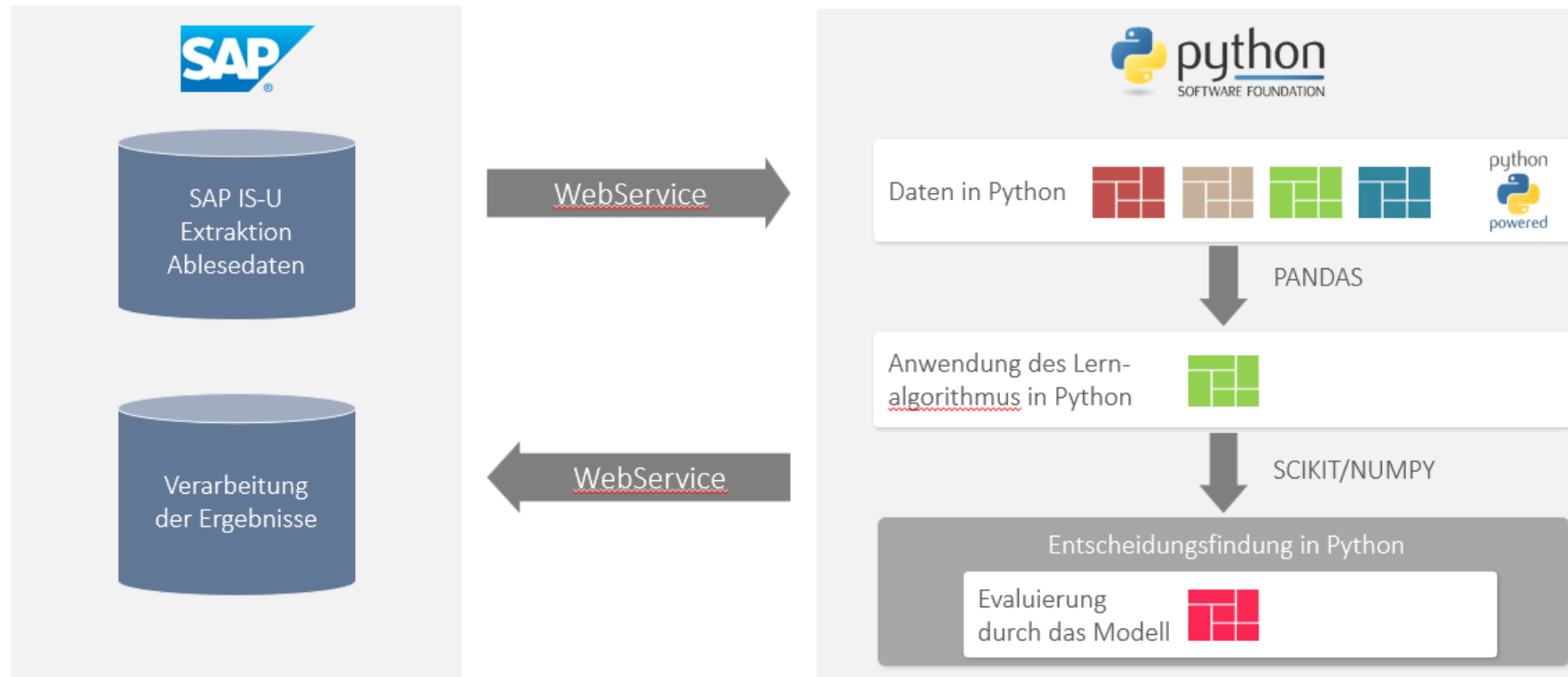
Methodisches Vorgehen:



Quelle: : Introduction to Data Science (Ch04), Prof. Drabant, Hochschule Mannheim

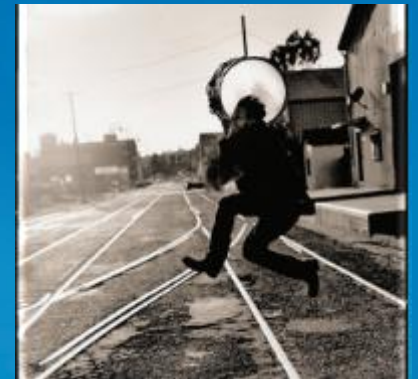
IMPLEMENTIERUNG VON KI-LÖSUNGEN

Beispiel:



Quelle: DSC GmbH

THE END!



Please refer any questions to:
Prof. Dr. Florian Kauffeldt
Faculty of International Business
florian.kauffeldt@hs-heilbronn.de