Machine Learning Renaissance

1. Introduction

Das Konzept **“Machine Learning”** hat sich über mehrere Jahrzehnte entwickelt und basiert auf Beiträgen aus verschiedenen wissenschaftlichen Disziplinen, insbesondere der Statistik, Mathematik, Informatik und KI.

Machine Learning ist Teilgebiet der Künstlichen Intelligenz (KI), das sich darauf konzentriert, Algorithmen und Modelle zu entwickeln, die aus Daten lernen und Vorhersagen oder Entscheidungen treffen können, anstatt explizit dafür programmiert zu werden. Das Hauptziel besteht darin, Muster und Zusammenhänge in Daten zu erkennen, um zukünftige Datenpunkte präzise vorherzusagen oder zu klassifizieren.

2. Machine Learning Types

Machine Learning (ML) lässt sich in mehrere Arten einteilen, die auf der Art und Weise basieren, wie Modelle aus Daten lernen. Die drei Hauptarten sind überwacht **(supervised learning)**, unüberwacht **(unsupervised learning)** und bestärkendes Lernen **(reinforcement learning)** [Fig.1].

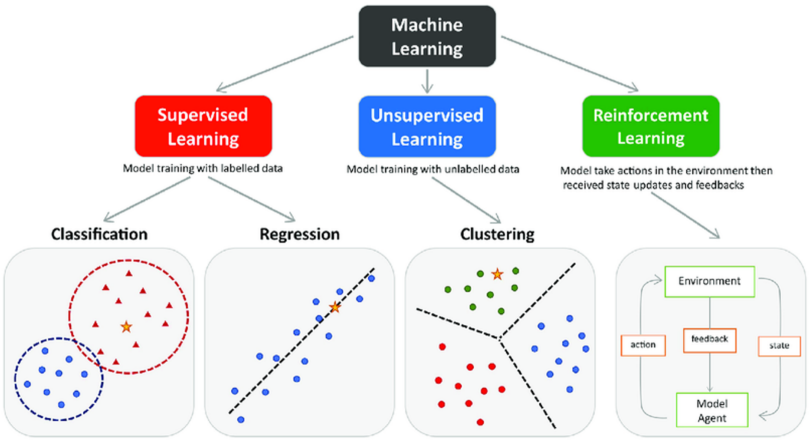

Fig.1 Die Abbildung zeigt die drei Hauptkategorien des maschinellen Lernens: überwachtes Lernen, unüberwachtes Lernen und verstärkendes Lernen. Überwachtes Lernen trainiert Modelle mit beschrifteten Daten für Klassifikation und Regression, während unüberwachtes Lernen unbeschriftete Daten verwendet, um Muster durch Clustering zu erkennen. Verstärkendes Lernen lässt Modelle durch Interaktionen mit der Umgebung lernen, um durch Feedback Belohnungen zu maximieren. [1]

2.1 Supervised Learning

Supervised Learning ist eine Art des Machine Learnings, bei der das Modell von Daten mit Labels trainiert wird. Das bedeutet, dass für jeden Datenpunkt ein bekanntes Ergebnis existiert. Das Ziel ist es, eine Funktion zu lernen, die Eingaben den richtigen Ausgaben zuordnet.

Typen	Beschreibung	Algorithmen
Klassifikation	Zuweisung von Datenpunkten zu vordefinierten Kategorien.	**Logistic Regression** **Decision Tree** **Naive Bayes** **SVM** (Support Vector Machine) **KNN** (K-Nearest Neighbors)

Typen	Beschreibung	Algorithmen
Regression	Vorhersage kontinuierlicher Werte.	**Linear Regression** **Polynomial Regression** **Decision Tree**

Beispiel: **Logistic Regression**

- Formula: $p(y = 1|x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$
  - $\beta_0 + \beta_1 x$ → Linear Regression | β_0 → Bias | β_1 → Intercept
  - e → Basis des natürlichen Logarithmus (ungefähr 2.71828).

Angenommen, wir möchten vorhersagen, ob ein Student eine Prüfung basierend auf der Anzahl der Stunden, die er zum Lernen gebracht hat, besteht. In diesem Fall könnten wir ein Supervised Modell, speziell eine Logistic Regression, verwenden.

- Dataset:
  - "Stunden gelernt" die **unabhängige Variable** $x = [30, 35, 50, 90, 110, 115]$ (oder Eingabe)
  - "Prüfung bestanden" wäre die **abhängige Variable** $y = [0, 1, 0, 1, 1, 1]$ (oder Ausgabe), die wir vorhersagen wollen.
- Berechnung:
  - Angenommen, wir erhalten nach dem Training folgende Koeffizienten: $\beta_0 = 0.039$ | $\beta_1 = 0.008$
  - Jetzt können wir für einen Student, der 80 Stunden gelernt hat, vorhersagen, ob er die Prüfung bestehen wird → $p(y = 1|80) = \frac{1}{1+e^{-(0.039+0.008*80)}} = 0.66$. Es ist also wahrscheinlich, dass er die Prüfung bestehen wird.

2.2 Unsupervised Learning

Unsupervised Learning ist eine Art des Machine Learnings, bei der das Modell auf Daten ohne Labels trainiert wird. Das Ziel ist es, Muster oder Strukturen in den Daten zu finden.

Typen	Beschreibung	Algorithmen
Clustering	Gruppierung ähnlicher Datenpunkte.	**K-Means Clustering** **Hierarchical Clustering** **Anomaly Detection**
Dimensionality Reduction	Reduzierung der Anzahl von Variablen.	**PCA** (Principal Component Analysis)

Beispiel: **K-Means Clustering** [3]

- Formulas:
  - **Initialize →** $\mu_1, \mu_2, ..., \mu_k \in \mathbb{R}^d$
    - μ → Centroid
  - **Assignment step →** $C_i^{(t)} = argmin||x_i - \mu_k||^2$
    - C_i → Cluster | x → Data Point | μ_k → *k*-th Centroid
  - **Update step →** $\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$
    - $|C_k|$ → Anzahl der Datenpunkte, die einem Cluster k zugewiesen sind

Angenommen, wir möchten Blumen “Iris” in Gruppen basierend auf ihren physischen Merkmalen (sepal length, sepal width, petal length, petal width) einteilen [2]. In diesem Fall könnten wir ein Unsupervised Modell, speziell ein K-Means Clustering, verwenden.

- Dataset:
  - “Merkmale” (**unabhängige Variablen**) $X = \begin{bmatrix} 5.0 & 3.5 & 1.6 & 0.6 \\ 5.7 & 2.6 & 3.5 & 1.0 \\ 6.9 & 3.2 & 5.7 & 2.3 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$
  - “Species” (**abhängige Variable**) $y = [\text{”Iris-setosa”}, \text{”Iris-versicolor”}, \text{”Iris-virginica”}, ...]$
- Berechnung:
  - Wir definieren ein K-Means-Modell mit 3 Clustern (da es drei Blumenarten im Dataset gibt).
  - Nach dem Training wir erhalten die $\text{Centers} = \begin{bmatrix} 5.9016129 & 2.7483871 & 4.39354839 & 1.43387097 \\ 5.006 & 3.428 & 1.462 & 0.246 \\ 6.85 & 3.07368421 & 5.74210526 & 2.07105263 \end{bmatrix}$
  - So für eine neue Blume, mit $x = [6.4, 2.8, 5.6, 2.1]$ nach der Berechnung der euklidischen Distanz $d = \sqrt{(x - \overline{x}) + (y - \overline{y}) + (z - \overline{z})}$, wir bekommen, dass die Blume zur **Iris-virginica** gehört.

2.3 Reinforcement Learning

Reinforcement Learning (RL) ist ein Teilgebiet des Machine Learning, das sich mit dem Training von Agenten beschäftigt, die in einer Environment Actions durchführen, um Belohnungen zu maximieren. RL basiert sich auf der Idee, dass ein Agent durch Versuch und Irrtum lernen kann, welche Actions am besten für eine bestimmte Situation geeignet sind, indem er Belohnungen (Zuckerbrot) oder Strafen (Peitsche) für seine Actions erhält.

Typen	Beschreibung	Algorithmen
Model-Free RL	Methoden erfordern kein explizites Modell der Umgebung. Sie versuchen nicht vorherzusagen, was in der Zukunft als Reaktion auf die Aktionen des Agenten passieren wird. Stattdessen konzentrieren sie sich auf die Optimierung des Verhaltens des Agenten basierend auf der erworbenen Erfahrung.	**Q-Learning** **SARSA (State-Action-Reward-State-Action)** **Deep Q-Networks (DQN)** **PPO (Proximal Policy Optimization)**
Model-Based RL	Dieses Modell prognostiziert, wie die Environment auf die Aktionen des Agenten reagieren wird, was eine Planung mehrere Schritte im Voraus ermöglicht.	**Dyna-Q** **AlphaGo**

Beispiel: **Q-Learning** [4]

- Formula: **Update Rule** → $Q(s, a) = Q(s, a) + a[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
  - $Q(s, a)$ → Current Q Values | a → Learning Rate | r → Reward | γ → Discount Factor
  - $\max_{a'} Q(s', a')$ → Maximum Q-value for the next state s' over all possible actions a'

Angenommen, wir haben einen Agenten in einer Gridworld-Umgebung. Das Ziel des Agenten ist es, das Ziel (Goal) zu erreichen, während er Hindernissen ausweicht und dabei die kürzeste Route nimmt.

- Environment:

- Gridworld-Layout → $\begin{bmatrix} S & 0 & 0 & 0 \\ 0 & X & 0 & X \\ 0 & 0 & G & 0 \end{bmatrix}$ (S: Startpostion, G: Zielposition, X: Hindernisse)

- Training:
  - Zuerst wird Q-Table $Q(s, a)$ mit zufälligen Werten initialisiert.
  - Aktion a wird basierend auf ϵ-greedy Politik (mit Wahrscheinlichkeit ϵ, dass eine a zufällig gewählt wird oder $\arg\max Q(s, a)$).
  - Nachdem Aktion ausgeführt wird, wird Q-Wert aktualisiert ($Q(s, a) = ...$).
- Nach der n-Anzahl der Episoden sollen wir einen besten Weg zum G bekommen, entweder (2x RIGHT + 2x DOWN) oder umgekehrt.

3. Data

Allgemein sind Daten Informationen, die gesammelt und analysiert werden können. Diese Informationen können Zahlen, Texte, Bilder, Videos usw. sein. Die organisierte Sammlung von Daten heißt Datensatz, das könnte eine Tabelle sein, in der jede Zeile ein Datenpunkt und jede Spalte ein Merkmal (Feature) ist. Datensätze sind oft als Excel-Tabelle, CSV-Datei oder SQL-Datenbank gespeichert.

Im Gegensatz zur Datensätze gibt es Datenkorpus, das weniger strukturiert und textuell ist. Während Datensätze spezifische, oft numerische oder kategoriale Datenpunkte enthalten, haben Datenkorpora große Mengen an rohen Texten oder Sprachdaten.

Um besser zu verstehen, was Data ist, betrachten wir Titanic Dataset.

Survived	Pclass	Name	Sex	Age	SibSP	Parch
0	3	Kelly, Mr. James	male	34.5	0	0
1	3	Wilkes, Mrs. James	female	47	1	0
...	...	...	...	...	...	...

3.1 Data Types

Es gibt verschiedene Arten von Daten, die in Maschinellem Lernen verwendet werden können. Am häufigsten sind:

1. **Numerische Daten** oder **Quantitative Daten**: Dies sind Zahlen, die messbare Mengen repräsentieren.
   - Continuous → können jeden Wert innerhalb eines bestimmten Bereichs annehmen (Age)
   - Discrete → können nur bestimmte, getrennte Werte annehmen. (SibSP, Parch)
2. **Kategoriale Daten** oder **Qualitative Daten**: Diese Daten repräsentieren Eigenschaften und können daher in Kategorien eingeteilt werden.
   - Nominal → es gibt keine geordnete Beziehung zwischen den Werten (Sex: male, female)
   - Ordinal → es gibt eine geordnete Beziehung zwischen den Werten (Bildungsgrad: High school, bachelor ...)

3.2 Features and Feature Characteristics

Merkmale (Features) sind einzelne Eigenschaften oder Attribute, die zur Beschreibung der Daten verwendet werden. Merkmalsausprägungen sind spezifische Werte, die ein Merkmal annehmen kann.

- **Merkmale**: Alter, Geschlecht, Name, Ticketklasse, Überlebensstatus, Anzahl der Geschwister/Ehepartner an Bord, Anzahl der Eltern/Kinder an Bord.

- **Merkmalsausprägungen**: z.B. Geschlecht hat die Ausprägungen "männlich" und "weiblich".

3.3 Requirements for ML Features

1. **Relevanz**: Merkmale sollten relevant für die Aufgabe sein.
2. **Qualität und Sauberkeit:**
   - **Fehlende Werte**: Fehlende Daten sollen angemessen behandelt werden, entweder durch Imputation, Entfernung oder mit Algorithmen, die fehlende Werte von sich bearbeiten können.
   - **Ausreißer**: Die Ausreißer sollen identifiziert und behandelt werden, die das Verständnis des Modells von den Daten verzerren können.
3. **Unabhängigkeit**: Merkmale sollten nicht stark miteinander korreliert sein, um Multikollinearität zu vermeiden.
4. **Skalierung**: Merkmale sollten auf ähnliche Skalen gebracht werden, um bestimmte Algorithmen zu unterstützen.

3.4 Procedure for selecting features and data cleaning

Kehren wir uns zu unserem Dataset zurück und wählen basierend auf den oben genannten Anforderungen, die für das Training und zukünftige Vorhersagen, notwendigen Daten aus und bereinigen sie.

1. Columns Name und Ticket können entfernt werden `df.drop('Name', axis=1, inplace=True)`, weil diese Features irrelevant für Training sind (da sie keine bedeutende Auswirkung auf das Überlebensergebnis haben).
2. Die fehlende Daten in Age und Fare können mit Mean Value ersetzt werden `df['Age'].fillna(titanic['Age'].mean(), inplace=True)`.
3. Cabin Feature kann entfernt werden `df.drop('Name', axis=1, inplace=True)`, da es dort zu viele fehlende und unique Werte gibt, die nicht rational ersetzt werden können.

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

3. Nach der Korrelationen Berechnung **[Fig.2]** können wir eine zwei Features (SibSP und Parch) kombinieren, da die stark miteinander korreliert und ziemlich ähnlich sind.

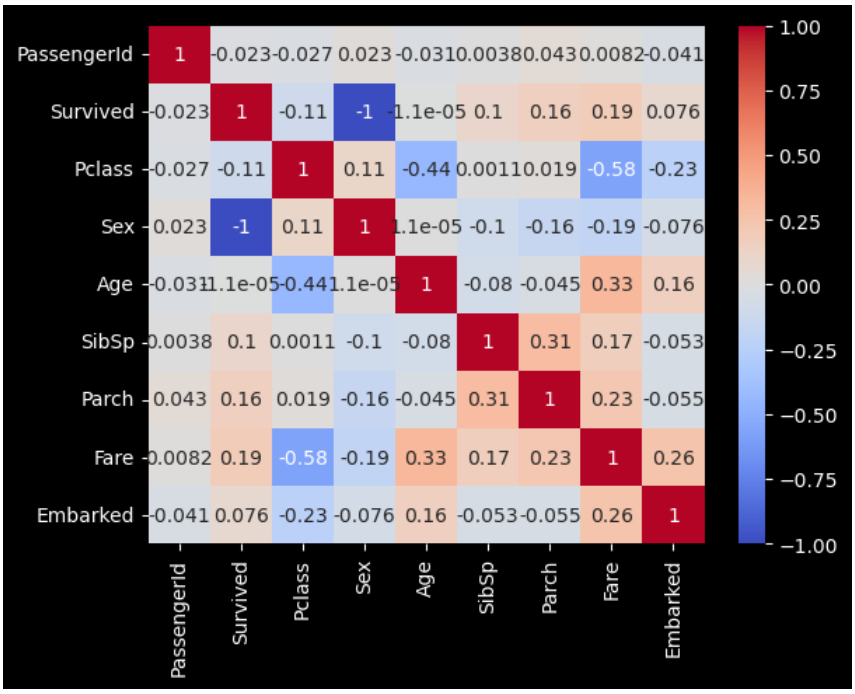

Fig.2 zeigt eine Korrelationsmatrix in Form einer Heatmap, die die Korrelationen zwischen verschiedenen Merkmalen eines Datensatzes darstellt. Die Matrix enthält sowohl positive als auch negative Korrelationswerte, die durch unterschiedliche Farben repräsentiert werden: rote Farbtöne stehen für positive Korrelationen und blaue Farbtöne für negative Korrelationen.

4. Machine Learning Pipeline

Der Algorithmus zur Training eines Maschinenlernmodells besteht aus mehreren Schichten **[Fig.3]**, die jeweils mehrere Schritte enthalten.

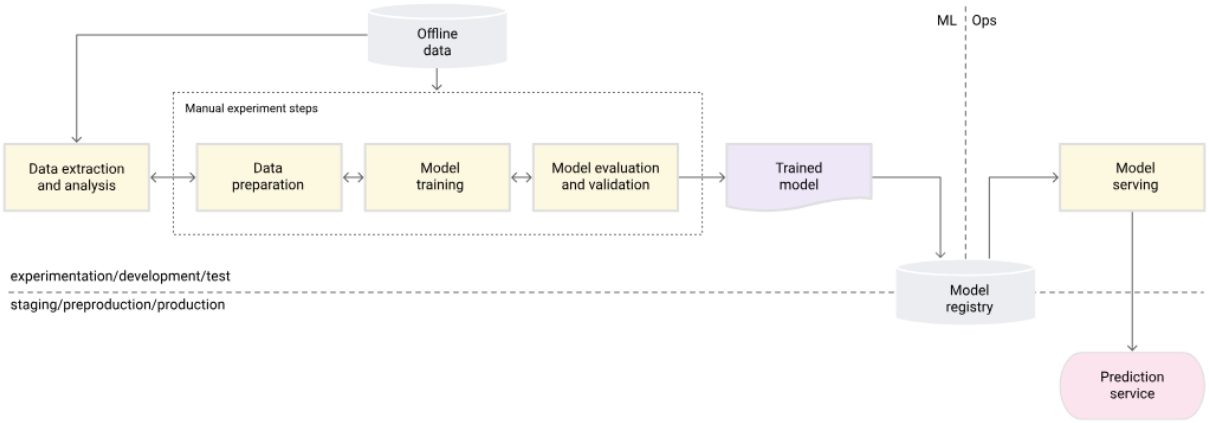

Fig.3 illustriert den End-to-End-Workflow einer Machine-Learning-Pipeline, von der Datenaushebung bis zur Modellbereitstellung. Der Prozess ist in mehrere Stadien unterteilt, wobei jedes Stadium eine kritische Phase im Lebenszyklus des maschinellen Lernens darstellt. **[5]**

Wir werden nur “Manual experiment steps” Bereich für obengenanntes Dataset (Titanic) betrachten.

1) Data preparation: Hierbei geht es um die Säuberung (was wurde bereits ober gemacht) von Daten von verschiedenen Arten von Outliers, die das Training des Modells beeinflussen könnten, ebenso wie von stark miteinander korrelierten Features, die Umwandlung von Daten in das benötigte Format (zu quantitativen oder qualitativen Daten) abhängig von der Modellauswahl.

```
import pandas as pd

df = pd.read_csv('titanic_train.csv')

# Data Cleaning
df = df.drop(['Name', 'Ticket', 'Cabin'], axis=1)

# Fill missing values
df['Age'] = df['Age'].fillna(df['Age'].mean())
df['Fare'] = df['Fare'].fillna(df['Fare'].mean())
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# Convert categorical values to numerical values
df['Sex'].replace({'male', 'female'}, {0, 1}, inplace=True)
df['Embarked'].replace({'S', 'C', 'Q'}, {0, 1, 2}, inplace=True)
```

2) Model Training: Vor dem Training eines Modells ist es gute Praxis, die Daten in Trainings- und Testdaten zu unterteilen, um zukünftiges Overfitting oder Underfitting zu vermeiden und eine bessere Möglichkeit zur Bewertung der Modellqualität zu haben.

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

# Split data
X = df.drop('Survived', axis=1)
y = df['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

3) Model Evaluation and Validation: Nachdem das Modell trainiert ist, müssen wir seine Leistung evaluieren und validieren.

- **Evaluation:** Bewertung der Modellleistung anhand geeigneter Metriken, z.B. Mean Squared Error (MSE) für Regressionsmodelle.
- **Cross-Validation:** Anstatt nur eine einzelne Aufteilung in Trainings- und Testdaten zu verwenden, kann die Kreuzvalidierung verwendet werden, um die Leistung des Modells stabiler und verlässlicher zu messen. Ein verbreiteter Ansatz ist die k-fache Kreuzvalidierung.

```
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score

# Evaluate model
print('Train accuracy:', accuracy_score(y_train, model.predict(X_train))) # Train accuracy:
```

```
print('Test accuracy:', accuracy_score(y_test, model.predict(X_test))) # Test accuracy: 0.8

# Cross validation
scores = cross_val_score(model, X, y, cv=5)
print('Cross validation scores:', scores) # Cross validation scores: [0.72067039 0.79775281
```

Nun haben wir für unseres Modell die Daten vorbereitet, das Modell mit der Daten trainiert und bewertet.

References

1. Peng, Junjie & Jury, Elizabeth & Dönnes, Pierre & Ciurtin, Coziana. (2021). Machine Learning Techniques for Personalised Medicine Approaches in Immune-Mediated Chronic Inflammatory Diseases: Applications and Challenges. Frontiers in Pharmacology. 12. 10.3389/fphar.2021.720694.
2. Fisher,R. A.. (1988). Iris. UCI Machine Learning Repository. https://doi.org/10.24432/C56C76.
3. Morissette, Laurence & Chartier, Sylvain. (2013). The k-means clustering technique: General considerations and implementation in Mathematica. Tutorials in Quantitative Methods for Psychology. 9. 15-24. 10.20982/tqmp.09.1.p015.
4. Jang, Beakcheol & Kim, Myeonghwi & Harerimana, Gaspard & Kim, Jong. (2019). Q-Learning Algorithms: A Comprehensive Classification and Applications. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2941229.
5. MLOps: Continuous delivery and automation pipelines in machine learning