

Projekt – Künstliche Intelligenz, Machine Learning

Entwickeln Sie eine KI-Anwendung, die einen ausgewählten Algorithmus (z.B. Logistische Regression, KNN, Entscheidungsbäume, Naiver Bayes, Neuronale Netze, ...) verwendet, um Vorhersagen zu treffen. Es steht Ihnen frei, einen Regressor oder Klassifikator zu entwickeln.

Dokumentation: Sie dokumentieren Ihre Arbeit in LaTeX in je einem Kapitel zu den einzelnen folgend aufgelisteten Schritten.

1. **Ideenfindung und Auswahl eines geeigneten Datensatzes:** Überlegen Sie sich einen Anwendungsfall und suchen Sie einen dazu passenden Datensatz. Das könnte z.B. die Klassifizierung von Tierarten, Lebensmittel, Immobilienpreise oder dergleichen sein. Hier eine Auswahl an Quellen, wo sie kostenlose Datensätze finden können:

Google Dataset Search	https://datasetsearch.research.google.com/
UCI Machine Learning Repository	https://archive.ics.uci.edu/
Kaggle	https://www.kaggle.com/datasets
Awesome Public Datasets	https://github.com/awesomedata/awesome-public-datasets

Mögliche Ideen: Schrifterkennung z.B. mit dem mit MNIST-Datensatz (Ziffern von 0-9), Bildklassifizierung, Wettervorhersage, Songvorschläge in Player wie Spotify etc.)

2. **Datensatzaufbereitung:** Bereinigen Sie den Datensatz von fehlenden Werten, kodieren Sie kategorische Variablen und skalieren Sie numerische Werte falls erforderlich. Nutzen Sie dafür auch *LLMs* (*Large Language Models* wie Copilot, BlackboxAI, ChatGPT, usw.) und dokumentieren Sie die Verwendung entsprechend. Visualisieren Sie den Datensatz mit geeigneten Tools/Bibliotheken wie z.B. Matplotlib, um Muster zu erkennen.
3. **Aufteilung des Datensatzes:** Teilen Sie den Datensatz in Trainings- und Testdaten auf.
4. **Algorithmus implementieren:** Wählen Sie einen Algorithmus (z.B. Logistische Regression, KNN, Entscheidungsbäume, Naiver Bayes, Neuronale Netze ...) und implementieren Sie diesen mithilfe von Bibliotheken wie z.B. scikit-learn, PyTorch oder TensorFlow.
5. **Modelltraining:** Trainieren Sie das Modell mit den Trainingsdaten und optimieren Sie die Parameter, um die Leistung zu verbessern.
6. **Modellbewertung:** Testen Sie das trainierte Modell mit den Testdaten, berechnen Sie Metriken wie Genauigkeit und visualisieren Sie die Ergebnisse.
7. **Anwendung des Modells:** Implementieren Sie eine Funktion, die es ermöglicht, neue Daten in das trainierte Modell einzuspeisen und Vorhersagen zu generieren.

Mindestanforderungen je nach gewählter Gruppengrösse:

Einzelarbeit: Modellbewertung kann «kurz» gehalten werden. Präsentationszeit 4-6 Min.

Zweiergruppen: Ausführliche Modellbewertung erfordert inkl. theoretischem Vergleich mit anderen KI-Modellen.

Dreiergruppen: Zwei Modelle im Vergleich (inkl. Auswertung der Metriken).

Folgende Bibliotheken können Ihnen bei der Modellerstellung behilflich sein.

- Für die Modelle: diverse Libraries, welche im ki-kurs genutzt wurden. (Neuronale Netze: *torch*, *pytorch*, *torchvision*, ...)
- Plots: *matplotlib*
- Graphical User Interface (GUI) für die Demo: *tkinter*, *customtkinter*
- Arbeit mit Bildern: *PIL (Pillow)*

Die Nutzung von generativer KI ist erlaubt, soll aber in einer entsprechenden Tabelle im Quellenverzeichnis der Dokumentation genügend vermerkt sein.

Bewertungskriterien für die KI-Anwendung:

Die Bewertung des KI-Projektes setzt sich folgendermassen zusammen:

Inhalt	Bewertung
Gruppenbewertung: <ul style="list-style-type: none"> - Programmierung einer Anwendung, die KI verwendet inkl. Dokumentation der Arbeit mit LaTeX - Präsentation des Projektes 	Punkte (max. 120)
Einzelbewertung: <ul style="list-style-type: none"> - Code Review der entwickelten Anwendung 	Persönlicher Faktor 0 -1.0

Die gruppenweisen erreichten Punkte werden mit einem persönlichen Faktor des Code-Reviews multipliziert, was ihre Individuellen Punkte und die daraus entstehende Note ergibt.

Gruppenbewertung:

Kriterium	Punkte max.	Erreichte Punkte
Datensatz: <ul style="list-style-type: none"> - Auswahl: Eignung des gewählten Datensatzes für den Anwendungsfall. Aufbereitung: <ul style="list-style-type: none"> - Vollständige Bereinigung von fehlenden Werten. - Korrekte Kodierung kategorischer Variablen. - Angemessenes Skalieren numerischer Werte. 	10	
Aufteilung des Datensatzes: <ul style="list-style-type: none"> - Korrekte Aufteilung in Trainings- und Testdaten. - Verwendung einer angemessenen Aufteilungsrate (z.B. 70-30). 	5	
Implementation des Algorithmus: <ul style="list-style-type: none"> - Klar strukturierte und ausführlich kommentierte Implementierung des Algorithmus. - Verwendung geeigneter Bibliotheken und Funktionen. 	20	
Modelltraining: <ul style="list-style-type: none"> - Erfolgreiches Training des Modells mit Trainingsdaten. - Optimieren von Parametern für verbesserte Leistung. 	15	
Modellbewertung: <ul style="list-style-type: none"> - Genauigkeitsberechnung unter Verwendung von Testdaten. - Visualisierung von Ergebnissen 	10	
Anwendung des Modells: <ul style="list-style-type: none"> - Erfolgreiche Implementierung der Funktion zur Vorhersage neuer Daten. - Demonstration der Anwendung anhand von Beispieldaten. 	20	
Dokumentation LaTeX <ul style="list-style-type: none"> - Die einzelnen Schritte des Projektes wurden sinnvoll dokumentiert. - Die ML-Theorie wurde sinnvoll miteinbezogen. 	20	

Präsentation <ul style="list-style-type: none"> - Das Datenset und der Algorithmus werden präsentiert und anhand einer Demo veranschaulicht. - Die Zeitbegrenzung von 8-10 Minuten wird eingehalten. - Die Fragen werden sinnvoll beantwortet. 	20	
--	----	--

Bewertungskriterien für das Code Review (Faktor 0-1):

Kriterium	Wert	Erreichter Wert
Standards und Struktur: <ul style="list-style-type: none"> - Der Code sollte gut kommentiert sein, um die Programmlogik und Funktionalität zu erklären. - Debugging von Fehlern wird an geeigneten Stellen im Code kommentiert. - Der Code sollte bewährten Programmierpraktiken (z.B. OOP oder Funktionaler Programmierung ...) folgen, einschliesslich guter Benennung und Strukturierung. 	0.3	
Wissen: <ul style="list-style-type: none"> - Der Code kann verständlich erklärt werden und Rückfragen können adäquat beantwortet werden. - Anpassungen können vorgenommen werden. - Die Orientierung im Code ist gegeben. 	0.7	
Erreichter individueller Faktor:		

Abgabe:

Die Abgabe erfolgt über den Abgabeordner auf Teams in Form einer Zip-Datei (*projektname_abgabe.zip*), welche die Dokumentation (*projektname_dokumentation.pdf*) alle Codes, Auswertungen, und das Datenset enthalten.

Note:

Die Note 6 wird mit 105 von maximal 120 Punkten erreicht. Für die Berechnung der individuellen Punktzahl werden die erreichten Gruppenpunkte mit dem Review-Faktor multipliziert. Die Note wird anhand der folgenden Formel errechnet:

$$\text{Note} = \frac{\text{Anzahl Punkte}}{\text{Punktemaximum}} \times 5 + 1$$