KI1: Einführung in die künstliche Intelligenz

Wintersemester 2022/23

Klausur

HOCHSCHULE ANSBACH

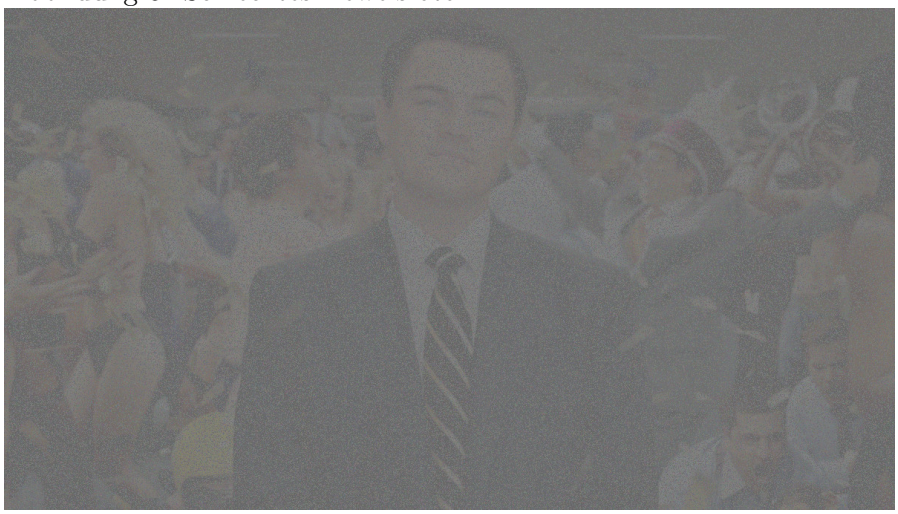
Die vorgefertigten Codeteile liegen im aus den Übungen bekannten Gitlab unter https://git-kik.hs-ansbach.de/2022_wise_ki1/2022_ki1_klausur. Der Projektname darf beim Fork in die Gruppe <benutzername>_and_sg nicht geändert werden.
Alternativ kann das Projekt vor Klausurbeginn per USB Stick oder auf Papier bezogen werden.

Zur Beantwortung der Theoriefragen ist die Datei abgabe.csv zu verwenden. Die Programmieraufgaben sind im jeweiligen Code zu bearbeiten.

Aufgaben:

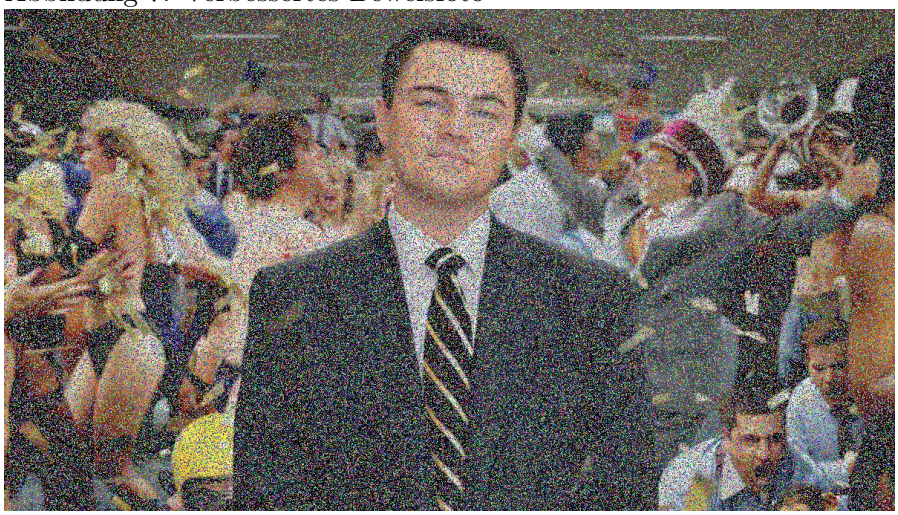
1. Fülle die allgemeinen Angaben in *abgabe.csv* unter Aufgabe1_Organisatorisches korrekt aus. [1 Punkt]
2. Welche der folgenden Punkte sind für die in der Vorlesung besprochene Definition von Machine Learning notwendig? [2 Punkte]
   - A: Neuronale Netze
   - B: Parameter
   - C: Anpassen der Parameter an Trainingsdaten
   - D: Modell
3. Ein hypothetisches Nachtsichtgerät nimmt gleichzeitig ein Farbbild und ein Bild von einem Infrarotkanal auf und speichert diese Informationen in einem kombinierten Bild. Alle Bilder werden mit einer Auflösung von 1000x500 Pixeln aufgenommen. Aus wie vielen Werten besteht ein solches kombiniertes Farb- und Infrarotbild? [Eine Zahl oder ganze Rechnung, 2 Punkte]
4. Was unterscheidet Klassifikation und Regression? [2-3 Sätze, 2 Punkte]
5. Warum evaluiert man die Performance eines Machine Learning Algorithmus möglichst nie auf Daten, auf denen der Algorithmus trainiert wurde? Wie ist die korrekte Herangehensweise? [Ca. drei Sätze, 3 Punkte]

Abbildung 6: Schlechtes Beweisfoto



6. Abbildung 6 zeigt das einzige Foto der Person, die viel Geld auf unserer Weihnachtsfeier verschwendet hat. Es hat nur Pixelwerte zwischen 0 und 100 anstatt der üblichen 0 bis 255. Dadurch ist die Person nicht ausreichend zu erkennen.
   Nenne und beschreibe eine in der Vorlesung besprochene Vorverarbeitungsmethode, mit der das Foto verbessert werden kann. [Ca. 2 Sätze, 2 Punkte]

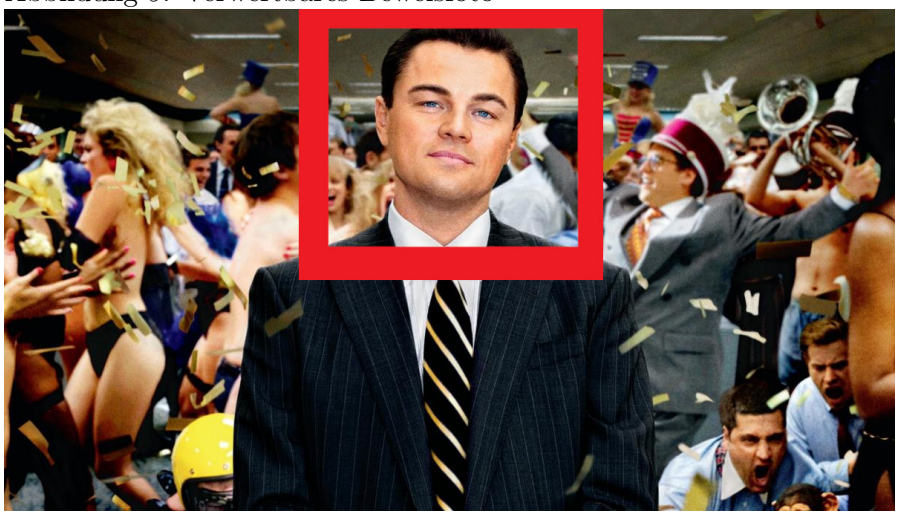
Abbildung 7: Verbessertes Beweisfoto



7. Abbildung 7 zeigt das Ergebnis. Besser, aber Bildfehler machen eine Identifizierung unsicher.
   Es handelt sich bei den Fehlern immer um einzelne Pixel, die zufällige Werte haben. Wir wollen die Ausreißer in jeder Nachbarschaft ersetzen, ohne die korrekten Pixelwerte durch Anteile der Bildfehler zu verfälschen.
   Nenne und beschreibe einen Filter, den wir in der Vorlesung kennen gelernt haben, der dafür geeignet ist. [Ca. 2 Sätze, 2 Punkte]

8. Handelt es sich bei dem in Aufgabe 7 benötigten Filter um einen linearen oder nichtlinearen Filter? Warum? [Stichpunkt, 1 Punkt]

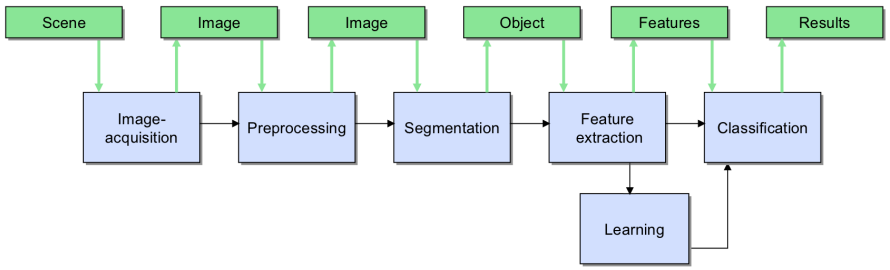
Abbildung 9: Verwertbares Beweisfoto



9. Abbildung 9 zeigt das stark verbesserte Beweisfoto nach unserer und anderen Vorverarbeitungen.
   Beschreibe, welche Information hier nach der Bildverbesserung durch die rote Markierung noch hinzugefügt wurde. [1-2 Sätze, 1 Punkt]

10. Zu welchem Schritt der Pipeline in Abbildung 10 gehört das Hinzufügen der roten Markierung in Abbildung 9? [Ein Wort, 1 Punkt]

Abbildung 10: Die ML Pipeline



11. Der Abgleich mit einem auf Personenerkennung vortrainierten Neuronalen Netz hat ergeben, dass Leonardo nicht das Budget der nächsten Weihnachtsfeier der Hochschule plant.
    Das dabei verwendete Netz liefert den passenden Namen zu einem Bild(ausschnitt). Wurde das Netz mit höchster Wahrscheinlichkeit supervised, semi-supervised oder unsupervised trainiert? [Ein Wort, 1 Punkt]

12. Begründe die Antwort von Aufgabe 11. [Ca. 2 Sätze, 2 Punkte]

13. In einer Sportart ohne Unentschieden soll Sieg und Niederlage vorhergesagt werden. Tabelle 13 zeigt die beobachteten Ergebnisse. Die "positive" Klasse ist als Sieg definiert.

    Tabelle 13: Confusion Matrix

	Sieg	Niederlage	Gesamt
Sieg vorhergesagt	90	10	100
Niederlage vorhergesagt	20	180	200
Gesamt	110	190	300

    Berechne die False Positive Rate. [Eine Zahl oder ganze Rechnung, 1 Punkt]

14. Berechne die Accuracy von Tabelle 13. [Eine Zahl oder ganze Rechnung, 1 Punkt]

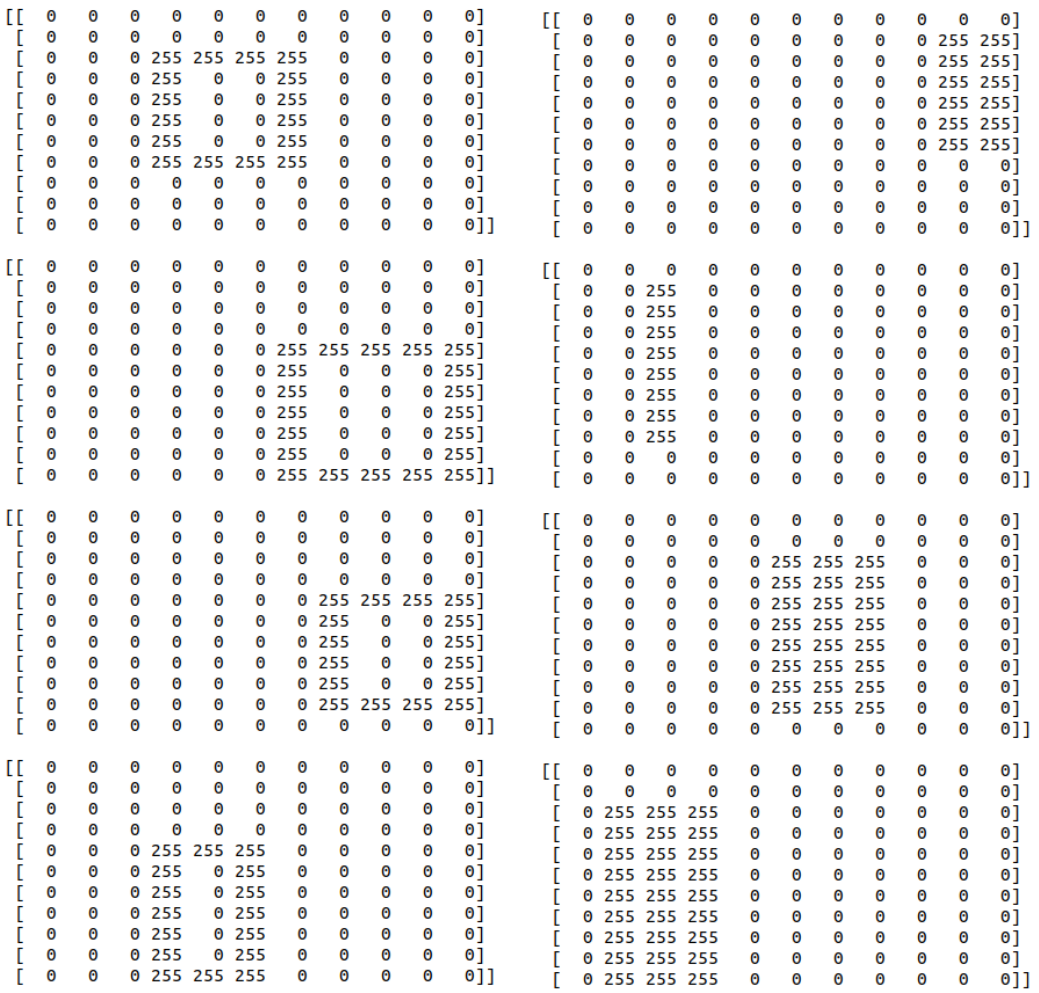
15. Programmieren in *split_train_test.py*: Schreibe eine Funktion *split_data*, die als Argumente eine Liste von Beispielen und eine Zahl *ratio* zwischen 0 und 1 bekommt. Die Funktion teilt die Beispiele zufällig gemäß des *ratios* in zwei Teile auf. Die Funktion gibt die beiden Listen zurück. Ein Ratio von 0.9 führt dazu, dass 90% der Daten in der ersten Liste enthalten sind.
    Hinweise: *split_main.py* verwendet den zu schreibenden Code korrekt und sollte zum Testen ausgeführt werden. [4 Punkte]

16. Programmieren in *features_compute.py*: Es müssen Bilder von "I" und "O" unterschieden werden. Die Funktion *compute_feature* soll ein einzelnes Feature berechnen, das diese Unterscheidung ermöglicht. Die Funktion bekommt ein zweidimensionales Numpy Array als Argument übergeben und gibt das Feature als float zurück. Hinweise: Beispiele der Inputdaten sind in Abbildung 16 visualisiert. *feature_main.py* verwendet den zu schreibenden Code korrekt und sollte zum Testen ausgeführt werden. [6 Punkte]

17. Programmieren in *features_compute.py*: Schreibe als erste Zeile in der Funktion *compute_features* einen Kommentar, der die Idee des Features beschreibt. Auch möglich ohne Aufgabe 16 zu bearbeiten! [1 Punkt]

18. Programmieren in *features_compute.py*: Die Funktion *good_threshold* legt einen Schwellwert fest. Dieser Schwellwert wird in *feature_main.py* verwendet, um Beispiele anhand des in Aufgabe 16 berechneten Features einzuteilen. Alle Beispiele unter dem Schwellwert werden als Klasse 1 ("I") erkannt, alle Beispiele über dem Schwellwert als Klasse 0 ("O"). Passe den Schwellwert so an, dass möglichst viele Beispiele richtig klassifiziert werden.
    [> 60% Trefferquote: 1 Punkt]
    [> 95% Trefferquote: 2 Punkte]

    Abbildung 16: Links Beispiele für Klasse "O" - Rechts Beispiele für Klasse "I"



Für volle Punktzahl muss jeder Code ohne Fehler importier- und ausführbar sein [1 Punkt]. Die Abgabe des Source Codes sowie der Datei abgabe.csv soll wie aus den Übungsaufgaben bekannt als Commit im geforkten Gitlabprojekt erfolgen [1 Punkt]. Alternativ kann auf Wunsch per USB Stick oder in Papierform abgegeben werden.

Viel Erfolg! Bearbeitungszeit 60 Minuten