Einführung in die künstliche Intelligenz

Wintersemester 2022/23

Übung 6 - Neuronales Netz und Confusion Matrix

HOCHSCHULE ANSBACH

Das Projekt liegt unter https://git-kik.hs-ansbach.de/2023_ki1/2022_ki1_a6.
Der Projektname darf beim Fork in die Gruppe <benutzername>_and_dozent nicht geändert werden.

In der letzten Übung werden wir ein neuronales Netz trainieren, um Bilder von Bananen zu klassifizieren. Ebenfalls implementieren wir einige verbreitete Metriken zur bewertung von Klassifikationsergebnissen. Die benötigten Definitionen aller Größen wie Confusion Matrix, False Positive Rate, etc. finden sich unter https://en.wikipedia.org/wiki/Confusion_matrix und im Skript. Zur Veranschaulichung der Werte ist die Klassifikation absichtlich nicht perfekt eingestellt und soll auch so bleiben.

1. in *machine_learning.py*
   - Die Funktion *featureberechnung_alle_bilder* soll von allen Bildern die Features und Label berechnen und jeweils in einer Liste zurückgeben.
   - Die Funktion *model_anwenden* soll so ergänzt werden, dass das Sklearn Model auf die Feature angewendet wird und ein Ergebnis herauskommt. Die Dokumentation unter https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html zeigt mehrere Beispiele. Wir benötigen den Funktionsaufruf, der nicht die Wahrscheinlichkeiten, sondern die wahrscheinlichsten Klassen als Ergebnis zurückliefert.

2. in *auswertung.py*
   - Die Funktion *confusion_matrix* berechnet aus den wahren Labeln und den Ergebnissen der Klassifikation eine Confusion Matrix (selten auch Wahrheitsmatrix). Die Konvention der Matrix ist im Kommentar vorgegeben.
   - *false_positive_rate* berechnet, wie viele der Nicht-Bananenbilder als Bananen klassifiziert wurden. Sie kann, falls gewünscht, ähnlich der vorgegebenen Funktion *true_positive_rate* implementiert werden.
   - *true_negative_rate* berechnet, wie viele der Nicht-Bananen korrekt als Nicht-Bananen klassifiziert wurden.
   - *false_negative_rate* berechnet, wie viele der Bananen fälschlicherweise als Nicht-Bananen erkannt wurden.
   - *precision* berechnet, wie viele der als Bananen erkannten Beispiele wirklich Bananen waren.

3. OPTIONAL in *featureberechnung.py*

   - Hier kann ein eigenes Feature ergänzt werden. Es kann auch das eigene Feature aus der letzten Übung verwendet werden, falls vorhanden

4. OPTIONAL in *machine_learning.py*

   • Hier kannst du auch einen anderen Algorithmus trainieren.
   Überlege dazu, welchen du verwenden möchtest (welche wären geeignet?).
   Importiere den Algorithmus aus sklearn und schreibe eine neue Funktion zum Trainieren der Methode.

Abgabeschluss ist der 18.12.2022 um 23:59:59