Aufgabe 1: Praxisbeispiel KNN, Rastersuche und Kreuzvalidierung

In dieser Aufgabe lernen Sie den Iris Datensatz kennen, trainieren eine Reihe von KNN-Modellen mit variierender Nachbarschaftsgröße und setzen dabei Kreuzvalidierung ein.

1. Laden Sie den Iris-Datensatz mit Hilfe `sklearn.datasets.load_iris` in die Variablen `X` (Features) und `y` (Zielgröße, Target).
2. Was sind die einzelnen Features und was sind die Target-Klassen?
3. Schreiben Sie eine Schleife, in welcher Sie über die Werte für K von 1 bis 100 iterieren (Rastersuche). Im Schleifenkörper erstellen Sie jeweils ein `sklearn.neighbors.KNeighborsClassifier` Modell und berechnen mit Hilfe von `sklearn.model_selection.cross_val_score` die Genauigkeit mit Hilfe von 5-facher Kreuzvalidierung. Nehmen Sie hierfür den Mittelwert. Speichern Sie diese Werte für jede Belegung von K in eine Liste.
4. Plotten Sie mit Hilfe von `matplotlib` die Scores auf die K-Werte.
5. Für welche(n) Wert(e) von K ist die Genauigkeit maximal?
6. Interpretieren Sie den Plot hinsichtlich möglicher Über- und Unteranpassung.

Aufgabe 2: Praxisbeispiel Entscheidungsbäume

Wir wollen nun einen Entscheidungsbaum trainieren, welcher von den Features im `Auto.csv` Datensatz nach der Anzahl der Zylinder klassifiziert.

1. Laden Sie dazu den `Auto.csv` Datensatz in einen DataFrame und korrigieren Sie ggf. fehlende bzw. fehlerhafte Daten.
2. Laden Sie in den Zielvektor `y` die Zylinderzahlen und wählen Sie als Features `X` alle vorhandenen Features außer `name` und `cylinders`.
3. Trainieren Sie einen `sklearn.tree.DecisionTreeClassifier` mit Hilfe 3-facher Kreuzvalidierung.
4. Welche Genauigkeit können Sie erreichen?

Aufgabe 3: Praxisbeispiel Random Forest

Verwenden Sie den Iris Datensatz, den Sie bereits aus vorhergehenden Übungsaufgaben kennen.

1. Laden Sie den Iris-Datensatz mit Hilfe `sklearn.datasets.load_iris` in die Variablen `X` (Features) und `y` (Zielgröße, Target).
2. In der Hilfe von Scikit-Learn finden Sie Informationen zum Tracken des OOB-Fehler, siehe `https://scikit-learn.org/stable/auto_examples/ensemble/plot_ensemble_oob.html`.
3. Erstellen Sie jeweils ein `sklearn.ensemble.RandomForestClassifier` Modell für verschiedene Anzahlen an Bäumen (`n_estimators`) zwischen 15 und 150 und tracken Sie dabei den OOB-Fehler.
   Hinweis: Das Argument `max_features` können Sie mit dem Default-Wert verwenden und brauchen dies nicht zu variieren.
4. Wählen Sie einen sinnvollen Wert für `n_estimators` und geben Sie für dieses Modell die Feature Importances (*property* `feature_importances_`) aus und visualisieren Sie diese.