Hochschule Bochum
Bochum University
of Applied Sciences

Fachbereich Wirtschaft
Prof. Dr. Christian Bockermann

Data Science

Wintersemester 2021/2022

Übungsblatt 5

Aufgabe 1 (Klassifikation mit Bäumen)

1. Starten Sie mit dem Notebook `V5-Entscheidungsbaum-sklearn`. Das Notebook finden Sie in `Kurse/DataScience1`. Beim Erstellen des Modells können sie dem Entscheidungsbaum eine maximale Tiefe mitgeben:

   ```
   m = DecisionTreeClassifier(max_depth=2)
   ```

   Probieren Sie verschiedene Werte für `max_depth` auf - wie verhält sich das Modell? (Trainingsfehler? Aussehen des Baumes?)

   Hinweis: Das Modul `datascience` enthält die Funktion `show_tree`, mit der Sie sich leicht den entstandenen Baum anzeigen lassen können.

2. Erzeugen Sie im Iris-Datensatz eine neue Spalte `label`. Diese Spalte soll eine 1 enthalten, wenn `species` gleich `versicolor` ist, eine 2, für die Klasse `virginica` und 0 sonst (`setosa`).

   Was passiert, wenn Sie die Spalte `label` mit in den Datensatz `X` zum Trainieren des Modells einbauen? Probieren Sie es aus!

   Hinweis: Eine Spalte kann `df[neueSpalte] = df[alteSpalte]` kopiert werden. Auf Übungsblatt 4 hatten wir mit `replace(..)` Werte einer Spalte in einem Datensatz ersetzt.

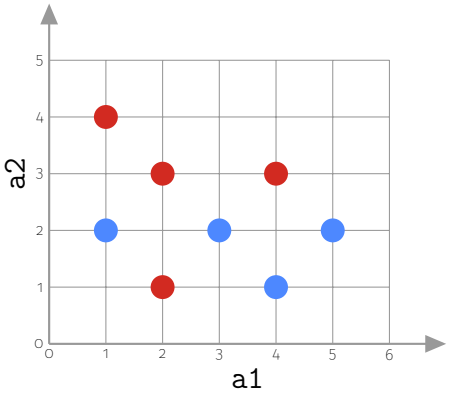
3. Erzeugen Sie in `X` eine Spalte mit dem Namen `ID`. Die Spalte soll einfach für jedes Beispiel eine Nummer enthalten (z.B. die Zeilennummer).

   Was passiert, wenn Sie das Modell jetzt auf den ursprünglichen 4 Spalten und der ID Spalte trainieren? Probieren Sie es aus!

   Hinweis: Ein neues Series-Objekt können Sie ja aus einer Liste von Werten erzeugen. Mit `range(n)` erzeugen Sie eine Sequenz von 0 bin $n-1$. Mit `list(seq)` erzeugen Sie eine Liste aus einer Sequenz.

Aufgabe 2 (Noch mehr Bäume)

Betrachten Sie die Daten in der folgenden Abbildung. Es sind lediglich die Attribute `a1` und `a2` gegeben. Die Farben markieren die Klasse – entweder **rot** oder **blau**.

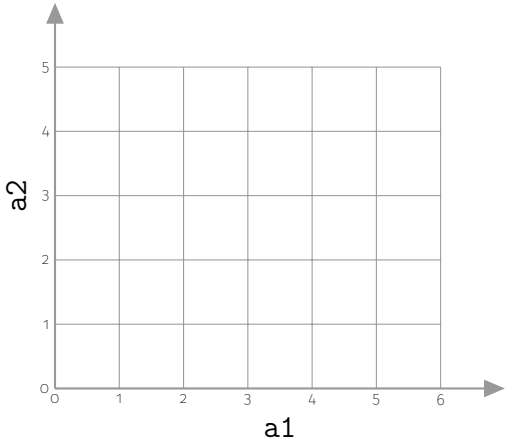


1. Wo wird ihr Entscheidungsbaum den ersten Split machen? Wie sieht der Gini-Index für den ersten Split aus?

   Nutzen Sie dazu ggf. ein Jupyter Notebook und die Funktion auf Folie 14 (Foliensatz 5).

2. Was ist der *schlechteste* Fall für einen Split in einem Attribut `a` mit Wert `v`?

   Überlegen Sie sich einen Datensatz, der für einen Entscheidungsbaum maximal schwer ist. Tragen Sie dazu im folgenden Koordinatensystem 16 Datenpunkte ein, von denen acht zur Klasse **Kreis** und die anderen acht zur Klasse **Quadrat** gehören.



   Vorschlag: Tragen Sie die Punkte als kleinen Kreis bzw. kleines Quadrat im Koordinatensystem ein und nutzen Sie dafür ganzzahlige Werte für `a1` bzw. `a2`.

Aufgabe 3 * (Entscheidungsbaum mit Datenkonvertierung)

Ein weiterer klassischer Datensatz ist der *Golf* Datensatz (manchmal auch mit anderen Spielen/Aktivitäten benutzt). Es geht bei dem Datensatz darum ob das Spiel (*Golf, Seglen, etc.*) aufgrund der aktuellen Wetterlage möglich ist `Play=yes` oder nicht `Play=no`. Der vollständige Datensatz als DataFrame ist:

	Outlook	Temperature	Humidity	Wind	Play
0	sunny	85	85	False	no
1	sunny	80	90	True	no
2	overcast	83	78	False	yes
3	rain	70	96	False	yes
4	rain	68	80	False	yes
5	rain	65	70	True	no
6	overcast	64	65	True	yes
7	sunny	72	95	False	no
8	sunny	69	70	False	yes
9	rain	75	80	False	yes
10	sunny	75	70	True	yes
11	overcast	72	90	True	yes
12	overcast	81	75	False	yes
13	rain	71	80	True	no

Den Datensatz finden Sie wie gewohnt unter `Kurse/DataScience1/data`.

1. Trainieren Sie mit `sklearn` einen Entscheidungsbaum auf dem Golf Datensatz. Das Attribut `Outlook` können Sie dabei zunächst weglassen. Testen Sie ihr Modell auf Test-Daten. nutzen Sie dazu 10% des Datensatzes als Test-Daten.

2. Die `sklearn` Modelle können nicht mit nicht-numerischen Attributen umgeben, also insbesondere nominalen Werten wie in der Spalte `Outlook`. Ändern Sie das Attribut `Outlook` um in ein *numerisches* Attribut und trainieren Sie ihr Modell erneut.

   Hat sich die Modell-Güte verbessert?

3. Der Datensatz hat nur sehr wenige Werte, was die Aufteilung in Trainings- und Test-Daten erschwert. Das `sklearn`-Modul unterstützt die *Kreuzvalidierung* (vgl. Foliensatz 4, Folie 33):

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

# Modell anlegen
m = DecisionTreeClassifier()

scores = cross_val_score(m, X, y, cv=5)
print("Accuracy: {} (+/- {}".format(scores.mean(), scores.
                                    std() * 2))
```

   Wie ist die gemittelte `accuracy` bei der Kreuzvalidierung? Probieren Sie verschiedene Parameterwerte für `max_depth` aus!