1. (4 Punkte) Berechne die Gini Impurity der Gruppen.
   (a) 6-mal Ja, 2-mal Nein
   (b) 5-mal Ja, 5-mal Nein
   (c) 2-mal A, 1-mal B, 1-mal C
   (d) Bei zwei Klassen ist die Gini Impurity höchstens 0,5. In c) ist sie größer. Wie groß kann sie bei drei Klassen höchstens werden?

Lösung:

(a) $1 - \left(\frac{6}{8}\right)^2 - \left(\frac{2}{8}\right)^2 = 1 - 0{,}5625 - 0{,}0625 = 0{,}375$

(b) $1 - 0{,}5^2 - 0{,}5^2 = 0{,}5$

(c) $1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{4}\right)^2 - \left(\frac{1}{4}\right)^2 = 0{,}625$

(d) Am größten ist sie, wenn alle drei Klassen gleich häufig sind: $1 - 3 \cdot \left(\frac{1}{3}\right)^2 = \frac{2}{3} \approx 0{,}667$.

2. (3 Punkte) In einer Urne liegen 4 rote, 4 blaue und 2 grüne Kugeln. Man zieht zweimal mit Zurücklegen.
   (a) Wie groß ist die Wahrscheinlichkeit, zwei Kugeln verschiedener Farbe zu ziehen?
   (b) Was hat das mit der Gini Impurity zu tun?

Lösung:

(a) Gleiche Farbe: $0{,}4^2 + 0{,}4^2 + 0{,}2^2 = 0{,}36$. Verschiedene Farben: $1 - 0{,}36 = 0{,}64$.

(b) Das ist genau die Gini Impurity der Sammlung: $1 - p_1^2 - p_2^2 - p_3^2$ mit den Anteilen p_i der Farben. Je gemischter die Sammlung, desto wahrscheinlicher zieht man zwei verschiedene.

3. (6 Punkte) Eine Schülerin hat an acht Tagen notiert, ob es geregnet hat, ob ihr Ziel nah oder weit war und ob sie mit dem Fahrrad gefahren ist.

Tag	Regen	Entfernung	Fahrrad
1	nein	nah	ja
2	nein	weit	ja
3	nein	nah	ja
4	nein	weit	ja
5	ja	nah	ja
6	ja	weit	nein
7	ja	weit	nein
8	ja	weit	nein

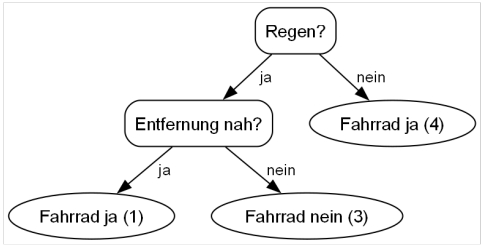
(a) Berechne den gewichteten Gini für die Fragen „Regen?“ und „Entfernung nah?“. Welche Frage kommt in die Wurzel des Entscheidungsbaums?

(b) Vervollständige den Entscheidungsbaum, bis alle Blätter rein sind. Zeichne ihn.

Lösung:

(a) Regen ja: ja, nein, nein, nein, Gini $= 1-\left(\frac{1}{4}\right)^2-\left(\frac{3}{4}\right)^2 = 0{,}375$. Regen nein: viermal ja, Gini $= 0$. Gewichtet: $\frac{4}{8}\cdot 0{,}375 + \frac{4}{8}\cdot 0 = 0{,}1875$.
Entfernung nah: dreimal ja, Gini $= 0$. Entfernung weit: ja, ja, nein, nein, nein, Gini $= 1-0{,}4^2-0{,}6^2 = 0{,}48$. Gewichtet: $\frac{5}{8}\cdot 0{,}48 = 0{,}3$.
„Regen?“ hat den kleineren Wert und kommt in die Wurzel.

(b) Ohne Regen ist die Gruppe rein (Fahrrad ja). Bei Regen trennt die Entfernung vollständig (gewichteter Gini 0):



4. (5 Punkte) Für sechs Schüler ist bekannt, wie viele Stunden sie für einen Test gelernt haben und ob sie bestanden haben (J/N).

Stunden	1	2	3	5	6	8
bestanden	N	N	J	J	N	J

(a) Welche Schwellen s kommen für eine Frage „Stunden $< s$?“ in Frage?

(b) Berechne für jede Schwelle den gewichteten Gini. Welche Frage kommt in die Wurzel?

Lösung:

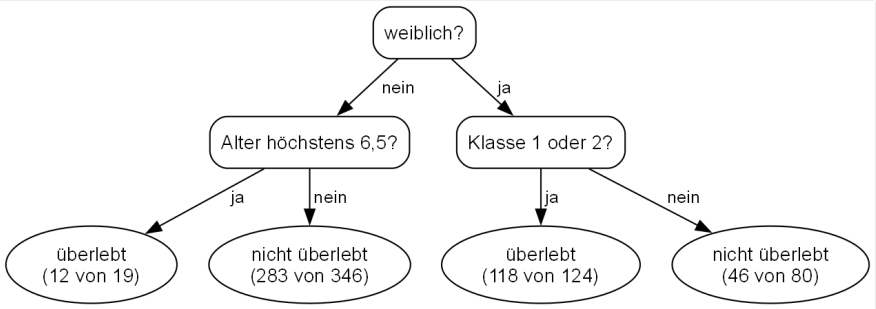
(a) Die Mitten zwischen benachbarten Werten: 1,5, 2,5, 4, 5,5, 7.

(b)

Schwelle	Aufteilung	gewichteter Gini
1,5	N | NJJNJ	0,4
2,5	NN | JJNJ	0,25
4	NNJ | JNJ	0,444
5,5	NNJJ | NJ	0,5
7	NNJJN | J	0,4

Am besten ist „Stunden $< 2{,}5$?“ mit 0,25: Wer weniger als 2,5 Stunden gelernt hat, hat nicht bestanden (rein), die anderen sind gemischt (J, J, N, J).

5. (5 Punkte) Ein Entscheidungsbaum wurde mit den Daten von 569 Passagieren der Titanic trainiert. In den Blättern steht, wie viele der Trainingsbeispiele dort zur vorhergesagten Klasse gehören.



(a) Was sagt der Baum voraus für einen 30-jährigen Mann in der 1. Klasse, ein 5-jähriges Mädchen in der 3. Klasse und einen 4-jährigen Jungen in der 2. Klasse?

(b) Schreibe den Baum als Python-Funktion `ueberlebt(weiblich, alter, klasse)`, die `True` oder `False` zurückgibt.

(c) Wie sicher ist die Vorhersage für Frauen in der 3. Klasse? Woran sieht man das?

Lösung:

(a) Mann, 30: nicht überlebt. Mädchen, 3. Klasse: nicht überlebt (bei Frauen zählt nur die Klasse). Junge, 4: überlebt.

(b)
```
def ueberlebt(weiblich, alter, klasse):
    if weiblich:
        return klasse <= 2
    else:
        return alter <= 6.5
```

(c) Wenig sicher: In diesem Blatt haben 46 von 80 nicht überlebt, also nur gut die Hälfte (Gini fast 0,5). Der Baum sagt die häufigere Klasse voraus, liegt aber bei 34 der 80 Trainingsbeispiele falsch.

6. (5 Punkte) Für die Titanic-Daten wurden Entscheidungsbäume mit verschiedener größter Tiefe `max_depth` auf 80 % der Daten trainiert und auf den übrigen 20 % getestet:

`max_depth`	1	2	4	8	ohne Grenze
Training	0,777	0,807	0,833	0,891	0,947
Test	0,79	0,783	0,797	0,811	0,783

(a) Beschreibe, was man sieht. Wie nennt man das?

(b) Warum misst man die Genauigkeit nicht einfach auf den Trainingsdaten?

(c) Die Testdaten sind 143 Passagiere. Wie viel macht ein einziger Passagier mehr oder weniger richtig bei der Genauigkeit aus? Was folgt daraus für den Vergleich der Tiefen?

(d) Welchen Baum würdest du nehmen? Begründe.

Lösung:

(a) Die Genauigkeit auf den Trainingsdaten steigt mit der Tiefe bis fast 95 %, auf den Testdaten bleibt sie um 78 bis 81 %. Der große Baum lernt die Trainingsdaten auswendig, auch Zufälligkeiten, die für neue Passagiere nicht gelten: Overfitting.

(b) Auf den Trainingsdaten kann ein Baum ohne Grenze fast alles richtig machen, das sagt nichts darüber, wie gut er bei neuen Fällen ist. Dafür braucht man Daten, die beim Training nicht benutzt wurden.

(c) $1/143 \approx 0{,}007$, also 0,7 Prozentpunkte. Die Unterschiede in der Testzeile sind nur wenige Passagiere und können Zufall sein. Besser vergleicht man mit Kreuzvalidierung (mehrere Aufteilungen, Mittelwert).

(d) Einen kleinen Baum, z.B. Tiefe 2: Er ist auf den Testdaten etwa so gut wie die großen, aber man kann ihn lesen und er überträgt sich eher auf neue Daten.

7. (5 Punkte) Schreibe eine Funktion `beste_schwelle(werte, klassen)`. `werte` ist eine Liste von Zahlen, `klassen` die Liste der zugehörigen Klassen. Die Funktion soll alle Schwellen (Mitten zwischen benachbarten verschiedenen Werten) ausprobieren und die Schwelle mit dem kleinsten gewichteten Gini zusammen mit diesem Wert zurückgeben. Teste mit `werte = [1, 2, 3, 5, 6, 8]` und `klassen = ['N', 'N', 'J', 'J', 'N', 'J']`.

Lösung:

```
def gini(klassen):
    n = len(klassen)
    return 1 - sum((klassen.count(k) / n) ** 2 for k in set(klassen))

def beste_schwelle(werte, klassen):
    sortiert = sorted(set(werte))
    beste, bester_gini = None, 2
    for a, b in zip(sortiert, sortiert[1:]):
        s = (a + b) / 2
        ja = [k for w, k in zip(werte, klassen) if w < s]
        nein = [k for w, k in zip(werte, klassen) if w >= s]
        g = len(ja) / len(werte) * gini(ja) + len(nein) / len(werte) * gini(nein)
        if g < bester_gini:
            beste, bester_gini = s, g
    return beste, bester_gini

print(beste_schwelle([1, 2, 3, 5, 6, 8], ['N', 'N', 'J', 'J', 'N', 'J']))
```

Ausgabe: `(2.5, 0.25)`