To prepare for the next session of the course, students are expected to complete the following exercises. At the beginning of the session, a handout will be distributed containing a list of exercises. Students will be required to tick off their individual achievements as evidence of their work. During the session, students will be randomly selected from those present and will be asked to present (defend) their solutions. A brief interview will follow, during which the student's solution will be discussed. If any doubts arise regarding a student's presentation that they are unable to explain, no credit will be awarded for that exercise in the final grading. To ensure a collective learning experience, the instructor will also review and discuss the presented work with the plenum.

1 Decision Tree

In this part, we are going to have a hands-on view on Decision Trees. Recap what we have discussed during class.

Ex. 24 Decision Tree using the ID.3 method The dataset to be used is taken from [RN21] and can be downloaded using this url.

This dataset illustrates the problem of entering a restaurant and deciding on 9 parameter if one shall wait for a table or not. The following features are given:

- Alternate: Is there a suitable alternative restaurant nearby?
- Bar: does the restaurant have a comfortable bar area to wait in while waiting for the table?
- Fri/Sat: is it Friday or Saturday?
- Hungry: are we hungry at the moment?
- Patrons: How many people are in the restaurant?
- Price: Price range
- Raining: is it raining outside
- Reservation: was a reservation made beforehand
- Type: kind of restaurant
- WaitEstimate: estimation of time by the host to be seated

Besides the Shannon-Entropy, another criterion on which a gain of a feature can be calculated is the so-called Gini Index. With the Gini Index, the probability of all labels on a certain class is multiplied by its negative:

$$H(X) = \sum_{j \in C} p_j \cdot p'_j = \sum_{j \in C} p_j \cdot (1 - p_j)$$

First of, use the given dataset and construct the Decision Tree by hand using the Gini Index. Proceed as done during the lecture. Use your Python skills to speed up (count the probabilities, etc.).

(Hint: The solution to this task can be found in [RN21], Figure 19.3.)

Ex. 25 Decision Tree using sklearn Use the same dataset as above and train a Decision Tree Classifier using sklearn and compare your results. (No reporting of any performance metrics necessary.)

Ex. 26 Regression Tree In this task, train and visualize a regression tree based on the sklearn implementation. Take care, this is a regression task, so the output is a numeric value. Use the following code to generate the training data:

Listing 1: Python example

```
import numpy as np

num_points = 80
interval_dist = 3
```

```
rng = np.random.RandomState(1)
X = np.sort(5 * rng.rand(num_points, 1), axis=0)
y = np.sin(X).ravel()
y[::interval_dist] += 3 * (
    0.5 - rng.rand(int(np.ceil(num_points/interval_dist)))
)
```

- Implement at least two different decision trees with different settings of their maximal depth. Plot the training data alongside the classified testing data. For the testing data, use all values within the trained range. So all decision steps will be visible
- Varying the maximum depth of the decision trees, what do you observe?
- Compare your results to another Decision Tree, where Pruning is performed. An explanation about Pruning Regression Trees can be found here.
- What are the depths of the trees after pruning? What does change with a varying CCP_ALPHA parameter value?
- Compare the visualizations of the Decision Trees. What differences do you observe and what may be the reason for those?

2 Ensemble Methods

Although Decision Trees are not too powerful, they build the foundation for other ensemble methods. In the following you are going to have a look on some ensemble methods.

Let's consider the following datasets:

Listing 2: Python example

```
import numpy as np
from matplotlib.colors import ListedColormap
from sklearn.datasets import make_moons, \
    make_circles, make_classification

X, y = make_classification(
    n_features=2, n_redundant=0, n_informative=2,
    random_state=1, n_clusters_per_class=1
)
rng = np.random.RandomState(2)
X += 2 * rng.uniform(size=X.shape)
linearly_separable = (X, y)

datasets = [
    make_moons(noise=0.3, random_state=0),
    make_circles(noise=0.2, factor=0.5, random_state=1),
    linearly_separable,
]
```

Ex. 27 Visualize the datasets Show, how the datasets are behaving in a two-dimensional plot.

Ex. 28 Decision Boundaries For each of the three datasets, print the decision boundaries in the feature space. Do this for

- a simple Decision Tree,
- a Random Forest Classifier,
- a Bagging Classifier, and
- an AdaBoost Classifier.

Use a mesh grid from matplotlib to encode the feature space. For some of the classifiers, you can predict their probabilities. Use this, wherever possible.

Hint: `predict_proba(...)` has to be preferred over `decision_function(...)`.

Do Short Research about the stated methods (not on a mathematical level) and answer the following questions:

- How do the decision boundaries differ visually and
- how does these relate to the methods itself?

References

[RN21] Stuart Russell and Peter Norvig. *Artificial Intelligence, Global Edition A Modern Approach*. Pearson, 2021. ISBN: 978-1-292-40113-3 (cit. on p. 1).