Assignment 0:

Each one of the datasets has properties which makes them hard to learn. Motivate which of the three problems is most difficult for a decision tree algorithm to learn.

Table 1: True concepts behind the MONK datasets

MONK-1	$(a_1 = a_2) \vee (a_5 = 1)$
MONK-2	$a_i = 1$ for exacly two $i \in \{1, 2, \ldots, 6\}$
MONK-3	$(a_5 = 1 \wedge a_4 = 1) \vee (a_5 \neq 4 \wedge a_2 \neq 3)$

Assignment 1:

The file dtree.py defines a function entropy which calculates the entropy of a dataset. Import this file along with the monks datasets and use it to calculate the entropy of the training datasets.

```
Entropy of training sets:
MONK-1: 1.000000
MONK-2: 0.957117
MONK-3: 0.999806
```

Assignment 2:

Explain entropy for a uniform distribution and a non-uniform distribution, present some example distributions with high and low entropy.

For a **uniform distribution**, all classes are equally likely, which maximizes uncertainty. In the binary case p and 1−p, the entropy becomes

$$H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$$

and it reaches its maximum at p=0.5, where H=1 . Thus, a 50/50 split is the most uncertain and has the highest entropy.

For a **non-uniform distribution**, one class dominates and the label becomes more predictable, so the uncertainty decreases and entropy becomes lower. In the extreme case of a pure node (p=1 and 1−p=0), entropy is 0.

Examples of **high entropy** distributions include (0.5,0.5) with H=1.0 and (0.6,0.4) with H≈0.971. Examples of **low entropy** distributions include (0.9,0.1) with H≈0.469, (0.99,0.01) with H≈0.081, and a pure distribution (1,0) with H=0.

Assignment 3:

Use the function averageGain (defined in dtree.py) to calculate the expected information gain corresponding to each of the six attributes. Note that the attributes are represented as in- stances of the class Attribute (defined in monkdata.py) which you can access via m.attributes[0], ..., m.attributes[5]. Based on

the results, which attribute should be used for splitting the examples at the root node?

Information Gain

Dataset	*a1*	*a2*	*a3*	*a4*	*a5*	*a6*
MONK-1	0.0752725556	0.0058384299	0.0047075666	0.0263116965	**0.2870307497**	0.0007578557
MONK-2	0.0037561773	0.0024584986	0.0010561477	0.0156642472	**0.0172771769**	0.0062476222
MONK-3	0.0071208683	**0.2937361735**	0.0008311140	0.0028918172	0.2559117246	0.0070770260

At the root node, we should choose the attribute with the highest information gain, since it yields the largest reduction in entropy (i.e., the greatest decrease in the weighted average impurity after the split).

Therefore:

- **MONK-1:** choose **a5**
- **MONK-2:** choose **a5**
- **MONK-3:** choose **a2**

Assignment 4:

For splitting we choose the attribute that maximizes the information gain, Eq.3. Looking at Eq.3 how does the entropy of the subsets, Sk , look like when the information gain is maximized? How can we motivate using the information gain as a heuristic for picking an attribute for splitting? Think about reduction in entropy after the split and what the entropy implies.

When information gain is maximized, the resulting subsets Sk tend to be more **pure**, meaning their entropies H(Sk)are typically small (ideally close to 0), which implies that the class distribution within each subset is more certain. Information gain is a natural heuristic for choosing a splitting attribute because entropy measures uncertainty about the class label; information gain directly quantifies how much this uncertainty is reduced by the split. Thus, selecting the attribute with the highest information gain yields the greatest expected reduction in entropy and produces the most informative split at that node.

Assignment 5:

Build the full decision trees for all three Monk datasets using buildTree. Then, use the function check to mea-sure the performance of the decision tree on both the training and test datasets.
For example to built a tree for monk1 and compute the performance on the test data you could use

```
import monkdata as m
import dtree as d
t=d.buildTree(m.monk1, m.attributes);
print(d.check(t, m.monk1test))
```

Compute the train and test set errors for the three Monk datasets for the full trees. Were your assumptions about the datasets correct? Explain the results you get for the training and test datasets.

	E_{train}	E_{test}

MONK-1	0.000000	0.171296
MONK-2	0.000000	0.307870
MONK-3	0.000000	0.055556

All three full decision trees achieve zero training error because an unrestricted ID3 tree can keep splitting until the leaves become (almost) perfectly pure, effectively memorizing the training set. This high model capacity typically leads to overfitting, which is reflected in the test errors. MONK-2 has the largest test error (0.307870) since its target concept is a global counting rule ("exactly two attributes equal 1"), for which single-attribute splits provide little reduction in uncertainty; the greedy information-gain criterion therefore tends to build deep and complex trees that generalize poorly. MONK-1 yields an intermediate test error (0.171296): the underlying rule is relatively simple, but the full tree can still overfit to idiosyncrasies of the training data. MONK-3 shows the best test performance (0.055556), indicating that its structure is easier for a decision tree to capture; however, because MONK-3 contains noise, a full tree may still fit noise, and pruning is expected to further improve or stabilize generalization. Overall, the observed test errors are consistent with the expectation that MONK-2 is the most difficult problem for greedy decision-tree learning.

Assignment 6:

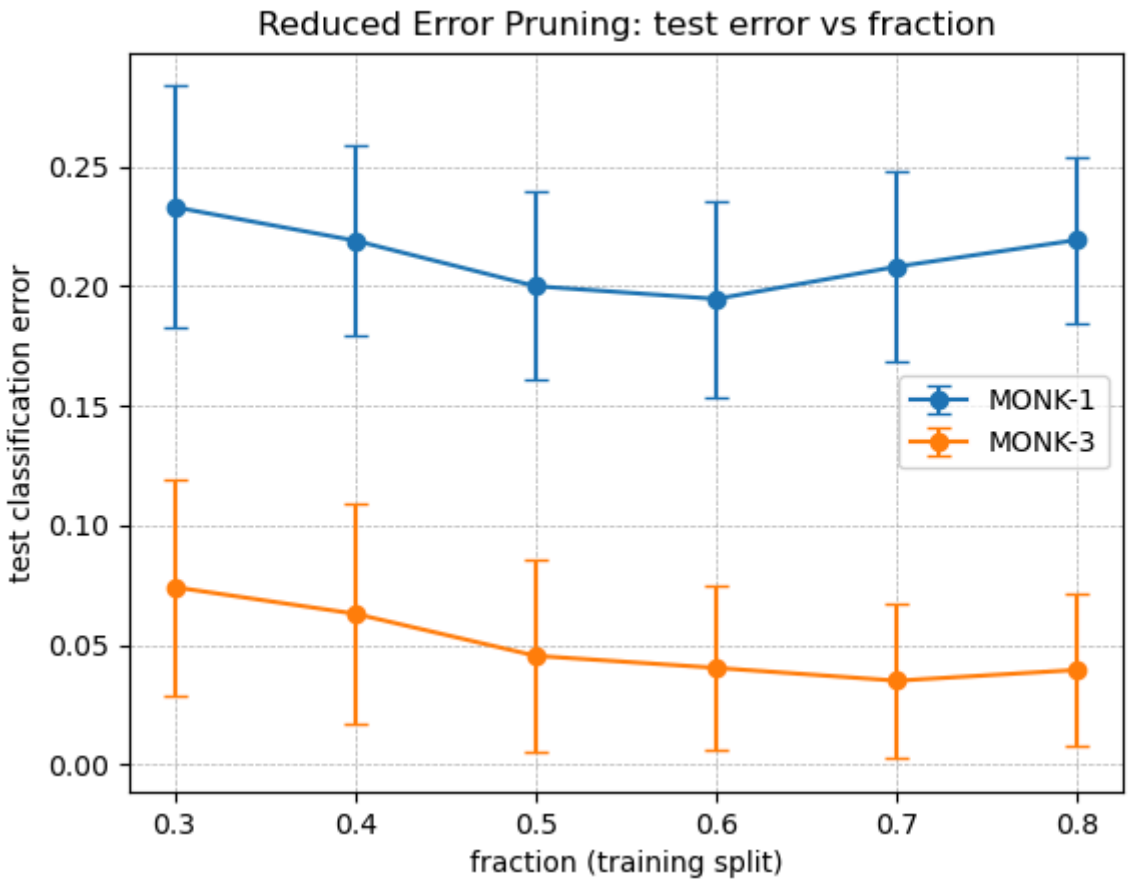
Explain pruning from a bias variance trade-off per-spective.

From a bias–variance trade-off perspective, a fully grown decision tree has very high capacity and can fit the training data extremely well, resulting in **low bias**. However, such a complex tree is highly sensitive to fluctuations and noise in the training set, which leads to **high variance** and often poorer generalization on unseen data. Pruning reduces the depth and complexity of the tree by replacing subtrees with leaf nodes, which acts as a form of regularization: it typically **reduces variance** (making predictions more stable and less noise-sensitive), at the cost of a possible **increase in bias** (reduced flexibility and potential underfitting). Reduced error pruning selects pruned trees based on validation performance, aiming to find the model complexity that minimizes generalization error. If removing a subtree does not degrade validation accuracy, that subtree likely captures noise or idiosyncrasies of the training set rather than true structure, so pruning it can improve or stabilize test performance.

Assignment 7:

Evaluate the effect pruning has on the test error for the monk1 and monk3 datasets, in particular determine the optimal partition into training and pruning by optimizing the parameter fraction. Plot the classification error on the test sets as a function of the parameter fraction $\in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$.

Note that the split of the data is random. We therefore need to compute the statistics over several runs of the split to be able to draw any conclusions. Reasonable statistics includes mean and a measure of the spread. Do remember to print axes labels, legends and data points as you will not pass without them.



We evaluated reduced error pruning on MONK-1 and MONK-3 by varying the training fraction in {0.3,…,0.8} and repeating random splits multiple times. We report the mean test error with a spread measure (error bars). The best fraction is around 0.6 for MONK-1 and around 0.7 for MONK-3. The curves show a trade-off: small fractions underfit due to too little training data, while large fractions yield unstable pruning decisions due to too little validation data.