Decission Trees

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.

- **MONK-1:** It is necessary to first check if $a_5 = 1$, otherwise, it is necessary to check if value of $a_1 = 1$ and $a_2 = 1$ too, $a_1 = 2$ and $a_2 = 2$, and so on, for all the possible patterns of a_1 and a_2.
- **MONK-2:** The resulting tree must check almost all the possible combinations of the attributes to find those having $a_i = 1$ for only two values of i. The attributes by themselves carry no information about the class, since the only thing that matters is the combination as a whole. Therefore, splitting the dataset by maximizing entropy gain will not capture this feature. This is the most difficult problem for a decision tree to learn.
- **MONK-3:** It is necessary either to check if a_5 and a_4 are equal to 1 or if $a_5 \neq 4$ and $a_2 \neq 3$. This property will be the easiest to learn because classes depends only on values of single attributes. Here we can have better learning by splitting the dataset maximizing entropy gain.

Therefore, once analysed the three MONK problems, the second one, MONK-2, is the one with higher difficulty for making a decision.

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.*

Dataset	Entropy
MONK-1	1.0
MONK-2	0.0.957117428264771
MONK-3	0.0.9998061328047111

Table 1: Datasets Entropy

Assignment 2:

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

Entropy is the way of measuring uncertainty. It is defined as the expected value of the information content of a random process.

$$Entropy = E[I(X)] = \sum_i -p_i log_2 p_i$$

The more an event is unlikely, the more information it brings. It can be proven that Entropy is maximized for a uniform distribution. For a discrete uniform distribution of K values with $p = 1/K$, the entropy is given by

$$Entropy = -\sum_{i=1}^{K} \frac{1}{K} \log_2 \frac{1}{K} = -\log_2 \frac{1}{K}$$

For a non-uniform distribution, the entropy will be lower.

An example of uniform distribution is a fair dice, where all faces have the same probability of $0.1\overline{6}$. Entropy is then

$$Entropy = -\log_2 \frac{1}{6} = 2.58$$

On the other hand, if we consider a unfair dice X, with $P(X = 1) = 0.5$ and $P(X = i) = 0.1$, $\forall\ i = 2$ to 6, the entropy becomes

$$Entropy = -\sum_{k=1}^{6} P(W = k) \log_2 P(W = k) = -0.5 \log_2 0.5 - 0.5 \log_2 0.1 = 2.16$$

which is lower than the one of the fair dice.

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?*

Dataset	A1	A2	A3	A4	A5	A6
MONK-1	0.075	0.006	0.005	0.026	0.287	0.001
MONK-2	0.004	0.002	0.001	0.016	0.017	0.006
MONK-3	0.007	0.294	0.001	0.003	0.256	0.007

Table 2: Entropy gain for attributes and datasets

The gain attribute is translated into the biggest reduction of entropy from the global set to a subset of it when selecting a certain attribute, so, biggest the value, better the attribute.

In this case, we select $A5$ for **MONK-1**, $A5$ for **MONK-2**, and $A2$ for **MONK-3**.

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, S_k, 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.

The Entropy gain, for the dataset S and attribute A is defined as

$$Gain(S, A) = Entropy[S] - \sum_{k \in A} \frac{|S_k|}{|S|} Entropy[S_k]$$

where S_k is the subset of S with attribute $A = k$. The maximized gain is

$$Gain_{max}(S) = \max_{A_i} Gain(S, A_i)$$

and the attribute that maximizes the gain is

$$A_{max} = arg \max_{A_i} Gain(S, A_i)$$

Since Entropy(S) doesn't change when maximizing over different A_i, maximizing the gain corresponds to minimizing the term

$$\sum_{k \in A} \frac{|S_k|}{|S|} Entropy[S_k]$$

Therefore, the maximum gain corresponds to the subset $\{S_k\}\ \forall\ k \in A$ with the lowest weighted average entropy. Since entropy can be seen as a measure of uncertainty, reducing the entropy by splitting for the attribute with the maximum gain results in a lower degree of uncertainty of the dataset.

Assignment 5:

Build the full decision trees for all three Monk datasets using `buildTree`*. Then, use the function*

check to measure 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.

Dataset	Error for training dataset	Error for testing dataset
MONK-1	0.0	0.171
MONK-2	0.0	0.308
MONK-3	0.0	0.056

The results match our expectations: **MONK-2** is the dataset with the highest error rate in training set. **MONK-3** is the one with the lower. The error in the training dataset is zero for all three datasets. This means that the decision trees perfectly fit the datasets they were trained on. On the other hand, we obtain errors in the training set, up to 30% in **MONK-2**.

Assignment 6:

Explain pruning from a bias variance trade-off perspective.

Without pruning, we can always build a decision tree which is perfect for our dataset, by splitting the data for each possible combination of the attributes. This, however, results in an high variance: a decision tree trained this way will make very different predictions for the same testing dataset depending on which training dataset was used. The bias is minimum, since the model perfectly captures all the feature of the dataset it was trained on. We can reduce the variance at the cost of increasing the bias by pruning the tree. The pruned tree will be a more general model less bounded to the training dataset, therefore with lower variance, but will miss particular features of the training dataset, increasing the bias.

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.

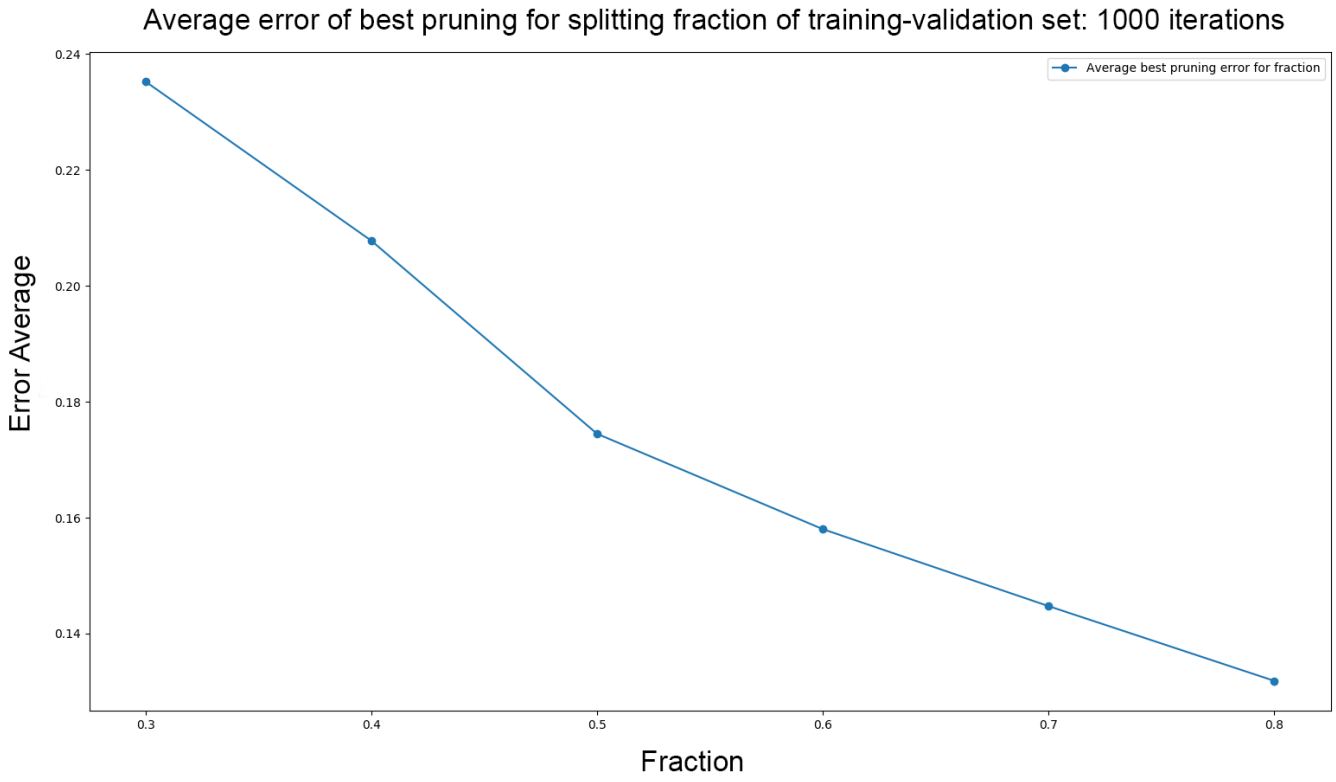

Figure 1: MONK-1

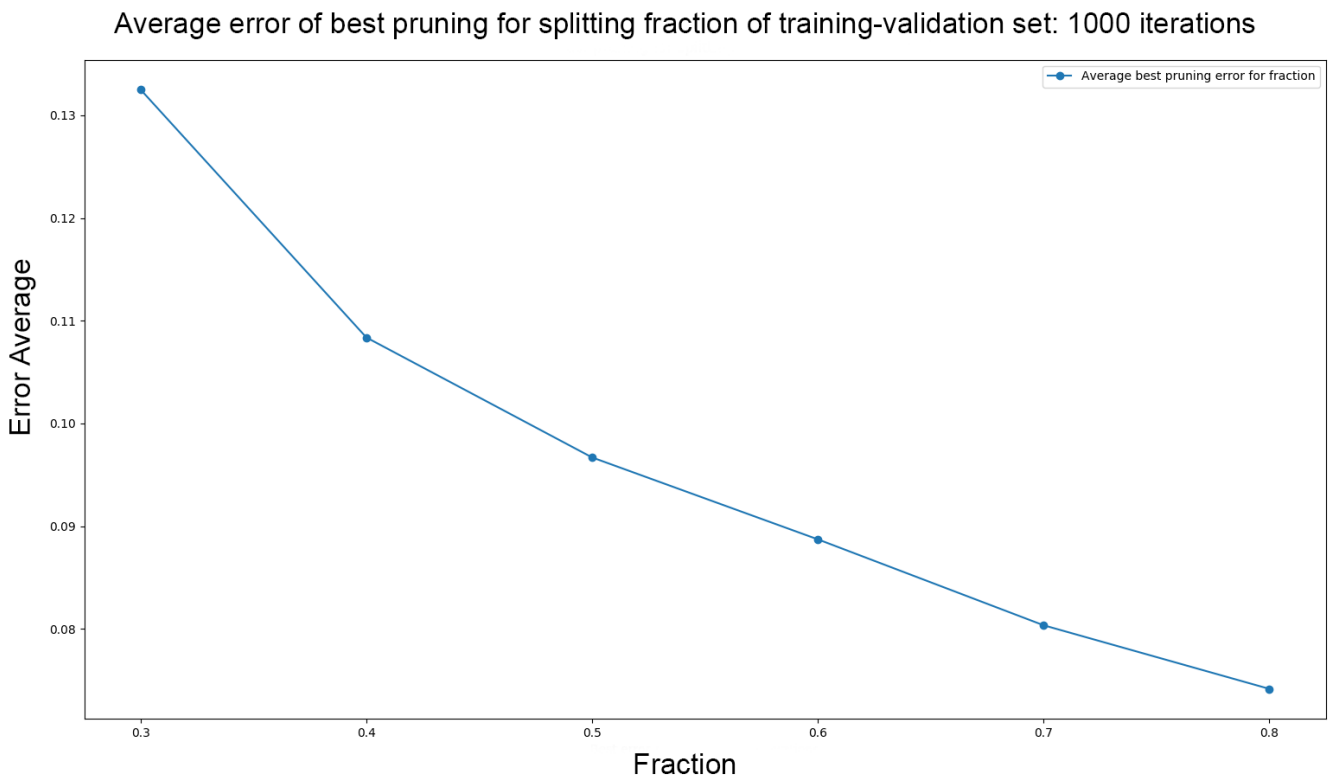

Figure 2: MONK-3

Frequency	Variance
0.3	0.0029
0.4	0.0034
0.5	0.0041
0.6	0.0044
0.7	0.0046
0.8	0.0059

Table 3: MONK-1 variances

Frequency	Variance
0.3	0.0031
0.4	0.0016
0.5	0.0013
0.6	0.0014
0.7	0.0014
0.8	0.0024

Table 4: MONK-3 variances

Figure 1 shows the plot of the average best pruning error over 1000 iterations for each fraction (splitting point of training-validation set) in **MONK-1**. Table 3 shows the variance of these 1000 best pruning errors for each fraction. The same, for **MONK-2**, is shown in Figure 2 and Table 4.

Once pruning has be performed for different fractions over a dataset, it is possible to conclude the following:

For a fixed number of elements in a dataset, the tradeoff between the number of elements selected for the training set and the number selected for the validation set implies that the bigger the training set the lower the error rate, since the model is trained with a bigger number of elements and is validated by a lower number, so the possibility of a misprediction is reduced. This can be checked in Figure **??**.