Report 1: Dectrees

Group XXX: XXX, Riccardo Fragale

January 25, 2026

1 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.

The bigger challenge for all the three datasets is the number of training samples compared to test samples. Let's consider tha dataset **MONK1** as example to justify that: it has exactly 431 test samples while the number of training samples is 123. Considering that we don't have a validation set and we don't use a k-fold cross-validation, we have a really limited amount of data to train the model on. This reasoning is valid also for **MONK2** and **MONK3**. This makes the decision-tree less able to generalise and have a complete picture of the classification problem to be solved.

Going in detail to the single datasets we can say that:

- **MONK1** appears to be the less hard of the three to model as there is a clearer rule that depends only on 3 attributes while the others are mostly irrelevant to predict the classification as true or false
- **MONK2** is the harder to classify because the pattern to be identified is more complex as discrete attributes have value 1 that is not repeated but simply two of them, randomically, get the value 1
- **MONK3** has the lower number of training samples so it is harder for the model to get an accurate classification. Moreover there is a 5% additional classification noise in the training set which makes the work even harder.

2 Assignment 2

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	0.957117428264771
MONK-3	0.9998061328047111

3 Assignment 2

Explain entropy for a uniform distribution and a non-uniform distribution, present some example distributions with high and low entropy. When we tak about a uniform distribution, we are describing a situation where every possible outcome has the same probability to appear in a single sample. This implies bigger unpredictability. A good example of it is the case of a fair coin flip where each of the 2 outcomes has 1/2 of probability. This is the maximum level of entropy possible for a set of outcomes as no outcome is more likely than any other. INSERIRE FOTO Then we have non-uniform distributions where some outcomes are more likely than others. In this case we have lower unpredictability and less surprise when certain outcomes appear. As a consequence the entropy is lower than a uniform distribution case. If the distribution is highly unbalanced towards a certain outcome the entropy might be very very low. An example could be the case of a very unfair coin where the outcome HEAD appears 99% of the times. INSERIRE FOTO

4 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?

Just for the sake of clarity, this is the equation corresponding to the gain we just talked about.

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{k \in \text{values}(A)} \frac{|S_k|}{|S|} \text{Entropy}(S_k) \tag{1}$$

Information Gain

Dataset	a_1	a_2	a_3	a_4	a_5	a_6
`MONK-1`	0.0753	0.0058	0.0047	0.0263	0.2870	0.0008
`MONK-2`	0.0038	0.0025	0.0011	0.0157	0.0173	0.0062
`MONK-3`	0.0071	0.2937	0.0008	0.0029	0.2559	0.0071

The attribute that carries the highest information gain is the one to be selected for the splitting of the dataset in the most effective way. For *MONK1* dataset, attribute a_5 carries an information gain of 0.2870 which is a lot more than any other attribute so it's the one to be chosen. A similar approach leads us to select attribute a_5 again for *MONK2* dataset even though here the difference is far less than in the previous case. As we individuated correctly in *Assignment 0* this dataset is the harder to classify and we have a further proof as no single attribute has a very big information gain when known. Finally, for *MONK3* dataset the one that has to be chosen to split the dataset is attribute a_2 with an information gain of 0.2937. In this case we have a big gain but there is another relecant attribute which is again a_5 but the gain is lower than a_2 by 0.04.

5 Assignment 4

For splitting we choose the attribute that maximizes the information gain, Eq.1. Looking at Eq.1 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.

Since we want a high gain, and the value of the entropy of S is fixed, then necessarily maximizing the gain implies that the entropy of the subsets S_k is minimized. After the split, the entropy of the single subsets is very low while the entropy of the whole dataset remains the same. As entropy is a measure of the unpredictability of a dataset, when we minimize it in the subsets (by choosing a splitting attribute) we are actually on the right path towards identifying small groups of the data that have to be classified with the same value as in each single subset S_k we have higher predictability than before the split. Hence, we are optimizeing the complexity of the tree (and as a consequence of the learning procedure) by selecting the attribute with the maximum gain.

6 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.

7 Assignment 6

Explain pruning from a bias variance trade-off perspective.

8 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` $\in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$.