DD2421: MACHINE LEARNING LAB1

Author
Adel Abdelsamed

Contents

1 Solutions

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.

Solution Assignment 0:

The concepts behind the MONK datasets are presented in Table 1.

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	$(a_5 = 1 \wedge a_4 = 1) \vee (a_5 \neq 4 \wedge a_2 \neq 3)$

MONK-3 has 5% additional noise (misclassification) in the training set.

The first data set MONK-1 is relatively simple and one can show that a minimum depth of three is required to encode the true concepts behind it.

The complexity of the third data set without noise is comparable with that of MONK-1, one could also show that it could be solved with a minimum tree depth of three. However, as it is corrupted with a 5% misclassification error, learning the true concept would be more difficult than MONK-1.

Of the three data sets the most difficult to learn would be MONK-2. We would need to go over all six attributes a_i of the tree, which would mean a bigger tree than the first and third data set.

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.

Solution Assignment 1: The entropies of the three data sets are computed and presented in Table 2.

Table 2: Entropy of the data sets.

Dataset	Entropy
MONK-1	1.0
MONK-2	0.95712
MONK-3	0.99981

Assignment 2: Explain entropy for a uniform distribution and a non-uniform distribution, present some example distributions with high and low entropy.

Solution Assignment 2:

Recall the Shannon information content of an outcome can be computed by $log_2(\frac{1}{p_i})$, where p_i is the probability of that outcome occurring. Thus, the entropy is the expected value of the Shannon information content

$$Entropy = \sum_i -p_i \cdot log_2(p_i) \tag{1}$$

We use entropy to quantify the unpredictability of the data set. Intuitively, this means if an outcome is very uncertain, then the entropy should be high and vice versa. The following example from the lecture illustrates this:

Example 1: Coin Toss

$p_{head} = 0.5$ and $p_{tail} = 0.5$, then the entropy can be computed

$$Entropy = -p_{head} \cdot log_2(p_{head}) - p_{tail} \cdot log_2(p_{tail}) = 1 \tag{2}$$

If the probabilities were $p_{head} = 0.8$ and $p_{tail} = 0.3$, then Head is the likely outcome and thus the entropy decreases to 0.72.

We can extend the definition of entropy easily to general probability distributions.

If we consider the uniform probability density function on a finite set $x_1, x_2, ..., x_n$, where $p(x_j) = \frac{1}{n}$ (meaning each outcome of the random variable $X = x_j$ is equally likely), then the entropy is log(n). On the other hand, if we consider a non-uniform distribution on the same finite set such as the gaussian distribution as the one in Figure 1(normal distributions can not be defined on finite sets but we assume that they do for the sake of the argument).

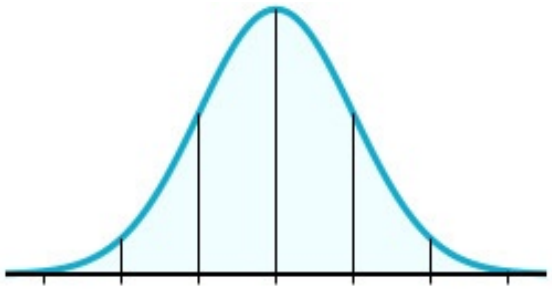

Figure 1: Normal Distribution.

Clearly, it is more likely to obtain the X-value where the peak is, thus if we compute the entropy it should decrease compared to the uniform distributions. It is known that the higher the variance the more spread the curve and lower the peak. Thus, by tuning the variance we could reach both extreme ends, where the variance is very high that we approach approximately a uniform distribution or the variance is zero that we have a $p(k) = \delta[n] - n_0$. In this case, the outcome becomes deterministic and thus the entropy would be 0.

Distributions with high entropy:

1. Uniform Distribution
2. Normal Distribution with high variance

3. Bernoulli Distribution with p = 0.5 (Coin Toss Example with Entropy = 1)

Distributions with low entropy:

1. Normal distribution with low variance
2. Dirac delta distribution (Entropy = 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 instances 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?

Solution Assignment 3: The information gain for all data sets are computed for all attributes from the root of the tree. The results are summarized in Table 3.

Table 3: Information Gain

Dataset	a_1	a_2	a_3	a_4	a_5	a_6
MONK-1	0.07527	0.00583	0.00471	0.02631	0.28703	0.000757
MONK-2	0.003756	0.0024584	0.001056	0.015664	0.017277	0.006247
MONK-3	0.007121	0.29374	0.000831	0.002892	0.2559117	0.007077

The attributes that maximize the information gain and thus should be used for splitting at the root node are colored in red.

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.

Solution Assignment 4:

Picking the attribute that maximizes the information gain means we pick the attribute that minimizes the second term in

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{k \in \text{values}(A)} \frac{|S_k|}{|S|} \text{Entropy}(S_k). \tag{3}$$

The second term sums over the entropy in all subsets after the split weighted by the fraction of data points in each subset. Thus, it can be regarded as the expected entropy after the split. Hence, we choose the attribute that results in the smallest expected entropy on average after the split. Again, if we recall entropy as a measure for unpredictability, we essentially split our original set into subsets with less expected entropy than before the split. Since the entropy of the data sets after the split decreases, the distribution within each new subset becomes more non-uniform, which means we become more certain in classifying the data sets.

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.

Solution Assignment 5:

The training and test set errors for the three data sets are computed and presented in Table 4. All models achieved a zero error value on the training sets. As assumed the second

Table 4: Training and Test set errors.

	E_{train}	E_{test}
MONK-1	0	0.1713
MONK-2	0	0.30787
MONK-3	0	0.05556

data set MONK-2 exhibits the highest error on the test set. Although, the output values of the third data set MONK-3 were corrupted with 5% classification error, this data set exhibits a lower average error on the test set than the first data set.

The results indicate that it is very likely that the trained trees have overfitted. This means they are overly specialized for the training set and don't capture the true pattern of the underlying data set (no generalization).

Assignment 6: Explain pruning from a bias variance trade-off perspective.

Solution Assignment 6: During the process of pruning, we first divide the available set into three data sets: training, validation and test set. Then we remove every possible node and the ones below that node and evaluate the performance of the pruned tree on the validation set. Once the node is found that leads to best possible improvements on the validation set, it is greedily removed.

The idea is that training the model tends to lead to high variance as seen in Assignment 5. This means the learned model overfits the training data capturing also some patterns that are irrelevant for the true concept behind the trees (e.g. noise). By evaluating the removal of each possible node, we are opting for a simplified model (hence lowering variance), but are effectively introducing some bias, where the model oversimplifies the true relationship

of the underlying data. Hence, by choosing the pruned model that leads to an increased accuracy on the validation set, we are trading some bias for less variance to achieve the smallest version of the most accurate tree .

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.

Solution Assignment 7:

The mean and standard deviation are shown in Figure 2. As we increase the fraction of training data points the mean error on the test set for both data sets decreases. The best performance on the test set is achieved when we use 70% of the dataset for training and the remaining for validation. This percentage results also in the lowest standard deviation for MONK-3. However, the standard deviation for MONK-1 remains approximately constant across all fractions.

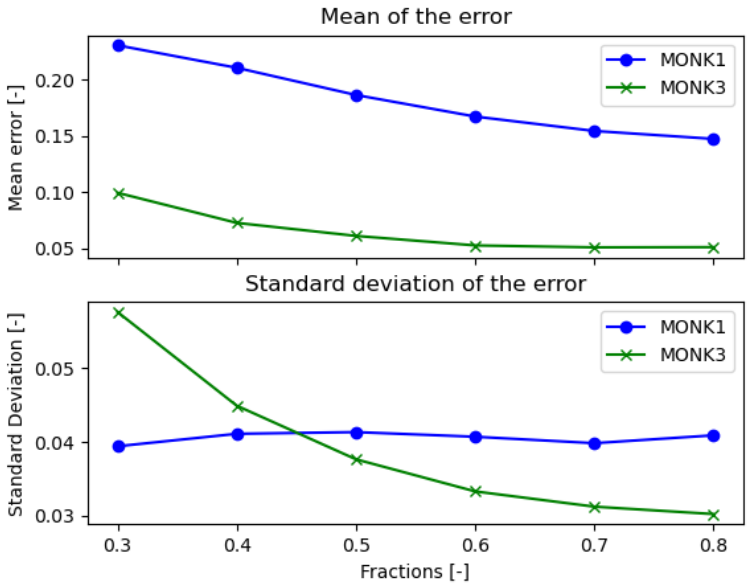

Figure 2: Effect of pruning under 300 instances.

It is obvious that pruning reduces the mean error on the test set as shown in Table 5.

Table 5: Test set errors pre- and post-pruning.

	E_{test} pre Pruning	E_{test} post Pruning
`MONK-1`	0.1713	0.1476
`MONK-3`	0.05556	0.04907