DD2421 - Lab 1

Pierre Rudin

October 9, 2017

1 Introduction

In this lab we will take a look at decision trees using the MONK data-sets. We will also implement and look at the effects of pruning.

2 Assignments

2.1 Assignment 0

Simply because of the greater amount of possible combinations MONK-2 is the hardest dataset to learn. If by hardest we mean the one that needs the most data points. Otherwise it is MONK-3 since that has noise (which makes it impossible to achieve 100 % correct classification).

2.2 Assignment 1

Data-set	Entropy
MONK-1	1.0
MONK-2	.957117428264771
MONK-3	.9998061328047111

Table 1: Calculated entropy for the training datasets.

2.3 Assignment 2

A uniform distribution would be something like a coin toss or a fair dice roll. Every outcome is equally likely. Contrary any distribution that doesn't describe an event where all outcomes are equally likely would be considered to be non-uniform. A example of such should be throwing two dices and sum the results from both dices, it will be much more likely to score 7 than 2 or 12.

The entropy of a uniform distribution are $log_2(n)$ where n is the number of possible outcomes. But in general we calculate the entropy with this formula

$$H = -\sum_{i=1}^{n} P(x_i)log_2(P(x_i))$$

To visualize how the entropy changes depending on the distribution we will construct three different distributions, one uniform, with 10 outcomes all equally likely, one where all but the last are equally likely (the one with higher probability has the same probability as all the other outcomes combined), and a final example where the probability increases as we count up (1/55, 2/55, ..., 10/55).

$$H = -\sum_{i=1}^{10}(\frac{1}{10}log_2(\frac{1}{10})) \approx 3.32$$

$$H = -\sum_{i=1}^{9}(\frac{1}{18}log_2(\frac{1}{18})) - \frac{1}{2}log_2(\frac{1}{2}) \approx 2.58$$

$$H = -\sum_{i=1}^{10}(\frac{i}{55}log_2(\frac{i}{55})) \approx 3.1$$

In all these examples 10 different events can occur, but the distribution varies and so does the entropy. We get the highest entropy when all events are equally likely to happen, and the lowest when there is one event that is 9 times as likely to happen as any of the other events. If we would like to find a distribution with higher entropy we could easily do this by choosing a uniform distribution with more events than 10 as in this example.
A example of a distribution with low entropy would be something like this:

$$H = -\frac{1}{100}log_2(\frac{1}{100}) - \frac{99}{100}log_2(\frac{99}{100}) \approx .08$$

Where out of 100 tries we expect one event to occur 99 times and another event to happen 1 time.

2.4 Assignment 3

Data-set	a_1	a_2	a_3	a_4	a_5	a_6
MONK-1	.07527256	.00583843	.00470757	.0263117	.28703075	.00075786
MONK-2	.00375618	.0024585	.00105615	.01566425	.01727718	.00624762
MONK-3	.00712087	.29373617	.00083111	.00289182	.25591172	.00707703

Table 2: Calculated entropy for the training datasets.

For MONK-1 and MONK-2 the best attribute to split at the root node would be a_5, and while a_5 could do fine on MONK-3 as well, a split on a_2 would be better.

2.5 Assignment 4

We get our predicted information gain from the following equation,

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

and since $Entropy(S)$ is fixed what we need to do is to maximize the rest of the equation, and since we subtract the sum what we really need to do is to minimize it. $|S|$ is also constant so we can drag that out of the sum as well. That gives us the mission to minimize the following term:

$$\sum_{k \in VALUES(A)} |S_k| Entropy(S_k)$$

So in order to maximize the gain, we need to minimize $Entropy(S_k)$.
The idea to split on the attribute with the highest information gain, is that this likely will give the fastest "road" to the solution. This attribute would be suggested to be the one with the highest relevance.
A high entropy implies high uncertainty, hence if we reduce the entropy we know more than we did before.

2.6 Assignment 5

Data-set	E_{train}	E_{test}
MONK-1	1.0	.828703703704
MONK-2	1.0	.69212962963
MONK-3	1.0	.944444444444

Table 3: Calculated entropy for the training datasets.

That MONK-2 would be tough comes as no surprise, with all possible combinations of rules (120) that defined a positive output, with only 169 sets of learning data, it would be nearly impossible to learn all possible combinations.
On the other hand I was expecting better results on the MONK-1 set. Why this didn't happen is probably because the rule $a_1 = a_2$ is somewhat giving false expectations, for my untrained eye this looks like a simple condition. But what is actually happening is that the tree has to try $a_1 = 1$ and thereafter $a_2 = 1$ and if not it have to test if $a_1 = 2$ and if so if $a_2 = 2$ and so on. This means the tree has to learn 4 different rules. Still doesn't sound that hard if you ask me. What is probably happening here is that if we look at the results from the function averageGain(), we see that it first is quite clear that we gain most information from a_5, but then it is not so clear, if we remove all records where $a_5 = 1$ we get the following average gains:

Data-set	Entropy
a_1	.05830506
a_2	.00515945
a_3	.00248375
a_4	.04241942
a_5	.00108335
a_6	.00116152

Table 4: Average gain for all attributes in the MONK-1 data-set when all records where $a_5 = 1$ are removed.

As we can see no single attribute gives us that much information, and this is probably what causes the error. We would for instance expect a_2 to be of more relevance than a_4, but this table suggests the opposite.
The results on the MONK-3 data-set is probably a bit better than I expected, I thought that the noise would effect the learning a bit more and make the tree to learn false patterns. This seems not to be the case so I guess that the rules are simple enough for the tree to not be effected by the noise.

2.7 Assignment 6

Pruning is a tool to decomplexify a decisison tree. In terms of bias and variance, bias goes up whit less complexity while variance grows. The goal is to minimize the expected combined bias/variance error. This does probably sound simpler than it actually is, while we can say that we want to minimize the function $MSE = bias^2 + variance$, its not that simple to quantify bias and variance. A simpler method is to stop pruning when it no longer improves the performance of the tree, simply by using a data-set for verification, with which the tree is first tested before pruning and then again afterwards. If the tree performes better we can assume that we are closing in on the MSE minima.

2.8 Assignment 7

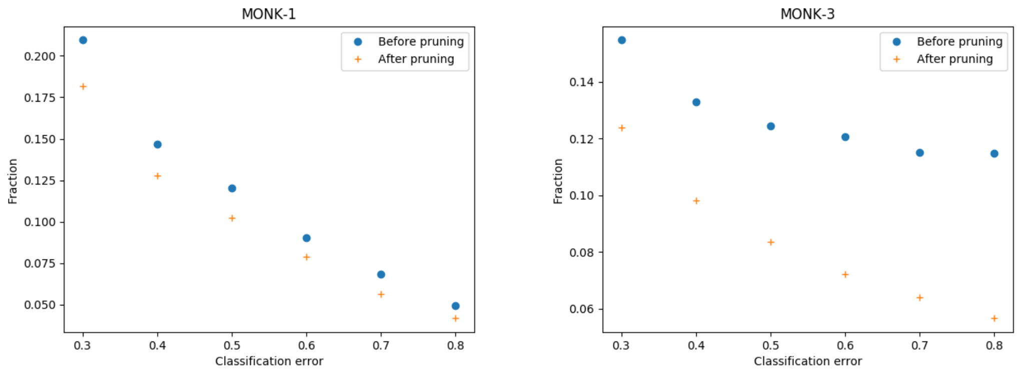

Figure 1: Classification errors

The error drops as the fraction increases, and that is not that surprising. As a bigger part of the data-set is used for training the error should decrease for relatively simple data-sets. And both MONK-1 and MONK-3 would be considered as relatively simple data-sets.