**Assignment 0**

The difficulty of learning depends on the number of questions to be asked to decide. So the question is how time consuming the training of the decision tree is. If the training data is noisy the training of the true model is more difficult since there a false-true values and vice versa.

* MONK-1: Decision possible after 2 boolean questions.
* MONK-2: Decision possible after 6 boolean questions.
* MONK-3: Decision possible after 4 boolean questions.

**Assignment 1**

Entropy for the MONK Data

* MONK-1: 1.0
* MONK-2: 0.9571
* MONK-3: 0.9998

**Assignment 2**

For uniform distribution the probability of an event is the same for every event. .

A typical example is a fair dice where the probability of every event is .

For a nonuniform distribution the probability at least for one event differs. So, the sum of every possible event is needed.

An example for a nonuniform distribution is an unfair dice where one number occurs with a higher probability, e.g. . As a result, the entropy decreases since the information decays because it’s likely to predict the outcome.

**Assignment 3**

The information gain is calculated by the given script. The following output is generated.

Information Gain Monk 1

* Attribute 1: 0.0752
* Attribute 2: 0.0058
* Attribute 3: 0.0047
* Attribute 4: 0.0263
* Attribute 5: 0.2870
* Attribute 6: 0.0007

Since Attribute results in the highest information gain it should be used to split at the root node.

Information Gain Monk 2

* Attribute 1: 0.0037
* Attribute 2: 0.0024
* Attribute 3: 0.0010
* Attribute 4: 0.0156
* Attribute 5: 0.0172
* Attribute 6: 0.0062

Since Attribute 5 results also in the highest information it should be used to split at the root node.

Information Gain Monk 3

* Attribute 1: 0.0071
* Attribute 2: 0.2937
* Attribute 3: 0.0008
* Attribute 4: 0.0028
* Attribute 5: 0.2559
* Attribute 6: 0.0070

Since Attribute 2 results in the highest information gain the tree should be splitted with it as a root node.

**Assignment 4**

For the attribute of the highest information gain the is reduced the most. So, this subset has the lowest weighted sum of entropies. So the most lowest uncertainty.

**Assignment 5**

We built up a decision without full knowledge of the samples, e.g. for monk we have 124 samples for learning and 432 samples for testing.

The Check-Function return the fraction of correctly classified samples.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Monk-1 | 0 | 0.17 |
| Monk-2 | 0 | 0.31 |
| Monk-3 | 0 | 0.06 |

**Assignment 6 - Preparation**

To get an overview the decision trees are plotted.

**Monk 1**

**Ein Bild, das Entwurf, Reihe, weiß, Diagramm enthält.

Automatisch generierte Beschreibung**

**Monk 2**

**Ein Bild, das Entwurf, Reihe, weiß, Diagramm enthält.

Automatisch generierte Beschreibung**

**Ein Bild, das Entwurf, Reihe, weiß, Diagramm enthält.

Automatisch generierte Beschreibung**

**Ein Bild, das Entwurf, Reihe, weiß, Diagramm enthält.

Automatisch generierte Beschreibung**

**Monk 3**

**Ein Bild, das Entwurf, Diagramm, Zeichnung, Reihe enthält.

Automatisch generierte Beschreibung**

**Assignment 6**

**>> Explain Pruning from a bias variance trade-off perspective.**

For pruning the tree every node is subject of an accuracy increase check. To prune compare the accuracy of the decision with and without the node. If the accuracy does not decrease by pruning, the node should be pruned since the simplest tree is desired. This simplest tree will have less edge cases. As a result, the decision is more generalized, which will decrease the variance of the decision because little deviation in the attributes will not lead to different results. By pruning the bias of the decision will increase since the decision is more general and may be taking with to few attributes.

**Assignment 7**

For pruning we start with Monk 3 because the tree is small to be handled easy.

Analysis of Monk 3:

As a general procedure the most forked branch is checked first. It is to be checked if the decision without the node “A1” has the same precision.

The function *allPruned()* returns all possibilities to prune a tree:

**Ein Bild, das Entwurf, Diagramm, Zeichnung, Reihe enthält.

Automatisch generierte Beschreibung**

1. -
2. A2(+A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+A1(--+))--))
3. A2(A5(+++-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+A1(--+))--))
4. A2(A5(++A4(-++)-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+A1(--+))--))
5. A2(A5(++A4(A1(--+)++)-)+A4(A5(--+A1(--+))--)) **[The hole middle branch is pruned at A5]**
6. A2(A5(++A4(A1(--+)++)-)A5(++A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+A1(--+))--))
7. A2(A5(++A4(A1(--+)++)-)A5(+A1(+++)A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+A1(--+))--))
8. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)+-)A4(A5(--+A1(--+))--))
9. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(++)-)A4(A5(--+A1(--+))--))
10. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(A4(+-A1(+--))+)-)A4(A5(--+A1(--+))--))
11. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)-)+)-)A4(A5(--+A1(--+))--))
12. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)-)
13. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)A4(---))
14. A2(A5(++A4(A1(--+)++)-)A5(+A1(+A4(+-+)+)A3(A4(+A1(--+)A1(+--))+)-)A4(A5(--+-)--))

Algorithm see Python.