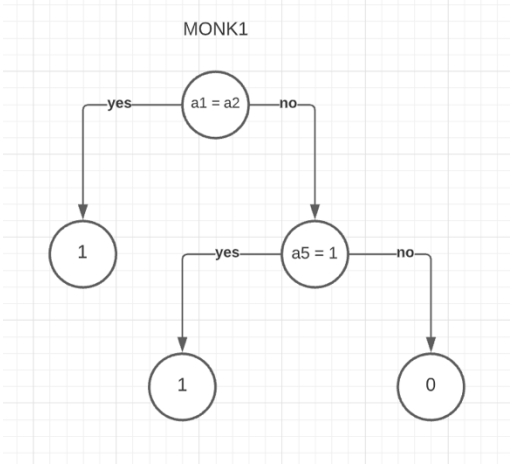
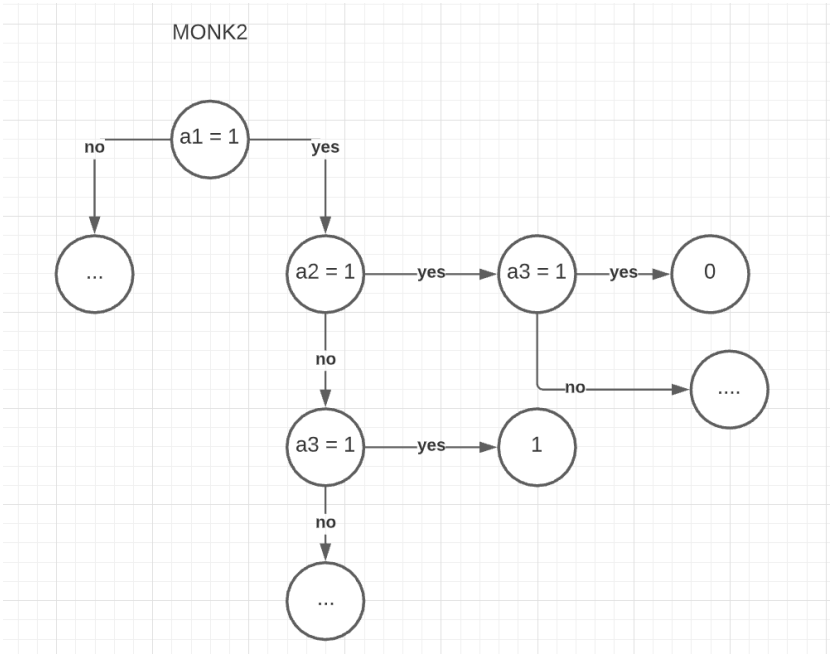
Machine Learning Exercise 1

Assignment 0

Monk2 is the most complex dataset as it might take up to 6 steps to determine the answer.
For Monk2 the number of splits is much larger than the other ones, that's why it will be harder to determine.



MONK1 for example is easy to solve, as it requires a maximum of two splits.



Monk 2 on the other hand will take six splits, making it extremely complex to solve just for two of the attributes to be equal to one. The decision tree is far more complex than the pictured part of it: in any case, we must check for all attributes if their equal 1, and there is no scenario in which we might be able to determine the answer before checking all the attributes. This makes the decision tree quite big.

Assignment 1

Entropy of the respective dataset:

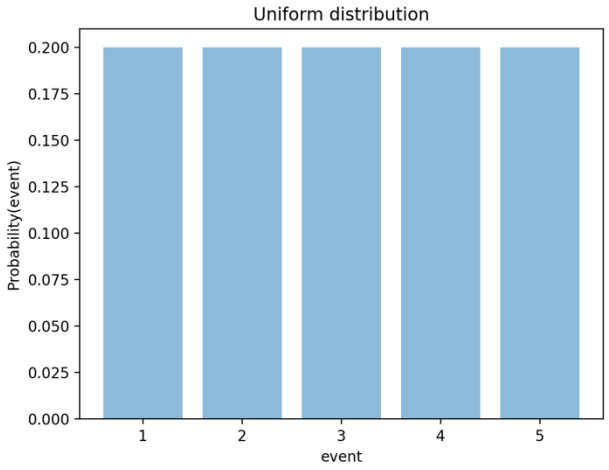
Output:

```
Monk1:  1.0
Monk2:  0.957117428264771
Monk3:  0.9998061328047111
```

Code:

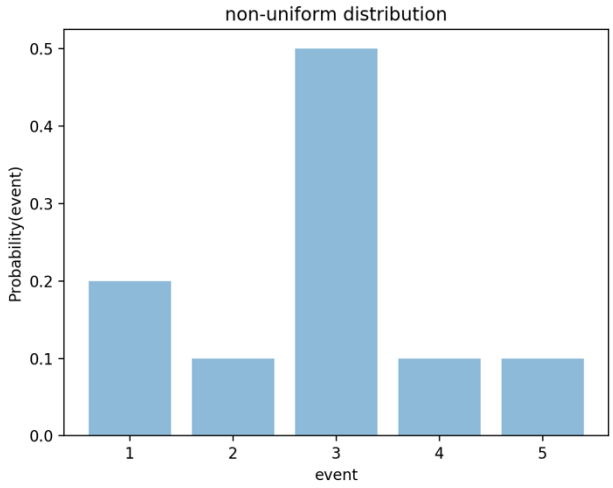
```
import monkdata as m
import dtree as d

print("Monk1: ", d.entropy(m.monk1))
print("Monk2: ", d.entropy(m.monk2))
print("Monk3: ", d.entropy(m.monk3))
```

Assignment 2:



In the case the uniform distribution, all events have the same probability, so the Entropy is maximum, and the information needed to make a guess will be as high as it can be.

This is the case of a fair die in real life, for instance.



Contrarily, in the case of a non-uniform distribution, there are some events that are more likely to take place: In other words, we do not need that much info to make a guess, since one event is clearly far more likely to be.

This is the case of a biased die in real life, for instance.

Assignment 3:

Output:

```
Monk1, attribute 0 0.07527255560831925
Monk1, attribute 1 0.005838429962909286
Monk1, attribute 2 0.00470756661729721
Monk1, attribute 3 0.02631169650768228
Monk1, attribute 4 0.28703074971578435
Monk1, attribute 5 0.0007578557158638421
Monk2, attribute 0 0.0037561773775118823
Monk2, attribute 1 0.0024584986660830532
Monk2, attribute 2 0.0010561477158920196
Monk2, attribute 3 0.015664247292643818
Monk2, attribute 4 0.01727717693791797
Monk2, attribute 5 0.006247622236881467
Monk2, attribute 0 0.007120868396071844
Monk2, attribute 1 0.29373617350838865
Monk2, attribute 2 0.0008311140445336207
Monk2, attribute 3 0.002891817288654397
Monk2, attribute 4 0.25591172461972755
Monk2, attribute 5 0.007077026074097326
```

We always choose the attribute with the highest information gain as a root node. For instance, in the case of Monk1, we should choose attribute 4, which minimizes the entropy for the sub-datasets that will be formed after the split.

Assignment 4:

The higher the entropy, the more difficult it is to discriminate the data. After the split, the entropy of the subsets should be as low as possible in order to get a high information gain. The main goal is to lower the entropy.

Assignment 5:

Output:

```
Monk1 Test:  0.17129629629629628
Monk1 Train:  0.0
Monk2 Test:  0.30787037037037035
Monk2 Train:  0.0
Monk3 Test:  0.05555555555555558
Monk3 Train:  0.0
```

Our assumption that the Monk2 is the most complex was correct, as it has the highest error of the three datasets.

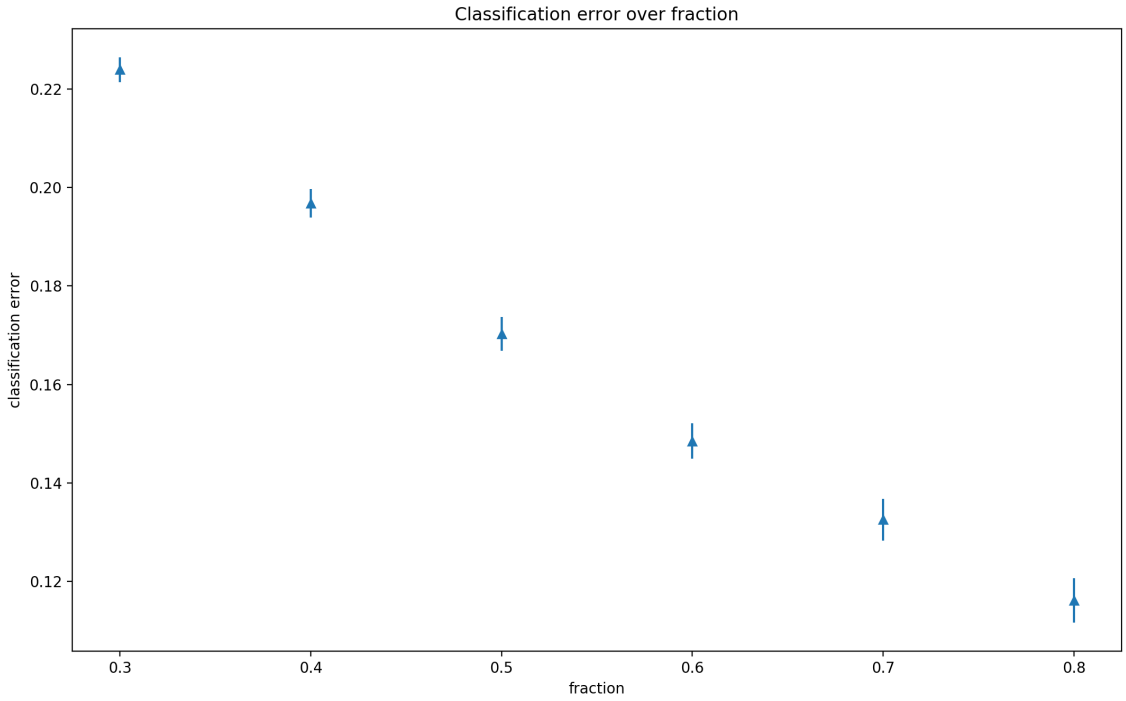
Monk3 has low errors, i.e. the algorithm pre-learns the patterns and is able to perform well on data that he has not seen before. The classifier for the second dataset is overfitting, as the training data implies a zero error, while the test data is not performed well.

To make it perform better, we might have to change the hyperparameters.

Assignment 6:

We want to keep the bias as high as possible. The goal of pruning is to reduce the variance while keeping the bias high. Pruning works in such way that a node is taken away from a tree, then the new tree is compared to the old tree. If the new tree performs better, the node is taken away and replaced by a leaf node. The higher the depth of the tree, the higher the variance, so usually the trees loose depth. ☺

Assignment 7:



The figure includes the mean (as points) and variance (error boxes) for all of the 6 fractions in which the dataset was divided (train and test). As more data is used for training, the algorithm finds it easier to learn the pattern, as reflected by a low error rate performed upon the test set.