LAB1
Decision Trees

DD2421 – Machine Learning
21st September 2023

FLANDRE Corentin
MADEC Steven

Table of Contents

1. Introduction

During this mandatory laboratory, who handle various important points of the principles of Decision Trees in Machine Learning, we are going to use MONK, an artificial dataset created by UC Irvine repository. Different key concepts will be covered:

- Dataset (with both training and testing set)
- Entropy
- Information Gain
- Decision Trees (you can also refer to lecture number 2)
- Pruning

You can find our GitHub repository link containing initial files, our codes and the report needed to complete the lab: *https://github.com/flandrecorentin/DD2421-ML* (the important file created to respond to the assignments is the python file lab1_FLANDRE_MADEC.py)

2. Dataset

To introduce this laboratory, we have to explain a bit the dataset. It consists of three binary (0 or 1) classification (called MONK-1, MONK-2, MONK-3) and features six attributes (denoted a_1, a_2, a_3, a_4, a_5, a_6). These attributes can take discrete values up to 4 (depending on the attribute).

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.

Table 1: Concept behind MONK datasets

Each separate dataset is divided into a **training set** and a **testing set**.

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.

In view of the model usage of tree algorithm, the dataset **MONK-2** is probably the harder to learn. Decision tree is very well adapted for Arbitrary boolean functions. MONK-2 consists of more complex binary expression. In that case, we have to check all attributes one by one to calculate one binary value of the dataset. Moreover, it implies to use probably a sum operator and a temporary variable. On the contrary, MONK-1 and MONK-3 consist of very simple Boolean expressions (using disjunction, conjunction and not logic operators). As a reminder, in terms of how a computer works, these operations are very quick. To summary very roughly, decision trees algorithms are slightly less adapted to concepts who do not depend on the values of attributes but relation between them.

3. Entropy

3.1 Concept of Entropy

As we see in lecture 2, **entropy is a good way to measure the unpredictability** (measure of uncertainty) using bits of information. For instance, a coin tass corresponding to a 1 bit of information. Higher the entropy is, more unpredictable is the result. Applied to datasets, entropy is the measure of the unpredictability of values corresponding to a classification. We will see it again in the 'Information gain' part but with decision trees, if we are capable to reduce entropy, we will be more easily able to predict the classification of a set/tuple.

The entropy is measured with the theoretical formula following:

$$Entropy(S) = -\sum p_i \, log_2 p_i$$

in which pi denotes the proportion of examples of class i in S.

Applied to our dataset (with the specific binary classification), the entropy can me measure with the formula:

$$Entropy(S) = -p_0 \, log_2 p_0 - -p_1 \, log_2 p_1$$

Where p_0 and $p_1 = 1 - p_0$ are the proportions of the 0 or 1 in the binary classification.

3.2 Applied Entropy on dataset

In the case of a binary classification, the maximum entropy is 1 (corresponding at an uniform distribution of class -> as many 1 as 0 (the distribution is uniform)) and the minimum entropy is 0 (corresponding at always the same distribution -> only 1 or only 0 (the distribution is absolutely non-uniform))

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.

Dataset	Entropy
MONK-1	1.0000000
MONK-2	0.9571174
MONK-3	0.9998061

Values are rounded to the nearest 10^{-7}

In the case of our binary classification, we can say that the 3 data sets are well distributed because entropy are value close to 1 (maximum value if it's binary class like a non-faked toss coin)

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

In an uniform distribution, all class are equally likely (as example of the MONK-1 who represents perfectly an uniform distribution because there are as many 0 than 1 class in the monk1 dataset), it's more difficile to predict the class of a data. For example, a non-faked toss coin is a uniform distribution,

the entropy represents 1 bit of information because initially the probability of wining is the same as loosing. In the case of binary classification, it represents a high entropy.

In an non-uniform distribution, some class are more likely to others. There are more certain outcomes and less about others. For example, a faked toss coin who win with a probability of 0.95 (and loose with 0.05) has only an entropy of 0.286 bit because initially there are more probable than the toss coin will win. In the case of binary classification, it represents a low entropy.

For same number of classes, it's sure that an uniform distribution represents a higher entropy.

Even with another game, it's possible to have non-uniform distribution. For example, a 52 cards-game non-faked represents an entropy of 5.7bits ($-52 * \frac{1}{52} * log_2(\frac{1}{52})$). On the contrary, a possible non-uniform distribution could be probability of 0.5 for one card and $1/102$ for others. This implies an entropy of 3.836bits (less than before) ($-51 * \frac{1}{102} * log_2\left(\frac{1}{102}\right) - \frac{1}{2} log_2\left(\frac{1}{12}\right)$) because the game is less unpredictable. We have to relied on the notion of comparison (higher/less entropy) with the distribution but not really the value of entropy.

4. Information Gain

4.1 Concept of Information Gain

As we see in lecture 2, the information gain is related to the entropy. **Information gain is the measure who indicates the effectiveness of an attribute in classifying the training data**. Compared to the entropy, the information gain takes into consideration attributes because it's a way to measure the quality (the gain) of the classification (as though we were comparing before and after the classification).

The entropy is measured with the theoretical formula following:

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

Where S_k is the subset of examples in S for the attribute A

Actually, in the next part (application of Information gain concept to monks dataset) we are going to study the classification split for one attribute at a time but it's possible to involve more than one attribute for one classification (for example: combination of attributes)

4.2 Applied Information Gain to dataset

For the classification using decision trees algorithms, for a specific dataset we must choose the attribute which tell us the most about the class of tuples (0 or 1 because we are in the case of a binary classification)

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?

Dataset	a_1	a_2	a_3	a_4	a_5	a_6
MONK-1	0.0752726	0.0058384	0.0047076	0.0263117	**0.2870307**	0.0007579
MONK-2	0.0037562	0.0024585	0.0010561	0.0156642	**0.0172772**	0.0062476
MONK-3	0.0071209	**0.2937362**	0.0008311	0.0028918	0.2559117	0.0070770

Values are rounded to the nearest 10^{-7}

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, Sk, 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.

To have a decision tree more performant on the classification models, we must split regarding the information gain (there are other criteria as seen in lecture 2: Gini impurity). To get the information gain maximized, the entropy of the subsets Sk reduced. If the value of entropy reduces after the split, it means that we are easily more able to deduce the class of a tuple into the training set (it's less unpredictable). Of course, we want a model capable to detect the class using attributes, so we want

in each step/split reduce the entropy. Without talking about overfitting in the training (see part 6 talking about pruning), a split step must be carried out on the attribute who give the most in terms of information on its class.

Regarding the formula, and taking into consideration the weight ponderation of subset, lower the entropy of the subset is, higher is the gain information. The weight ponderation allows for example to don't split into a singleton subset when there is certainly a true relation in but only for one data and it's useless (maybe dangerous) if the rule is not applicable on the testing set.

In our cases, to split (in depth 1) our decision tree, we will split with values possibles of the attribute with the higher gain information (a5 for monk1 and monk2 and a2 for monk3). Already now, we realise that the monk2 dataset is likely to have a less efficient decision tree model because all gains values are very low, and we don't really learn information with the first level of leaves.

5. Decision Trees

The laboratory subject asks us to create our own decision tree up to the first two levels (depth 2) with the monk1 dataset. So, we just must reiterate the process of calculation gain information on the second level because we already have the first one corresponding to different values possibles of a5.

- Personal monk1 tree obtained (textual format): $\boldsymbol{A5}(+\boldsymbol{A4}(---)\boldsymbol{A6}(--)\boldsymbol{A1}(--+))$

This tree corresponds exactly to the ID3 (depth=2) result. Now that creation of decision tree model is understand, we can use the function buildTree already implemented into initial files who can build the all tree.

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.

The check validity corresponds to the value returned by the d.check function. So, error corresponding at the probability that a tuple is incorrectly classified (error = 1 – check validity)

	E_{train}		E_{test}	
	Check validity	Error	Check validity	Error
MONK-1	1.0000000	0.0000000	0.8287037	0.1712963
MONK-2	1.0000000	0.0000000	0.6921296	0.3078704
MONK-3	1.0000000	0.0000000	0.9444444	0.0555555

Values are rounded to the nearest 10^{-7}

Regarding results, it's possible to have perfect validity on the training sets (but risks of overfitting, see part 6). That's understandable because all combinations are easy to find (max 169 data in monk2 corresponding of 7bits so 7 level maximums on the tree)

The assumption that MONK-2 is the hardest to learn in that decision tree lab seems real. The link between attributes relations is less indirect than other datasets. The percentage of errors is very high (0.3078) even if it's only 6 attributes. On the contrary the MONK-3 dataset is very well classed even if there are 5% of additional noise (due to misclassification), probably because there are not strong relationships between attributes compared to MONK-1 rule (who don't really care about values of a1 and a2 but only to the egality between them)

6. Pruning

6.1 Concept of Pruning in context of Decision trees

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

Before to look at the bias variance trade-off perspective of pruning, it's important to make a little resume of what's bias and variance. Bias refers to the approximation of a problem which may be more complex than what we represent, it can cause underfitting issues. On the contrary, the variance refers to the small fluctuations of noise than can impact on the learning, it can lead to overfitting if the model is too adapted to the training set like some rules will be created when they don't have to be.

For now, before pruning our monks dataset, we probably have a problem of overfitting because there are no errors by ours models with the training set. Sometimes we must authorize some errors in the learning step to have better results on test set. Pruning will be reducing the leaf of the tree (in other terms we reduce some levels of the tree) to be more representative of the real dataset (especially the test testing) and we reduce the overfitting effect that we probably created with the full depth tree.

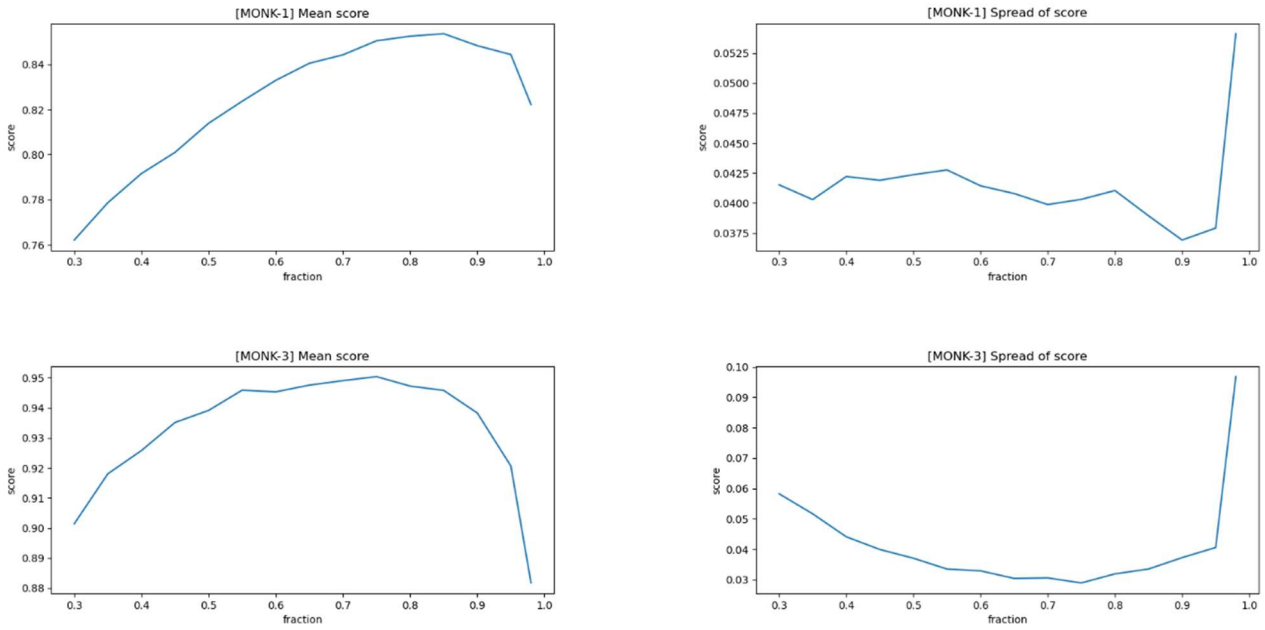
Basically, in our case, reduced error pruning corresponding to reduce the variance (and respectively increase the bias).

6.2 Applied Pruning

During the operation of pruning, for our datasets, it's preferable to use the new training set (this one partitioned) to create a decision tree and see all pruned trees and use the validation set to decided which one of them is better. Then we use the testing set to see what's the result of our partition choice. We reiterate these operations several times because due to the random repartition it's better to understand the best fraction in general (who can be sometimes different with a random specific repartition). We also take in consideration the spread during the statistics analysis.

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.

The objective of the pruning is to obtain a model of decision tree with a mean score higher (signifies a lower number of errors) than before pruning and to minimize the spread of score.



Plots 1: Means score on test set and spread score [MONK-1 & MONK-3]

Values used for the are:

- Datasets used: MONK-1, MONK-3
- Fractions: 0.3 to 0.95 with step 0.05 (more precisely than the precision asked by assignment for plots)
- Number iterations computed: 1000

For monk1, regarding the mean score on the test set, the best fraction (after 1000 iterations) is **0.8** and corresponds to a lower spread. The mean score is **0.8525278** which is a higher score than before pruning (0.8287037 see table part 5)

For monk3, regarding the mean score on the test set, the best fraction is **0.75** corresponds to a quite lower spread. The mean score is **0.9503842** and that's not so far of the value before pruning (the check validity score was 0.9444444 see table part 5). We can't really have a better value of that because there still have initially 5% of misclassification so we can consider that the result is just as good than before pruning.

In the case of the dataset MONK-1, the overfitting effect was present and with the pruning we can pass from around 17,1% to around 14,7% of errors of tuples classification.

7. Conclusion

During the laboratory about decision trees, we worked with a scientific approach regarding machine learning methods. This approach combines, pre-analysis of datasets, choice of algorithms and method of learning, reflection on how it's possible to improve the model and draw conclusions on our work. The pre-analysis is to understand a little the dataset before anything else and make statistics analysis if the dataset is not so complex like monks datasets. This step implies to understand well the interest to have a training set and a testing set. Of course, during this lab, the algorithms used for the classification was decision trees but it's important to understand what's others possibility to make another classification model. Different ways to improve a model are possibles but the lab made us work in depth the notion of pruning in the specific case of decision trees. Now, we are able to adapt the reflection of improving in other cases. It's important to well understand what's the impacts are of all parameters.

In more technical terms, the laboratory enabled us to understand the reality of a decision tree model construction with the calculation of entropy and gain information. As proof, the laboratory subject asks us to draw our personal tree in depth 2 and to compare it with functions already in place (as ID3). The concept of pruning is also acquired because the method of pruned a tree is understood and we understand that the impact of parameters is huge (as we can see with the specific case of the fraction repartition between training and validation set). The importance of partitioning into a training set and a validation independently to the testing set is well explained.