Assignment 2

part a

Console Output:

A screenshot of a cell phone

Description automatically generated

Visualized Learned Decision Tree:

A close up of a map

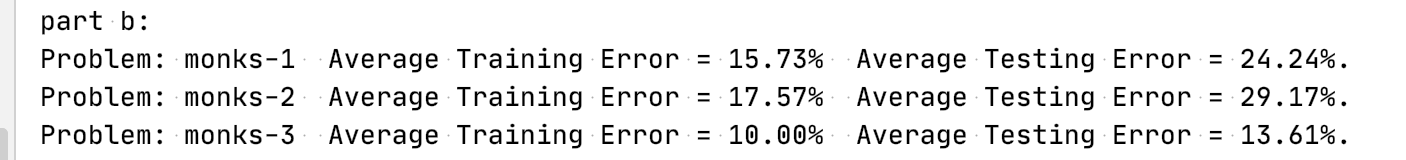
Description automatically generated

I coded with the given frame and designed my own ID3 Algorithm. In the following part b-e, I will import these functions directly from decision\_tree.py.

For part b-e, you should execute Assignment2b-e.py instead.

part b

Console Output:



Plots implemented by matplotlib:

A screenshot of a cell phone

Description automatically generatedA close up of a map

Description automatically generatedA screenshot of a cell phone

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| Max Depth | 1 | 3 | 5 |
| AVG Training Error | 15.73% | 17.57% | 10.00% |
| AVG Testing Error | 24.24% | 29.17% | 13.61% |

From these three plots can we see, it is wise to choose the maximum depth with 3. Because a decision tree with depth more than 3 is often suffered from overfitting.

part c

Console Output:

A screenshot of a cell phone

Description automatically generated

Visualized Learned Decision Tree:

Max Depth = 1

A drawing of a person

Description automatically generated

Max Depth = 3

A close up of a map

Description automatically generated

Max Depth = 5

A close up of a map

Description automatically generated

part d

Console Output:

A screenshot of a cell phone

Description automatically generated

Visualized Learned Decision Tree:

Max Depth = 1

A screenshot of a cell phone

Description automatically generated

Max Depth = 3

A screenshot of a cell phone

Description automatically generated

Max Depth = 5

A close up of a piece of paper

Description automatically generated

The result I get from part d is slightly different from part c because of the way that I split the nodes. But the overall shapes are same.

part e

ID3 on the left (top if too large) and Scikit-learn on the right (below).

Console Output:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Visualized Learned Decision Tree:

Max Depth = 1

A picture containing clock, drawing, meter

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Max Depth = 3

A close up of a map

Description automatically generated

A close up of a map

Description automatically generated

Max Depth = 5

A close up of a map

Description automatically generated

A close up of a map

Description automatically generated

At the last part, I use the Glass Identification Data Set from UCI. I did not want to make y which had 7 different values into 0 and 1. I think that way is illogical. So I optimized a little with my code in decision\_tree.py and made it work by only transform xi’s values into 0 and 1.

I did not find a way to make Scikit-learn use equal to split. Hence, the learned decision trees obtained by ID3 have some differences from the ones by Scikit-learn. However, if you observe the confusion matrix, you would find the matrixes are similar correspondingly. I think this proved the rationality of decision tree learning algorithm.