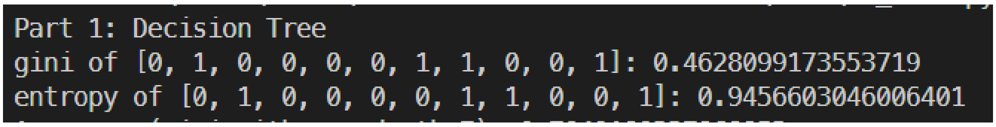
NYCU Introduction to Machine Learning, Homework 3

110705013, 沈昱宏

Part. 1, Coding (50%):

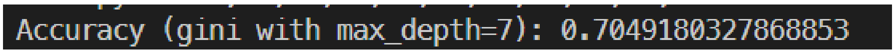
(30%) Dicision Tree

1. (5%) Compute the gini index and the entropy of the array [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1].



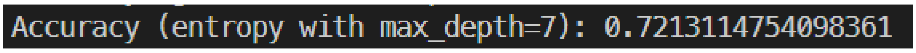
```
Part 1: Decision Tree
gini of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.4628099173553719
entropy of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.9456603046006401
```

2. (10%) Show the accuracy score of the testing data using criterion="gini" and max_depth=7. Your accuracy score should be higher than 0.7.



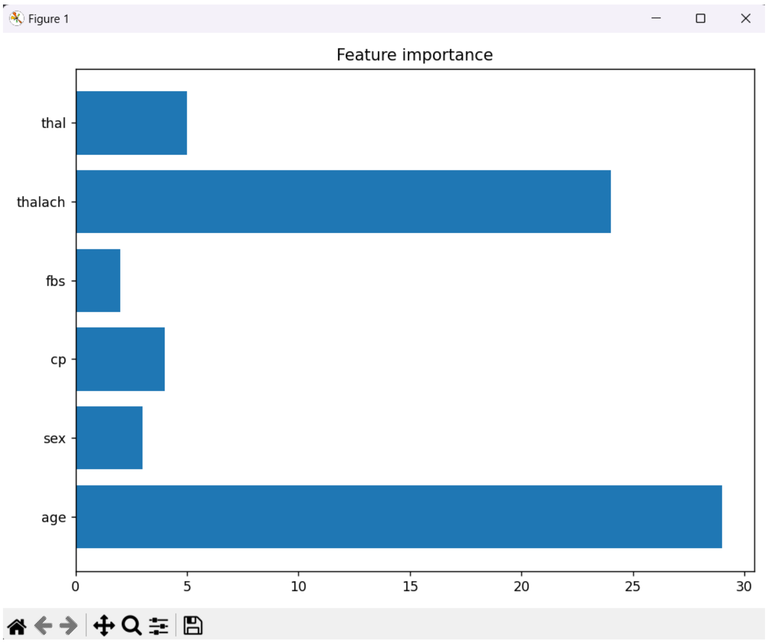
```
Accuracy (gini with max_depth=7): 0.7049180327868853
```

3. (10%) Show the accuracy score of the testing data using criterion="entropy" and max_depth=7. Your accuracy score should be higher than 0.7.



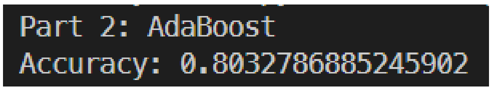
```
Accuracy (entropy with max_depth=7): 0.7213114754098361
```

4. (5%) Train your model using criterion="gini", max_depth=15. Plot the feature importance of your decision tree model by simply counting the number of times each feature is used to split the data.



(40%) Adaboost

5. (20%) Tune the arguments of AdaBoost to achieve higher accuracy than your Decision Trees



```
Part 2: AdaBoost
Accuracy: 0.8032786885245902
```

Part. 2, Questions (50%):

1. (10%) True or False. If your answer is false, please explain.

   a. (5%) In an iteration of AdaBoost, the weights of misclassified examples are increased by adding the same additive factor to emphasize their importance in subsequent iterations.

   b. (5%) AdaBoost can use various classification methods as its weak classifiers, such as linear classifiers, decision trees, etc.

Ans:

a. False. It does not add the same additive factor. It update by :

$$\varepsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$$

- Weight classifier: $\alpha_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$
- Update distribution:

$$D_{t+1}(i) = \frac{D_t(i)\exp[-\alpha_t y_i h_t(x_i)]}{Z_t}$$

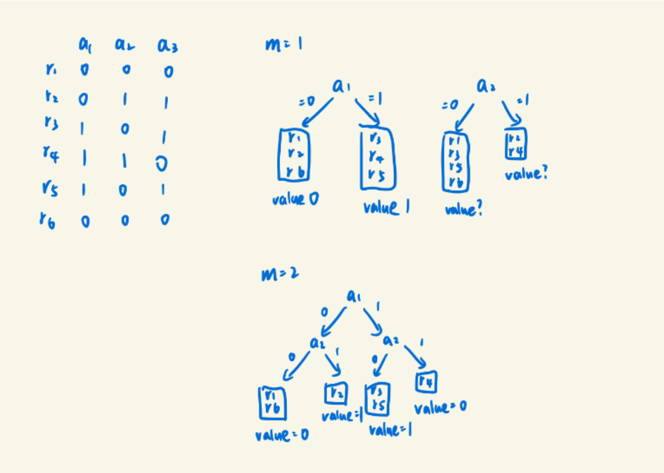
b. True

2. (10%) How does the number of weak classifiers in AdaBoost influence the model's performance? Please discuss the potential impact on overfitting, underfitting, computational cost, memory for saving the model, and other relevant factors when the number of weak classifiers is too small or too large.

When the number of classifiers is too small, the model might underfit, and the computational cost and memory usage is relatively low. When the number of classifiers is too large, the model might overfit, the computational cost will grow since you have to select from more classifiers, and the memory for saving the model will be larger.

3. (15%) A student claims to have a brilliant idea to make random forests more powerful: since random forests prefer trees which are diverse, i.e., not strongly correlated, the student proposes setting m = 1, where m is the number of random features used in each node of each decision tree. The student claims that this will improve accuracy while reducing variance. Do you agree with the student's claims?

No, this will not increase the accuracy, and it is very likely that the accurray will drop. Considering only one random feature per node limits the information available for splitting, making each tree to become very simple. Let’s consider a simple dataset shown below. Here use the whole dataset instead of bagging for simplicity. Using m = 1 random forest, the accuracy is doomed to be low because the outcome is not predictable by only one column. Using m = 2 can solve the problem easily (the below example I draw is a decision tree, but it can be extended to random forest through bagging).



	a_1	a_2	a_3
r_1	0	0	0
r_2	0	1	1
r_3	1	0	1
r_4	1	1	0
r_5	1	0	1
r_6	0	0	0

4. (15%) The formula on the left is the forward process of a standard neural network while the formula on the right is the forward process of a modified model with a specific technique

   a. (5%) According to the two formulas, describe what is the main difference between the two models and what is the technique applied to the model on the right side.

   b. (10%) This technique was used to deal with overfitting and has many different explanations; according to what you learned from the lecture, try to explain it with respect to the ensemble method.

$$z^{(l+1)} = w^{(l+1)}y^l + b^{(l+1)}$$

$$y^{(l+1)} = f(z^{(l+1)})$$

$$r^l = Bernoulli(p)$$

$$\tilde{y}^l = r^l y^l$$

$$z^{(l+1)} = w^{(l+1)}\tilde{y}^l + b^{(l+1)}$$

$$y^{(l+1)} = f(z^{(l+1)})$$

Ans:

a. The formula on the left is a normal neural network. The formula on the right uses a bounoulli random variable of probability p to perform drop-out on the nueral network. y(l+1) is dropped out (=0) with probability p.

b. Ensemble methods deal with overfitting issues by leveraging the strengths of multiple models and making more generalized predictions by combining diverse information learned from different perspectives of the data.

   The Dropout mechanism is somehow like the concept of ensemble. In each iteration, we drop certain nodes as if removing them from the neural network. Therefore, the neural network we use have different structure during each iteration. The model learned from each iterations is like the weak classifiers in the ensemble methods, and the final neural network after training combines the results of weak classifiers to become the final classifiers.