NYCU Introduction to Machine Learning, Homework 3

110550168, 賴御安

Part. 1, Coding (50%):

For this coding assignment, you are required to implement the Decision Tree and Adaboost algorithms using only NumPy. After that, train your model on the provided dataset and evaluate the performance on the testing data.

(30%) Decision Tree

Requirements:

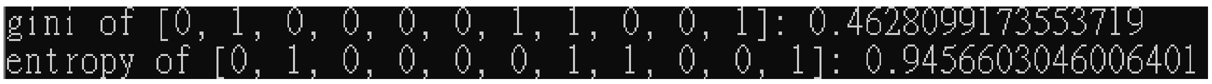
- Implement **gini index** and **entropy** for measuring the best split of the data.
- Implement the decision tree classifier [(CART, Classification and Regression Trees)](#) with the following two arguments:
- **criterion**: The function to measure the quality of a split of the data. Your model should support "gini" and "entropy".
- **max_depth**: The maximum depth of the tree. If max_depth=None, then nodes are expanded until all leaves are pure. max_depth=1 equals to splitting data once.

Tips:

- Your model should produce the same results when rebuilt with the same arguments, and there is no need to prune the trees.
- You can use the recursive method to build the nodes.

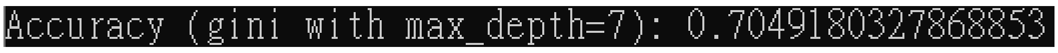
Criteria:

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



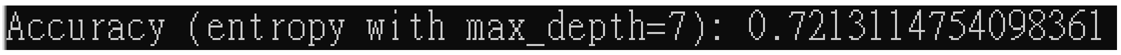
```
gini of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.4628099173553719
entropy of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.9456603046006401
```

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.



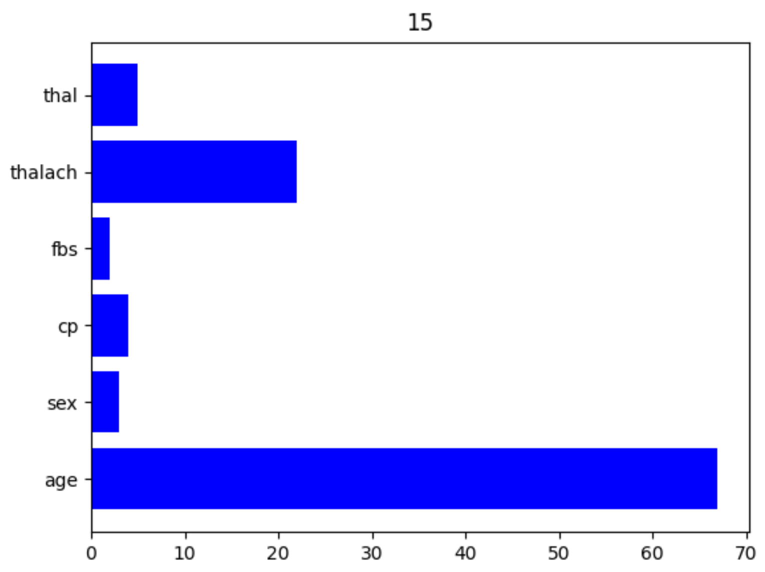
```
Accuracy (gini with max_depth=7): 0.7049180327868853
```

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. Your answer should look like the plot below:



(20%) Adaboost

Requirements:

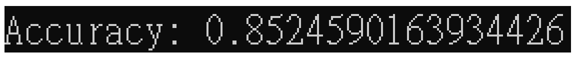
- Implement the Adaboost algorithm by using the decision tree classifier (max_depth=1) you just implemented as the weak classifier.
- The Adaboost model should include the following two arguments:
- **criterion**: The function to measure the quality of a split of the data. Your model should support "gini" and "entropy".
- **n_estimators**: The total number of weak classifiers.

Tips:

- You can set any random seed to make your result reproducible.

Criteria:

1. (20%) Tune the arguments of AdaBoost to achieve higher accuracy than your Decision Trees.



```
Accuracy: 0.8524590163934426
```

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 :

1. (a) false :

In Adaboost, the weight of misclassified examples is increased, but instead of adding the same additive factor, it is added by different weights. The misclassified examples will be assigned with higher weights in order to increase the importance during training.

1. (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.

A:

(1) underfitting / overfitting : If the number of weak classifiers is too small, this may lead to underfitting, while the model cannot catch the complexity of the given data. And if the number of the weak classifiers is too large, this may lead to overfitting, while the model will memorize the training data but perform worse when new data comes.

(2) computational cost : If the number of weak classifiers is too small, the computation cost is small but this cannot get a high prediction accuracy. And if the number of weak classifiers is too large, the computation cost is high since we need to create lots of models and keep updating the weight.

(3) memory for saving the model : If the number of weak classifiers is too small, this saves memory for storing the models, but we have less models when predicting. On the other hand, it requires lots of space to store the models.

(4) other relevant factors : If the number of weak classifiers is too small, we can train fast, but the accuracy may not perform well. And if the number of weak classifiers is too large, we might need a long time especially when we have computationally expensive cases.

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? Clearly explain your answer.

A:
pros :

(1) This is easier to implement, since we only need to pick one of the values in the chosen feature.
(2) When the number of features is small, using one random feature may be simple and effective.

cons:

(1) This might lose some information. Since we only consider one of the features, we cannot figure out the relationship between different features. So that this might reduce the predicted power.
(2) Using only one feature at each split might make the model more sensitive to noise in the training data and lead to weak performance on new, unseen data.

Because of these reasons, we cannot say that reducing the variance may lead to improved performance. We still need to consider the number of the features and if there exist relations between different features.

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.

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

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

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

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

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

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

4.a.A : The main difference between the two models is that the left one is the standard neural network that contains all neurons, and the right one is using a Bernoulli random variable with the probability of p to let some of the neurons to be closed, which is called drop out. This can deal with overfitting.

4.b.A :

(1) When we are training, this technique randomly drops out a fraction of neurons in each layer at each iteration. This means that different subsets of neurons are active during different training iterations.

(2) This technique in neural networks can be viewed as a form of implicit ensemble method during training. It encourages the network to learn a diverse set of features and improves the model's ability to generalize by effectively training and combining an ensemble of subnetworks. This contributes to the overall regularization of the model and helps prevent overfitting