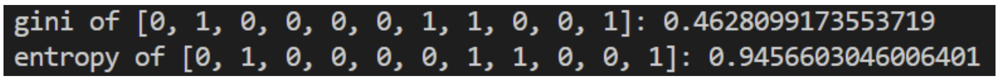
NYCU Introduction to Machine Learning, Homework 3

110550108 施柏江

Part. 1, Coding (50%):

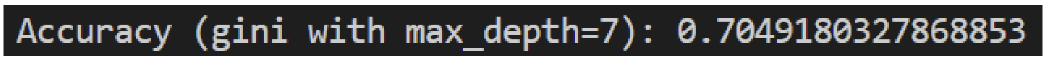
(30%) Decision Tree

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



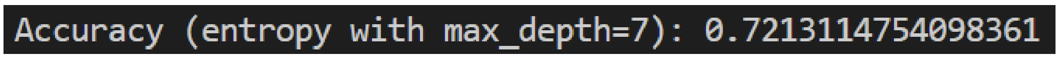
```
gini of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.4628099173553719
entropy of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.9456603046006401
```

2. (10%) Show the accuracy score of the testing data using criterion="gini" and max_depth=7.



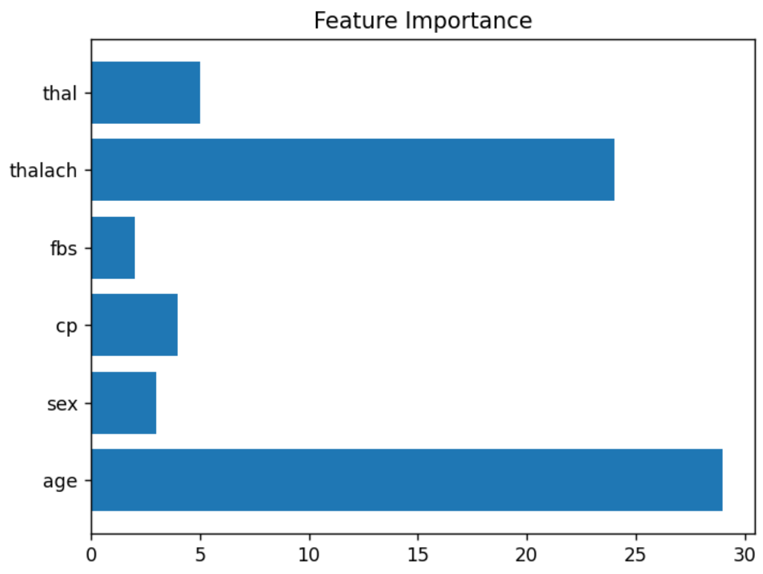
```
Accuracy (gini with max_depth=7): 0.7049180327868853
```

3. (10%) Show the accuracy score of the testing data using criterion="entropy" and max_depth=7.



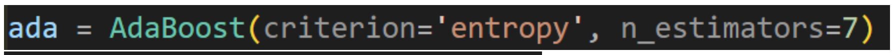
```
Accuracy (entropy with max_depth=7): 0.7213114754098361
```

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.

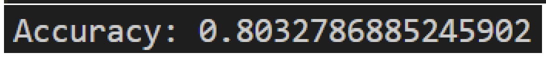


(20%) Adaboost

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



```
ada = AdaBoost(criterion='entropy', n_estimators=7)
```



```
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. In AdaBoost, the weights of misclassified examples are increased, but the emphasis is not achieved by adding the same additive factor to all misclassified examples. Instead, AdaBoost assigns higher weights to the examples that are misclassified more often in previous iterations. The idea is to focus on the examples that are harder to classify correctly.

   b. True. AdaBoost is a meta-algorithm that can be used with various classification methods as its weak classifiers.

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.

Ans:

Too few weak classifiers in AdaBoost can lead to underfitting, causing the model to oversimplify and struggle with complex relationships in the data. This results in high bias and poor adaptation to the dataset intricacies. Conversely, an excessive number of weak classifiers poses the risk of overfitting. The model may memorize training examples, including noise, and struggle to generalize to new data, leading to poor performance on the test set. The computational cost increases with more weak classifiers, as each iteration involves adjusting weights and training a new classifier. Storing many weak classifiers raises memory requirements, impacting resource usage. Too few weak classifiers can result in a less robust model, sensitive to noise and outliers. Increasing their number enhances robustness by reducing the impact of individual misclassifications.

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.

Ans:

The student's proposal to set the number of random features used in each node of each decision tree to 1 to make random forests more powerful might not be a sound approach. In fact, this way may not align with the fundamental principles of random forests. Random forests are designed to be diverse ensembles, and limiting each tree to a single feature could undermine their effectiveness. It's generally recommended to set m to a value that allows for sufficient diversity among trees, striking a balance between bias and variance to achieve optimal generalization performance.

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.

Ans:

a. The main difference between the two models lies in the modified model, which incorporates a technique to introduce randomness during the forward process. In the modified model, a random binary mask is generated from a Bernoulli distribution with probability p. This mask is then element-wise multiplied with y, creating a stochastic version y~. The modified model then uses this stochastic y~ in the computation of z.

b. This technique is a form of dropout. In the context of ensemble methods, dropout can be viewed as creating an ensemble of diverse subnetworks within the overall neural network. Each subnetwork is trained with a random subset of neurons active, and during testing, the predictions are averaged over multiple runs with different sets of active neurons. This ensemble effect helps to improve generalization by reducing overfitting, as the network learns to make predictions that are less dependent on specific neurons. The randomness introduced during training forces the network to learn more robust and distributed representations, making it less likely to memorize noise in the training data.