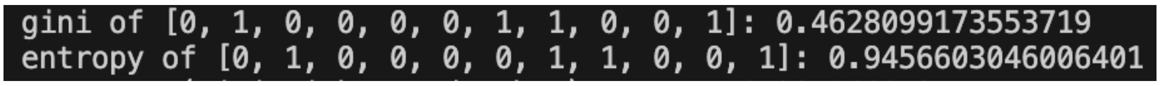
NYCU Introduction to Machine Learning, Homework 3

110550085 房天越

Part. 1, Coding (50%):

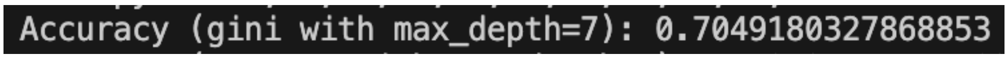
(30%) Decision Tree

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



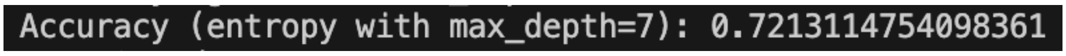
```
gini of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.4628099173553719
entropy of [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1]: 0.9456603046006401
```

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.



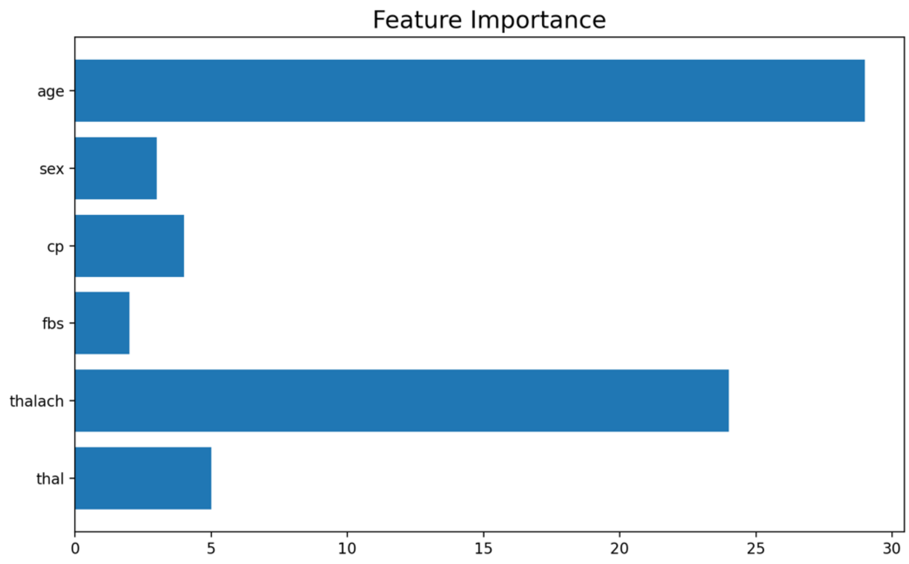
```
Accuracy (gini with max_depth=7): 0.7049180327868853
```

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.



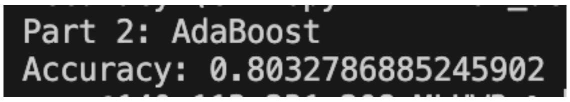
```
Accuracy (entropy with max_depth=7): 0.7213114754098361
```

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:



(20%) 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.

    a. False
    The weights of misclassified examples are indeed increased, but not by adding the same additive factor. Instead, the amount of emphasis is determined by the weak classifier of the weak classifier in the current iteration.
    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.
    I. Underfitting:
    If there are too few weak classifiers, then the performance might be bad because Adaboost relies on enough weak classifiers to build a strong classifier, it might not be able to capture enough underlying features.
    II. Overfitting:
    If there are too many weak classifiers, then the model might start to capture the noises of the training data instead of capturing the overall features. This could also lead to bad performance.
    III. Computational Cost:
    If there are a lot of classifiers, then the computational cost of the model might be very high, while if we do not want it to cost that much, we might have to sacrifice the performance of the model.
    IV. Memory for saving the model
    If there are too many classifiers to use, we would have to save a lot of decision trees of other weak classifiers in the memory, which leads to a very high memory usage, while if we use fewer, the performance might be sacrificed.

To sum up, the number of weak classifiers would affect the model in a lot of factors, when training our Adaboost model, we would have to make sure that we use the most appropriate way to do that.

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.

   No, I do not agree with this student’s claims.
   The student’s proposal contradicts the typical behavior of Random Forest, in which the algorithm randomly selects a subset of features at each node, and m is usually set to the square root of the total number of features to encourage diversity. Setting m to only one would not increase diversity but make the trees too specialized and less robust. Also, it would also not increase accuracy but make the trees too simplistic and would not be able to capture the complex relationship in data. These points all show that the student’s claims are not realistic and applicable.

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.

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

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

   1. (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.

2. (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.

    1. The main difference between the two formulas is that for the formula on the right, it would need to multiply by r before multiplying with w, where r is a random number generated with probability p in Bernouli distribution. This technique is called “Dropout”.
    2. In Ensemble Method, we train a lot of models, and consider the predicted result based on all the models. It can improve the overall performance by taking advantage. Dropout is like the way of Ensemble Method. In every iteration, we stop functioning some neurons. So in every iteration we get a neural network with different structures. It seems like we are training only one model, while actually we are training a model with different structures. So by applying Dropout method, we are not only outputting the result of a neural network, but the complex result of many neural networks. This is the spirit of Ensemble method.