NYCU Introduction to Machine Learning, Homework 3

Deadline: Nov. 15, 23:59

Part. 1, Coding (80%):

In this coding assignment, you need to implement the Decision Tree, AdaBoost and Random Forest algorithm by using only NumPy, then train your implemented model by the provided dataset and test the performance with testing data. Find the sample code and data on the GitHub page
https://github.com/NCTU-VRDL/CS_CS20024/tree/main/HW3

Please note that only NumPy can be used to implement your model, you will get no points by simply calling sklearn.tree.DecsionTreeClassifier.

1. (5%) Gini Index or Entropy is often used for measuring the "best" splitting of the data. Please compute the Entropy and Gini Index of this array `np.array([1,2,1,1,1,1,2,2,1,1,2])` by the formula below.

```
Gini of data is  0.4628099173553719
```

```
Entropy of data is  0.9456603046006402
```

2. (10%) Implement the Decision Tree algorithm (CART, Classification and Regression Trees) and train the model by the given arguments, and print the accuracy score on the test data. You should implement **two arguments** for the Decision Tree algorithm, 1) **Criterion**: The function to measure the quality of a split. Your model should support "gini" for the Gini impurity and "entropy" for the information gain.
   2) **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 split data once

   2.1. Using Criterion='gini', showing the accuracy score of test data by Max_depth=3 and Max_depth=10, respectively.
   Max_depth=3 :

```
Test-set accuarcy score:  0.92
```

   Max_depth=10 :

```
Test-set accuarcy score:  0.94
```

   2.2. Using Max_depth=3, showing the accuracy score of test data by Criterion='gini' and Criterion='entropy', respectively.
   Criterion='gini':

```
Test-set accuarcy score:  0.92
```

Criterion='entropy':

```
Test-set accuarcy score:  0.8933333333333333
```

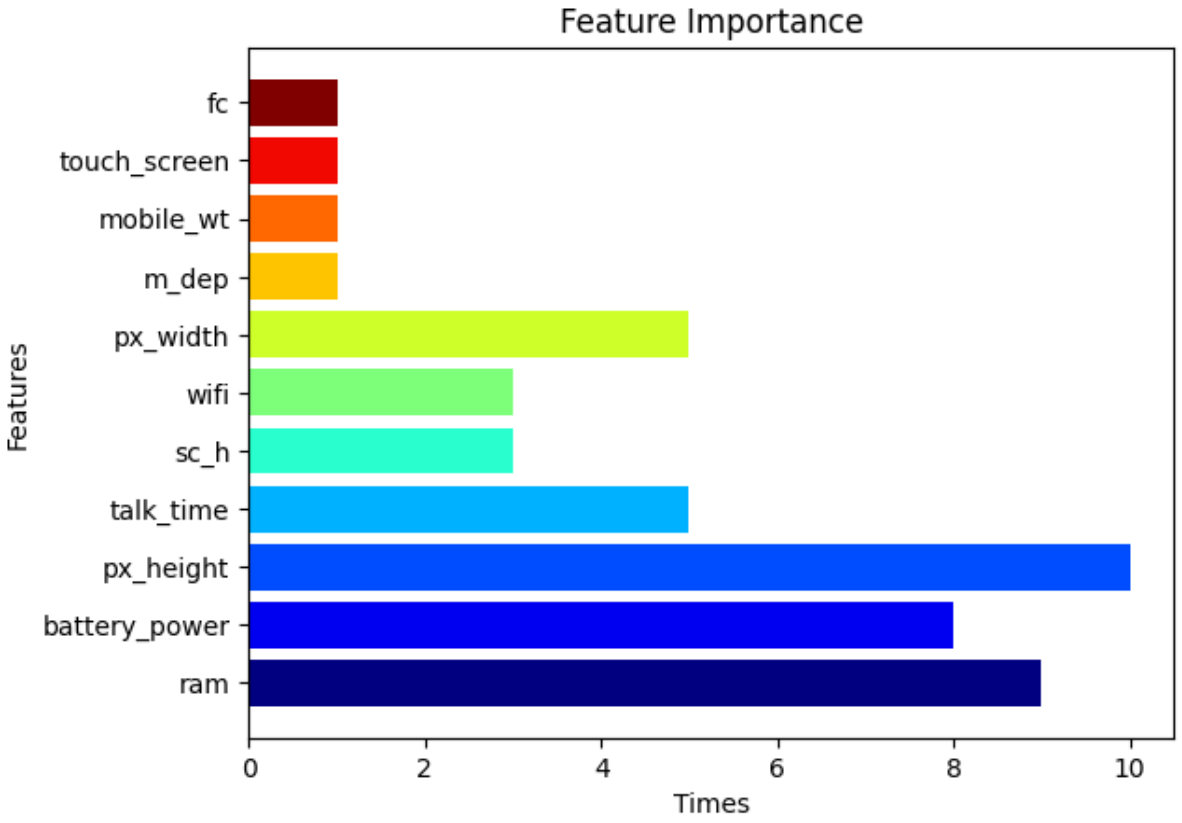
*Note: Your decision tree scores should be over **0.9**. It may suffer from overfitting, if so, you can tune the hyperparameter such as `max_depth`*

*Note: You should get the same results when re-building the model with the same arguments, **no need to prune the trees***

Note: You can find the best split threshold by both methods. First one: 1) Try N-1 threshold values, where the i-th threshold is the average of the i-th and (i+1)-th sorted values. Second one: Use the unique sorted value of the feature as the threshold to split

Hint: You can use the recursive method to build the nodes

3. (5%) Plot the feature importance of your Decision Tree model. You can use the model from Question 2.1, max_depth=10. (You can use simply counting to get the feature importance instead of the formula in the reference, more details on the sample code. **Matplotlib** is allowed to be used)



4. (15%) Implement the AdaBoost algorithm by using the CART you just implemented from question 2. You should implement **one argument** for the AdaBoost.

1) **N_estimators**: The number of trees in the forest.

4.1. Showing the accuracy score of test data by n_estimators=10 and n_estimators=100, respectively.

n_estimators=10 :

```
Test-set accuarcy score:  0.8933333333333333
```

n_estimators=10 :

```
Test-set accuarcy score:  0.9233333333333333
```

5. (15%) Implement the Random Forest algorithm by using the CART you just implemented from question 2. You should implement **three arguments** for the Random Forest.
   1) **N_estimators**: The number of trees in the forest.
   2) **Max_features**: The number of features to consider when looking for the best split
   3) **Bootstrap**: Whether bootstrap samples are used when building trees

   5.1. Using Criterion='gini', Max_depth=None, Max_features=sqrt(n_features), Bootstrap=True, showing the accuracy score of test data by n_estimators=10 and n_estimators=100, respectively.
   n_estimators=10 :

   ```
   Test-set accuarcy score:  0.87
   ```

   n_estimators=100 :

   ```
   Test-set accuarcy score:  0.9133333333333333
   ```

   5.2. Using Criterion='gini', Max_depth=None, N_estimators=10, Bootstrap=True, showing the accuracy score of test data by Max_features=sqrt(n_features) and Max_features=n_features, respectively.
   Max_features=sqrt(n_features) :

   ```
   Test-set accuarcy score:  0.9
   ```

   Max_features=n_features :

   ```
   Test-set accuarcy score:  0.93
   ```

   Note: Use majority votes to get the final prediction, you may get different results when re-building the random forest model

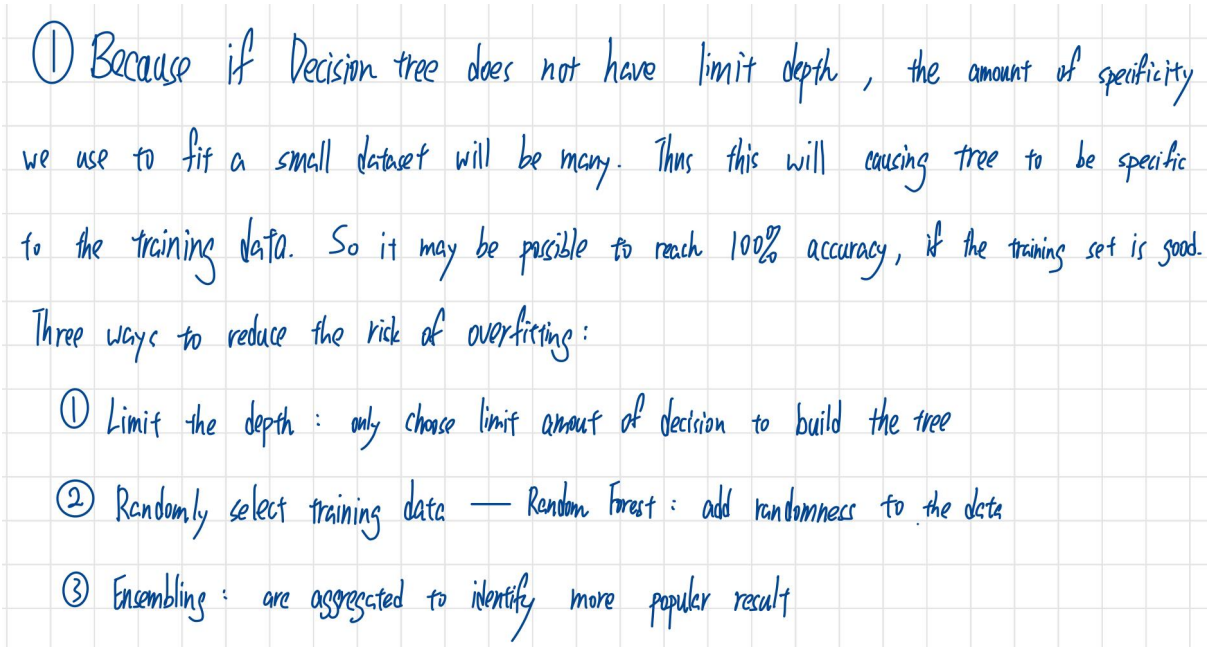
6. (20%) Tune the hyperparameter, perform feature engineering or implement more powerful ensemble methods to get a higher accuracy score. Please note that only the ensemble method can be used. The neural network method is not allowed.

Accuracy	Your scores
acc > 0.975	20 points
0.95 < acc <= 0.975	15 points
0.9 < acc <= 0.95	10 points

acc < 0.9	0 points

Part. 2, Questions (30%):

1. Why does a decision tree have a tendency to overfit to the training set? Is it possible for a decision tree to reach a 100% accuracy in the training set? please explain. List and describe at least 3 strategies we can use to reduce the risk of overfitting of a decision tree.



2. This part consists of three True/False questions. Answer True/False for each question and briefly explain your answer.
   a. In AdaBoost, weights of the misclassified examples go up by the same multiplicative factor.
   b. In AdaBoost, weighted training error ε_t of the t_{th} weak classifier on training data with weights D_t tends to increase as a function of t.
   c. AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

(2)

a. True, follows from the update weight function

b. True, Since boosting aims to classify more difficult examples. The weights will increase when repeated misclassify data by weak classifiers. The weighted training error ϵ_t of the t^{th} weak classifier on the training data therefore trends are performed.

C. Not if the data in the training set cannot be seperated by a linear combination of the specific weak classifiers we are using.

3. Consider a data set comprising 400 data points from class C_1 and 400 data points from class C_2. Suppose that a tree model A splits these into (200, 400) at the first leaf node and (200, 0) at the second leaf node, where (n, m) denotes that n points are assigned to C_1 and m points are assigned to C_2. Similarly, suppose that a second tree model B splits them into (300, 100) and (100, 300). **Evaluate the misclassification rates for the two trees and hence show that they are equal**. Similarly, **evaluate the cross-entropy** $Entropy = -\sum_{k=1}^{K} p_k \log_2 p_k$ and **Gini index** $Gini = 1 - \sum_{k=1}^{K} p_k^2$ **for the two trees**. Define p_k to be the proportion of data points in region R assigned to class k, where k = 1, . . . , K.

3.

Tree A:

left

Gini: $1-(\frac{1}{3})^2-(\frac{2}{3})^2 = 1-\frac{5}{9} = \frac{4}{9}$

right

$1-1=0$

cross-entropy:

left

$-\frac{1}{3}\log_2\frac{1}{3}-\frac{2}{3}\log_2\frac{2}{3} = -\log_2(\frac{1}{3})^{\frac{1}{3}}(\frac{2}{3})^{\frac{2}{3}} = -\log_2(\frac{1}{3})\cdot(\frac{1}{3})^{\frac{2}{3}} = -\frac{2}{3}\cdot\log_2\frac{1}{3} \doteq 0.918$

right

$-\log_2 1 = 0$

misclassification rates: $\frac{200}{200+200+1200} = \frac{1}{4}$

Tree B:

left

Gini: $1-(\frac{5}{6})^2-(\frac{1}{6})^2 = 1-\frac{10}{16} = \frac{6}{16} = \frac{3}{8}$

right

$1-(\frac{3}{4})^2-(\frac{1}{4})^2 = \frac{3}{8}$

Cross Entropy: left

$-\frac{3}{4}\log_2\frac{3}{4}-\frac{1}{4}\log_2\frac{1}{4} = -\log_2\frac{1}{4}-\frac{3}{4}\log_2 3 = 0.811$

right

$-\frac{3}{4}\log_2\frac{3}{4}-\frac{1}{4}\log_2\frac{1}{4} = -\log_2\frac{1}{4}-\frac{3}{4}\log_2 3 = 0.811$

misclassification rates: $\frac{100+100}{100+300+100+300} = \frac{200}{800} = \frac{1}{4}$