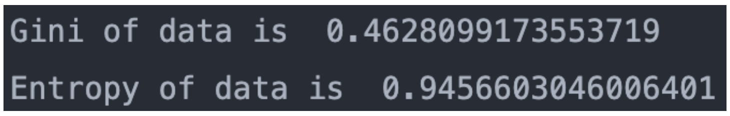
NYCU Introduction to Machine Learning, Homework 3

陳存佩 109550171

Part. 1, Coding (70%):

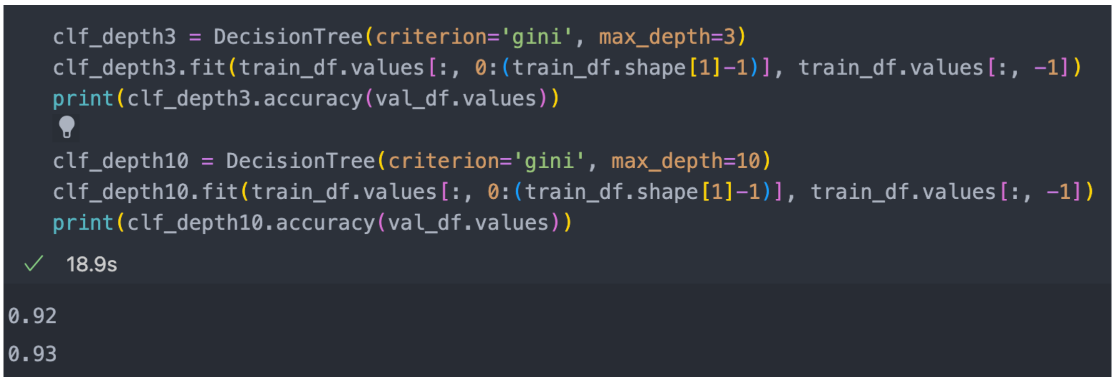
1. (5%) Gini Index or Entropy is often used for measuring the "best" splitting of the data. Please compute the Entropy and Gini Index.



```
Gini of data is  0.4628099173553719
Entropy of data is  0.9456603046006401
```

2. (10%) Implement the Decision Tree algorithm

1.

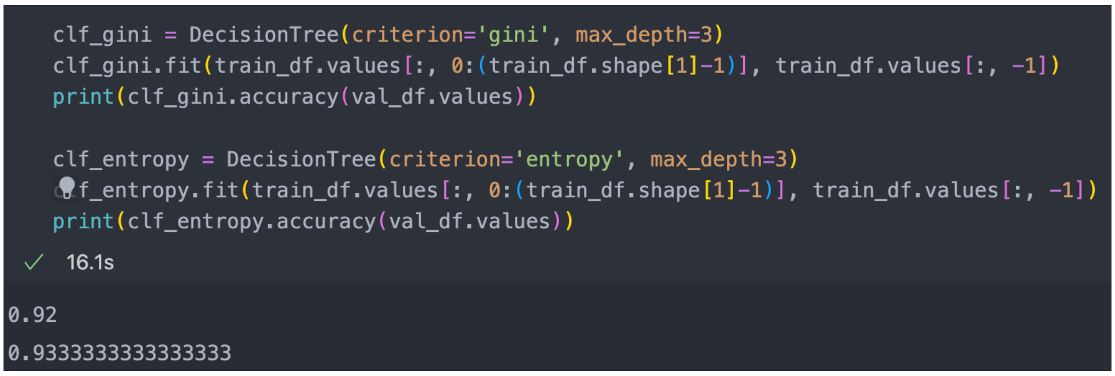


```
clf_depth3 = DecisionTree(criterion='gini', max_depth=3)
clf_depth3.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_depth3.accuracy(val_df.values))

clf_depth10 = DecisionTree(criterion='gini', max_depth=10)
clf_depth10.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_depth10.accuracy(val_df.values))
```

✓ 18.9s

```
0.92
0.93
```

2.

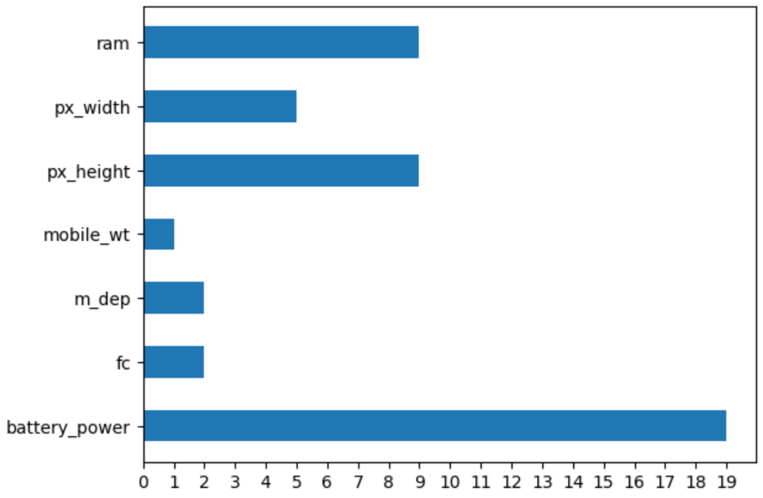


```
clf_gini = DecisionTree(criterion='gini', max_depth=3)
clf_gini.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_gini.accuracy(val_df.values))

clf_entropy = DecisionTree(criterion='entropy', max_depth=3)
clf_entropy.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_entropy.accuracy(val_df.values))
```

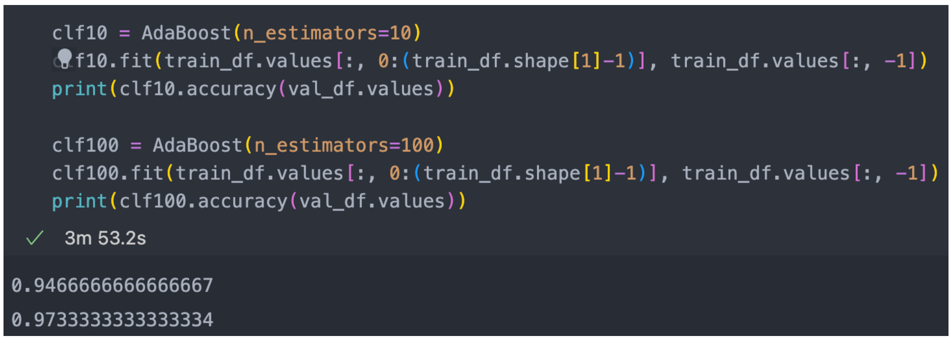
✓ 16.1s

```
0.92
0.9333333333333333
```

3. (5%) Plot the feature importance of your Decision Tree model.



4. (15%) Implement the AdaBoost algorithm by using the CART you just implemented from question 2.



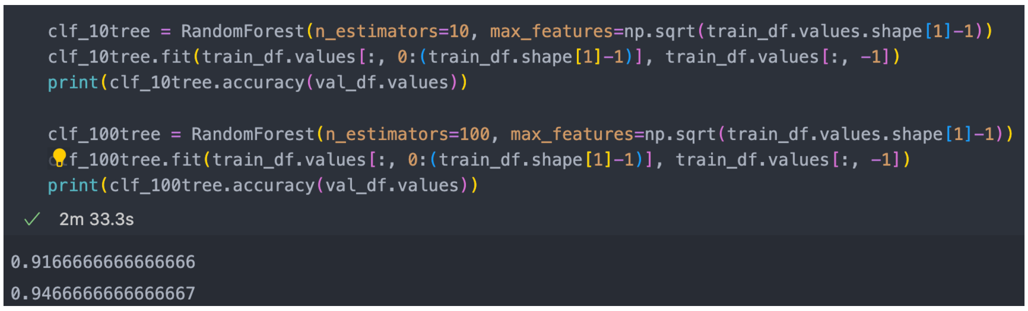
```
clf10 = AdaBoost(n_estimators=10)
clf10.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf10.accuracy(val_df.values))

clf100 = AdaBoost(n_estimators=100)
clf100.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf100.accuracy(val_df.values))
```

✓ 3m 53.2s

```
0.9466666666666667
0.9733333333333334
```

5. (15%) Implement the Random Forest algorithm by using the CART you just implemented from question 2.

1.

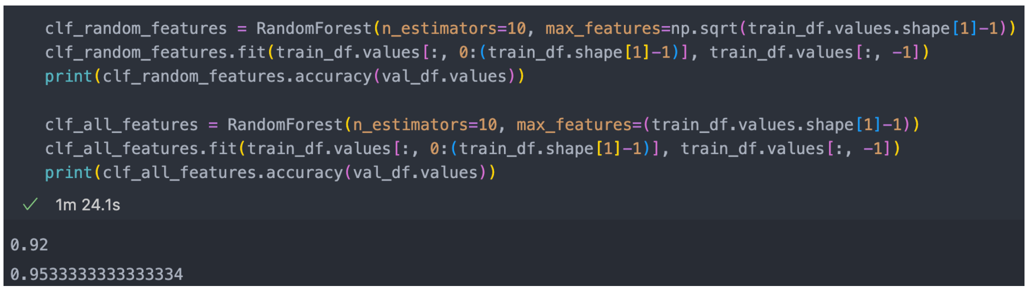


```
clf_10tree = RandomForest(n_estimators=10, max_features=np.sqrt(train_df.values.shape[1]-1))
clf_10tree.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_10tree.accuracy(val_df.values))

clf_100tree = RandomForest(n_estimators=100, max_features=np.sqrt(train_df.values.shape[1]-1))
clf_100tree.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_100tree.accuracy(val_df.values))
```

✓ 2m 33.3s

```
0.9166666666666666
0.9466666666666667
```

2.



```
clf_random_features = RandomForest(n_estimators=10, max_features=np.sqrt(train_df.values.shape[1]-1))
clf_random_features.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_random_features.accuracy(val_df.values))

clf_all_features = RandomForest(n_estimators=10, max_features=(train_df.values.shape[1]-1))
clf_all_features.fit(train_df.values[:, 0:(train_df.shape[1]-1)], train_df.values[:, -1])
print(clf_all_features.accuracy(val_df.values))
```

✓ 1m 24.1s

```
0.92
0.9533333333333334
```

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.

   The decision tree examines the data in lots of ways. At each node, it looks every possible split of every distinct value, resulting in memorizing the noise of the training data and failing to capture important patterns.

   It is possible to reach a 100% accuracy decision tree using enough depth and features, but it increases the probability of overfitting.

   1. **Pre-pruning using max_depth:** It stops the split when reaching the max_depth, preventing the tree grows to its full depth.

   2. **Pre-pruning using min_sample_leaf:** It sets the minimum number of samples required to be a leaf node (default = 1)

   3. **Post-pruning:** It removes some branches from the tree with full depth. CCP(cost complexity pruning) is a kind of post-pruning technique. It finds the optimal value for alpha which means the number of nodes pruned.

   4. **Random forest:** It uses bagging(random sample dataset) and random input vectors(each node only considers some random chosen features) techniques.

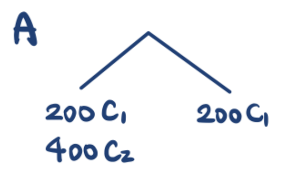
2. This part consists of three True/False questions. Answer True/False for each question and briefly explain your answer.
   1. In AdaBoost, weights of the misclassified examples go up by the same multiplicative factor.
   2. In AdaBoost, weighted training error $\boldsymbol{\varepsilon}_t$ of the t_{th}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

   1. True, it follows the same update equation: $D_{t+1}(i) = \frac{D_t(i)\exp[-\alpha_t y_i h_t(x_i)]}{Z_t}$,for misclassified data, $y_t h_t(x_t) = -1$

   2. True, D_t tends to increase as a function of t because the weak classifiers try to put more emphasis on data which misclassified many times.

   3. False, if the data cannot be separated by a linear combination of the weak classifiers, no matter how many iterations provided, it cannot reach zero training error.

3. Consider a data set comprising 400 data points from class C1 and 400 data points from class C2. 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 C1 and m points are assigned to C2. Similarly, suppose that a second tree model B splits them into (300, 100) and (100, 300). **Evaluate the <u>misclassification rates</u> for the two trees and hence show that they are equal**. Similarly, **evaluate the cross-entropy Entropy=** and **Gini index for the two trees**. Define pk to be the proportion of data points in region R assigned to class k, where k = 1, . . . , K.



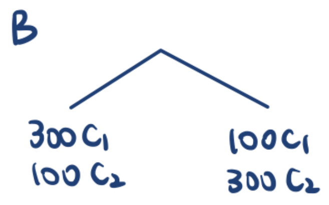
<u>misscassification rate</u> $= \frac{200}{800} = \frac{1}{4}$

<u>entropy</u> first : $-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} \approx 0.9183$

second: $-1 \cdot \log_2 1 = 0$

<u>gini</u> first : $1 - \frac{1}{3}^2 - \frac{2}{3}^2 = \frac{4}{9}$

second: $1 - 1^2 - 0^2 = 0$



<u>misscassification rate</u> $= \frac{200}{800} = \frac{1}{4}$

<u>entropy</u> first : $-\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} \approx 0.81128$

second: $-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4} \approx 0.81128$

<u>gini</u> first : $1 - \frac{3}{4}^2 - \frac{1}{4}^2 = \frac{3}{8}$

second: $1 - \frac{1}{4}^2 - \frac{3}{4}^2 = \frac{3}{8}$