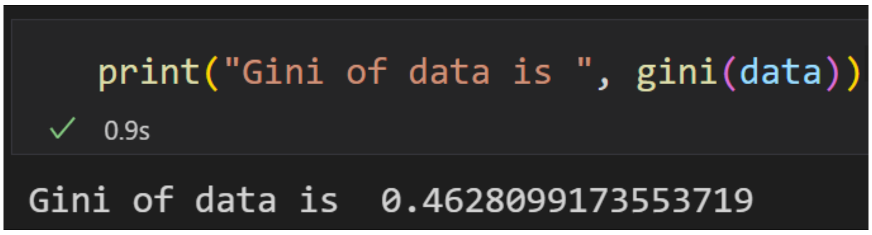
109550031_HW3

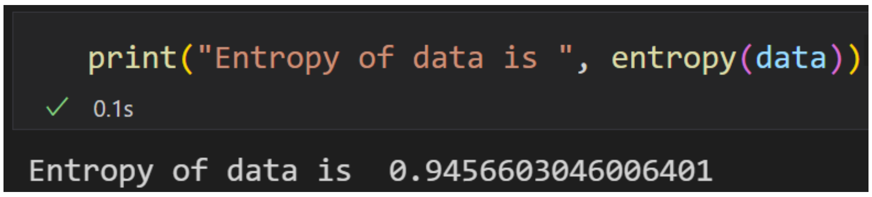
Part. 1, Coding

1. Please compute the Entropy and Gini Index of this array np.array([1,2,1,1,1,1,2,2,1,1,2]).



```
print("Gini of data is ", gini(data))
✓ 0.9s
Gini of data is  0.4628099173553719
```

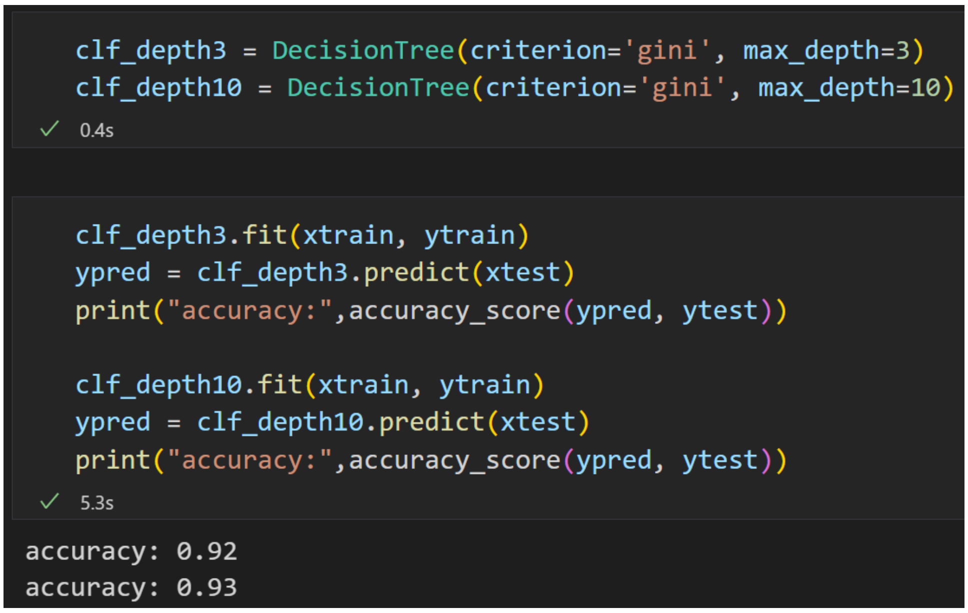
Gini Index



```
print("Entropy of data is ", entropy(data))
✓ 0.1s
Entropy of data is  0.9456603046006401
```

Entropy

2. Implement the Decision Tree algorithm and train the model by the given arguments, and print the accuracy score on the test data.

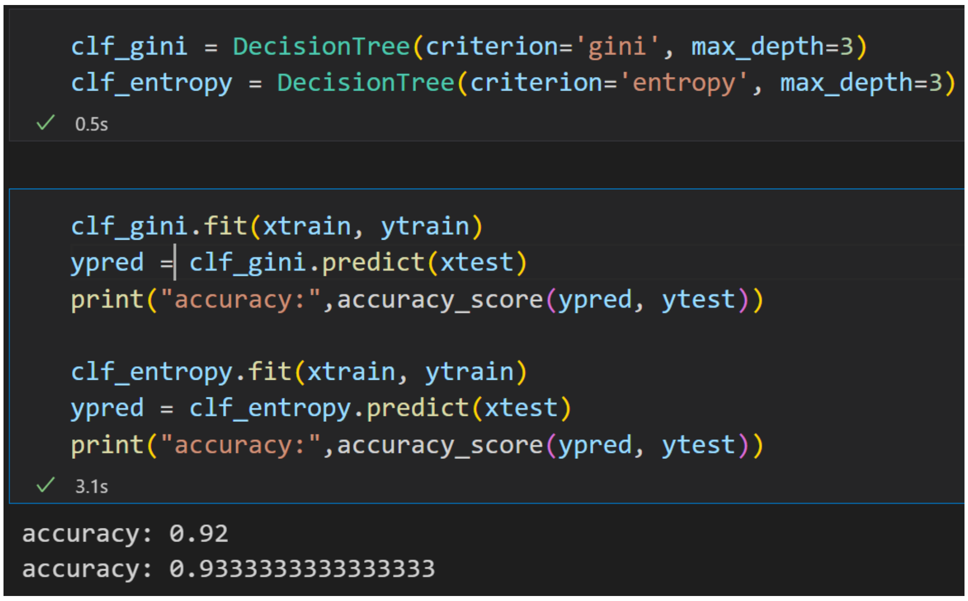


```
clf_depth3 = DecisionTree(criterion='gini', max_depth=3)
clf_depth10 = DecisionTree(criterion='gini', max_depth=10)
✓ 0.4s
```

```
clf_depth3.fit(xtrain, ytrain)
ypred = clf_depth3.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))

clf_depth10.fit(xtrain, ytrain)
ypred = clf_depth10.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))
✓ 5.3s
accuracy: 0.92
accuracy: 0.93
```

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



```
clf_gini = DecisionTree(criterion='gini', max_depth=3)
clf_entropy = DecisionTree(criterion='entropy', max_depth=3)
```
✓ 0.5s

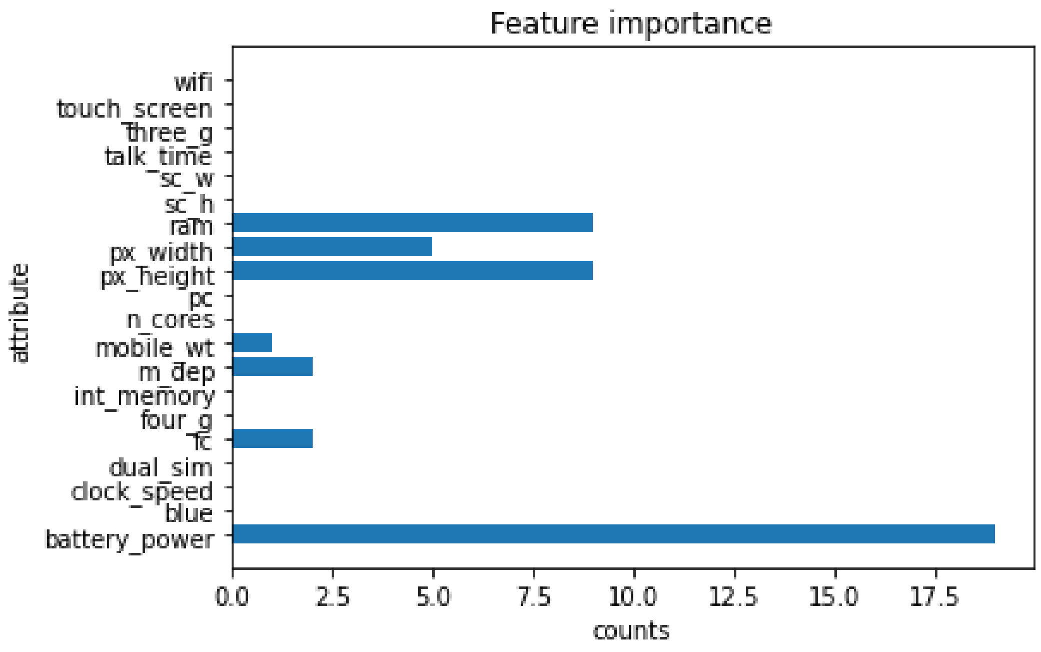
```
clf_gini.fit(xtrain, ytrain)
ypred = clf_gini.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))

clf_entropy.fit(xtrain, ytrain)
ypred = clf_entropy.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))
```
✓ 3.1s

```
accuracy: 0.92
accuracy: 0.9333333333333333
```

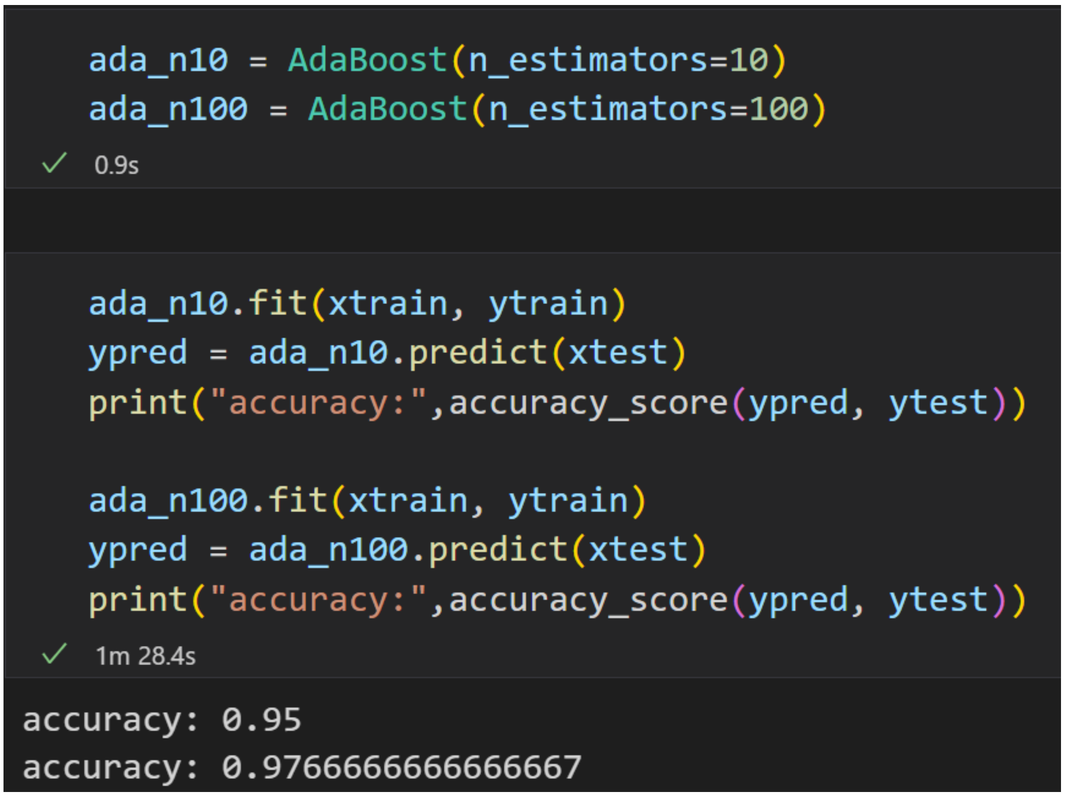
2.2 Using Max_depth=3, showing the accuracy score of test data by Criterion='gini' and Criterion='entropy', respectively

3. Plot the feautre importance of your Decision Tree model.



feature importance of Question 2.1, max_depth=10

4. Implement the AdaBoost algorithm by using the CART.



```
ada_n10 = AdaBoost(n_estimators=10)
ada_n100 = AdaBoost(n_estimators=100)
```
✓ 0.9s

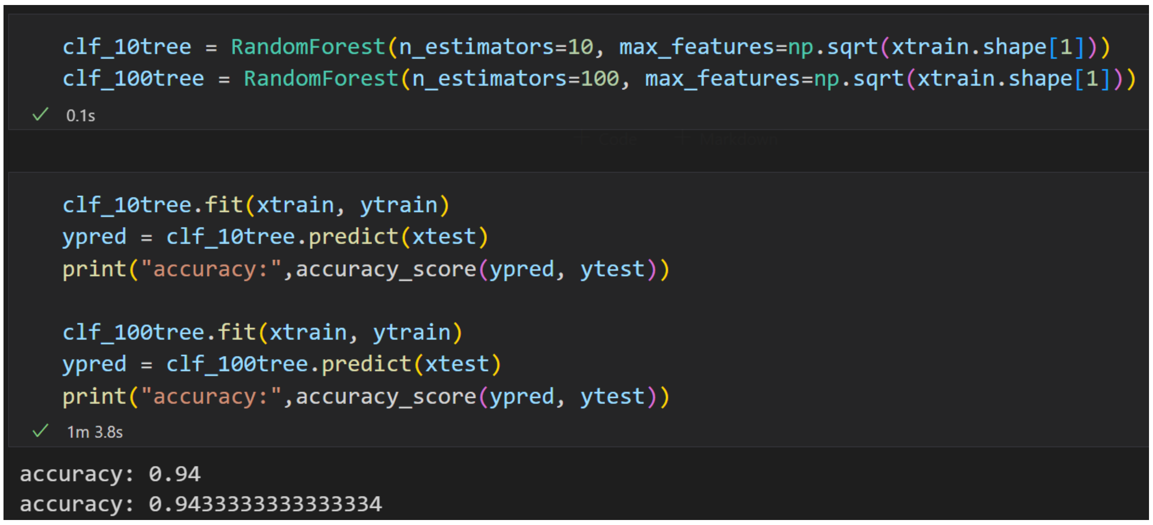
```
ada_n10.fit(xtrain, ytrain)
ypred = ada_n10.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))

ada_n100.fit(xtrain, ytrain)
ypred = ada_n100.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))
```
✓ 1m 28.4s

```
accuracy: 0.95
accuracy: 0.9766666666666667
```

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

5. Implement the Random Forest algorithm by using the CART.



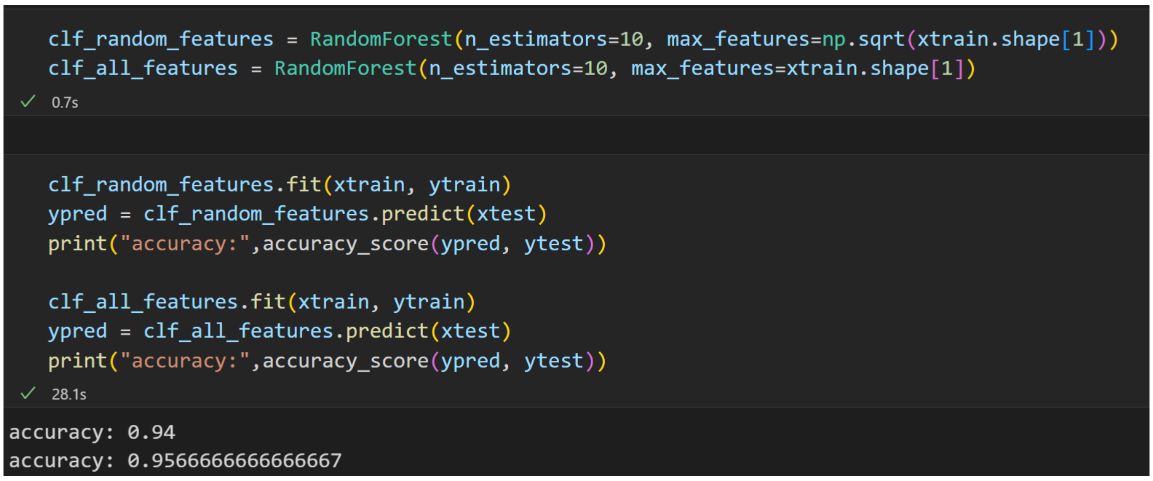
```
clf_10tree = RandomForest(n_estimators=10, max_features=np.sqrt(xtrain.shape[1]))
clf_100tree = RandomForest(n_estimators=100, max_features=np.sqrt(xtrain.shape[1]))
```
✓ 0.1s

```
clf_10tree.fit(xtrain, ytrain)
ypred = clf_10tree.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))

clf_100tree.fit(xtrain, ytrain)
ypred = clf_100tree.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))
```
✓ 1m 3.8s

```
accuracy: 0.94
accuracy: 0.9433333333333334
```

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.



```
clf_random_features = RandomForest(n_estimators=10, max_features=np.sqrt(xtrain.shape[1]))
clf_all_features = RandomForest(n_estimators=10, max_features=xtrain.shape[1])
```
✓ 0.7s

```
clf_random_features.fit(xtrain, ytrain)
ypred = clf_random_features.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))

clf_all_features.fit(xtrain, ytrain)
ypred = clf_all_features.predict(xtest)
print("accuracy:",accuracy_score(ypred, ytest))
```
✓ 28.1s

```
accuracy: 0.94
accuracy: 0.9566666666666667
```

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.

Part. 2, Questions

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.**

   In decision tree, overfit condition arises when the model memorizes the noise of the training data and fails to capture important patterns so that the decision tree performs well for training data but performs poorly for unseen test data.

   It is possible for a decision tree to reach a 100% accuracy in the training set. It occurs when the decision tree tries to cover all the training data points and the data in leaf nodes are pure. But, high accuracy measured on the training set is the result of overfitting.

   There are three strategies to reduce the risk of overfitting of a decision tree.

   a. Pruning

      Pruning removes the parts of the decision tree to prevent growing to its full depth. The hyperparameters of the decision tree such as `max_depth` can be tuned to early stop the growth of the tree and prevent the model from overfitting.

   b. Boosting

      Boosting focuses on converting a weak learner to a stronger one by improving the predictive flexibility of much simpler models. The weak learners can focus on learning from their previous mistakes. Once this is done, the weak learners are all combined into a strong learner.

c. Random Forest

Random Forest is an ensemble of decision trees. A large ensemble of uncorrelated decision trees in a random forest can produce more accurate results than any of the individual decision trees. In random forest, each tree can search for the best feature out of a random subset of features. This creates extra randomness when growing the trees inside a random forest to prevent overfitting.

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.**

      True, weights go up according to the wrong classification by the same multiplicative factor.

   b. **In AdaBoost, weighted training error εt of the t_{th} weak classifieron training data with weights D_t tends to increase as a function of t.**

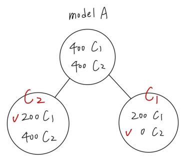
      True, the weights will increase for data which are repeatedly misclassified by the weak classifier, so the weighted training error of the t_{th} weak classifier on the training data therefore tends to increase.

   c. **AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.**

      False, if the data in the training set can't be separated by a linear combination of the specific type weak classifiers, AdaBoost will not give 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 misclassification rates for the two trees and hence show that they are equal. Similarly, evaluate the cross-entropy $Entropy = -\Sigma_{k=1}^{K} p_k log_2 pk$ and Gini index $Gini = 1 - \Sigma_{k=1}^{K} p_k^2$ for**

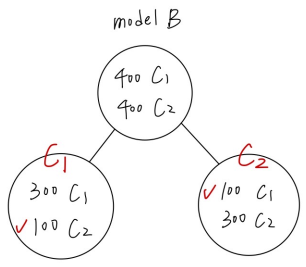
the two trees. Define p_k to be the proportion of data points in region R assigned to class k, where k = 1, . . . , K.



misclassification rates: $\frac{200}{400+400} = 0,25$

cross-entropy: $-\left[\frac{1}{3}\log_2(\frac{1}{3}) + \frac{2}{3}\log_2(\frac{2}{3})\right] \times \frac{3}{4} - \left[1\log_2 1 + 0\right] \times \frac{1}{4} \approx 0,6887$

gini index: $\left[1-(\frac{1}{3})^2 - (\frac{2}{3})^2\right] \times \frac{3}{4} + \left[1-1^2\right] \times \frac{1}{4} = \frac{1}{3} \approx 0,333$



misclassification rates: $\frac{100+100}{400+400} = 0,25$

cross-entropy: $-\left[\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right] \times \frac{1}{2} - \left[\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right] \times \frac{1}{2} \approx 0,8112$

gini index: $\left[1-(\frac{3}{4})^2 - (\frac{1}{4})^2\right] \times \frac{1}{2} + \left[1-(\frac{1}{4})^2 - (\frac{3}{4})^2\right] \times \frac{1}{2} = \frac{3}{8} = 0,375$

📢 the misclassification rates for the two trees are both equal to 0.25