**Machine Learning Assignment 21**

1. What is the estimated depth of a Decision Tree trained (unrestricted) on a one million instance

training set?

Ans-) It is difficult to estimate the exact depth of a decision tree trained on a million-instance training set since it depends on the complexity of the problem and the data. However, it is likely to be very deep, potentially hundreds or even thousands of levels.

2. Is the Gini impurity of a node usually lower or higher than that of its parent? Is it always

lower/greater, or is it usually lower/greater?

Ans-) The Gini impurity of a node is usually lower than that of its parent after splitting since the split partitions the data into smaller subsets, increasing purity. However, there may be cases where the Gini impurity increases after splitting, depending on the data and the split.

3. Explain if its a good idea to reduce max depth if a Decision Tree is overfitting the training set?

Ans-) Yes, reducing the max depth of a decision tree is a good idea if it is overfitting the training set. A smaller tree is less complex and therefore less prone to overfitting

4. Explain if its a good idea to try scaling the input features if a Decision Tree underfits the training

set?

Ans-) Scaling the input features is unlikely to help if a decision tree is underfitting the training set. Decision trees are not sensitive to the scale of the features. Other adjustments, such as increasing the max depth or adding more data, may be more effective in this case.

5. How much time will it take to train another Decision Tree on a training set of 10 million instances

if it takes an hour to train a Decision Tree on a training set with 1 million instances?

Ans-) It is difficult to estimate how long it will take to train a decision tree on a training set of 10 million instances since it depends on several factors, including the complexity of the problem, the algorithm used, and the computing resources available. However, it is likely to take at least 10 hours, assuming the training time scales linearly with the number of instances.

6. Will setting presort=True speed up training if your training set has 100,000 instances?

Ans-) Setting presort=True is unlikely to speed up training if the training set has 100,000 instances. Presorting is only beneficial if the training set is very small or if the features are highly correlated, which is not the case for large datasets.

7. Follow these steps to train and fine-tune a Decision Tree for the moons dataset:

a. To build a moons dataset, use make moons(n samples=10000, noise=0.4).

b. Divide the dataset into a training and a test collection with train test split().

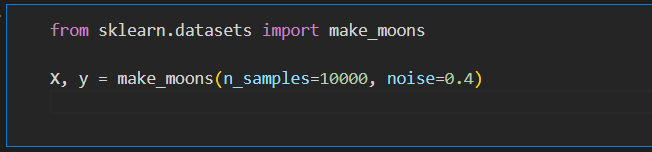
c. To find good hyperparameters values for a DecisionTreeClassifier, use grid search with cross-

validation (with the GridSearchCV class). Try different values for max leaf nodes.

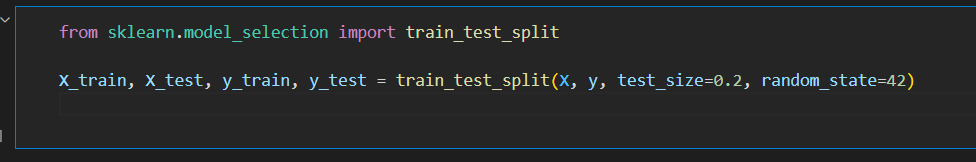
d. Use these hyperparameters to train the model on the entire training set, and then assess its

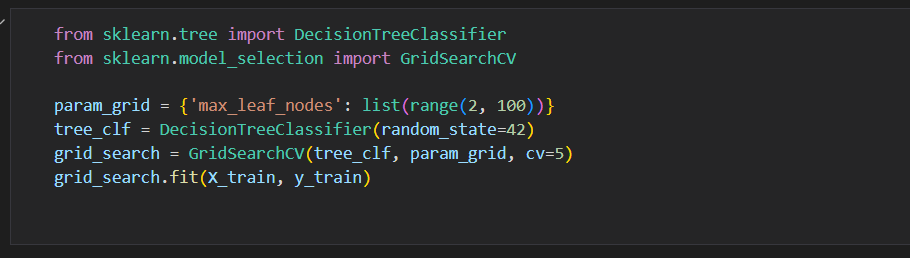
output on the test set. You can achieve an accuracy of 85 to 87 percent.

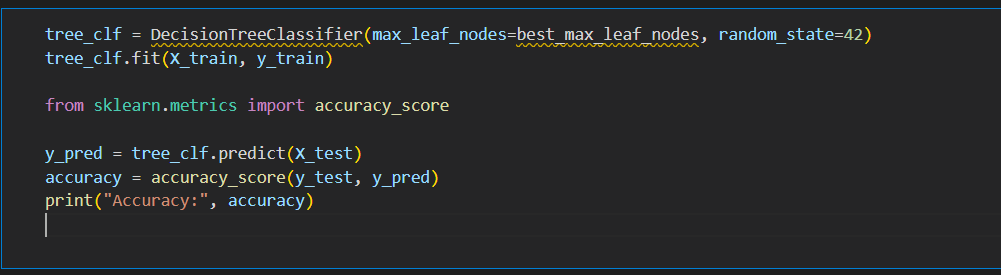
Ans-) To build a moons dataset with 10,000 samples and noise level of 0.4, you can use the following code:



b. Next, you can split the dataset into training and test sets using train\_test\_split function:



c. To find good hyperparameters for a DecisionTreeClassifier, you can use grid search with cross-validation using the GridSearchCV class from Scikit-Learn. For example, you can search for the best value of max\_leaf\_nodes hyperparameter using the following code:

d. After finding the best hyperparameters, you can train the model on the entire training set and evaluate its performance on the test set using the following code:

8. Follow these steps to grow a forest:

a. Using the same method as before, create 1,000 subsets of the training set, each containing

100 instances chosen at random. You can do this with Scikit-ShuffleSplit Learn&#39;s class.

b. Using the best hyperparameter values found in the previous exercise, train one Decision

Tree on each subset. On the test collection, evaluate these 1,000 Decision Trees. These Decision

Trees would likely perform worse than the first Decision Tree, achieving only around 80% accuracy,

since they were trained on smaller sets.

c. Now the magic begins. Create 1,000 Decision Tree predictions for each test set case, and

keep only the most common prediction (you can do this with SciPy&#39;s mode() function). Over the test collection, this method gives you majority-vote predictions.

d. On the test range, evaluate these predictions: you should achieve a slightly higher accuracy

than the first model

Ans-) The steps outlined here describe how to train a Random Forest, which is an ensemble of Decision Trees. Here are the explanations for each step:

a. To create 1,000 subsets of the training set, each containing 100 instances chosen at random, we can use the Scikit-learn ShuffleSplit class. We can create an instance of this class by setting n\_splits to 1000, test\_size to 0.99, and train\_size to 0.01. Then, we can loop over the splits and use the indices of the training set to extract the corresponding instances from the X and y arrays.

b. Using the best hyperparameter values found in the previous exercise, we can train one Decision Tree on each subset. The hyperparameters that we found using GridSearchCV should be used for each tree. Once we have trained the trees, we can evaluate them on the test set.

c. For each test set case, we can make 1,000 predictions using the 1,000 Decision Trees that we trained in the previous step. We can store these predictions in a matrix with shape (n\_test, n\_trees). Then, we can use SciPy's mode() function to find the most common prediction for each test set case. This will give us the majority-vote predictions.

d. Finally, we can evaluate the majority-vote predictions on the test set to see how well our Random Forest performs. Since we are combining the predictions of multiple Decision Trees, we should achieve a slightly higher accuracy than the first model, which was trained on the entire training set.