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

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?

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

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

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?

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

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.

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'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'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 (approx 0.5 to 1.5 percent higher). You've successfully learned a Random Forest classifier!  
    
  **Estimated Depth of Decision Tree:**
* The estimated depth of a decision tree trained on a one million instance training set would depend on factors like the complexity of the data and the hyperparameters chosen during training. However, with a large dataset, the tree could potentially grow deep to capture intricate patterns.
* **Gini Impurity of a Node:**
* The Gini impurity of a node is usually lower than that of its parent. The algorithm seeks to split nodes in a way that minimizes impurity, leading to child nodes that are purer than the parent.
* **Reducing Max Depth to Prevent Overfitting:**
* Yes, it's generally a good idea to reduce the maximum depth if a Decision Tree is overfitting the training set. Reducing the depth limits the tree's complexity, preventing it from capturing noise in the data and improving generalization to unseen instances.
* **Scaling Input Features for Underfitting Decision Tree:**
* Scaling input features is typically not necessary for decision trees. Decision trees are not sensitive to the scale of input features, and underfitting in decision trees is more likely due to other factors such as insufficient tree depth or inadequate features.
* **Training Time for 10 Million Instances:**
* The training time for a Decision Tree is generally proportional to the number of instances and features. Therefore, training on 10 million instances might take approximately 10 times longer, so it could be around 10 hours.
* **Presort=True to Speed Up Training:**
* No, setting presort=True is unlikely to speed up training for a dataset with 100,000 instances. Presorting is beneficial for smaller datasets but becomes computationally expensive for larger datasets.
* **Training and Fine-Tuning Decision Tree for Moons Dataset:**
* a. Create Moons Dataset: **make\_moons(n\_samples=10000, noise=0.4)**
* b. Split into Training and Test Sets: **train\_test\_split()**
* c. Hyperparameter Tuning: Use GridSearchCV to find optimal max\_leaf\_nodes.
* d. Train and Evaluate: Train the model on the entire training set and assess performance on the test set.
* **Growing a Forest (Random Forest):**
* a. Create 1,000 Subsets: Use ShuffleSplit to create subsets with 100 instances.
* b. Train Decision Trees: Train one Decision Tree on each subset with the best hyperparameter values.
* c. Aggregate Predictions: For each test set case, create 1,000 Decision Tree predictions and keep the most common prediction.
* d. Evaluate Random Forest: Evaluate the aggregated predictions on the test set to achieve a slightly higher accuracy than a single Decision Tree.