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

**Ans:** The estimated depth of a decision tree trained on a one million instance training set is likely to be relatively deep, as decision trees tend to become deeper with more data. However, the exact depth can vary depending on the complexity of the data and the specific features used for the decision tree.

**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 because the purpose of the decision tree is to split nodes in a way that minimizes impurity. However, it is not always guaranteed to be strictly lower, as certain splits may not decrease the impurity, especially in cases where there are ties in the feature values.

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

**Ans:** It can be a good idea to reduce the max depth if a decision tree is overfitting the training set. By reducing the depth, the model becomes simpler, reducing the risk of overfitting. This can lead to improved generalization performance on unseen data, making the model more reliable.

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

**Ans:** Scaling input features is generally not required for decision trees because they are not sensitive to the scale of the features. If a decision tree underfits the training set, it's better to investigate other potential causes, such as insufficient depth or lack of informative features, rather than focusing on feature scaling.

**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:** If it takes an hour to train a decision tree on a training set with 1 million instances, training a decision tree on a training set with 10 million instances would approximately take 10 hours. This estimate assumes that 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 significantly with a training set of 100,000 instances. The presorting option is beneficial for small datasets but can significantly slow down training for larger datasets. Therefore, it might not be advantageous for datasets with 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.**

**Ans:** These steps involve creating a moons dataset, splitting it into a training and test set, performing grid search with cross-validation for finding optimal hyperparameters, training the model, and evaluating its performance on the test set. The expected accuracy ranges from 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!**

**Ans:** The process involves creating 1,000 subsets of the training set, training one decision tree on each subset, aggregating the predictions using the mode function, and evaluating the random forest's accuracy. The random forest is expected to achieve slightly higher accuracy compared to the initial decision tree, typically around 0.5 to 1.5 percent higher.