Train and fine-tune a Decision Tree for the moons dataset by following these steps:

Use make_moons(n_samples=10000, noise=0.4) to generate a moons dataset. Use train_test_split() to split the dataset into a training set and a test set. Use grid search with cross-validation (with the help of the GridSearchCV class) to find good hyperparameter values for a DecisionTreeClassifier. Hint: try various values for max_leaf_nodes. Train it on the full training set using these hyperparameters, and measure your model's performance on the test set. You should get roughly 85% to 87% accuracy.

In [ ]:
```
# 1. Use make_moons(n_samples=10000, noise=0.4) to generate a moons dataset.
from sklearn.datasets import make_moons
X, y = make_moons(n_samples=10000, noise=0.4, random_state=42)
```

In [ ]:
```
print(X)

print(y)
```

```
[[ 0.9402914   0.12230559]
 [ 0.12454026 -0.42477546]
 [ 0.26198823  0.50841438]
 ...
 [-0.24177973  0.20957199]
 [ 0.90679645  0.54958215]
 [ 2.08837082 -0.05050728]]
[1 0 0 ... 1 0 1]
```

In [ ]:
```
print(X.shape)
print(y.shape)
```

```
(10000, 2)
(10000,)
```

In [ ]:
```
# 2. Split it into a training set and a test set using train_test_split()
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
```

In [ ]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(8000, 2)
(2000, 2)
(8000,)
(2000,)
```

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
```

In [ ]:
```
# 3. Use grid search with cross-validation (with the help of the GridSearchCV class


params = {'max_leaf_nodes': list(range(2, 100)), 'min_samples_split': [2, 3, 4]}
grid_search_cv = GridSearchCV(DecisionTreeClassifier(random_state=42), params, verb

grid_search_cv.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 294 candidates, totalling 882 fits
```

Out[ ]:
```
GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'max_leaf_nodes': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                                            13, 14, 15, 16, 17, 18, 19, 20, 21,
                                            22, 23, 24, 25, 26, 27, 28, 29, 30,
                                            31, ...],
                         'min_samples_split': [2, 3, 4]},
             verbose=1)
```

In [ ]:
```
grid_search_cv.best_estimator_
```

Out[ ]:
```
DecisionTreeClassifier(max_leaf_nodes=17, random_state=42)
```

In [ ]:
```
# 4. Train it on the full training set using these hyperparameters, and measure you
from sklearn.metrics import accuracy_score

y_pred = grid_search_cv.predict(X_test)
accuracy_score(y_test, y_pred)
```

Out[ ]:
```
0.8695
```

In [ ]:
```

```