```
import pandas as pd


df = pd.read_excel('/content/car.xlsx')
df
```

	brand	model	transmission	age	fuel	price	mileage	power	seats
0	18	244	1	4	1	1231000.0	19.01	4.496471	5
1	10	263	1	6	4	786000.0	19.01	4.496471	5
2	31	123	1	2	1	1489000.0	19.01	4.496471	5
3	9	55	0	1	4	1227000.0	19.01	4.496471	5
4	8	82	1	3	1	887000.0	19.01	4.496471	5
...	...	...	...	...	...	...	...	...	...
32009	5	199	1	6	4	292000.0	19.01	4.496471	5
32010	32	295	1	6	4	534000.0	19.01	4.496471	5
32011	33	25	1	8	4	424000.0	19.01	4.496471	5
32012	10	120	0	5	4	685000.0	19.01	4.496471	5
32013	31	247	1	2	4	392000.0	19.01	4.496471	5

32014 rows × 9 columns

```
X = df.drop('price',axis=1)
y = df['price']


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_state=True)


from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, r2_score

# Initialize the model with default params
dt_reg = DecisionTreeRegressor(random_state=42)

# Fit on training data
dt_reg.fit(X_train, y_train)

# Predict on train and test
y_train_pred = dt_reg.predict(X_train)
y_test_pred = dt_reg.predict(X_test)

# Evaluate
print('Train R2:', r2_score(y_train, y_train_pred))
print('Test R2:', r2_score(y_test, y_test_pred))
print('Train MSE:', mean_squared_error(y_train, y_train_pred))
print('Test MSE:', mean_squared_error(y_test, y_test_pred))

# Cross-validation (5-fold) on the entire dataset X, y
cv_scores = cross_val_score(dt_reg, X, y, cv=5, scoring='r2')
print('Cross-Validation R2 Scores:', cv_scores)
print('Mean CV R2 Score:', cv_scores.mean())
```

```
Train R2: 0.9643323020743481
Test R2: 0.8561125124893902
Train MSE: 17986573269.737762
Test MSE: 65420433593.05073
Cross-Validation R2 Scores: [0.76931854 0.85220707 0.83311843 0.85433936 0.70270359]
Mean CV R2 Score: 0.8023373965991863
```

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, r2_score

# Initialize base model
dt_reg = DecisionTreeRegressor(random_state=42)

# Define hyperparameter grid
param_grid = {
    'criterion': ['squared_error', 'friedman_mse', 'absolute_error'],
    'max_depth': range(1, 16),
    'min_samples_split': [2, 5, 10],
```

```
    'min_samples_leaf': [1, 2, 4]
}

# Setup GridSearch with 5-fold CV
grid_search = GridSearchCV(dt_reg, param_grid, cv=5, scoring='r2', n_jobs=-1)

# Fit on training data
grid_search.fit(X_train, y_train)

# Best estimator from grid search
best_dt_reg = grid_search.best_estimator_
print('Best Parameters:', grid_search.best_params_)

# Predict using the best model
y_train_pred = best_dt_reg.predict(X_train)
y_test_pred = best_dt_reg.predict(X_test)

# Evaluate
print('Train R2:', r2_score(y_train, y_train_pred))
print('Test R2:', r2_score(y_test, y_test_pred))
print('Train MSE:', mean_squared_error(y_train, y_train_pred))
print('Test MSE:', mean_squared_error(y_test, y_test_pred))

# Cross-validation on entire dataset with best model
cv_scores = cross_val_score(best_dt_reg, X, y, cv=5, scoring='r2')
print('Cross-Validation R2 Scores:', cv_scores)
print('Mean CV R2 Score:', cv_scores.mean())
```

```
Best Parameters: {'criterion': 'squared_error', 'max_depth': 15, 'min_samples_leaf': 4, 'min_samples_split': 2}
Train R2: 0.9202757895790773
Test R2: 0.8572464126246068
Train MSE: 40203473605.0801
Test MSE: 64904890234.96953
Cross-Validation R2 Scores: [0.75314394 0.84872503 0.86234613 0.85558998 0.64100464]
Mean CV R2 Score: 0.792161941708647
```

Start coding or generate with AI.