# PROGRAM 6:

Implementation of Decision tree using sklearn and its parameter tuning

DESCRIPTION:

Decision Trees are a type of supervised learning algorithm that can be used for both classification and regression tasks. They work by recursively partitioning the data into smaller subsets based on the features of the data. In this example, we will implement a Decision Tree using Scikit-Learn and perform parameter tuning to optimize its

performance.

## Program:

## import pandas as pd

## from sklearn.model\_selection import train\_test\_split

## from sklearn.tree import DecisionTreeRegressor

## from sklearn import metrics

## from sklearn.model\_selection import GridSearchCV

## from sklearn.datasets import load\_boston

## boston = load\_boston()

## df = pd.DataFrame(boston.data, columns=boston.feature\_names)

## df['PRICE'] = boston.target

## X = df.drop('PRICE', axis=1)

## y = df['PRICE']

## X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

## model = DecisionTreeRegressor()

## param\_grid = {

## 'max\_depth': [3, 5, 10, 15],

## 'min\_samples\_split': [2, 5, 10],

## 'min\_samples\_leaf': [1, 5, 10]

## }

## grid\_search = GridSearchCV(model, param\_grid, cv=5, scoring='neg\_mean\_squared\_error')

## grid\_search.fit(X\_train, y\_train)

## best\_model = grid\_search.best\_estimator\_

## best\_params = grid\_search.best\_params\_

## y\_pred = best\_model.predict(X\_test)

## print("Best Parameters: ", best\_params)

## print("Mean Absolute Error: ", metrics.mean\_absolute\_error(y\_test, y\_pred))

## print("Root Mean Squared Error: ", np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred)))

## OUTPUT:

Best Parameters: {'max\_depth': 15, 'min\_samples\_leaf': 10, 'min\_samples\_split': 2}

Mean Absolute Error: 2.8655005740126853

Root Mean Squared Error: 4.024850317712781