How to implement Decision Tree Regression using sklearn

Decision Tree Regression:

Classification and regression trees are machine-learning methods for constructing prediction models from data. The models are obtained by recursively partitioning the data space and fitting a simple prediction model within each partition. As a result, the partitioning can be represented graphically as a decision tree.

Regression trees are for dependent variables that take continuous or ordered discrete values, with prediction error typically measured by the squared difference between the observed and predicted values.

Implementation:

We have created a random dataset using numpy and applied Random Forest Regression

Import required library and dataset

Include libraries like numpy, sklearn.tree, matplotlib,

```
import numpy as np
from sklearn.tree import DecisionTreeRegressor
import matplotlib.pyplot as plt

# Create a random dataset
rng = np.random.RandomState(1)
X = np.sort(5 * rng.rand(80, 1), axis=0)
y = np.sin(X).ravel()
y[::5] += 3 * (0.5 - rng.rand(16))
```

Apply Decision Tree Regressor:

We have used criterion for splitting is mean square error(mse) and max depth is 3,and trained the model on X(independent variable) and y(dependent variable).

```
# Fit regression model
tree = DecisionTreeRegressor(criterion='mse',max_depth=3)
tree.fit(X, y)
```

Prediction:

Applied model for prediction on X_test data variable

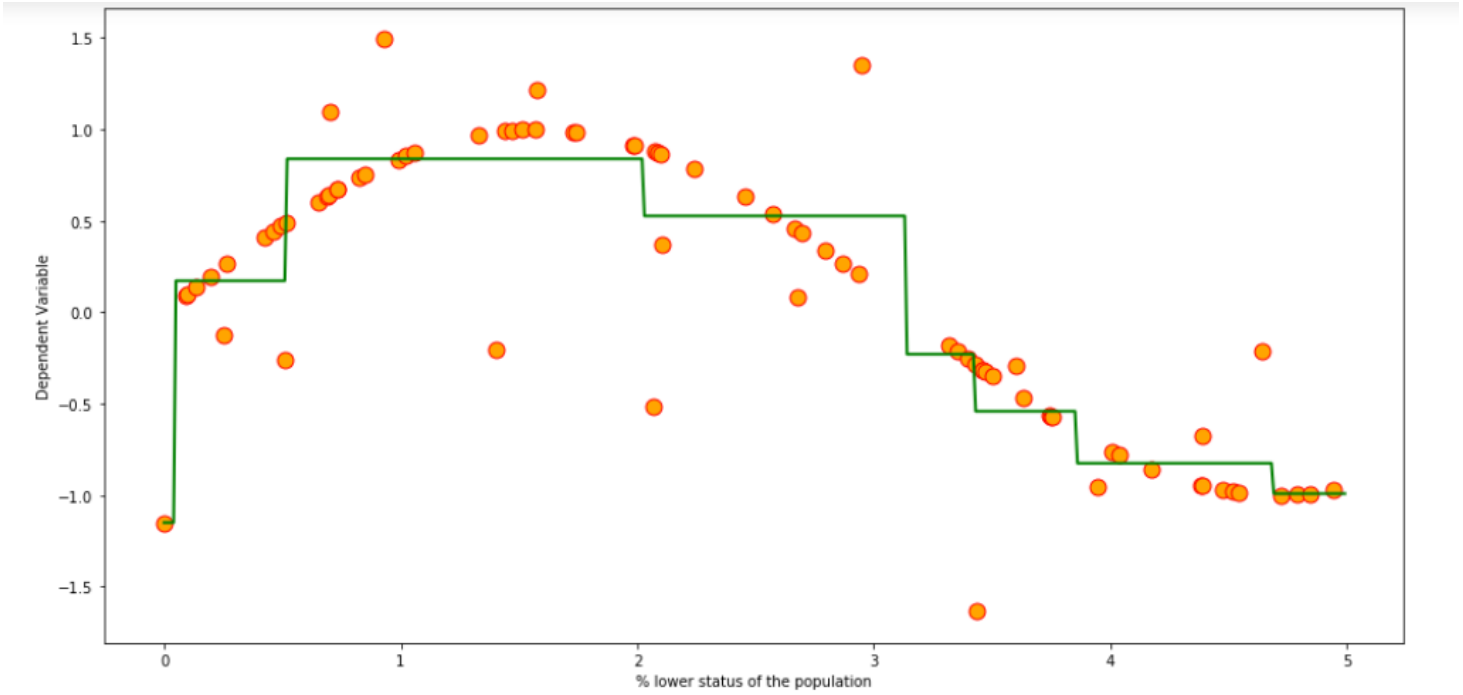
```
# Predict
X_test = np.arange(0.0, 5.0, 0.01)[:, np.newaxis]
y_pred = tree.predict(X_test)
```

Plotting Result:

Use scatterplot function for plotting

```
# Plot the results
plt.figure(figsize=(16, 8))
plt.scatter(X, y, c='orange',edgecolor='red', s=120)
plt.plot(X_test, y_pred,color='green',lw=2)
plt.xlabel('% lower status of the population')
plt.ylabel('Dependent Variable')
plt.show()
```

Output:



Now we can see in the resulting graph, the decision tree of the applied model of depth 3 captures the general trend in the random dataset.

Draw Regression Tree:

The regression tree is plotted and shown in the tree structure below, visualized using http://www.webgraphviz.com/ by copying the data from the 'reg_tree.dot' file.

```
from sklearn.tree import export_graphviz
# export the decision tree to a reg_tree.dot file
export_graphviz(tree, out_file ='reg_tree.dot')
```