Chapter 6: DecisionTreeRegressor

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
# %cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter6_Decision_
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

In [3]:
```
iris = pd.read_excel("Iris.xls")
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepallength    150 non-null float64
sepalwidth     150 non-null float64
petallength    150 non-null float64
petalwidth     150 non-null float64
iris           150 non-null object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

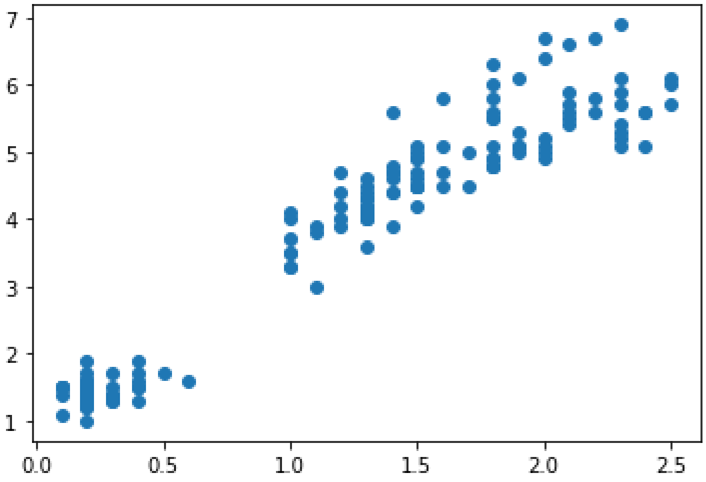
In [4]:
```
iris.head()
```

Out[4]:

	sepallength	sepalwidth	petallength	petalwidth	iris
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [5]:
```
petalwidth = iris[['petalwidth']] # input
pentallength = iris['petallength'] # output
```

In [6]:
```
plt.scatter(petalwidth, pentallength)
plt.show()
```



In [7]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(petalwidth,
                                                    pentallength,
                                                    test_size=0.20,
                                                    random_state = 42)
```

In [8]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [9]:
```
# Create decision tree regressor object
dtr = DecisionTreeRegressor()
# Train model
model = dtr.fit(X_train, y_train)
```

In [10]:
```
y_pred = model.predict(X_test)
y_pred
```

Out[10]:
```
array([4.2       , 1.35      , 5.625     , 4.69      , 4.61428571,
       1.58      , 4.2       , 5.625     , 4.69      , 4.2       ,
       5.45      , 1.4       , 1.436     , 1.4       , 1.35      ,
       5.13333333, 6.7       , 3.4       , 4.2       , 6.7       ,
       1.436     , 5.33      , 1.58      , 5.82      , 5.45      ,
       5.625     , 5.33      , 5.625     , 1.35      , 1.436     ])
```

In [11]:
```
# Train's Score
print("The Train R^2 score is: ", model.score(X_train,y_train))
```

```
The Train R^2 score is:  0.964853727798769
```

In [12]:
```
# Test's Score
print("The Test R^2 score is: ", model.score(X_test,y_test))
```

```
The Test R^2 score is:  0.9264886792633416
```

In [13]:
```
# Both training and testing have high R^2 scores => OK
```

In [14]:
```
from sklearn import metrics
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
```

```
Mean Squared Error: 0.24092518578987163
```

In [15]:
```
# MSE is low.
```

In [16]:
```
df = pd.DataFrame({'Actual': pd.DataFrame(y_test.values)[0].values,
                   'Prediction': pd.DataFrame(y_pred)[0].values})
df.head()
```

Out[16]:

	Actual	Prediction
0	4.7	4.200000
1	1.7	1.350000
2	6.9	5.625000
3	4.5	4.690000
4	4.8	4.614286

In [17]:
```
x_now = [[0.25]]
y_now = model.predict(x_now)
y_now
```

Out[17]:
```
array([1.436])
```