Chapter 7: RandomForestRegressor

In [1]:

```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
```

In [2]:

```
# %cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter7_Random_Fo
```

In [3]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

In [4]:

```
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 [5]:

```
iris.head(3)
```

Out[5]:

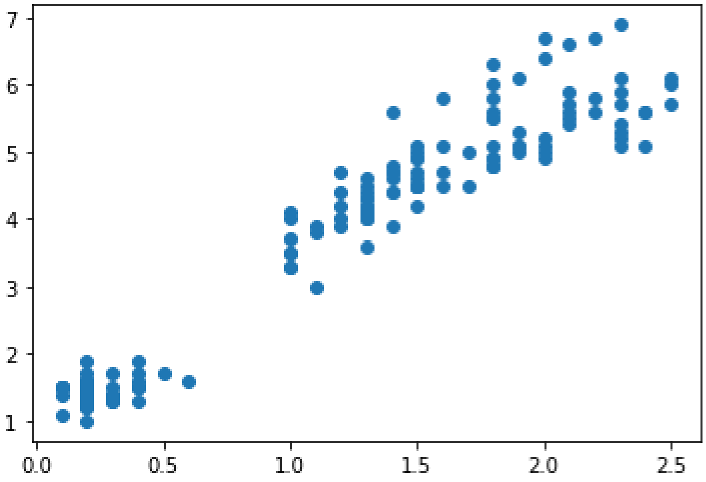
	sepallength	sepalwidth	petallength	petalwidth	iris
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa

In [6]:

```
petalwidth = iris[['petalwidth']]
pentallength = iris['petallength']
```

In [7]:

```
plt.scatter(petalwidth, pentallength)
plt.show()
```



In [8]:

```
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 [9]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [10]:

```
# Create random forest resgressor object
rrf = RandomForestRegressor(n_estimators=100)
# Train model
model = rrf.fit(X_train, y_train)
```

In [11]:

```
y_pred = model.predict(X_test)
y_pred
```

Out[11]:

```
array([4.19193269, 1.36013355, 5.64014206, 4.69483378, 4.61298634,
       1.58186227, 4.19755928, 5.64014206, 4.69483378, 4.19193269,
       5.51277778, 1.38911429, 1.43615429, 1.38911429, 1.36013355,
       5.09608532, 6.25373254, 3.39263153, 4.19755928, 6.25373254,
       1.43615429, 5.35125697, 1.58186227, 5.82300292, 5.51277778,
       5.64014206, 5.35125697, 5.64014206, 1.36013355, 1.43615429])
```

In [12]:
```
from sklearn.metrics import mean_squared_error
```

In [13]:
```
# Score
print("The R^2 train score is: ", model.score(X_train, y_train))
print("The R^2 test score is: ", model.score(X_test,y_test))
# tinh them mse
print("MSE (in train): ", mean_squared_error(y_train, model.predict(X_train)))
print("MSE (in test): ", mean_squared_error(y_test, y_pred))
```

```
The R^2 train score is:  0.9641699712422555
The R^2 test score is:  0.9417096100439368
MSE (in test):  0.10897083685754066
MSE (in test):  0.1910402763710021
```

In [14]:
```
# R^2 của cả train và test đều cao => model phù hợp với dữ liệu
# MSE thấp
```

In [15]:
```
df = pd.DataFrame({'Actual': pd.DataFrame(y_test.values)[0].values,
                   'Prediction': pd.DataFrame(y_pred)[0].values})
df.head()
```

Out[15]:

	Actual	Prediction
0	4.7	4.191933
1	1.7	1.360134
2	6.9	5.640142
3	4.5	4.694834
4	4.8	4.612986

In [16]:
```
x_now = [[0.25]]
y_now = model.predict(x_now)
y_now
```

Out[16]:
```
array([1.43615429])
```