**Bài 6: Supervised Learning: Regression - Lập trình AI bằng Python**

Hồi quy là một trong những công cụ quan trọng nhất. Mình sẽ không sai khi nói rằng hành trình của học máy bắt đầu từ hồi quy. Nó có thể được định nghĩa là kỹ thuật tham số cho phép chúng ta đưa ra quyết định dựa trên dữ liệu hoặc nói cách khác cho phép chúng ta đưa ra dự đoán dựa trên dữ liệu bằng cách tìm hiểu mối quan hệ giữa các biến đầu vào và đầu ra. Ở đây, các biến đầu ra phụ thuộc vào các biến đầu vào, là các số thực có giá trị liên tục. Trong hồi quy, mối quan hệ giữa các biến đầu vào và đầu ra rất quan trọng và nó giúp chúng ta hiểu được giá trị của biến đầu ra thay đổi như thế nào với sự thay đổi của biến đầu vào. Hồi quy thường được sử dụng để dự đoán giá cả, kinh tế, các biến thể, v.v.

**Xây dựng Regressors bằng Python :**

Trong phần này, chúng ta sẽ học cách xây dựng bộ hồi quy đơn cũng như đa biến.

**a. Single Variable Regressor :**

Import các thư viện cần thiết :

**import** numpy as np

from sklearn **import** linear\_model

**import** sklearn.metrics as sm

**import** matplotlib.pyplot as plt

Load dữ liệu bằng cách sử dụng hàm **np.loadtxt :**

input\_data = np.loadtxt(input, delimiter=',')

X, y = input\_data[:, :-1], input\_data[:, -1]

Bước tiếp theo sẽ là đào tạo mô hình. Tiến hành đưa bộ dữ liệu vào traning và testing

training\_samples = **int**(0.6 \* len(X))

testing\_samples = len(X) - num\_training

X\_train, y\_train = X[:training\_samples], y[:training\_samples]

X\_test, y\_test = X[training\_samples:], y[training\_samples:]

Tiếp theo ta sẽ tạo linear regressor object

reg\_linear = linear\_model.LinearRegression()

Đào tạo object bằng tập traning

**reg\_linear**.fit(X\_train, y\_train)

Sau đó ta cần dự đoán với dữ liệu testing :

y\_test\_pred = reg\_linear.predict(X\_test)

Vẽ và hiển thị dữ liệu :

**plt**.scatter(X\_test, y\_test, color = 'red')

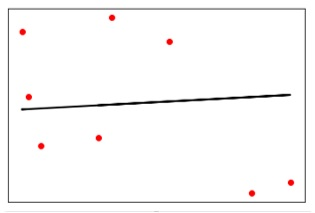
**plt**.plot(X\_test, y\_test\_pred, color = 'black', linewidth = 2)

**plt**.xticks(())

**plt**.yticks(())

**plt**.show()

Output :



Bây giờ , ta tiến hành tính toán các tham số về mô hình chúng ta đã xây dựng như sau :

print("Performance of Linear regressor:")

print("Mean absolute error =", round(sm.mean\_absolute\_error(y\_test, y\_test\_pred), 2))

print("Mean squared error =", round(sm.mean\_squared\_error(y\_test, y\_test\_pred), 2))

print("Median absolute error =", round(sm.median\_absolute\_error(y\_test, y\_test\_pred), 2))

print("Explain variance score =", round(sm.explained\_variance\_score(y\_test, y\_test\_pred),

2))

print("R2 score =", round(sm.r2\_score(y\_test, y\_test\_pred), 2))

Output :

Mean absolute error = 1.78

Mean squared error = 3.89

Median absolute error = 2.01

Explain variance score = -0.09

R2 score = -0.09

Ở đoạn code trên, mình đã sử dụng lượng nhỏ dữ liệu, nếu bạn muốn dùng các dataset lớn thì ta nên sử dụng sklearn.dataset để import

2,4.82.9,4.72.5,53.2,5.56,57.6,43.2,0.92.9,1.92.4,

3.50.5,3.41,40.9,5.91.2,2.583.2,5.65.1,1.54.5,

1.22.3,6.32.1,2.8

**b. Multivariable Regressor**

Đầu tiên, ta tiến hành import một vài thư viện cần thiết :

**import** numpy as np

from sklearn **import** linear\_model

**import** sklearn.metrics as sm

**import** matplotlib.pyplot as plt

from sklearn.preprocessing **import** PolynomialFeatures

Sau đó, cần cung cấp dữ liệu đầu vào và mình đã  lưu trữ dữ liệu với tên là linear.txt

input = 'D:/ProgramData/Mul\_linear.txt'

Load dữ liệu với hàm **np.loadtxt :**

input\_data = np.loadtxt(input, delimiter=',')

X, y = input\_data[:, :-1], input\_data[:, -1]

Bước tiếp theo cần phải train model, ta sẽ lấy mẫu traning và testing

training\_samples = **int**(0.6 \* len(X))

testing\_samples = len(X) - num\_training

X\_train, y\_train = X[:training\_samples], y[:training\_samples]

X\_test, y\_test = X[training\_samples:], y[training\_samples:]

Tạo linear regressor object :

reg\_linear\_mul = linear\_model.LinearRegression()

training object với traning samples

**reg\_linear\_mul**.fit(X\_train, y\_train)

Cuồi cùng ta sẽ dự đoán tới tập dữ liệu test

y\_test\_pred = reg\_linear\_mul.predict(X\_test)

print("Performance of Linear regressor:")

print("Mean absolute error =", round(sm.mean\_absolute\_error(y\_test, y\_test\_pred), 2))

print("Mean squared error =", round(sm.mean\_squared\_error(y\_test, y\_test\_pred), 2))

print("Median absolute error =", round(sm.median\_absolute\_error(y\_test, y\_test\_pred), 2))

print("Explain variance score =", round(sm.explained\_variance\_score(y\_test, y\_test\_pred), 2))

print("R2 score =", round(sm.r2\_score(y\_test, y\_test\_pred), 2))

Output :

Mean absolute error = 0.6

Mean squared error = 0.65

Median absolute error = 0.41

Explain variance score = 0.34

R2 score = 0.33

Bây giờ, chúng ta sẽ tạo một đa thức bậc 10 và huấn luyện bộ hồi quy

polynomial = PolynomialFeatures(degree = 10)

X\_train\_transformed = polynomial.fit\_transform(X\_train)

datapoint = [[2.23, 1.35, 1.12]]

poly\_datapoint = polynomial.fit\_transform(datapoint)

poly\_linear\_model = linear\_model.LinearRegression()

poly\_linear\_model.fit(X\_train\_transformed, y\_train)

print("\nLinear regression:\n", reg\_linear\_mul.predict(datapoint))

print("\nPolynomial regression:\n", poly\_linear\_model.predict(poly\_datapoint))

**Output**

Linear regression :

[2.40170462]

Polynomial regression −

[1.8697225]

Trong đoạn code trên, mình đã sử dụng dữ liệu nhỏ này. Nếu bạn muốn có một tập dữ liệu lớn, bạn có thể sử dụng sklearn.dataset để nhập một tập dữ liệu lớn hơn.

2,4.8,1.2,3.22.9,4.7,1.5,3.62.5,5,2.8,23.2,5.5,3.5,2.16,5,

2,3.27.6,4,1.2,3.23.2,0.9,2.3,1.42.9,1.9,2.3,1.22.4,3.5,

2.8,3.60.5,3.4,1.8,2.91,4,3,2.50.9,5.9,5.6,0.81.2,2.58,

3.45,1.233.2,5.6,2,3.25.1,1.5,1.2,1.34.5,1.2,4.1,2.32.3,

6.3,2.5,3.22.1,2.8,1.2,3.6