Thực hành Máy Học Ứng dụng
Buổi 4

Hồi quy tuyến tính và phương pháp tập hợp mô hình

Mục tiêu:

- Ứng dụng giải thuật hồi quy tuyến tính và phương pháp tập hợp mô hình
- Kiểm thử và đánh giá

A. HỒI QUY TUYẾN TÍNH

1. **Ví dụ dự đoán giá nhà (bài tập ví dụ trên lớp)**
   Cho tập dữ liệu gồm 3 phần tử như bảng bên dưới,

X	1	2	4
Y	2	3	6

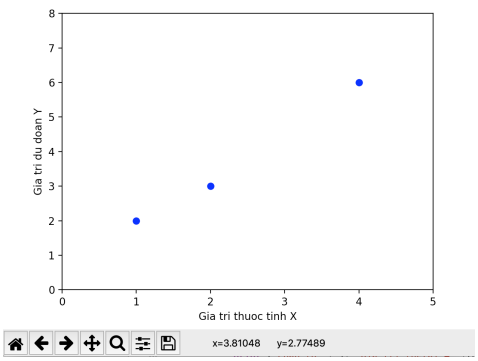
Anh/chị hãy thực hiện các yêu cầu sau:

- Biểu diễn tập dữ liệu lên mặt phẳng toạ độ Oxy
- Tìm hàm hồi quy h(x) với giá trị khởi tạo theta0=0, theta1=1, tốc độ học: 0.2, số bước lặp: 2
- Vẽ đường hồi quy lên mặt phẳng toạ độ
- Dự đoán giá trị y cho các phần tử có x có giá trị lần lượt là 0, 3, 5

Hướng dẫn

a. Biểu diễn dữ liệu lên mặt phẳng toạ đô

```
import numpy as np
import matplotlib.pyplot as plt

X = np.array([1,2,4])
Y = np.array([2,3,6])

plt.axis([0,5,0,8])
plt.plot(X,Y,"ro",color="blue")
plt.xlabel("Gia tri thuoc tinh X")
plt.ylabel("Gia tri du doan Y")
plt.show()
```



b. Tìm hàm hồi quy với theta0 = 0, theta1 =1, tốc độ học = 0.2, số lần lặp là 1

for i=1 to m, {

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$$

}

```
def LR1(X,Y,eta,lanlap, theta0,theta1):
    m = len(X) # so luong phan tu
    for k in range(0,lanlap):
        print("Lan lap: ", k)
        for i in range(0,m):
            h_i= theta0 + theta1*X[i]
            #theta0
            theta0 = theta0 + eta*(Y[i]-h_i)*1
            print ("Phan tu ", i, "y=", Y[i], "h=",h_i,"gia tri theta0 = ",round(theta0,3))
            #theta1
            theta1 = theta1 + eta*(Y[i]-h_i)*X[i]
            print ("Phan tu ", i, "gia tri theta1 = ",round(theta1,3))
    return [round(theta0,3),round(theta1,3)]

theta = LR1(X,Y,0.2,1,0,1)
theta
```

Kết quả cho **1 lần lặp** : `theta = LR1(X,Y,0.2,1,0,1)`

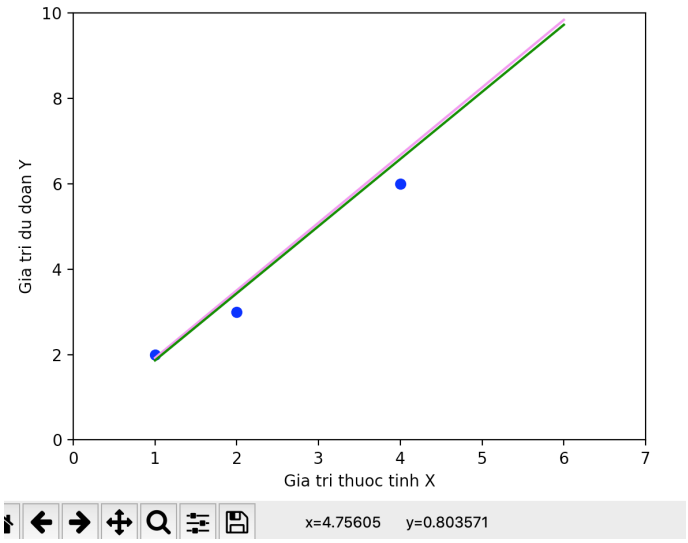
```
>>> theta = LR1(X,Y,0.2,1,0,1)
Lan lap:  0
Phan tu  0 y= 2 h= 1 gia tri theta0 =  0.2
Phan tu  0 gia tri theta1 =  1.2
Phan tu  1 y= 3 h= 2.6 gia tri theta0 =  0.28
Phan tu  1 gia tri theta1 =  1.36
Phan tu  2 y= 6 h= 5.72 gia tri theta0 =  0.336
Phan tu  2 gia tri theta1 =  1.584
>>> theta
[0.336, 1.584]
```

Kết quả cho 2 lần lặp : `theta2 = LR1(X,Y,0.2,2,0,1)`

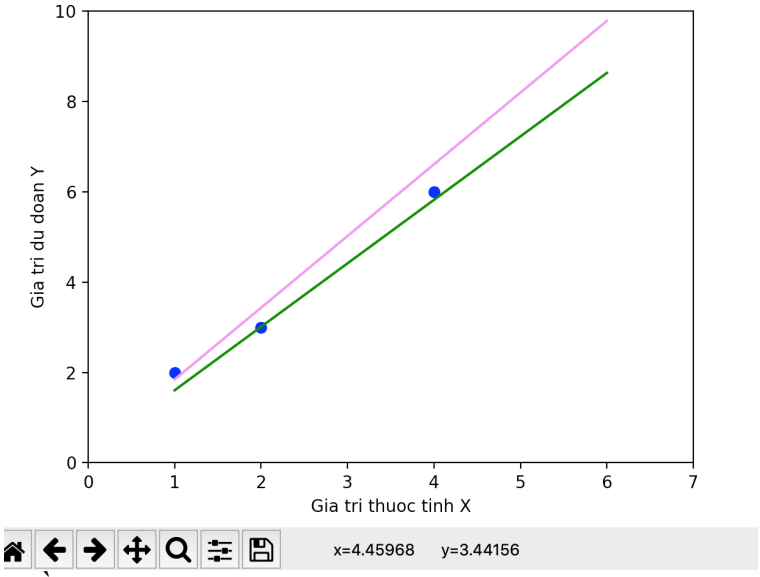
```
>>> theta
[0.29, 1.572]
```

c. Vẽ đường hồi quy

```
theta = LR1(X,Y,0.2,1,0,1) # theta 1 bước
X1= np.array([1,6])
Y1= theta[0] + theta[1]*X1

theta2 = LR1(X,Y,0.2,2,0,1) # theta 2 bước lap
X2= np.array([1,6])
Y2= theta2[0] + theta2[1]*X2

plt.axis([0,7,0,10])
plt.plot(X,Y,"ro",color="blue")

plt.plot(X1,Y1,color="violet") # duong hoi quy lan lap 1
plt.plot(X2,Y2,color="green")  # duong hoi quy lan lap 2

plt.xlabel("Gia tri thuoc tinh X")
plt.ylabel("Gia tri du doan Y")
plt.show()
```



d. Thay đổi tốc độ học bằng 0.1, anh/chị vẽ đường hồi quy màu hồng cho 1 lần lặp và màu xanh lá cây cho 2 lần lặp và so sánh với kết quả ở bước câu c.



e. Dự báo cho phần tử mới tới
Dự báo giá trị y cho 3 phần tử sau: x=0, x=3, x=5

```
# Du bao
y1 = theta[0] + theta[1]*0
y2 = theta[0] + theta[1]*3
y3 = theta[0] + theta[1]*5
```

Hoặc sử dụng vòng lặp for

```
# Du bao
XX = [0,3,5]
for i in range(0,3):
    YY = theta[0] + theta[1]*XX[i]
    print (round(YY,3))
```

Kết quả dự đoán 2 lần lặp:

```
theta = LR1(X,Y,0.2,2,0,1)
   X=0 => y=0.29
   X=3 => y=5.006
   X=5 => y=8.15
```

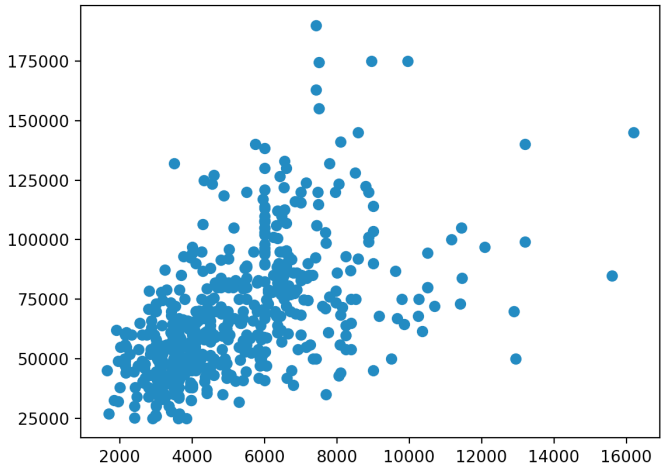
2. Sử dụng thư viện scikit-learn của Python để tìm các giá trị theta

- Đọc dữ liệu từ file csv

```
# đọc dữ liệu từ file Housing.csv
import pandas as pd
dt = pd.read_csv("Housing_2019.csv", index_col=0)
dt.iloc[2:4,]
X= dt.iloc[:,[1,2,3,4,10]]
X.iloc[1:5,]
y = dt.price
```

- Hiển thị dữ liệu tương quan giữa diện tích (lotsize) và giá nhà (price)

```
plt.scatter(dt.lotsize, dt.price)
plt.show()
```



- Sử dụng sklearn để tìm các giá trị theta, sinh viên thực hành đoạn code bên dưới và trả lời các câu hỏi sau:

- Có bao nhiêu thuộc tính, đó là những thuộc tính nào đã được sử dụng để dự đoán giá nhà?
- Xác định số lượng theta và các giá trị của nó.
- Dữ liệu được sử dụng để huấn luyện mô hình?
- Dữ liệu được sử dụng để dự báo mô hình?
- Độ chính xác được đánh giá bằng chỉ số gì và giá trị của nó?

```
# đọc dữ liệu từ file Housing.csv
import pandas as pd
dt = pd.read_csv("Housing_2019.csv", index_col=0)
dt.iloc[2:4,]
X= dt.iloc[:,[1,2,3,4,10]]
X.iloc[1:5,]
y = dt.price

plt.scatter(dt.lotsize, dt.price)
plt.show()

# huấn luyện mô hình
import sklearn
from sklearn import linear_model
lm = linear_model.LinearRegression()
lm.fit(X[1:520],y[1:520])

print (round(lm.intercept_,3))
print (lm.coef_)
# dự báo giá nhà cho 20 phần tử cuối cùng trong tập dữ liệu
y = dt.price
y_test = y[-20:]
X_test = X[-20:]
y_pred = lm.predict(X_test)
# so sánh giá trị thực tế và giá trị dự báo
y_pred
y_test
```

```
# danh gia
from sklearn.metrics import mean_squared_error
import numpy as np
err = mean_squared_error(y_test, y_pred)
print (round(err,3))
np.sqrt(err)
```

B. PHƯƠNG PHÁP TẬP HỢP MÔ HÌNH

Áp dụng giải thuật Bagging trên tập dữ liệu dự đoán giá nhà

Dự đoán giá nhà (bài toán hồi quy) với cây quyết định

```
import pandas as pd
from sklearn.metrics import mean_squared_error
import numpy as np

dt = pd.read_csv("Housing_2019.csv", index_col=0)
X= dt.iloc[:,[1,2,4,10]]
Y = dt.price

import sklearn
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=1.0/3, random_state=100)
len(X_train) #1199


from sklearn.ensemble import BaggingRegressor
from sklearn.tree import DecisionTreeRegressor
tree = DecisionTreeRegressor(random_state = 0)

bagging_regtree = BaggingRegressor(base_estimator=tree, n_estimators=10, random_state=42)
bagging_regtree.fit(X_train, y_train)
y_pred = bagging_regtree.predict(X_test)
err = mean_squared_error(y_test, y_pred)
err
np.sqrt(err)
```

Dự đoán giá nhà (bài toán hồi quy) với hồi quy tuyến tính

```
import pandas as pd
from sklearn.metrics import mean_squared_error
import numpy as np

dt = pd.read_csv("Housing_2019.csv", index_col=0)
X= dt.iloc[:,[1,2,4,10]]
Y = dt.price

import sklearn
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=1.0/3, random_state=100)
len(X_train) #1199

from sklearn.ensemble import BaggingRegressor
from sklearn import linear_model
lm = linear_model.LinearRegression()
bagging_reg = BaggingRegressor(base_estimator=lm, n_estimators=10, random_state=42)
bagging_reg.fit(X_train, y_train)
y_pred = bagging_reg.predict(X_test)
err = mean_squared_error(y_test, y_pred)
err #362862254.1975257
np.sqrt(err) #19048.943650437042
```