Tut 8 - Sử dụng các models Linear Regression của thư viện Scikit-Learn

- Tác giả: Nguyễn Hữu Vũ
- Nhóm nghiên cứu về AI Đại học Bách Khoa Thành Phố Hồ Chí Minh

1 Nhắc lại khái niệm

Cho hàm $y = a_1x_1 + a_2x_2 + ... + a_nx_n + b$, trong đó

- y là biến phụ thuộc (dependent variable)
- x_1, x_2, ..., x_n là các biến độc lập (independent variables).
- $a_1, .., a_2$ là các hệ số (coefficients)
- b: intercept

General problem to solve: Cho các bộ số $(y^1, x_1^1, x_2^1, ..., x_n^1)$, $(y^2, x_1^2, x_2^2, ..., x_n^2)$, ..., $(y^m, x_1^m, x_2^m, ..., x_n^m)$, hãy sử dụng model Linear Regression của thư viện sklearn để tìm ra bộ hệ số (coefficients) $(a_1, ..., a_n)$ và intercept b sao cho $y^i = a_1x_1^i + a_2x_2^i + ... + a_nx_n^i + b$ với mọi $1 \leq i \leq m$.

2 Sử dụng model Linear Regression của thư viện sklearn

Problem: Cho 3 bộ số $(0, 0, 0); (1, 1, 1); (2, 2, 2)$ có dạng (y, x_1, x_2), hãy sử dụng model Linear Regression của thư viện sklearn để tìm bộ hệ số (coefficients) (a_1, a_2) và intercept b sao cho $y = a_1x_1 + a_2x_2 + b$.

1. Chạy Python

- Tham khảo Tut1

2. Cài đặt thư viện Scikit-Learn

   - Tham khảo Tut1

3. Import thư viện cần thiết

```
1 >>> from sklearn import linear_model
2 >>> import numpy as np
```

   Trong đó:

   - Dòng 1: import thư viện sklearn để sử dụng model Linear Regression
   - Dòng 2: import thư viện numpy để khởi tạo dữ liệu array

4. Tạo dữ liệu đầu vào cho các models (dùng chung cho Linear Regression, Ridge, Lasso)

```
1 >>> X = np.array([[0, 0], [1, 1], [2, 2]])
2 >>> Y = np.array([0, 1, 2])
```

   Trong đó:

   - Dòng 1: mảng 2 chiều khởi tạo các bộ giá trị (x_1, x_2).
   - Dòng 2: mảng một chiều khởi tạo các giá trị y tương ứng (xem lại Problem ở đầu section này)

5. Tạo model Linear Regression

```
1 >>> reg = linear_model.LinearRegression()
```

6. Train model Linear Regression với bộ dữ liệu đầu vào X, Y để tính bộ hệ số (a_1, a_2) và intercept b

```
1 >>> reg.fit (X, Y)
```

7. Xuất ra bộ hệ số (a_1, a_2)

```
1 >>> reg.coef_
```

   Output: **array([0.5, 0.5])**

8. Xuất ra giá trị intercept b

```
>>> reg.intercept_
```

Output: **2.220446049250313e-16**

Conclusion: Như vậy, với 3 bộ số $(0,0,0);(1,1,1);(2,2,2)$ có dạng (y,x_1,x_2), mối liên hệ giữa biến phụ thuộc y và các biến độc lập x_1,x_2 là $y=0.5x_1+0.5x_2+0$.

3 Sử dụng các models Regularized Linear Regression của thư viện sklearn

3.1 Sử dụng Lasso Regression (L1)

1. Tạo model Lasso Regression

```
>>> lassoReg = linear_model.Lasso(alpha = 0.1)
```

2. Train model Lasso Regression với bộ dữ liệu đầu vào X,Y để tính bộ hệ số (a_1,a_2) và intercept b

```
>>> lassoReg.fit (X, Y)
```

3. Xuất ra bộ hệ số (a_1,a_2)

```
>>> lassoReg.coef_
```

Output: **array([0.85, 0.])**

4. Xuất ra giá trị intercept b

```
>>> lassoReg.intercept_
```

Output: **0.15000000000000002**

Conclusion: Như vậy, chúng ta có thể thấy nếu sử dụng model Lasso Regression, thuộc tính nào không quan trọng sẽ bị bỏ đi ,trong trường hợp này là thuộc tính x_2 sẽ bị bỏ đi với hệ số $a_2=0$.

3.2 Sử dụng Ridge Regression (L2)

1. Tạo model Ridge Regression

```
>>> ridgeReg = linear_model.Ridge (alpha = .5)
```

2. Train model Ridge Regression với bộ dữ liệu đầu vào X, Y để tính bộ hệ số (a_1, a_2) và intercept b

```
>>> ridgeReg.fit (X, Y)
```

3. Xuất ra bộ hệ số (a_1, a_2)

```
>>> ridgeReg.coef_
```

   Output: **array([0.44444444, 0.44444444])**

4. Xuất ra giá trị intercept b

```
>>> ridgeReg.intercept_
```

   Output: **0.11111111111111116**

4 Sử dụng dữ liệu từ dataset để train và test

Trong phần này, chúng ta sẽ sử dụng dữ liệu từ file "Dataset-Tut8-student_scores.csv" để train và test .

4.1 Bối cảnh

File "Dataset-Tut8-student_scores.csv" gồm 2 cột: cột "Hours" là thời gian mà một sinh viên dành để học một môn học, cột "Scores" là số điểm (tính theo phần trăm) mà sinh viên đó đạt được. Trong phần này, chúng ta sẽ tìm quan hệ tuyến tính giữa thời gian một sinh viên học một môn học với điểm số mà sinh viên đó đạt được.

4.2 Training

1. Copy Dataset-Tut8-student_scores.csv vào C:\Datasets

2. Import các thư viện cần thiết

```
1 >>> import pandas as pd
2 >>> import numpy as np
3 >>> import matplotlib.pyplot as plt
```

Trong đó:

- Dòng 1: import thư viện pandas để xử lí, đọc dữ liệu từ file .csv.
- Dòng 3: import thư viện matplotlib.pyplot để visualize dữ liệu

3. Sử dụng thư viện pandas để import dữ liệu từ file .csv

```
1 >>> dataset = pd.read_csv('C:\Datasets\Dataset-Tut8-student_scores
    .csv')
```

4. Có thể xem thông tin về dataset vừa import vào

```
1 >>> dataset.shape
```

Output: **(25, 2)** $\Longrightarrow$ 25 hàng và 2 cột

```
1 >>> dataset.head()
```

Output:

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

5. Chia dữ liệu thành 2 nhóm "attributes" và "labels": "attributes" là các biến độc lập $(x_1, ..., x_n)$, "labels" là biến phụ thuộc y. Trong dataset mẫu này, chúng ta có 2 cột là "Hours" và "Scores". Chúng ta muốn dự đoán "Scores" từ "Hours", do đó chúng ta sẽ chọn cột "Hours" làm "attributes", cột "Scores" làm "labels"

```
1 >>> X = dataset[['Hours']].values
2 >>> y = dataset['Scores'].values
```

Chú ý: $X = [(x_1^1, ..., x_n^1), ..., (x_1^m, ..., x_n^m)]$. Do đó X phải là một mảng 2 chiều. Vì vậy, khi lấy giá trị cột "Hours" gắn vào giá trị X, chúng ta phải dùng 2 dấu ngoặc vuông "[[ ]]" để kết quả trả về là một mảng 2 chiều.

6. Kiểm tra giá trị của X, y

```
>>> print('X = ', X)
>>> print('y= ', y)
```

Output:

```
X =  [[2.5]
 [5.1]
 [3.2]
 [8.5]
 [3.5]
 [1.5]
 [9.2]
 [5.5]
 [8.3]
 [2.7]
 [7.7]
 [5.9]
 [4.5]
 [3.3]
 [1.1]
 [8.9]
 [2.5]
 [1.9]
 [6.1]
 [7.4]
 [2.7]
 [4.8]
 [3.8]
 [6.9]
 [7.8]]
y=  [21 47 27 75 30 20 88 60 81 25 85 62 41 42 17 95 30 24 67 69 30 54 35 76
 86]
```

Chú ý: trong kết quả xuất ra phải đảm bảo X là một mảng 2 chiều, y là mảng một chiều.

7. Có thể sử dụng thư viện pyplot để vẽ đồ thị thể hiện sự phân bố dữ liệu một cách trực quan

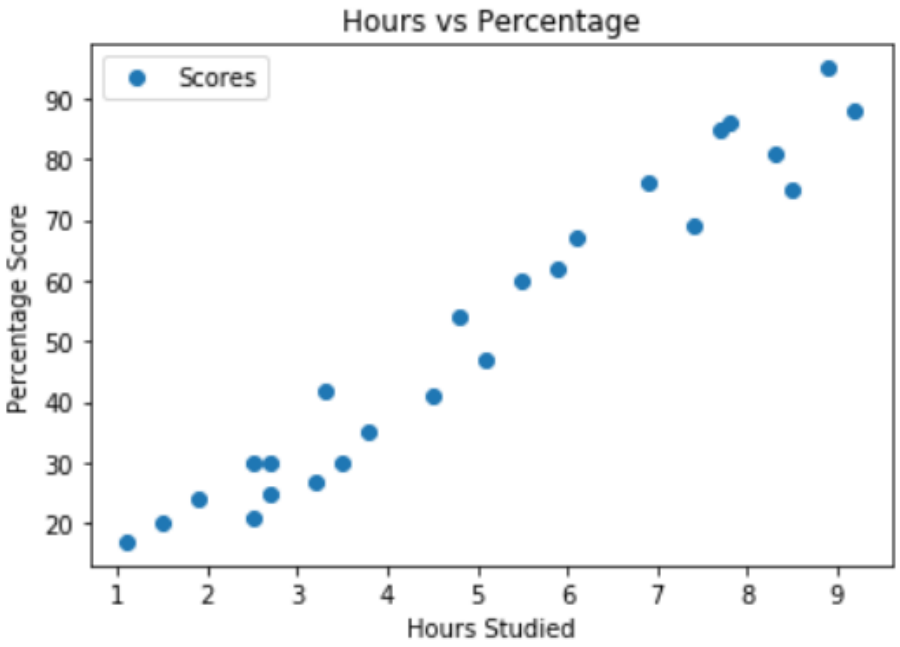
```
>>> dataset.plot(x='Hours', y='Scores', style='o')
>>> plt.title('Hours vs Percentage')
>>> plt.xlabel('Hours Studied')
>>> plt.ylabel('Percentage Score')
>>> plt.show()
```

Trong đó:

- Dòng 1: dùng dữ liệu của cột "Hours" trong tập dataset làm dữ liệu cho trục x, dữ liệu của cột "Scores" trong tập dataset làm dữ liệu cho trục y

- Dòng 2,3,4 : Gắn title cho đồ thị, nhãn cho trục x, nhãn cho trục y

Output:



8. Tiến hành chia tập dữ liệu ban đầu thành 2 tập: tập train và tập test. Tập train dùng cho training, tập test sẽ được dùng để test xem model có chính xác không

```
1 >>> from sklearn.model_selection import train_test_split
2 >>>  X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=0)
```

Trong đó:

- Dòng 2: $test\_size = 0.2$ để lấy 80% dữ liệu ban đầu dùng cho training, 20% còn lại dùng để test.

9. Sử dụng model Linear Regression để training

```
1 >>> from sklearn.linear_model import LinearRegression
2 >>> regressor = LinearRegression()
3 >>> regressor.fit(X_train, y_train)
```

Trong đó:

- Dòng 3: Sử dụng bộ dữ liệu dùng để train (X_train, y_train) để train .

10. Có thể xem các hệ số coefficients và intercept sau khi train xong

```
>>> print('coefs = ', regressor.coef_)
>>> print('intercept = ', regressor.intercept_)
```

4.3 Dự đoán dữ liệu (Making Predictions)

Ở section trước chúng ta đã sử dụng model Linear Regression để train. Trong mục này chúng ta sẽ sử dụng kết quản train được để dự đoán dữ liệu.

11. Dự đoán dữ liệu

```
>>> y_pred = regressor.predict(X_test)
```

Trong đó: kết quả trả về y_pred sẽ là một mảng chứa giá trị dự đoán tương ứng của các giá trị trong tập X_test

12. So sánh dữ liệu thật y_test với dữ liệu dự đoán y_pred của tập X_test

```
>>> df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
>>> df
```

Output:

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801
4	62	60.491033

Conclusion: Có thể thấy độ chính xác của model chúng ta chưa cao, nhưng về cơ bản thì giá trị dự đoán đã khá gần với giá trị thực tế. Các bạn có thể chỉnh sửa dataset, tăng giảm tỉ lệ dữ liệu dùng để train để tăng độ chính xác cho model.

5 References

- `http://scikit-learn.org/stable/modules/linear_model.html`
- `https://stackabuse.com/linear-regression-in-python-with-scikit-learn/`
- `http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`