TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
ĐHQG - TPHCM

Khoa công nghệ thông tin
Môn: Toán ứng dụng và thống kê cho Công nghệ thông tin

BÁO CÁO ĐỒ ÁN 3

LINEAR REGRESSION

Giáo viên hướng dẫn: Nguyễn Trọng Hiến
Nguyễn Văn Quang Huy
Nguyễn Đình Thúc
Võ Nam Thục Đoan

Sinh viên thực hiện: Phùng Nghĩa Phúc – 20127284

TP.Hồ Chí Minh, ngày 19 tháng 07 năm 2022

MỤC LỤC

GIỚI THIỆU

1. Sinh viên
   - **Tên**: Phùng Nghĩa Phúc
   - **MSV**: 20127284
   - **Lớp**: 20CLC07
   - **Môn**: Toán ứng dụng và thống kê cho Công nghệ thông tin
2. Đề tài
   - Đồ án 3: **Linear regression**
   - Nội dung: File "**wine.csv**" là cơ sở dữ liệu đánh giá chất lượng của 1200 chai rượu vang theo thang điểm 1 - 10 dựa trên 11 tính chất khác nhau.
3. Đề bài:

   Xây dựng mô hình đánh giá chất lượng rượu sử dụng phương pháp hồi quy tuyến tính.

   a. Sử dụng toàn bộ 11 đặc trưng đề bài cung cấp,

   y=theta_1x_1+theta_2x_2+...+theta_11x_11

   b. Sử dụng duy nhất 1 đặc trưng cho kết quả tốt nhất. (Gợi ý: Phương pháp Cross Validation)

   y=theta_ix_i (dùng mô hình lần lượt cho từng đặc trưng).

   c. Xây dựng một mô hình của riêng bạn cho kết quả tốt nhất.
4. Các thư viện sử dụng
   - **import pandas as pd:** được sử dụng để đọc data của file wine.csv
   - **import numpy as np:** được sử dụng để thực thi các phép toán liên quan đến ma trận
5. Lưu ý
   - Code được thi trên jupyter notebook.
   - Khi thực hiện code, chuyển đổi local của file “wine.csv” tại vị trí:

```
df = pd.read_csv('C:/Users/Decane/Desktop/toan thong ke/project 3- linear regression/wine.csv', sep=';')
```

   - Ngôn ngữ được sử dụng: Python 3

CHAPTER 1: MÔ TẢ DỮ LIỆU ĐẦU VÀO

```
import pandas as pd
df = pd.read_csv('C:/Users/Decane/Desktop/toan thong ke/project 3- linear regression/wine.csv', sep=';')
df
```

- Sử dụng cấu trúc được thực thi ở trên để thực hiện đọc file “wine.csv”
- Sau khi thực thi thì dữ liệu sẽ được mô tả như sau:

[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1194	7.0	0.745	0.12	1.8	0.114	15.0	64	0.99588	3.22	0.59	9.5	6
1195	6.2	0.430	0.22	1.8	0.078	21.0	56	0.99633	3.52	0.60	9.5	6
1196	7.9	0.580	0.23	2.3	0.076	23.0	94	0.99686	3.21	0.58	9.5	6
1197	7.7	0.570	0.21	1.5	0.069	4.0	9	0.99458	3.16	0.54	9.8	6
1198	7.7	0.260	0.26	2.0	0.052	19.0	77	0.99510	3.15	0.79	10.9	6

1199 rows × 12 columns

- Dựa vào hình trên có thể thấy được:
  - Dữ liệu có 1200 dòng và 12 cột
  - Phân loại dữ liệu:
    - X = 11 cột đầu tiên

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol

    - Y = cột cuối cùng (quality)

quality

CHAPTER 2: CẤU TRÚC CỦA PHƯƠNG TRÌNH HỒI QUY TUYẾN TÍNH

1. Cách tính nghiệm của phương trình hồi quy tuyến tính là:

$$\theta = (X^T X)^{-1}(X^T Y)$$

Trong đó: $(X^T X)^{-1}X^T$ là ma trận nghịch đảo của X

2. Cách tính độ lỗi:

$r = ||A\theta - B||$

Trong đó: r là độ lỗi

A là ma trận x nhưng được thêm cột chứa số 1 ở phía trước

B là ma trận của Y

θ là nghiệm của giải phương trình hồi quy tuyến tính

CHAPTER 3: Câu a. Sử dụng toàn bộ 11 đặc trưng đề bài cung cấp, y=theta_1x_1+theta_2x_2+...+theta_11x_11

1. Mô tả hàm
   - Sử dụng thư viện numpy để hỗ trợ việc tính toán

```
def getAB(a, b):
    return np.hstack((np.ones((a.shape[0], 1)), a)), b
```

   - Hàm này được dùng để tính A B
     - Với A là ma trận x nhưng được thêm cột chứa số 1 ở phía trước
     - Với B là ma trận của Y

```
x = np.array(df.iloc[:, :-1])
y = np.array(df.iloc[:, -1]).reshape(-1, 1)
```

   - 2 dòng này được dùng để lấy dữ liệu từ data file
     - x lấy dữ liệu từ đầu cho tới cột ancohol
     - y lấy dữ liệu của cột quality

```
theta = (np.linalg.pinv(a)@b)
```

   - Ở dòng này như tên của biến được dùng để tính theta θ

```
r_score = np.round((np.linalg.norm(a@theta - b)), 3)
```

   - Dòng này được dùng để tính độ lỗi bằng việc sử dụng thư viện numpy hỗ trợ và làm tròn đến chữ số thứ 3

2. Toàn bộ code

```
import numpy as np
```

```
def getAB(a, b):
    return np.hstack((np.ones((a.shape[0], 1)), a)), b
```

```
x = np.array(df.iloc[:, :-1])
y = np.array(df.iloc[:, -1]).reshape(-1, 1)

a, b = getAB(x, y)
theta = (np.linalg.pinv(a)@b)

ans = np.round(theta,1)
ans
```

```
array([[ 42.9],
       [  0. ],
       [ -1.1],
       [ -0.3],
       [  0. ],
       [ -1.6],
       [  0. ],
       [ -0. ],
       [-39.5],
       [ -0.2],
       [  0.8],
       [  0.3]])
```

```
r_score = np.round((np.linalg.norm(a@theta - b)), 3)

r_score
```

```
22.095
```

3. Kết quả của câu hỏi

```
'([[ 42.9],
   [  0. ],
   [ -1.1],
   [ -0.3],
   [  0. ],
   [ -1.6],
   [  0. ],
   [ -0. ],
   [-39.5],
   [ -0.2],
   [  0.8],
   [  0.3]])
```

Nghiệm của phương trình hồi quy tuyến tính là: θ =
Độ lỗi là 22.095

CHAPTER 4: Câu b. Sử dụng duy nhất 1 đặc trưng cho kết quả tốt nhất. (Gợi ý: Phương pháp Cross Validation) y=theta_ix_i (dùng mô hình lần lượt cho từng đặc trưng).

1. Mô tả hàm
   - Xây dựng 2 hàm chính thực hiện:
     - r_score(x, y): để thực hiện tính độ lỗi
     - cross_validation(df): thực hiện tính toán và tìm ra đặc trưng cho kết quả tốt nhất, ở đây kết quả tốt nhất là độ lỗi nhỏ nhất
2. Ý tưởng
   - Thực hiện tách các đặc trưng ra và thực hiện hồi quy tuyến tính. Để thực hiện được việc này cần xây dựng mô hình theo hướng huấn luyện và kiểm định
   - Chia bộ dữ liệu của mỗi đặc trưng ra làm 5 phần. Lần lượt cho từng phần đi kiểm định, và phần còn lại làm dữ liệu huấn luyện. Sau khi có đủ 5 độ lỗi -> thực hiện tính trung bình của của bộ độ lỗi
3. Toàn bộ code

```
def r_score(x, y):
    result = 0

    new_X = np.array_split(x, min(len(x), 5))
    new_Y = np.array_split(y, min(len(y), 5))

    for i in range(5):
        x_test = new_X[i]
        y_test = new_Y[i]

        x_train = np.array([])
        y_train = np.array([])

        for j in range(5):
            if i != j:
                x_train = np.append(x_train, new_X[j])
                y_train = np.append(y_train, new_Y[j])

        x_train = x_train.reshape(-1, 1)
        y_train = y_train.reshape(-1, 1)

        new_A, new_B = getAB(x_train, y_train)
        old_A, old_B = getAB(x_test, y_test)

        theta = np.linalg.pinv(new_A) @ new_B

        r_score = np.linalg.norm(old_A @ theta - old_B)
        result += r_score

    return result/5
```

```
def cross_validation(df):
    result = []
    y = np.array(df.iloc[:, -1]).reshape(-1, 1)

    for i in df.columns[:-1]:
        x = np.array(df[i]).reshape(-1, 1)
        r = r_score(x, y)
        result.append([r, i])
    return result[-1]
cross_validation(df)
```

```
[10.896303487242815, 'alcohol']
```

```
r_score, name = cross_validation(df)

print(f'Đặc trưng cho kết quả tốt nhất là {name} và r = {np.round(r_score,3)}')
```

```
Đặc trưng cho kết quả tốt nhất là alcohol và r = 10.896
```

4. Kết quả của câu hỏi
   - Đặc trưng cho kết quả tốt nhất là alcohol với độ lỗi là 10.896

CHAPTER 5: Câu c. Xây dựng một mô hình của riêng bạn cho kết quả tốt nhất.

1. Mô tả hàm
   - Xây dựng 2 hàm chính thực hiện:
     - r_score(x, y): để thực hiện tính độ lỗi
     - calculator(df): Thực hiện tính toán và tìm ra kết quả tốt nhất
2. Ý tưởng
   - Dựa vào ý tưởng của câu b thực hiện dựa trên việc tính độ lỗi trước và sau đó tìm ra kết quả tốt nhất
   - Nhìn vào dữ liệu đầu vào có 1200 dữ liệu, để có kết quả tốt nhất thì thực hiện chia dữ liệu ra làm 2 phần:
     - 1000 dữ liệu đầu tiên là dữ liệu huấn luyện
     - 200 dữ liệu còn lại là dữ liệu kiểm định
3. Toàn bộ code

```
def r_score(x, y):
    x_test = x[1000:]
    y_test = y[1000:]

    x_train = x[:1000]
    y_train = y[:1000]

    new_A, new_B = getAB(x_train, y_train)
    old_A, old_B = getAB(x_test, y_test)

    theta = np.linalg.pinv(new_A) @ new_B

    result = np.linalg.norm(old_A @ theta - old_B)
    return result
```

```
def calculator(df):
    result = []
    y = np.array(df.iloc[:, -1]).reshape(-1, 1)

    for i in df.columns[:-1]:
        x = np.array(df[i]).reshape(-1, 1)
        r = r_score(x, y)
        result.append([r, i])
    return result[-1]
calculator(df)
```

```
[9.862217648422789, 'alcohol']
```

```
r_score, name = calculator(df)

print(f'Đặc trưng cho kết quả tốt nhất là {name} và r = {np.round(r_score,3)}')
```

```
Đặc trưng cho kết quả tốt nhất là alcohol và r = 9.862
```

4. Kết quả của câu hỏi
   - Đây chưa phải là kết quả vì mô hình này chưa được kiểm định rõ ràng, chỉ dựa trên ý tưởng của cross_validation()
   - Kết quả cho kết quả tốt nhất sẽ là alcohol với độ lỗi là 9.862