Bài thực hành 2. MÔ HÌNH HỒI QUY (Phần 1)

1. MÔ HÌNH HỒI QUY TUYẾN TÍNH CƠ BẢN

- Khởi tạo dữ liệu: khởi tạo 100 điểm dữ liệu gồm X và y.

$$y = 4 + 3*X + \varepsilon$$

  - *Gợi ý:* Sử dụng hàm np.random.rand(100, 1). Hàm này sẽ tạo ra 1 vector với 100 phần tử ngẫu nhiên.
  - *Code mẫu:*

```
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)
```

- Thêm giá trị bias_term vào dữ liệu X ban đầu. Mặc định bias_term là 1.
  - *Gợi ý:*
    - Sử dụng thư viện np.c_[v1,v2] để nối 2 vector v1 và v2 lại với nhau. X ban đầu có 100 điểm dữ liệu, do đó vector bias_term sẽ là 1 vector có 100 phần tử có giá trị 1.
    - Để tạo ra 1 vector 100 phần tử có giá trị là 1, ta dùng lệnh: np.ones((100, 1))
  - *Code mẫu:*

```
X_b = np.c_[np.ones((100, 1)), X]
```

Câu 1. Vẽ biểu đồ cho dữ liệu (X,y) vừa tạo ở trên.

Gợi ý: Sử dụng hàm plot với 3 tham số: X, y và "b.".

Câu 2. Cho biết chiều (shape) của X_b và y.

Gợi ý: Dùng lệnh X_b.shape.

Câu 3. Hãy viết code Python tìm ra tham số tối ưu cho mô hình hồi quy tuyến tính. Tìm tham số tối ưu cho mô hình bằng phương trình sau:

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

Kết quả lưu vào biến theta_best.

Gợi ý:

- Sử dụng X_b thay cho X (X_b là dữ liệu X ban đầu đã thêm bias_term vào).
  - X^T là ma trận chuyển vị. Sử dụng lệnh X.T trong Python (điều kiện là X đã ép kiểu về *numpy_array*).

- $\mathbf{X^T X}$ là phép tích vô hướng (dot product) giữa 2 vector. Sử dụng lệnh np.dot() trong thư viện numpy.
- $(\mathbf{X^T X})^{-1}$ là phép tính nghịch đảo của ma trận. Sử dụng lệnh np.linalg.inv(A) để tính nghịch đảo của ma trận A. Thay A = $\mathbf{X^T X}$.

– Tạo ra dữ liệu mới X_new bằng đoạn lệnh sau (dùng để kiểm tra):

```
X_new = np.array([[0], [2]])
X_new_b = np.c_[np.ones((2, 1)), X_new]
```

Câu 4. Dùng tham số tối ưu của hồi quy tuyến tính vừa tìm được ở Câu 3, dự đoán kết quả cho **X_new_b**. Giá trị dự đoán lưu vào biến **y_pred**.

Gợi ý:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\theta}}$$

– Sử dụng tích vô hướng (dot product) để tìm ra $\hat{\mathbf{y}}$ (kết quả dự đoán).

– Tính tích vô hướng bằng lệnh np.dot().

Câu 5. Vẽ biểu đồ thể hiện kết quả dự đoán trên dữ liệu.

Gợi ý: Vẽ 2 biểu đồ bằng lệnh plot trong thư viện Matplotlib:

– *Biểu đồ 1:* Vẽ biểu đồ cho dữ liệu huấn luyện ban đầu X, y. Tham số gồm: X, y, "b." (chấm màu xanh dương - blue).

– *Biểu đồ 2:* Vẽ biểu đồ cho mô hình hồi quy tuyến tính dưới dạng đường thẳng. Tham số gồm: X_new_b, y_pred, "r-" (dấu gạch màu đỏ - red).

2. BÀI TẬP

Bài tập 1. Thực hiện lại các câu hỏi trong mục 1 để xây dựng mô hình hồi quy tuyến tính.

Bài tập 2. Thực hiện xây dựng mô hình hồi quy đa thức (Polynomial Regression) cho dữ liệu đa thức như sau:

- Dữ liệu huấn luyện:

```
m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.5 * X**2 + X + 2 + np.random.randn(m, 1)
```

- Dữ liệu dự đoán:

```
X_new = np.linspace(-3, 3, 100).reshape(100, 1)
```

- *Gợi ý:*
  - Sử dụng lớp ***PolynomialFeatures*** trong sklearn để chuyển giá trị trong X và X_new về thành dạng bậc 2 (dữ liệu phi tuyến) theo đoạn code mẫu như sau:

```
from sklearn.preprocessing import PolynomialFeatures
poly_features = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly_features.fit_transform(X)
```

  - Dữ liệu huấn luyện sẽ thành: **(X_poly, y).**
  - Các bước còn lại thực hiện tương tự Bài tập 1.

Bài tập 3. Xây dựng mô hình hồi quy Ridge. Thực hiện tương tự như Bài 1.

Gợi ý:

- Tham số tối ưu cho mô hình Ridge Regression được tính theo công thức sau:

$$\theta = (X^T X + \alpha I)^{-1} X^T y$$

- Chọn tham số α là 0.02. Để tạo ma trận đơn vị, ta sử dụng lệnh (với M là kích thước của ma trận đơn vị I):

```
np.identity(M)
```

Bài tập 4. So sánh tham số tối ưu tìm được bằng tay và bằng thư viện sklearn trong Bài 1 và Bài 3. So sánh kết quả thu được giữa thực hiện bằng tay và bằng thư viện.

Gợi ý: Tham khảo thư viện như sau:

- *Hồi quy tuyến tính*: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html.
- *Hồi quy Ridge*: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html.