AI VIET NAM – Warmup Course 2025

Simple Linear Regression - Exercise

Quang-Vinh Dinh

Ngày 20 tháng 1 năm 2025

Linear equation

$$\hat{y} = wx + b$$

where $\hat{y}$ is a predicted value,

w and b are parameters

and x is input feature

Error (loss) computation

Idea: compare predicted values $\hat{y}$ and label values y

Squared loss

$$L(\hat{y}, y) = (\hat{y} - y)^2$$

Find better w and b

Use gradient descent to minimize the loss function

Compute derivate for each parameter

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

Update parameters

$$w = w - \eta \frac{\partial L}{\partial w} \qquad b = b - \eta \frac{\partial L}{\partial b}$$

η is learning rate

Giới thiệu về tập data: Cho trước dữ liệu về quảng cáo có 200 samples (200 dòng) được lưu trong file advertising.csv, gồm 4 thông tin TV, Radio, Newspaper, và Sales (hình 1).

TV	Radio	Newspaper	Sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	12
151.5	41.3	58.5	16.5
180.8	10.8	58.4	17.9
8.7	48.9	75	7.2
57.5	32.8	23.5	11.8
120.2	19.6	11.6	13.2
8.6	2.1	1	4.8
199.8	2.6	21.2	15.6

Hình 1: Một vài sample data từ dữ liệu quảng cáo advertising.csv

Bài tập 1 (kỹ thuật đọc và xử lý dữ liệu từ file .csv): Cho trước file dữ liệu advertising.csv, hãy hoàn

thành function **prepare_data(file_name_dataset)** trả về dữ liệu đã được tổ chức (X cho input và y cho output).

```

# dataset
import numpy as np
import matplotlib.pyplot as plt
import random

def get_column(data, index):

    #your code here ***************************

    return result

def prepare_data(file_name_dataset):
  data = np.genfromtxt(file_name_dataset, delimiter=',', skip_header=1).tolist()
  N = len(data)

  # get tv (index=0)
  tv_data = get_column(data, 0)

  # get radio (index=1)
  radio_data = get_column(data, 1)

  # get newspaper (index=2)
  newspaper_data = get_column(data, 2)

  # get sales (index=3)
  sales_data = get_column(data, 3)

  # building X input  and y output for training
  X = [tv_data, radio_data, newspaper_data]
  y = sales_data
  return X,y
```

```
Question 1:
X,y = prepare_data('advertising.csv')
list = [sum(X[0][:5]), sum(X[1][:5]), sum(X[2][:5]), sum(y[:5])]
print(list)
Select one of the following answers:

a) [624.1, 175.1, 300.5, 78.9]

b) [625, 175.1, 75.0, 7.2]

c) [626, 175.1, 75.0, 7.2]

d) [627.8, 175.1, 75.0, 7.2]
```

Bài tập 2 (kỹ thuật huấn luyện data dùng 1 sample): Sử dụng kết quả dữ liệu đầu vào X, và dữ liệu đầu ra y từ bài 1, để phát triển chương trình dự đoán thông tin sales (y) từ X bằng cách dùng giải thuật linear regression with one sample-training với loss được tính bằng công thức Squared Error $L(\hat{y}, y) = (\hat{y} - y)^2$. Sơ đồ hoạt động của giải thuật được mô tả ở hình 2. Nhiệm vụ của bạn là hoàn thành function **implement_linear_regression(X_data, y_data, epoch_max, lr)** và trả về 4 tham số w1, w2, w3, b và lịch sử tính loss như bên dưới.

```
def implement_linear_regression(X_data, y_data, epoch_max = 50,lr = 1e-5):
  losses = []
```

1) Pick a sample (x_1, x_2, x_3, y) from training data

2) Compute the output $\hat{y}$

$$\hat{y} = w_1 * TV + w_2 * R + w_3 * N + b$$

$$\hat{y} = w_1 * x_1 + w_2 * x_2 + w_3 * x_3 + b$$

3) Compute loss

$$L = (\hat{y} - y)^2$$

4) Compute derivative

$$\frac{\partial L}{\partial w_1} = 2x_1(\hat{y} - y) \qquad \frac{\partial L}{\partial w_3} = 2x_3(\hat{y} - y)$$

$$\frac{\partial L}{\partial w_2} = 2x_2(\hat{y} - y) \qquad \frac{\partial L}{\partial b} = 2(\hat{y} - y)$$

5) Update parameters

$$w_1 = w_1 - \eta \frac{\partial L}{\partial w_1} \qquad w_3 = w_3 - \eta \frac{\partial L}{\partial w_3}$$

$$w_2 = w_2 - \eta \frac{\partial L}{\partial w_2} \qquad b = b - \eta \frac{\partial L}{\partial b}$$

Hình 2: Các bước để thực hiện train linear regression model

```
3
4   w1, w2, w3, b = initialize_params()
5
6   N = len(y_data)
7   for epoch in range(epoch_max):
8       for i in range(N):
9           # get a sample
10          x1 = X_data[0][i]
11          x2 = X_data[1][i]
12          x3 = X_data[2][i]
13
14          y  = y_data[i]
15
16          # compute output
17          y_hat = predict(x1, x2, x3, w1, w2, w3, b)
18
19          # compute loss
20          loss = compute_loss(y, y_hat)
21
22          # compute gradient w1, w2, w3, b
23          dl_dw1 = compute_gradient_wi(x1, y, y_hat)
24          dl_dw2 = compute_gradient_wi(x2, y, y_hat)
25          dl_dw3 = compute_gradient_wi(x3, y, y_hat)
26          dl_db  = compute_gradient_b(y, y_hat)
27
28          # update parameters
29          w1 = update_weight_wi(w1, dl_dw1, lr)
```

```
            w2 = update_weight_wi(w2, dl_dw2, lr)
            w3 = update_weight_wi(w3, dl_dw3, lr)
            b  = update_weight_b(b, dl_db, lr)

            # logging
            losses.append(loss)
    return (w1,w2,w3,b, losses)
```

Để hoàn thành function implement_linear_regression(X_data, y_data, epoch_max, lr) bạn cần phải hoàn thành cái sub-function sau đây:

2.1 Hoàn thành function **initialize_params()** để khởi tạo ngẫu nhiên giá trị ban đầu cho w1, w2, w3 theo gaussian random.gauss(mu=0.0, sigma=0.01) và b = 0. Ở bước này các bạn có thể dùng hàm sau để khởi tạo bốn tham số trên.

```
def initialize_params():
    w1 = random.gauss(mu=0.0, sigma=0.01)
    w2 = random.gauss(mu=0.0, sigma=0.01)
    w3 = random.gauss(mu=0.0, sigma=0.01)
    b  = 0
    return w1, w2, w3, b
```

Vì để thống nhất việc đánh giá kết quả cho toàn bài tập trong module này, các bạn được yêu cầu khởi tạo cố định wi, cũng như b như sau (trong thực tế bạn nên sử dụng hàm random phía trên nhé):

```
    def initialize_params():
        w1, w2, w3, b = (0.016992259082509283, 0.0070783670518262355,
                         -0.002307860847821344, 0)
        return w1, w2, w3, b
```

2.2 Hoàn thành function **predict(x1, x2, x3, w1, w2, w3, b)** để trả về kết quả dự đoán $\hat{y}$ tương ứng

```
def predict(x1, x2, x3, w1, w2, w3, b):

    # your code here ****************************

    return result
```

```
Question 2:
y_hat = predict(x1=1, x2=1, x3=1, w1=0, w2=0.5, w3=0, b=0.5)
print(y_hat)
Select one of the following answers:

a) 1.0

b) 2.0

c) 3.0

d) 4.0
```

2.3 Hoàn thành function **compute_loss(y_hat, y)** để tính loss giữa kết quả dự đoán y_hat và giá trị thực y, sử dụng Squared Error

```
def compute_loss(y_hat, y):

    # your code here ****************************

    return loss
```

```
Question 3:
l = compute_loss(y_hat=1, y=0.5)
print(l)
Select one of the following answers:

a) 0.25

b) 0.26

c) 0.27

d) 0.28
```

2.4 Hoàn thành function **compute_gradient_wi(xi, y, y_hat))** để tính đạo hàm của hàm loss $L = (\hat{y} - y)^2$ theo wi và function **compute_gradient_b(y, y_hat)** để tính đạo hàm của hàm loss $L = (\hat{y} - y)^2$ theo b.

```python
# compute gradient
def compute_gradient_wi(xi, y, y_hat):

    # your code here ****************************

    return dl_dwi

def compute_gradient_b(y, y_hat):

    # your code here ****************************

    return dl_db
```

```
Question 4:
# loss function
g_wi = compute_gradient_wi(xi=1.0, y=1.0, y_hat=0.5)
print(g_wi)
Select one of the following answers:

a) -1.0

b) -2.0

c) 1.0

d) 2.0
```

```
Question 5:
g_b = compute_gradient_b(y=2.0, y_hat=0.5)
print(g_b)
Select one of the following answers:

a) -1.0

b) -3.0

c) 1.0

d) 2.0
```

2.5 Hoàn thành function **update_weight_wi(wi, dl_dwi, lr)** để cập nhật wi sau khi tính đạo hàm hàm loss L theo wi, và function **update_weight_b(b, dl_db, lr)** để update bias (b) sau khi tính đạo hàm hàm loss L theo b.

```

# update weights
def update_weight_wi(wi, dl_dwi, lr):

  # your code here ****************************

    return wi

def update_weight_b(b, dl_db, lr):

  # your code here ****************************

    return b
```

```
Question 6:
after_wi = update_weight_wi(wi=1.0, dl_dwi=-0.5, lr = 1e-5)
print(after_wi)
Select one of the following answers:

a) 1.000005

b) -3.0

c) 1.0

d) 2.0
```

```
Question 7:
after_b = update_weight_b(b=0.5, dl_db=-1.0, lr = 1e-5)
print(after_b)
Select one of the following answers:

a) 0.50001

b) -3.0

c) 1.0

d) 2.0
```

2.6 Thực hiện huấn luyện data bằng cách gọi hàm **implement_linear_regression(X, y)** và vẽ đồ thị kết quả cho 100 giá trị loss đầu tiên (loss cho 100 lần cập nhật đầu tiên) như bên hình 3.

```
(w1,w2,w3,b, losses) = implement_linear_regression(X,y)
plt.plot(losses[:100])
plt.xlabel("#iteration")
plt.ylabel("Loss")
plt.show()
```

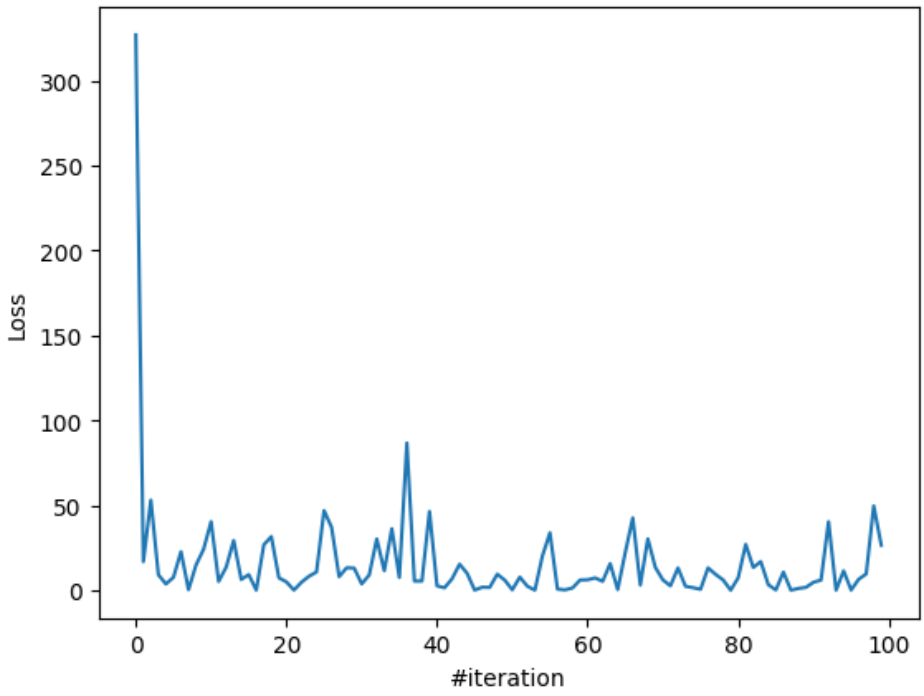
```
Question 8:
X,y = prepare_data('advertising.csv')
(w1,w2,w3,b, losses) = implement_linear_regression(X,y)
print(w1, w2, w3)
Select one of the following answers:

a) w1 = 0.074, w2 = 0.15, w3 = 0.017

b) w1 = 1.074, w2 = 1.15, w3 = 1.17

c) w1 = 2.074, w2 = 0.15, w3 = 2.17

d) w1 = 3.074, w2 = 0.15, w3 = 3.17
```



Hình 3: Kết quả loss $\mathrm{L} = (\hat{y} - y)^2$ sau 100 iteration cập nhật.

```
Question 9:

# given new data
tv = 19.2
radio = 35.9
newspaper = 51.3

X,y = prepare_data('advertising.csv')
(w1,w2,w3,b, losses) = implement_linear_regression(X, y, epoch_max=50, lr=1e-5)
sales = predict(tv, radio, newspaper, w1, w2, w3, b)
print(f'predicted sales is {sales}')

Select one of the following answers:

a) predicted sale is 6.18

b) predicted sale is 8.18

c) predicted sale is 7.18

d) predicted sale is 9.18
```

Phụ lục

1. **Hint:** File code gợi ý có thể được tải tại đây.
2. **Solution:** File code cài đặt hoàn chỉnh và phần trả lời nội dung trắc nghiệm có thể được tải tại đây (Lưu ý: Sáng thứ 7, ad mới copy các nội dung bài giải nêu trên vào đường dẫn).
3. **Rubric:**

Simple Linear Regression - Rubric		
Câu	**Kiến Thức**	**Đánh Giá**
1	Cách thức thao tác và truy xuất dữ liệu từ file .csv - Cách tổ chức dữ liêu đầu vào và đầu ra theo yêu cầu của giải thuật linear regression	- Biết cách thao tác, truy xuất, và tổ chức dư liệu đầu vào, đầu ra từ file .csv
2	- Cách thức tổ chức huấn luyện mô hình linear regress bằng các sử dụng one-sample với SGD - Cách xây dựng các thành phần cơ bản của một giải thuật machine learning: tính loss, tính gradient, cập nhật trọng số và bias. - Sử dụng các hàm loss khác nhau để đánh giá kết quả đầu ra: MSE và MAE	- Biết cách tổ chức huấn luyện mô hình linear regress bằng các sử dụng one-sample với gradient descent - Biết được cách xây dựng các thành phần cơ bản của một giải thuật machine learning gồm tính loss, tính gradient, cập nhật trọng số và bias - Hiểu được cách thức hoạt động của hàm loss MSE và MAE trên one-sample training
3	- Cách thức tổ chức dữ liệu đầu vào như là một danh sách các features (x1,x2,x3,...,xn) để hướng tới xử lý song song tăng tốc độ giải thuật - Cách thức hiện thực lại các bước trong giải thuật linear regresson sử dụng List trong python	- Vận dung cấu trúc dữ liệu List để tổ chức dữ liệu đầu vào như là một danh sách các features (x1,x2,x3,...,xn) để hướng tới xử lý song song tăng tốc độ giải thuật

- Hết -