REVIEW NUMPY (Vectorized Implemetation for Linear Regression)

Review-Numpy array

Game ôn tập kiến thức về numpy

Numpy

Outline

- Numpy Array Operations
- Broadcasting
- Vectorized Implementation for Linear Regression

Numpy Array Operations

❖ Addition

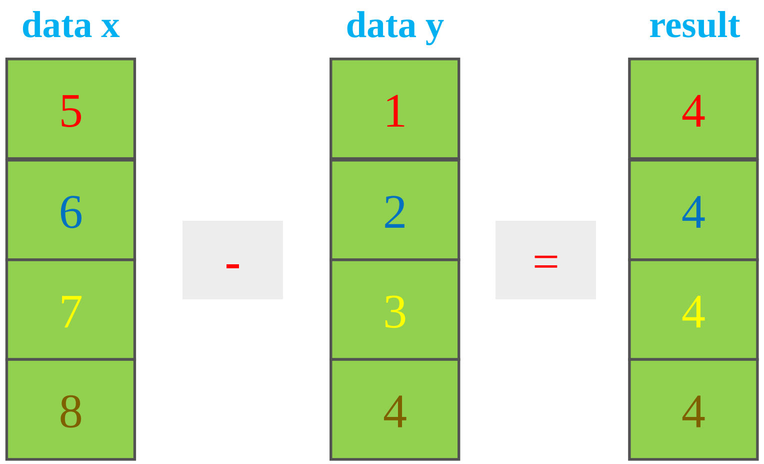
data x		data y		result
1		5		6
2	+	6	=	8
3		7		10
4		8		12

x + y

np.add(x, y)

Numpy Array Operations

❖ Subtraction



`x - y`

`np.subtract(x, y)`

Numpy Array Operations

❖ Multiplication

data x		data y		result
1		5		5
2	*	6	=	12
3		7		21
4		8		32

```
x * y
```

```
np.multiply(x, y)
```

Numpy Array Operations

❖ Division

data x		data y		result
1		5		0.2
2	/	6	=	0.33
3		7		0.42
4		8		0.5

```
x // y
```

```
x / y
```

```
np.divide(x, y)
```

Numpy Array Operations

❖ Square root

data
1
2
3
4

sqrt(data) =

result
1.0
1.4
1.7
2.0

```
np.sqrt(data)
```

Numpy Array Operations

❖ Inner product

$$\underset{v}{\begin{bmatrix} 1 & 2 \end{bmatrix}} \bullet \underset{w}{\begin{bmatrix} 2 \\ 3 \end{bmatrix}} = \underset{result}{\begin{bmatrix} 8 \end{bmatrix}}$$

```
v.dot(w)
```

```
np.dot(v, w)
```

Numpy Array Operations

❖ Vector-matrix multiplication

$$\underset{X}{\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}} \bullet \underset{v}{\begin{bmatrix} 1 \\ 2 \end{bmatrix}} = \underset{result}{\begin{bmatrix} 5 \\ 11 \end{bmatrix}}$$

X.dot(v)

$$\underset{v}{\begin{bmatrix} 1 & 2 \end{bmatrix}} \bullet \underset{X}{\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}} = \underset{result}{\begin{bmatrix} 7 \\ 10 \end{bmatrix}}$$

np.dot(v, X)

Numpy Array Operations

❖ Matrix-matrix multiplication

$$\overset{X}{\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}} \bullet \overset{Y}{\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}} = \overset{result}{\begin{bmatrix} 6 & 5 \\ 14 & 13 \end{bmatrix}}$$

X.dot(Y)

$$\overset{Y}{\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}} \bullet \overset{X}{\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}} = \overset{result}{\begin{bmatrix} 11 & 16 \\ 5 & 8 \end{bmatrix}}$$

np.dot(Y, X)

Numpy Array Operations

❖ Transpose

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^T \rightarrow \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

X.T

ND array-Creating Arrays

Vẽ mindmap tổng hợp kiến thức

Outline

- Numpy Array Operations
- Broadcasting
- Vectorized Implementation for Linear Regression

Broadcasting

❖ Vector and a scalar

data

0	1	2
1	2	3

* 2 =

result

0	1	2
2	4	6

data

0	1	2
1	2	3

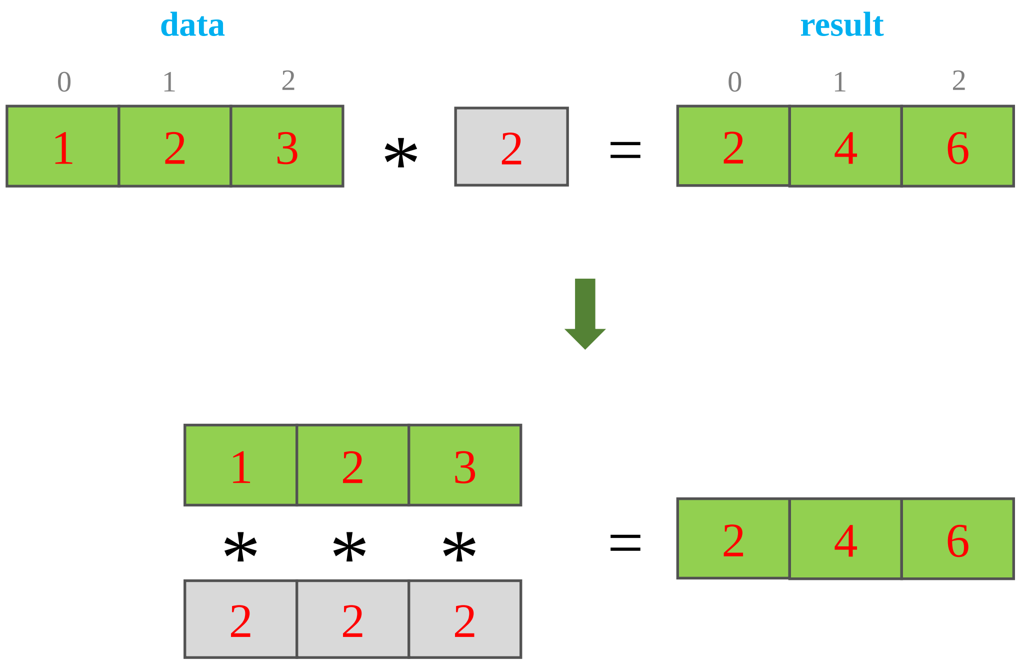
\- 2 =

result

0	1	2
-1	0	1

Broadcasting

❖ Vector and a scalar



Broadcasting

❖ Matrix and vector

$$
\mathbf{X} = \begin{array}{ccc} 0 & 1 & 2 \\ \hline 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{array}
\quad + \quad
\mathbf{v} = \begin{array}{ccc} 1 & 0 & 1 \end{array}
\quad = \quad
\mathbf{Y} = \begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 2 & 2 & 4 \\ 1 & 5 & 5 & 7 \\ 2 & 8 & 8 & 10 \\ 3 & 11 & 11 & 13 \end{array}
$$

Broadcasting

❖ Matrix and vector

$$
\mathbf{X} = \begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \\ 3 & 10 & 11 & 12 \end{array}
\quad + \quad
\mathbf{v} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}
$$

$$\Downarrow$$

$$
\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \\ 3 & 10 & 11 & 12 \end{array}
\quad + \quad
\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}
\quad = \quad
\mathbf{Y} = \begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 2 & 2 & 4 \\ 1 & 5 & 5 & 7 \\ 2 & 8 & 8 & 10 \\ 3 & 11 & 11 & 13 \end{array}
$$

Broadcasting

❖ Matrix and vector

$$\underset{\mathbf{X}}{\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \end{array}} + \underset{\mathbf{v}}{\begin{bmatrix} 1 & 0 & 1 \end{bmatrix}}$$

$$\Downarrow$$

$$\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \end{array} + \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} = \underset{\mathbf{Y}}{\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 2 & 2 & 4 \\ 5 & 5 & 7 \\ 8 & 8 & 10 \\ 11 & 11 & 13 \end{bmatrix} \end{array}}$$

ND array-Creating Arrays

Vẽ mindmap tổng hợp kiến thức

Creating Arrays

ABC Game

Score:

Team Sầu Riêng

Score:

Team Măng Cụt

Creating Arrays

ABC Game

Team Sầu Riêng

Bài 1: Viết một chương trình Python sử dụng thư viện numpy để thực hiện các phép toán cộng, trừ, nhân và chia trên hai vector vector1 = [1, 2, 3, 4, 5], vector2 = [6, 7, 8, 9, 10].

Bài 2: Cho ma trận matrix_A = [[1, 2, 3], [4, 5, 6]], vector vector_v = [1, 1, 0].

a) Thực hiện chuyển vị ma trận A và tính multiplication A*v và v*A
b) Thực hiện tính A+v, A*v theo cơ chế Broadcasting

Team Măng Cụt

Bài 1. Cho hai vector a = [1, 2, 3, 4], b = [9, 2, 3, 4] = Viết chương trình Python sử dụng thư viện numpy để tính căn bậc hai và tích vô hướng của hai vector.

Bài 2. Cho vector a = [8, 8, 8], b = 2. Viết chương trình tính tổng, hiệu, tích, thương của a và b theo cơ chế broadcasting.

Outline

- Numpy Array Operations
- Broadcasting
- Vectorized Implementation for Linear Regression

Linear Regression

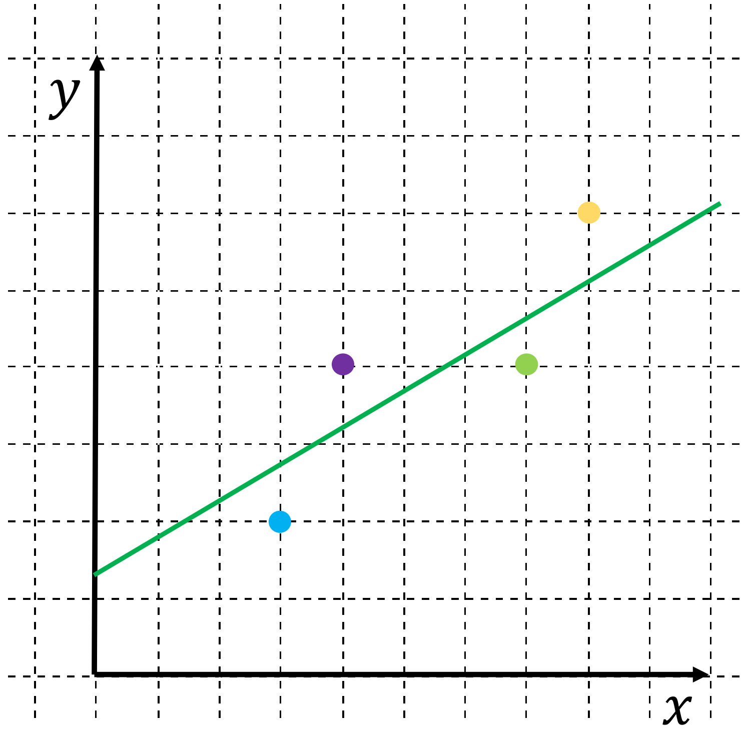
❖ Quick review

Linear regression models ← Linear equations

Linear equation = $w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$

Feature	Label
area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

House price data



Linear Regression

❖ Quick review

	Feature	Label
	area	price
	6.7	9.1
	4.6	5.9
	3.5	4.6
	5.5	6.7

House price data

Model: $\hat{y} = w_1 x_1 + b$

price = a $* \, area + b$

Features			Label
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

Advertising data

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

Sale = $w_1 * TV + w_2 * Radio + w_3 * Newspaper + b$

Linear Regression

❖ Quick review

Linear equation

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

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

w and b are parameters

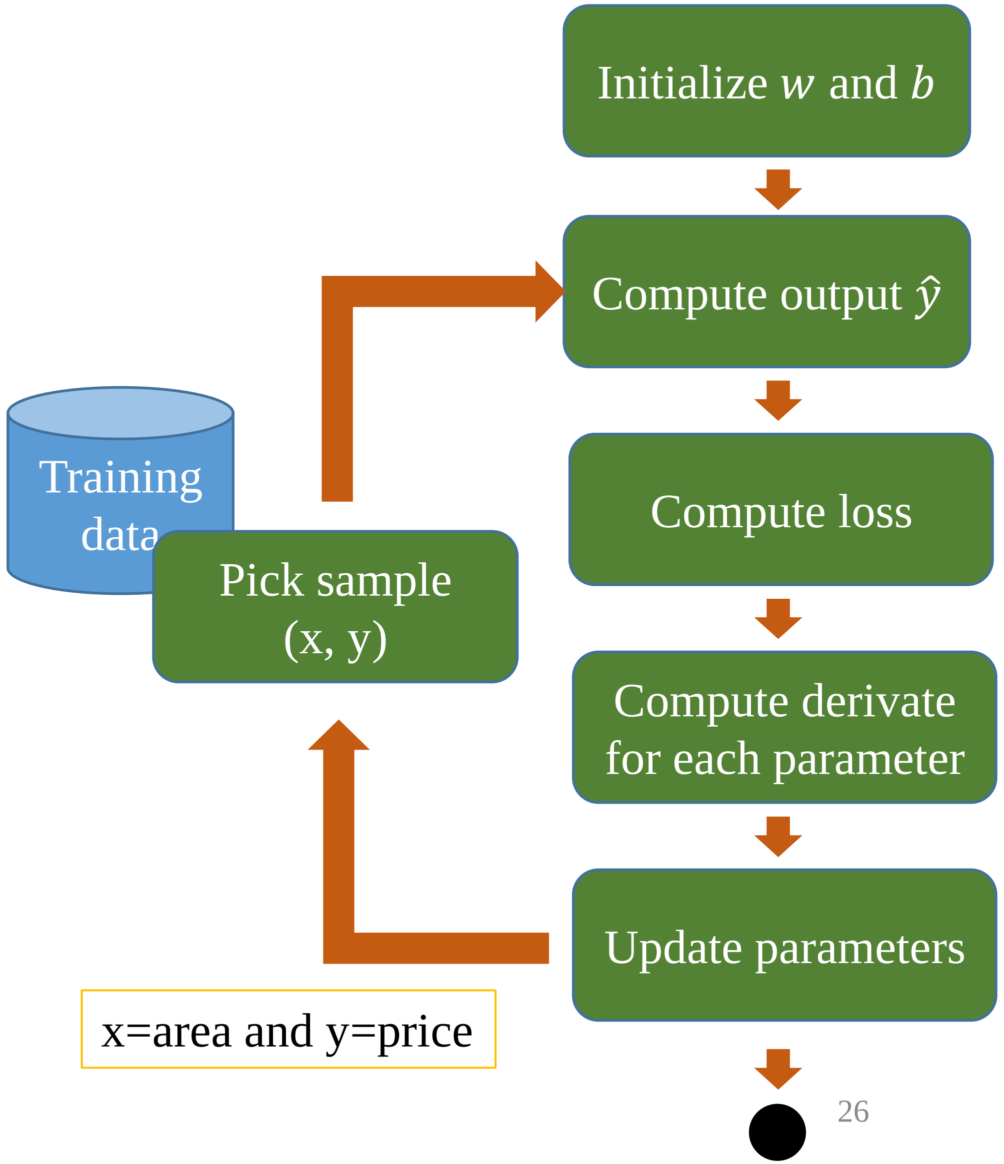
and x is input feature

Error (loss) computation

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

Squared loss

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



Linear Regression

❖ Quick review

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

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

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

$$w = w - \eta L'_w$$

$$b = b - \eta L'_b$$

η is learning rate

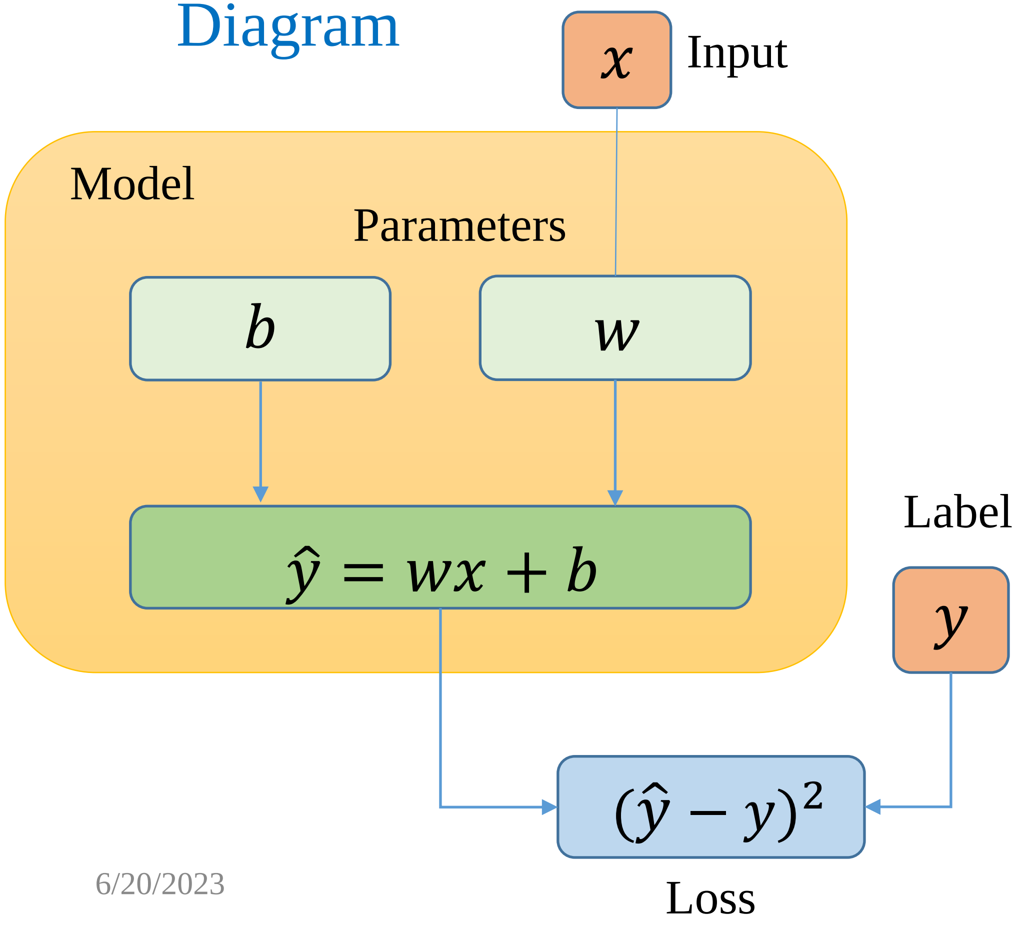
Linear Regression

❖ Quick review

Feature	Label
area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

House price data

Diagram



Cheat sheet

Compute the output $\hat{y}$

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

Compute the loss

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

Compute derivative

$$L'_w = 2x(\hat{y} - y)$$

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

Update parameters

$$w = w - \eta L'_w$$

$$b = b - \eta L'_b$$

Computational graph

❖ Quick review

1) Pick a sample (x, y) from training data

2) Tính output $\hat{y}$

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

3) Tính loss

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

4) Tính đạo hàm

$$L'_w = 2x(\hat{y} - y)$$

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

5) Cập nhật tham số

η is learning rate

$$w = w - \eta L'_w$$

$$b = b - \eta L'_b$$

```
# forward
def predict(x, w, b):
    return x*w + b

# compute gradient
def gradient(y_hat, y, x):
    dw = 2*x*(y_hat-y)
    db = 2*(y_hat-y)

    return (dw, db)

# update weights
def update_weight(w, b, lr, dw, db):
    w_new = w - lr*dw
    b_new = b - lr*db

    return (w_new, b_new)
```

Computational graph

❖ Quick review

1) Pick a sample (x, y) from training data

2) Tính output $\hat{y}$

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

3) Tính loss

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

4) Tính đạo hàm

$$L'_w = 2x(\hat{y} - y)$$

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

5) Cập nhật tham số

η is learning rate

$$w = w - \eta L'_w$$

$$b = b - \eta L'_b$$

```python
# init weights
b = 0.04
w = -0.34
lr = 0.01

# how long
epoch_max = 10
data_size = 4

for epoch in range(epoch_max):
    for i in range(data_size):
        # get a sample
        x = areas[i]
        y = prices[i]

        # predict y_hat
        y_hat = predict(x, w, b)

        # compute loss
        loss = (y_hat-y)*(y_hat-y)

        # compute gradient
        (dw, db) = gradient(y_hat, y, x)

        # update weights
        (w, b) = update_weight(w, b, lr, dw, db)
```

Problem and solution?

Feature	Label
area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

House price data

price = a $* \ area + b$

Features			Label
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

Advertising data

Model: $\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + b$

Sale = $w_1 * TV + w_2 * Radio + w_3 * Newspaper + b$

Features													Label
crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9

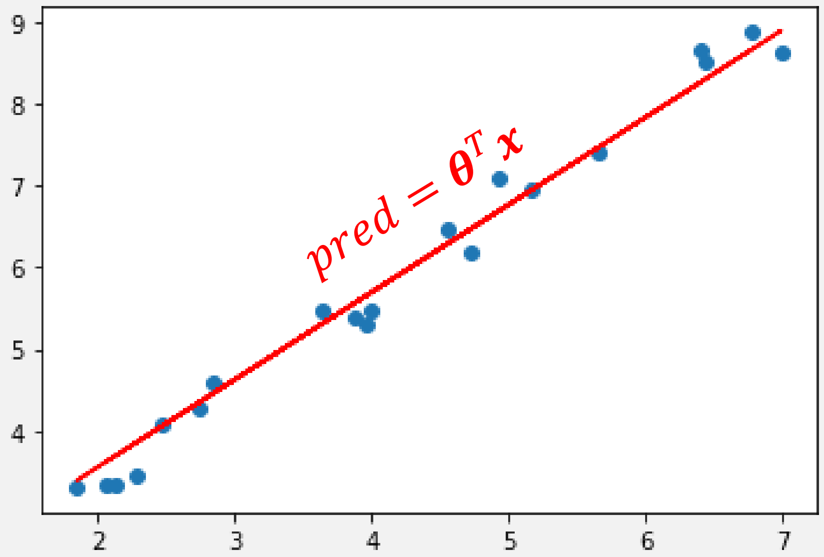
Boston House Price Data

medv = $w_1 * x_1 + \cdots + w_{13} * x_{13} + b$

Linear Regression

Model the relationship between feature x and label y

label y



feature **x**

Using a linear equation to fit data

Samples (x, y) are given in advance

Linear equation

$$\hat{y} = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

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

$\boldsymbol{\theta} = [b \;\; w_1 \;\; w_2 \; \ldots \; w_n]^T$ is parameter vector

and $\boldsymbol{x} = [1 \;\; x_1 \;\; x_2 \; \ldots \; x_n]^T$ is feature vector.

Error (loss) computation

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

Squared loss

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

Linear Regression

Linear equation

$$\hat{y} = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$

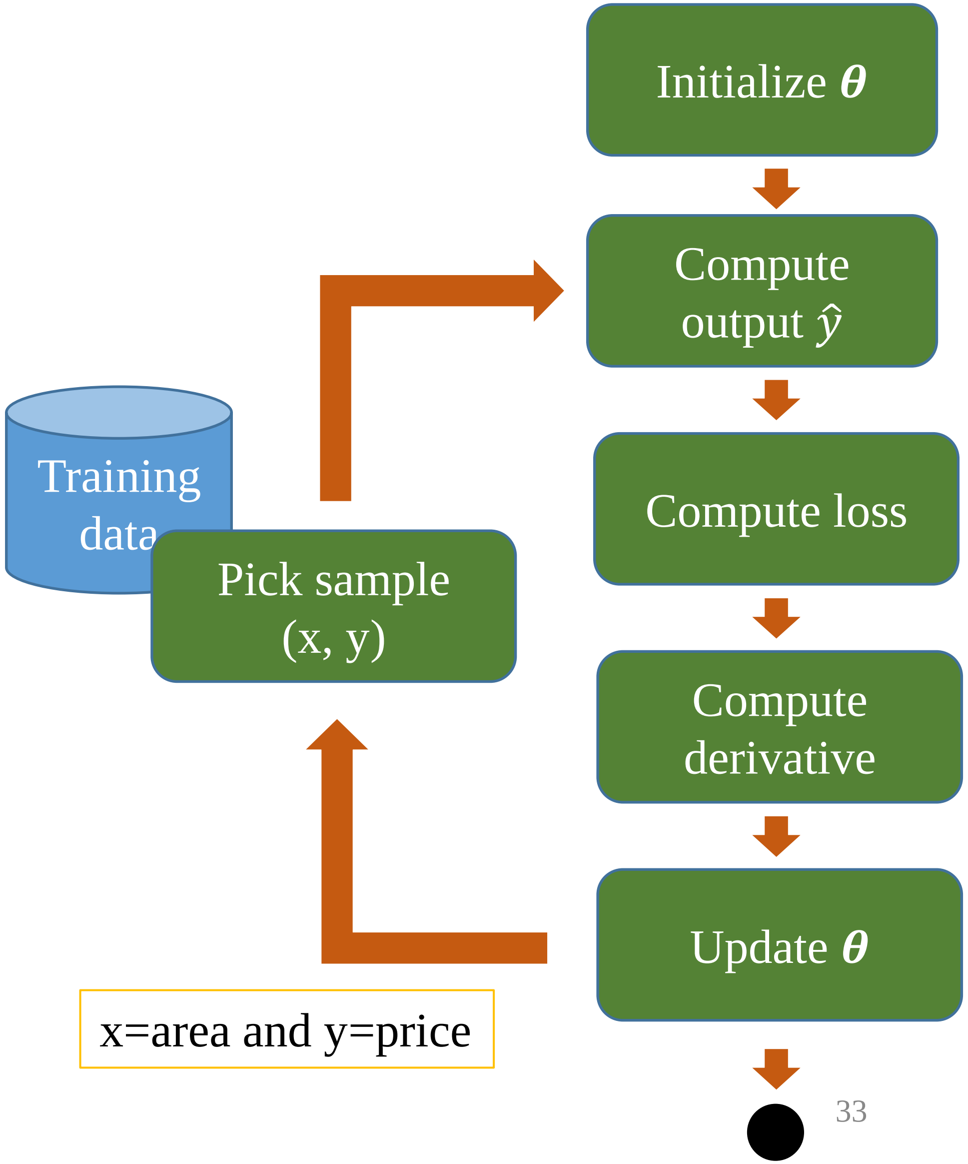
where $\hat{y}$ is a predicted value,
$\boldsymbol{\theta} = [b \;\; w_1 \;\; w_2 \; \ldots \; w_n]^T$ is parameter vector
and $\boldsymbol{x} = [1 \;\; x_1 \;\; x_2 \; \ldots \; x_n]^T$ is feature vector.

Error (loss) computation

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

Squared loss

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



Linear Regression: Vectorization

1) Pick a sample (x, y) from training data

2) Compute output $\hat{y}$

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

3) Compute loss

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

4) Compute derivative

$$L'_w = 2x(\hat{y} - y)$$

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

5) Update parameters

$$w = w - \eta L'_w$$

$$b = b - \eta L'_b$$

η is learning rate

Normal version

1) Pick a sample (x, y) from training data

2) Compute output $\hat{y}$

$$\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{\theta}$$

3) Compute loss

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

4) Compute derivative

$$L'_{\boldsymbol{\theta}} = 2\boldsymbol{x}(\hat{y} - y)$$

5) Update parameters

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

Vectorization

Linear Regression: Vectorization

1) Pick a sample (x, y) from training data

2) Compute output $\hat{y}$

$$\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{\theta}$$

3) Compute loss

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

4) Compute derivative

$$L'_{\boldsymbol{\theta}} = 2\boldsymbol{x}(\hat{y} - y)$$

5) Update parameters

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

$$\underset{\textbf{v}}{\begin{bmatrix} 1 & 2 \end{bmatrix}} \bullet \underset{\textbf{w}}{\begin{bmatrix} 2 \\ 3 \end{bmatrix}} = \underset{\textbf{result}}{\begin{bmatrix} 8 \end{bmatrix}}$$

```python
import numpy as np

v = np.array([1, 2])
w = np.array([2, 3])

# Tính inner product giữa v và w
print('method 1 \n', v.dot(w))
print('method 2 \n', np.dot(v, w))
```

```
method 1
8

method 2
8
```

Vectorization

Linear Regression: Vectorization

1) Pick a sample (x, y) from training data

2) Compute output $\hat{y}$

$$\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{\theta}$$

3) Compute loss

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

4) Compute derivative

$$L'_{\boldsymbol{\theta}} = 2\boldsymbol{x}(\hat{y} - y)$$

5) Update parameters

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

```python
import numpy as np

# forward
def predict(x, theta):
    return x.dot(theta)

# compute gradient
def gradient(y_hat, y, x):
    dtheta = 2*x*(y_hat-y)

    return dtheta

# update weights
def update_weight(theta, lr, dtheta):
    dtheta_new = theta - lr*dtheta

    return dtheta_new
```


Vectorization

Linear Regression Vectorization

❖ Implementation

```python
import numpy as np

# forward
def predict(x, theta):
    return x.dot(theta)

# compute gradient
def gradient(y_hat, y, x):
    dtheta = 2*x*(y_hat-y)

    return dtheta

# update weights
def update_weight(theta, lr, dtheta):
    dtheta_new = theta - lr*dtheta

    return dtheta_new
```

```python
# test sample
x = np.array([6.7, 1])
y = np.array([9.1])

# init weight
lr = 0.01
theta = np.array([-0.34, 0.04]) #[w, b]
print('theta', theta)

# predict y_hat
y_hat = predict(x, theta)
print('y_hat: ', y_hat)

# compute loss
loss = (y_hat-y)*(y_hat-y)
print('Loss: ', loss)

# compute gradient
dtheta = gradient(y_hat, y, x)
print('dtheta: ', dtheta)

# update weights
theta = update_weight(theta, lr, dtheta)
print('theta_new: ', theta)
```

Linear Regression Vectorization

Vẽ mindmap tổng hợp kiến thức