AI VIETNAM
AI Insight Course

Computational Graph and Linear Regression

Quang-Vinh Dinh
Ph.D. in Computer Science

Year 2020

Outline

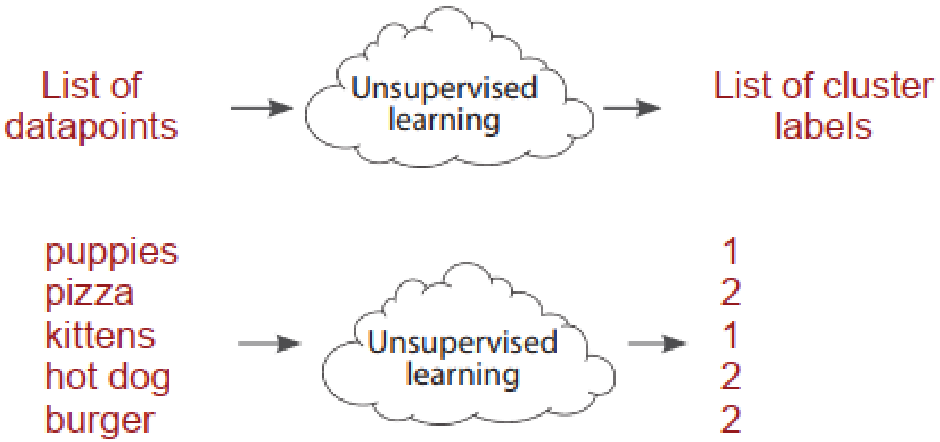
- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
- Summary

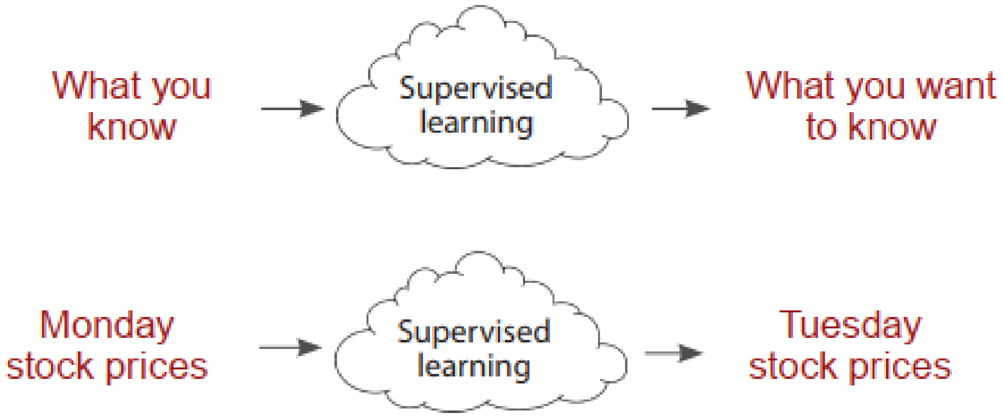
Machine Learning

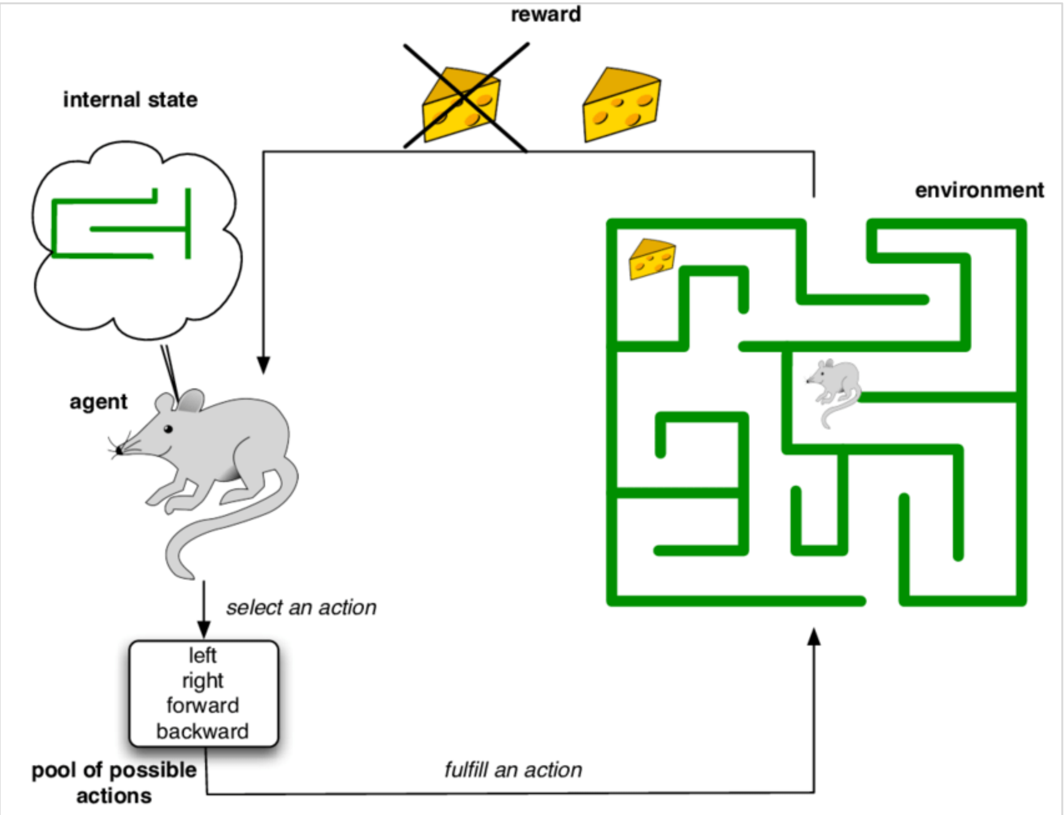
❖ Definition

What is machine learning?

> A field of study that gives computers the ability to learn without being explicitly programmed.
>
> —Attributed to Arthur Samuel

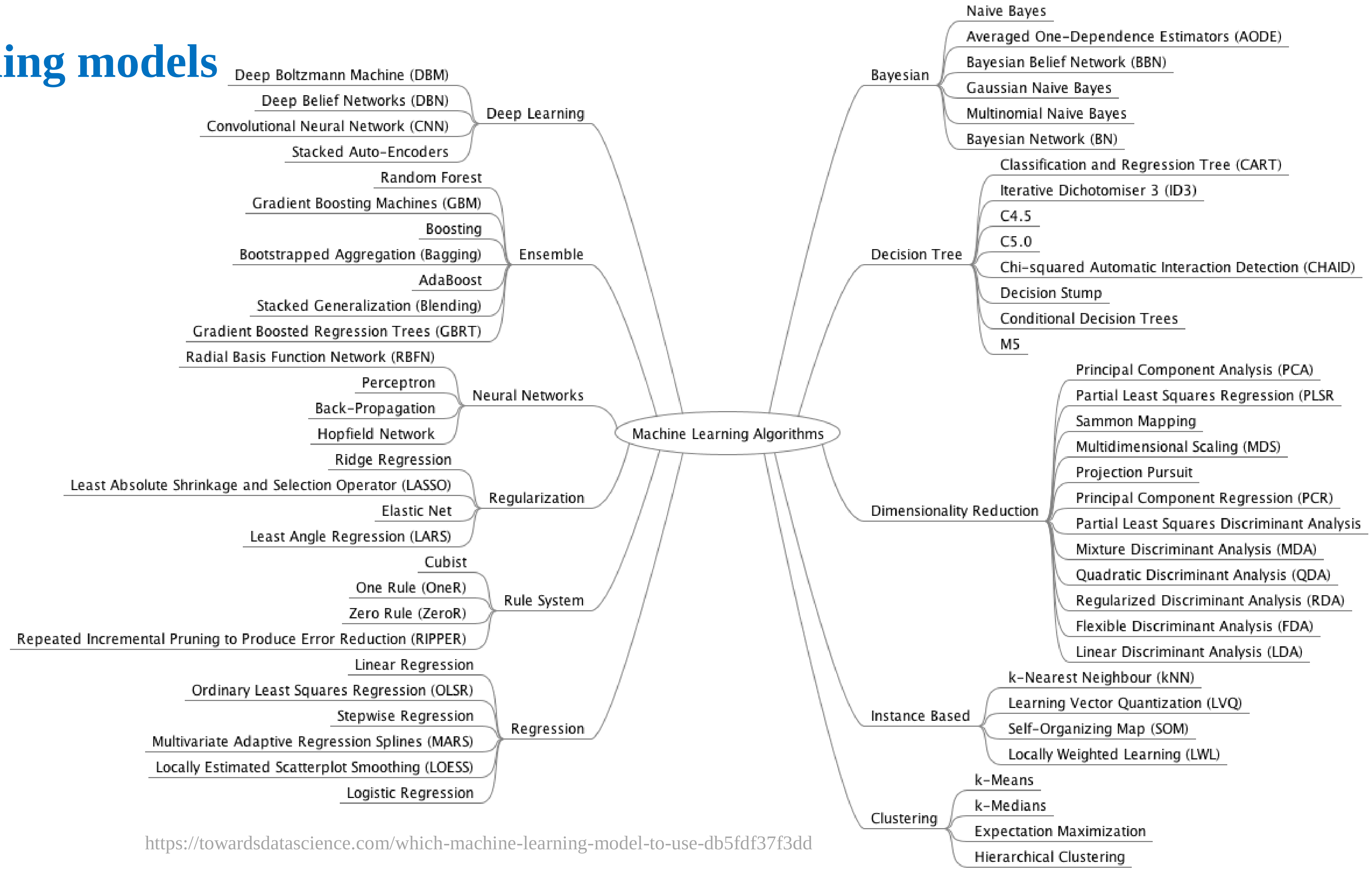






Machine Learning

Machine learning models



https://towardsdatascience.com/which-machine-learning-model-to-use-db5fdf37f3dd

Machine Learning

❖ Supervised learning

Input and output data is provided

Training data

Cats

Dogs

From Cat-Dog dataset

Machine Learning

❖ Supervised learning

Training data

Used to teach

Machine learning model

This is a cat

This is a dog

Machine learning model

Training phase

From Cat-Dog dataset

Testing data (≠ training data)

Machine learning model

Make decision

Cat or Dog?

Testing phase

Machine Learning

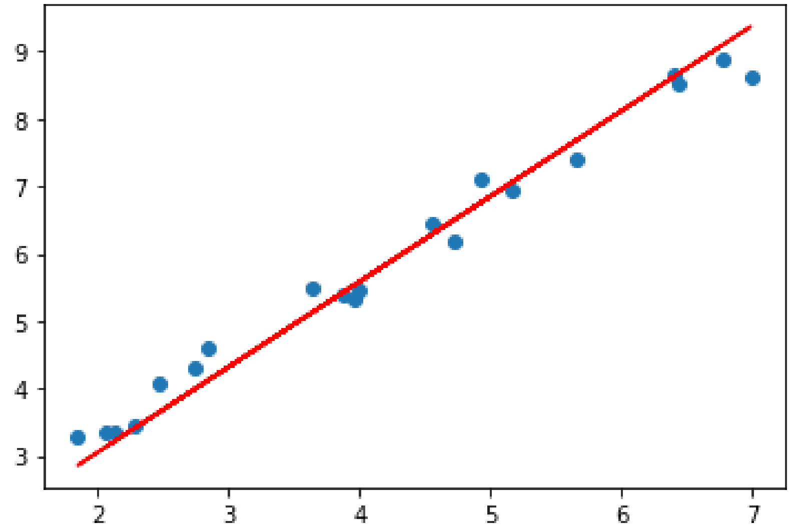
❖ Supervised learning

❖ Regression (prediction)

Linear regression models ← Linear equations

Linear equation = $w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$

where $\boldsymbol{w}$ is a weight vector
and $\boldsymbol{x}$ is feature vector



Machine Learning

❖ Supervised learning

❖ Linear Regression: Data processing

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

House price data

TV (Features)	Radio (Features)	Newspaper (Features)	Sales (Label)
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: y = $w_1x_1 + b$

price = a $* \, area + b$

Model: y = $w_1x_1 + w_2x_2 + w_3x_3 + b$

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

Machine Learning

❖ Supervised learning

❖ Linear Regression: Data processing

Boston House Price Data

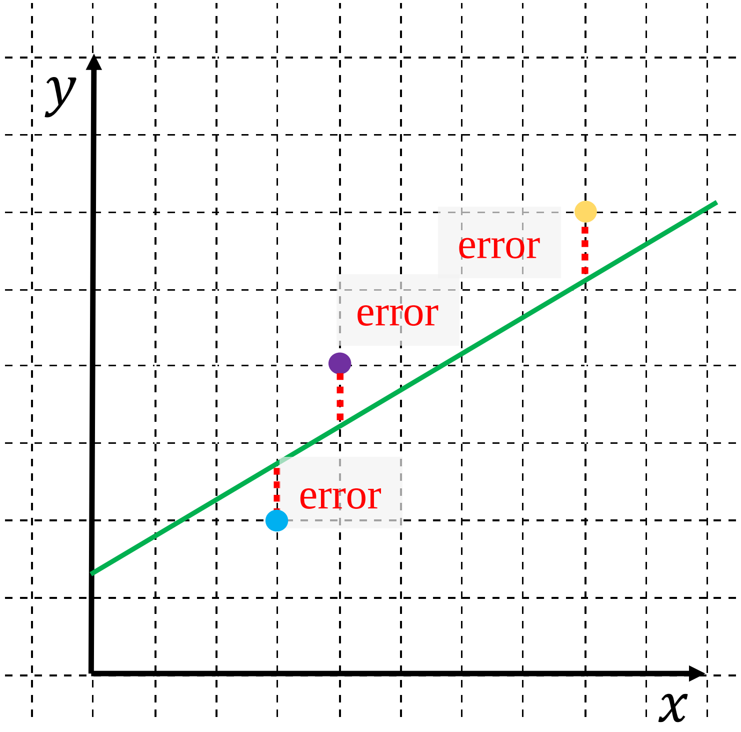
Features													Label
crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	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

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

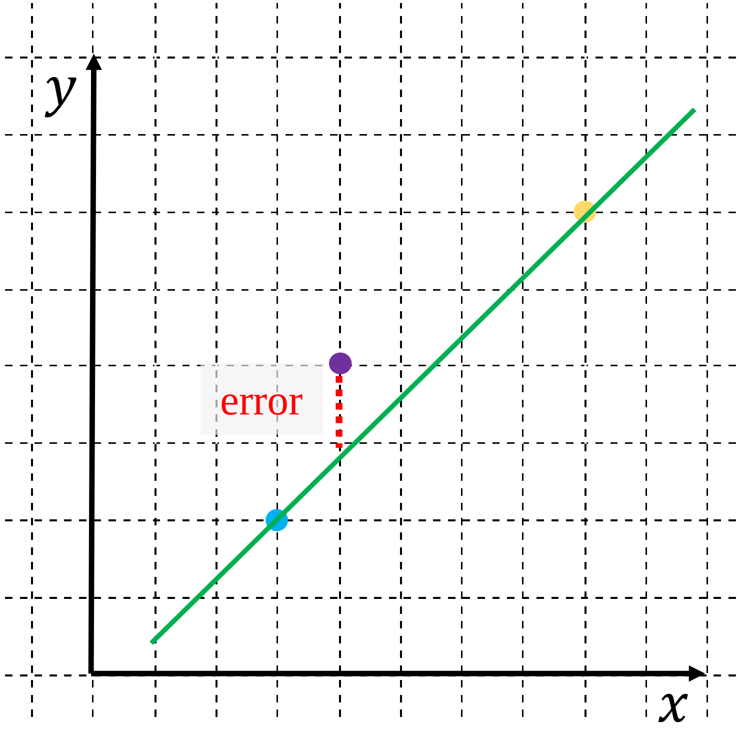
Machine Learning

- **Linear regression**
  - **Aim to fit data**

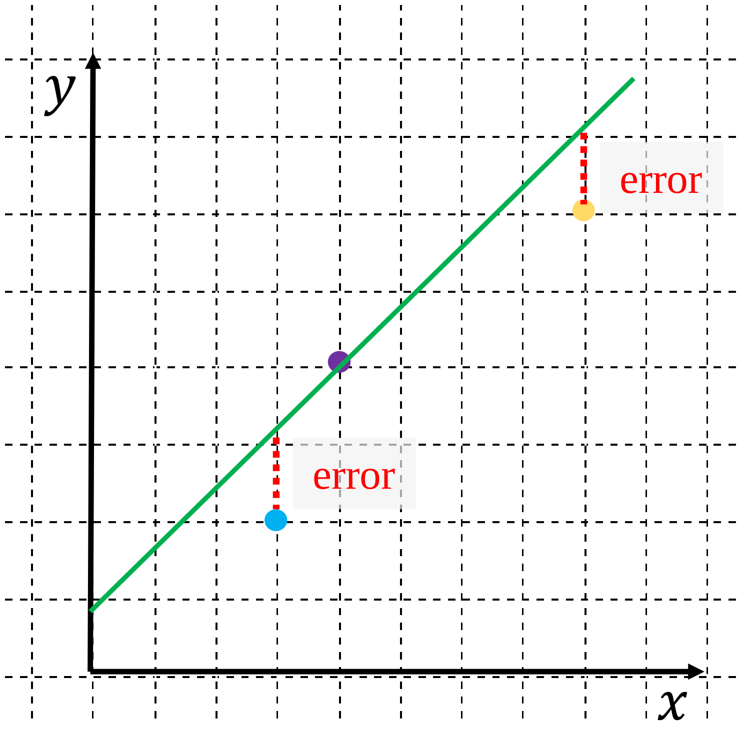
Training data



$y = a_1 x + b_1$



$y = a_2 x + b_2$



$y = a_3 x + b_3$

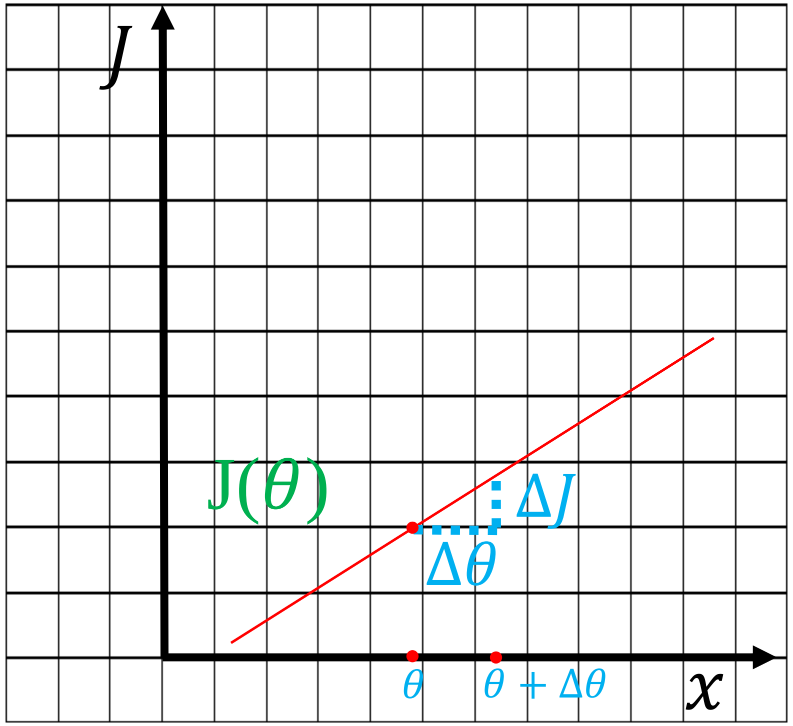
Find w and b that have the smallest error. How?

Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
- Summary

Derivative/Gradient

❖ A cue to optimize a function



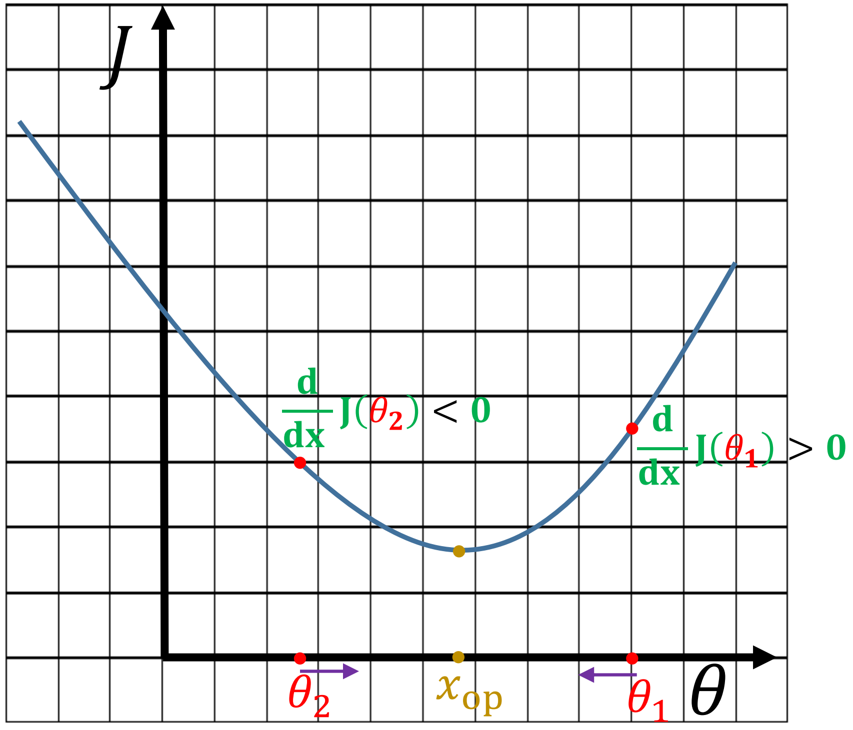
$$\text{Đạo hàm} = \frac{\textit{Thay đổi theo } J}{\textit{Thay đổi theo } \theta} = \frac{\Delta J}{\Delta\theta}$$

$$\frac{d}{d\theta} J(\theta) = \underbrace{\lim_{\Delta\theta \to 0}} \frac{J(\theta + \Delta\theta) - J(\theta)}{\Delta\theta}$$

$\Delta\theta$ cần tiến về 0 để đường tiếp tuyến tiến về hàm J(θ) trong vùng lân cận tại θ

Derivative/Gradient

❖ A cue to optimize a function



Quan sát: θ_{op} ở vị trí ngược hướng đạo hàm tại θ_1 và θ_2

Cách xử lý việc di chuyển ngược hướng đạo hàm cho θ_1 và θ_2 (để tìm θ_{op}) khác nhau hình thành các thuật toán tối ưu hóa khác nhau

Cách cập nhật giá trị x đơn giản

$$\theta = \theta - \eta \frac{d}{dx}J(\theta)$$

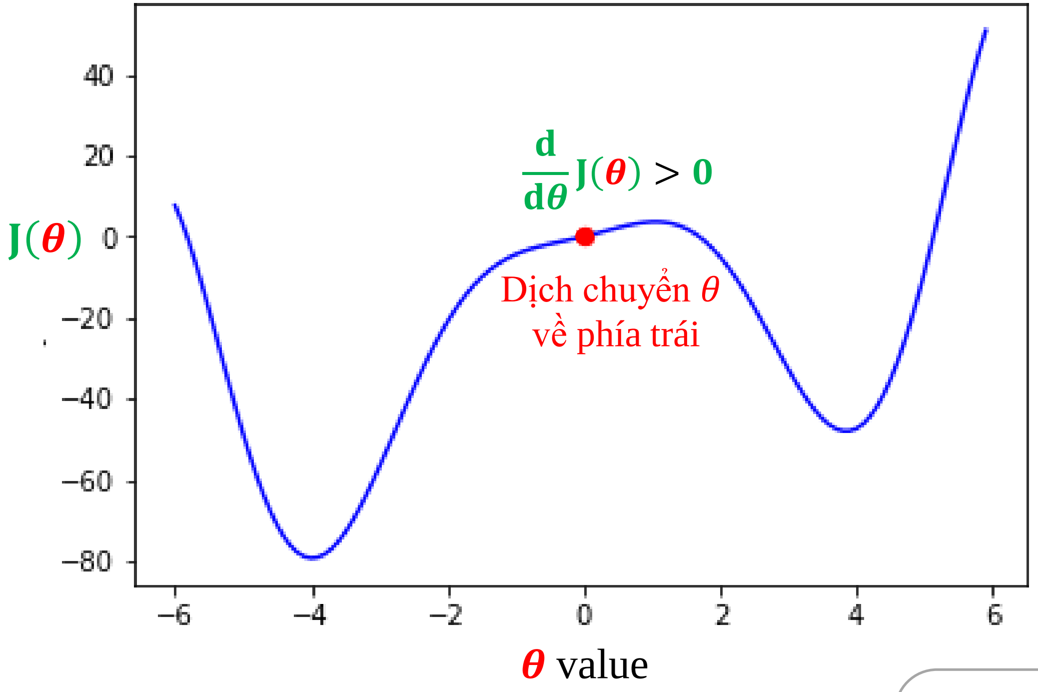
Đạo hàm tại θ

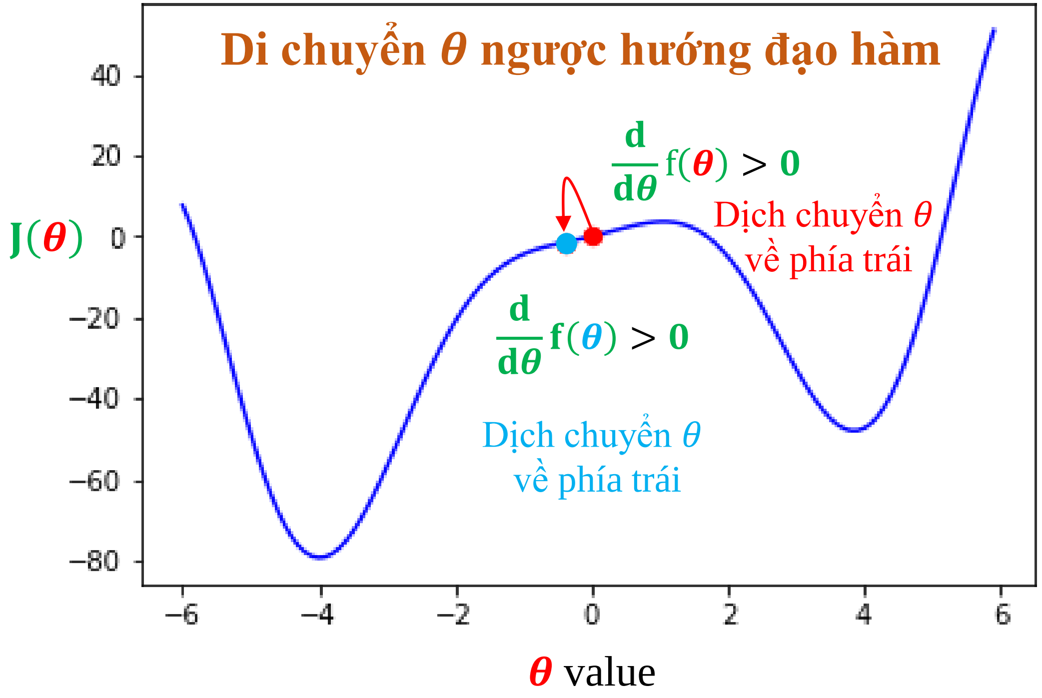
Trọng số

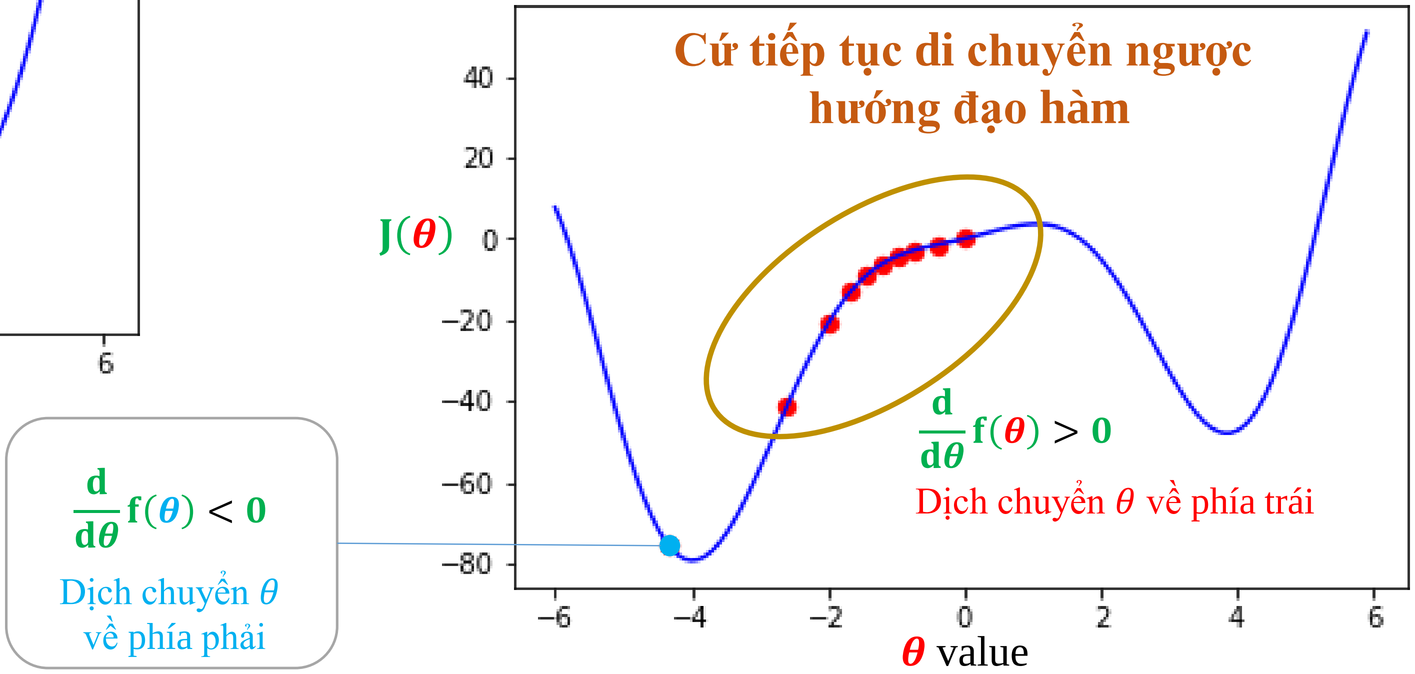
Derivative/Gradient

❖ **A cue to optimize a function**

Khởi tạo giá trị θ







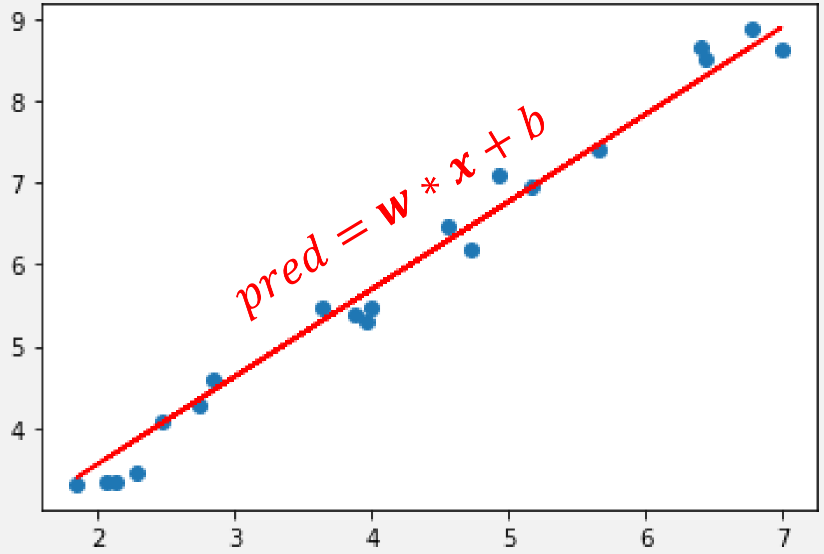
Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
- Summary

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

$$o = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

where o is a predicted value,
$w_1, w_2, \ldots, w_n$ and b are parameters
and $\boldsymbol{x} = [x_1 \;\; x_2 \;\; \ldots \;\; x_n]^T$ is feature vector.

Error (loss) computation

Idea: compare predicted values o and label values y

Squared loss

$$\mathrm{L}(\mathbf{w}, \mathrm{b}) = (o - y)^2$$

How to find optimal w and b?

Linear Regression

Linear equation

$$o = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

where o is a predicted value,
$w_1, w_2, \dots, w_n$ and b are parameters
and $\boldsymbol{x} = [x_1 \ \ x_2 \ \ \dots \ \ x_n]^T$ is feature vector.

Error (loss) computation

Idea: compare predicted values o and label values y

Squared loss

$$\mathrm{L}(\mathbf{w}, \mathrm{b}) = (o - y)^2$$

How to find optimal **w** and b?

Use gradient descent to minimize the loss function

Tính đạo hàm

$$\frac{\partial L}{\partial w_j} = \frac{\partial L}{\partial o}\frac{\partial o}{\partial w_j} = 2x_j(o - y)$$

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

Cập nhật tham số

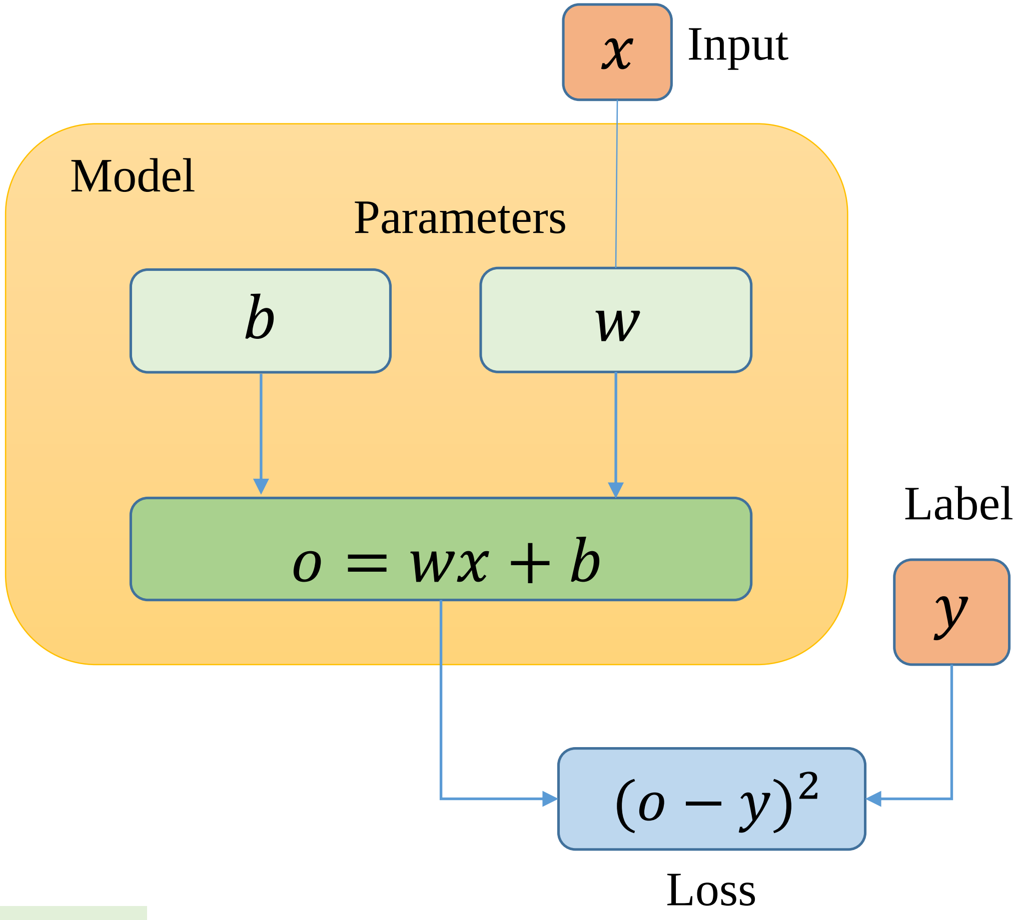
$$w_j = w_j - \eta L'_{w_j}$$

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

η is learning rate

Linear Regression

Diagram



Cheat sheet

Tính output o

$$o = \boldsymbol{wx} + b$$

Tính Loss

$$L = (o - y)^2$$

Tính đạo hàm

$$L'_{w_j} = 2x_j(o - y)$$

$$L'_b = 2(o - y)$$

Cập nhật tham số

$$w_j = w_j - \eta L'_{w_j}$$

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

Linear Regression

Cheat sheet

Tính output o

$$o = \boldsymbol{wx} + b$$

Tính Loss

$$L = (o - y)^2$$

Tính đạo hàm

$$L'_{w_j} = 2x_j(o - y)$$

$$L'_b = 2(o - y)$$

Cập nhật tham số

$$w_j = w_j - \eta L'_{w_j}$$

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

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

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

    return (dw, db)

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

    return (w_new, b_new)
```

Linear Regression

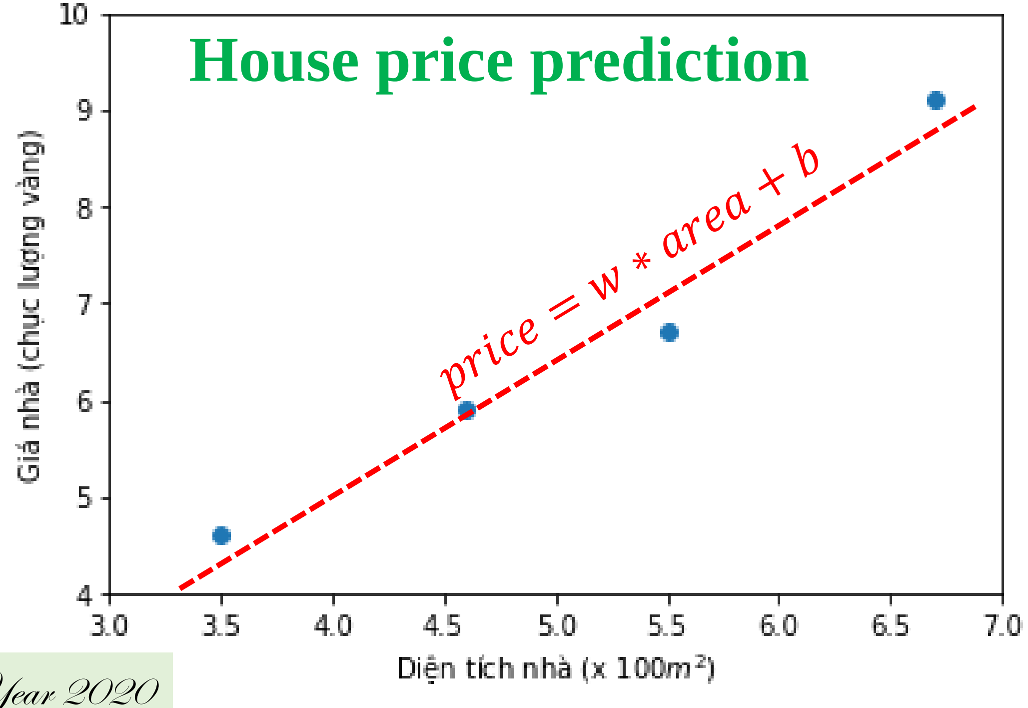
```python
# forward
def predict(x,w,b):
    return x*w + b

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

    return (dw, db)

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

    return (w_new, b_new)
```

```python
# test with a sample
x = 6.7
y = 9.1

# init weight
b = 0.04
w = -0.34
n = 0.01

# predict z
z = predict(x,w,b)
print('z: ', z)

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

# compute gradient
(dw, db) = gradient(z,y,x)
print('dw: ', dw)
print('db: ', db)

# update weights
(w_new, b_new) = update_weight(w,b,n,dw,db)
print('w_new: ', w_new)
print('b_new: ', b_new)
```

Linear Regression

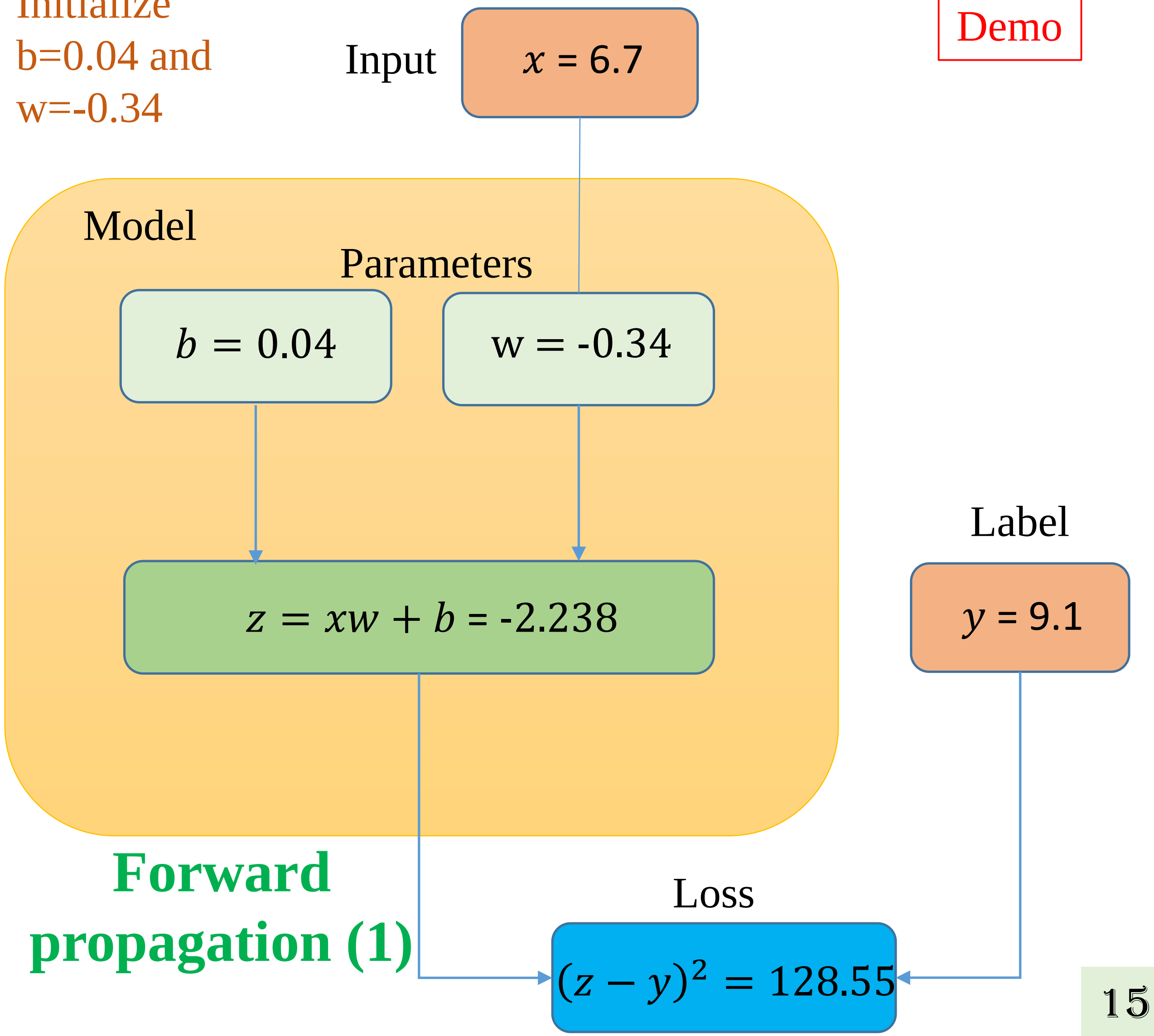
Given sample data

area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7



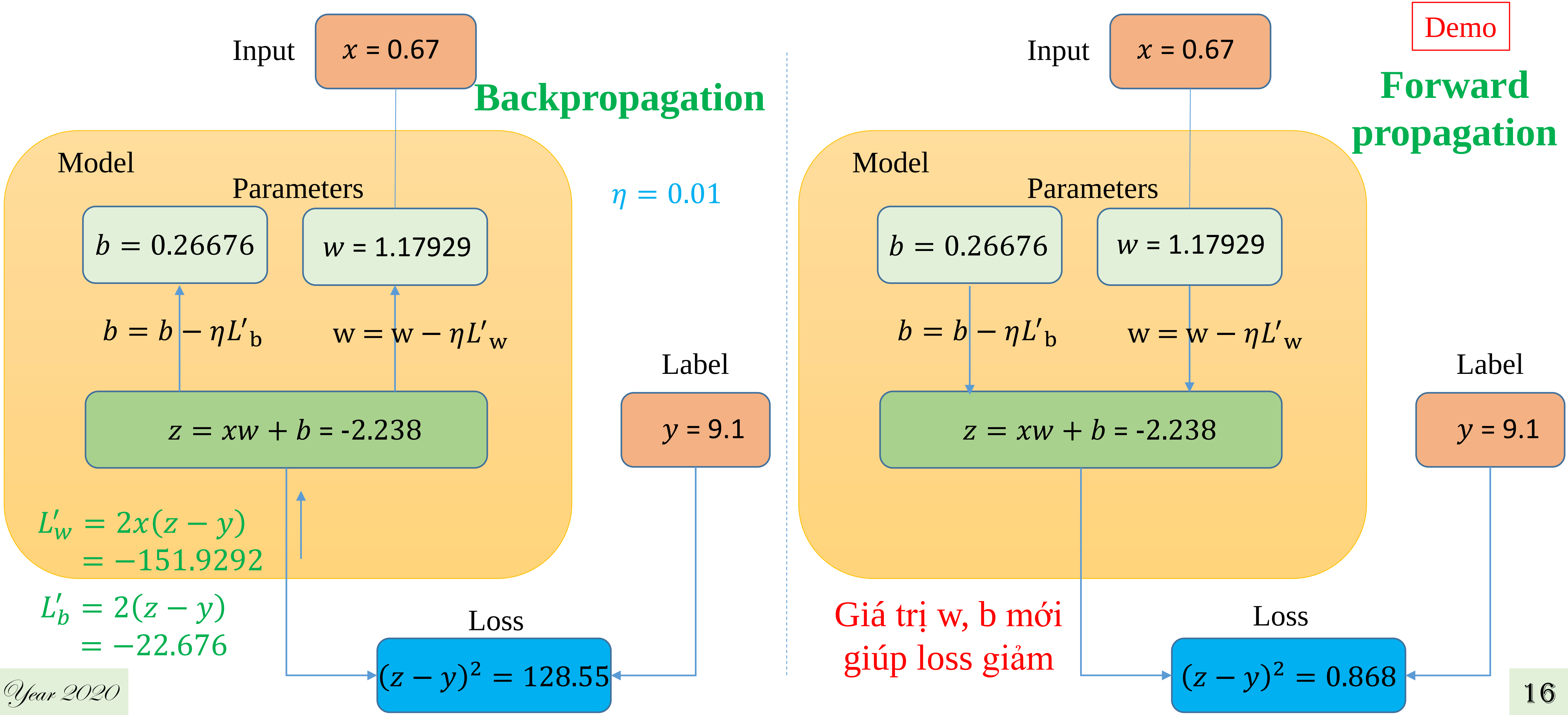
Initialize b=0.04 and w=-0.34

Demo



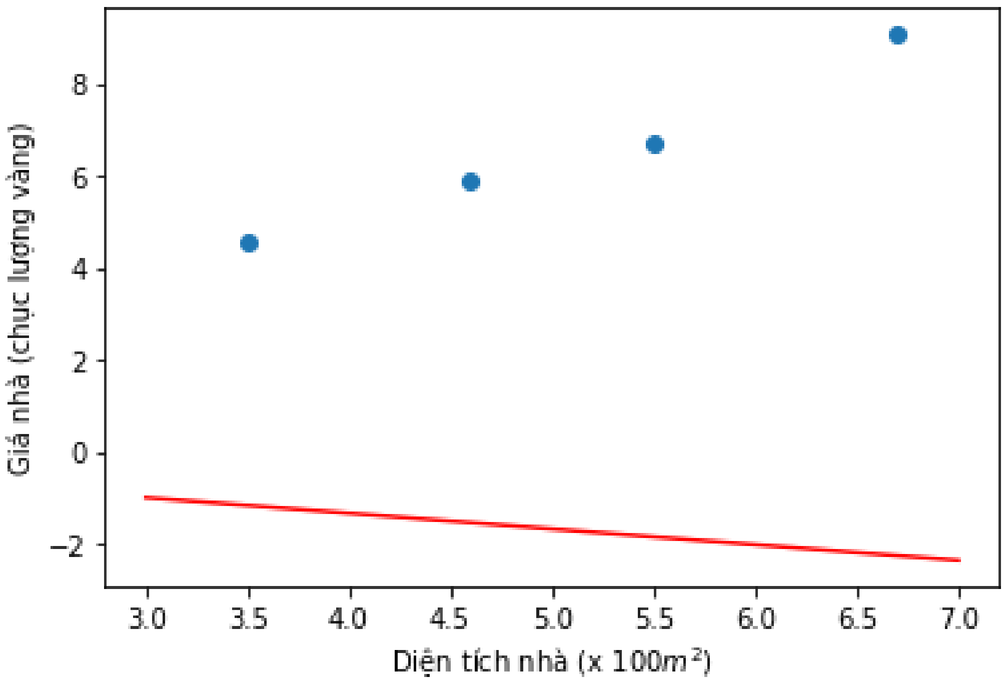
Forward propagation (1)

Linear Regression



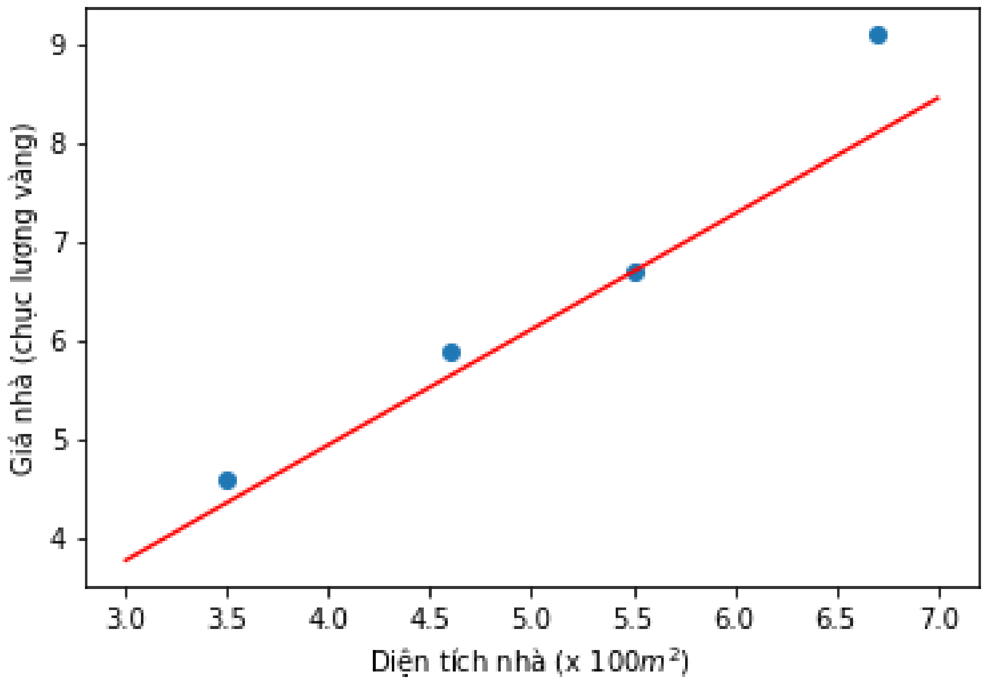
Linear Regression

Model prediction before and after the first update



w = -0.34 b = 0.04 L = 128.55

Before updating



w = 1.179292 b = 0.26676 L = 0.868

After updating

Linear Regression: Vectorization

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

5) Cập nhật tham số

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

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

η is learning rate

Friendly version

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

Generalized formula

Linear Regression: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

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
# aivietnam.ai
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
```

Generalized formula

Linear Regression: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

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

```python
# aivietnam.ai
import numpy as np

x = np.array([1,2,3,4])
y = np.array([5,6,7,8])

print('data x \n', x)
print('data y \n', y)

# Tích các phần tử tương ứng giữa x và y
print('method 1 \n', x*y)
print('method 2 \n', np.multiply(x, y))
```

Generalized formula

Linear Regression: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

```python
import numpy as np

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

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

    return dtheta

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

    return dtheta_new

```

Generalized formula

Linear Regression: Vectorization

```python
import numpy as np

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

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

    return dtheta

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

    return dtheta_new

```

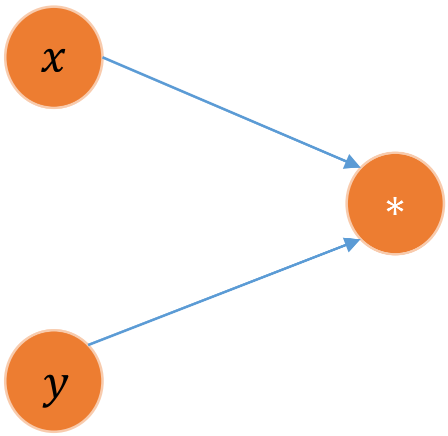
```python
# test with a sample
x = np.array([6.7, 1])
y = np.array([9.1])

# init weight
n = 0.01
# thetas = [w, b]
theta = np.array([-0.34, 0.04])


# predict z
z = predict(x, theta)
print('Input data: ', x)
print('Theta: ', theta)
print('z: ', z)

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

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

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

Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
- Summary

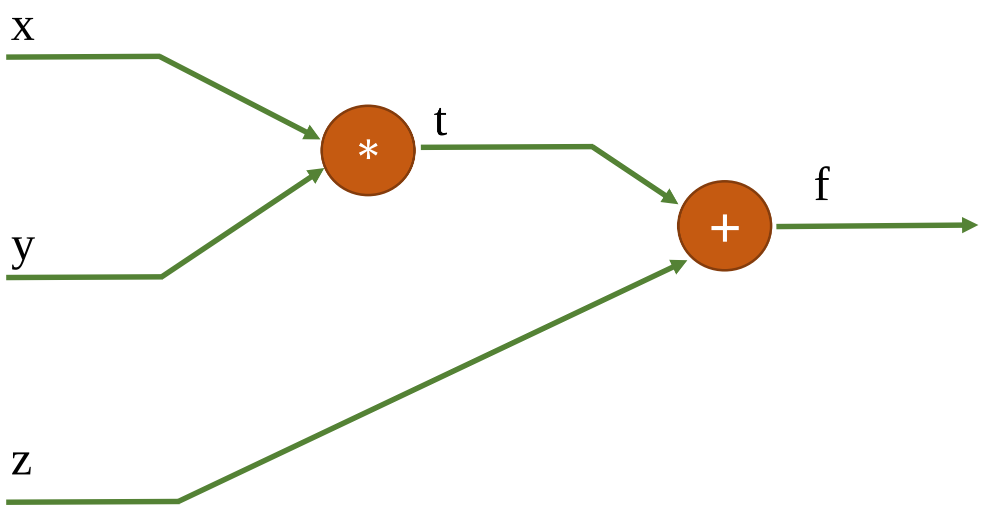
Computational graph

- ❖ A directed graph
- ❖ Nodes represent variables or operations



- ❖ Construct computational graph for $f(x, y, z) = x * y + z$
- ❖ Rewrite $f(x, y, z)$ as

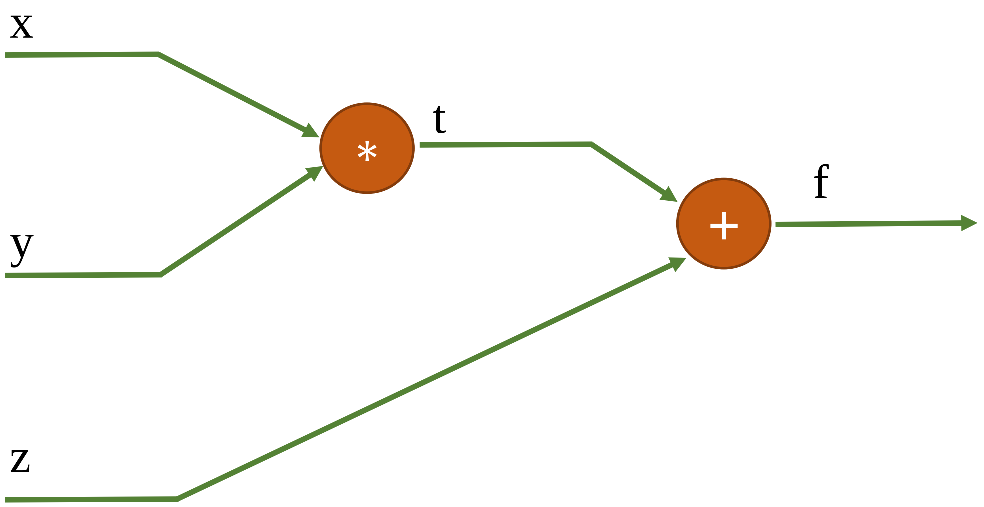
$f(t, z) = t + z$ where $t = x * y$



Computational graph

- **Construct computational graph for** $f(x, y, z) = x * y + z$
- **Rewrite** $f(x, y, z)$ **as**

$f(t, z) = t + z$ where $t = x * y$



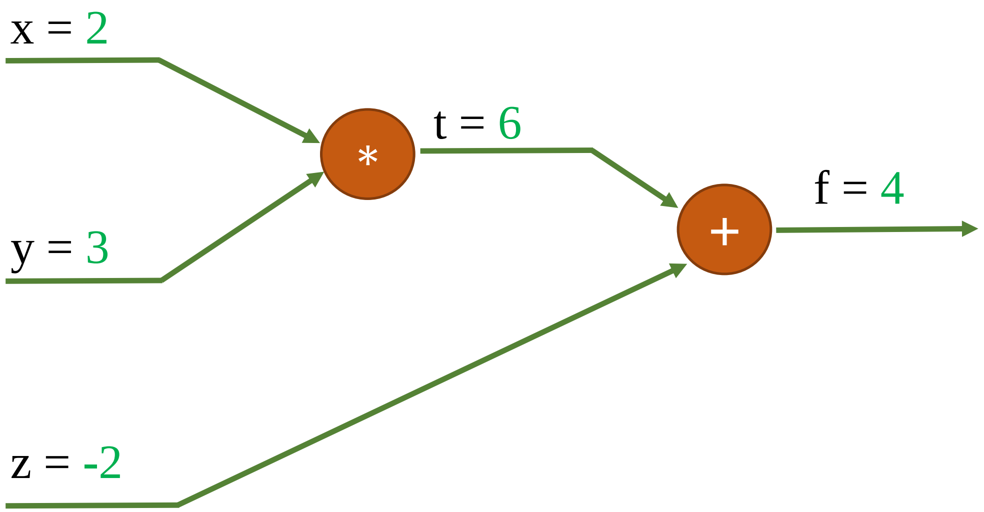
Partial derivative

$$\frac{\partial t}{\partial x} = y \qquad \frac{\partial f}{\partial z} = 1 \qquad \frac{\partial f}{\partial x} = y$$

$$\frac{\partial t}{\partial y} = x \qquad \frac{\partial f}{\partial t} = 1 \qquad \frac{\partial f}{\partial y} = x$$

Computational graph

❖ **Compute** $f(x, y, z)$ **với** $x = 2$, $y = 3$ **và** $z = -2$.



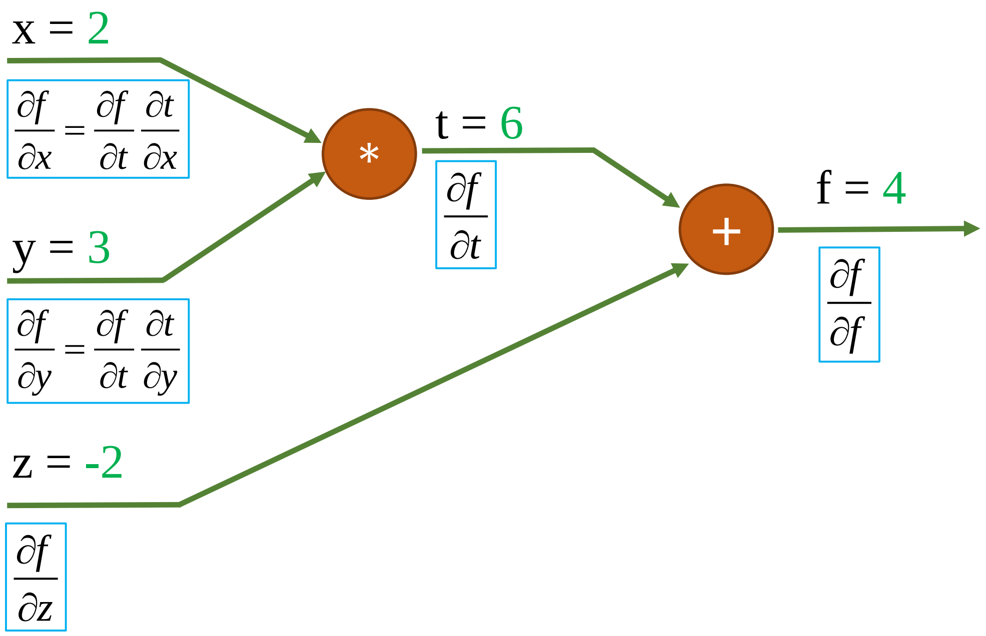
Partial derivative

$$\frac{\partial t}{\partial x} = y \qquad \frac{\partial f}{\partial z} = 1 \qquad \frac{\partial f}{\partial x} = y$$

$$\frac{\partial t}{\partial y} = x \qquad \frac{\partial f}{\partial t} = 1 \qquad \frac{\partial f}{\partial y} = x$$

Computational graph

❖ **Compute** $f(x, y, z)$ **với** $x = 2$**,** $y = 3$ **và** $z = -2$**.**



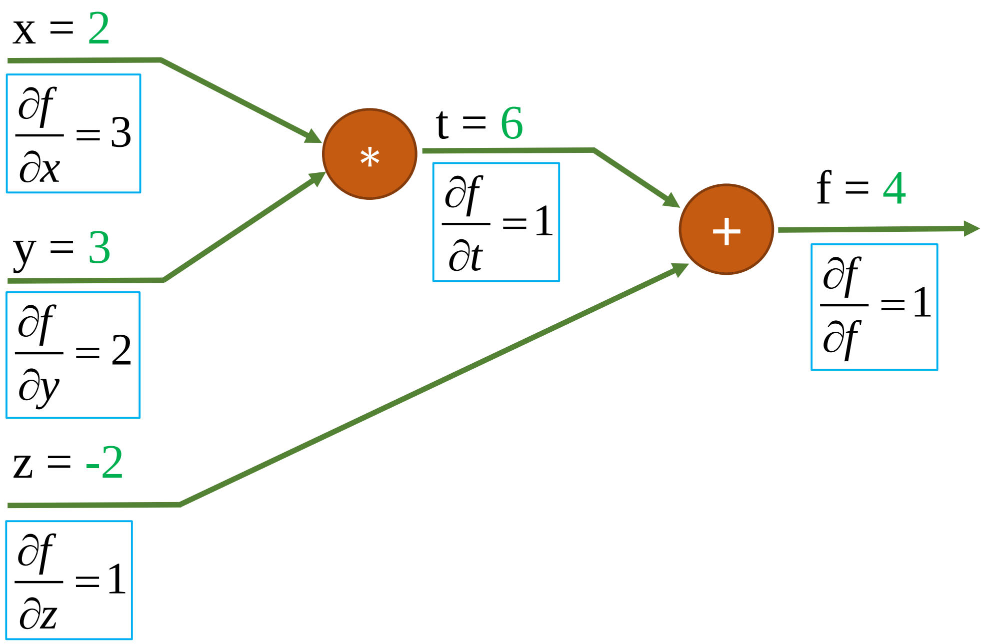
Partial derivative

$$\frac{\partial t}{\partial x} = y \qquad \frac{\partial f}{\partial z} = 1 \qquad \frac{\partial f}{\partial x} = y$$

$$\frac{\partial t}{\partial y} = x \qquad \frac{\partial f}{\partial t} = 1 \qquad \frac{\partial f}{\partial y} = x$$

Computational graph

❖ **Compute** $f(x, y, z)$ **với** $x = 2$, $y = 3$ **và** $z = -2$.



Partial derivative

$$\frac{\partial t}{\partial x} = y \qquad \frac{\partial f}{\partial z} = 1 \qquad \frac{\partial f}{\partial x} = y$$

$$\frac{\partial t}{\partial y} = x \qquad \frac{\partial f}{\partial t} = 1 \qquad \frac{\partial f}{\partial y} = x$$

Outline

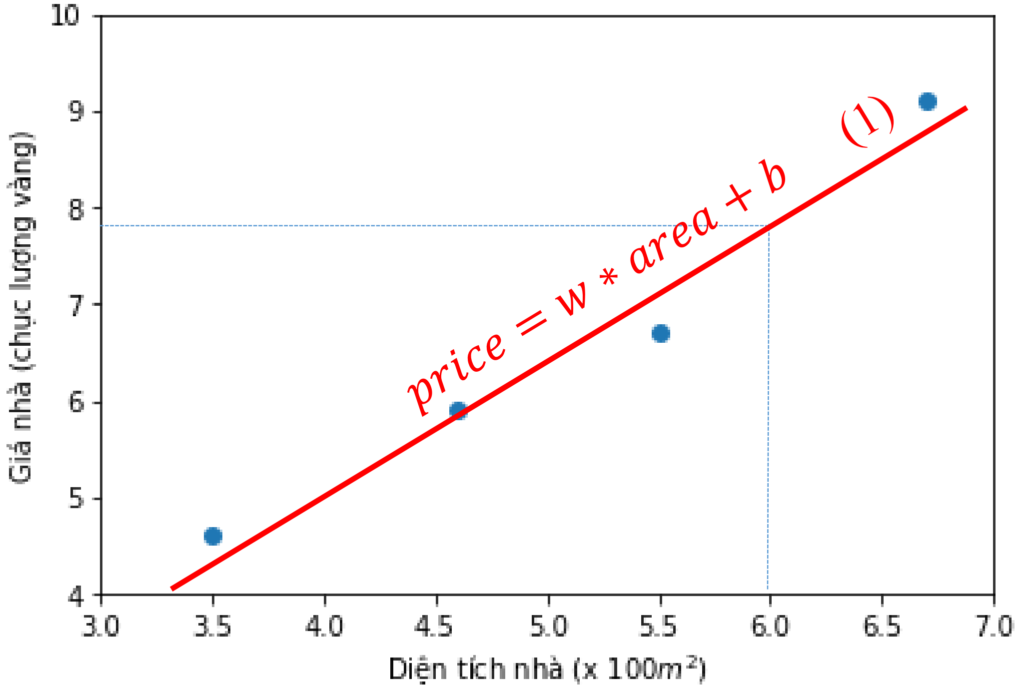
- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
    - 1-sample training
- Summary

Computational graph

- **House price predictions**
  - **How much for a 600-m^2 house?**

area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

Given sample data

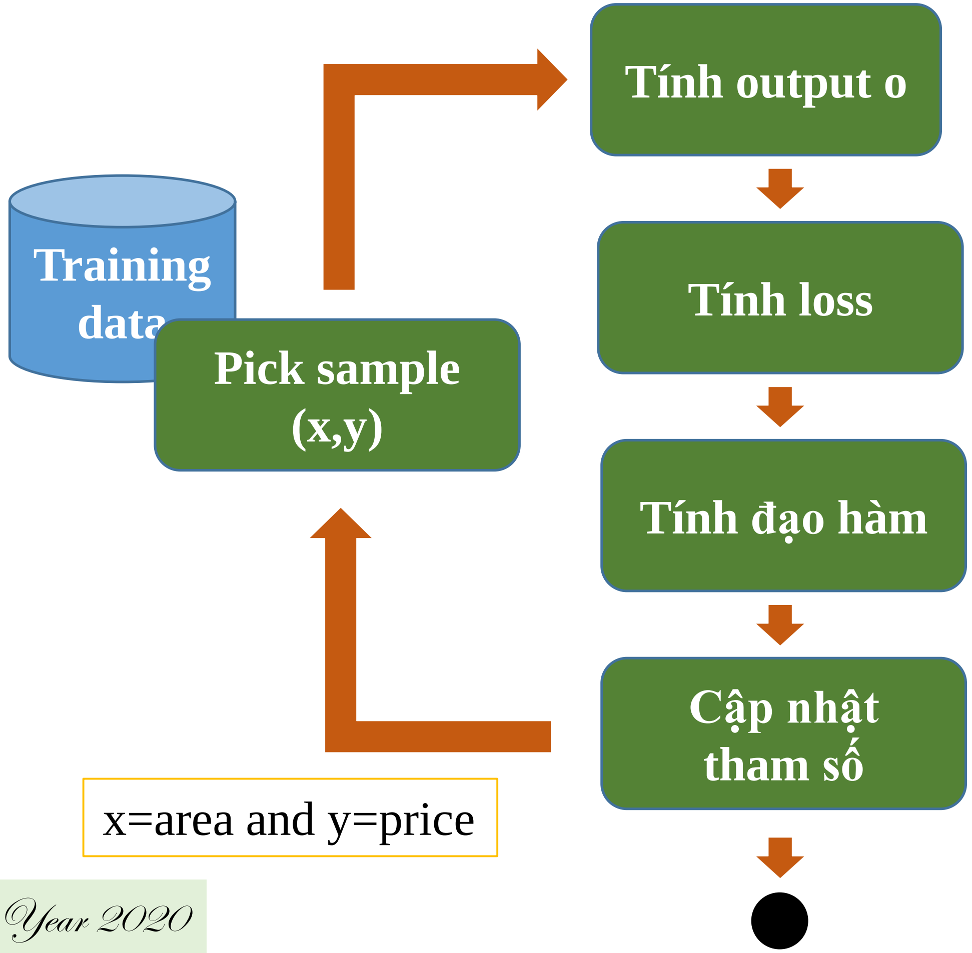


How to compute (1) using computational graph

Computational graph

❖ House price prediction

❖ One-sample training



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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$
$$L'_b = 2(o - y)$$

5) Cập nhật tham số

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

Learning rate η

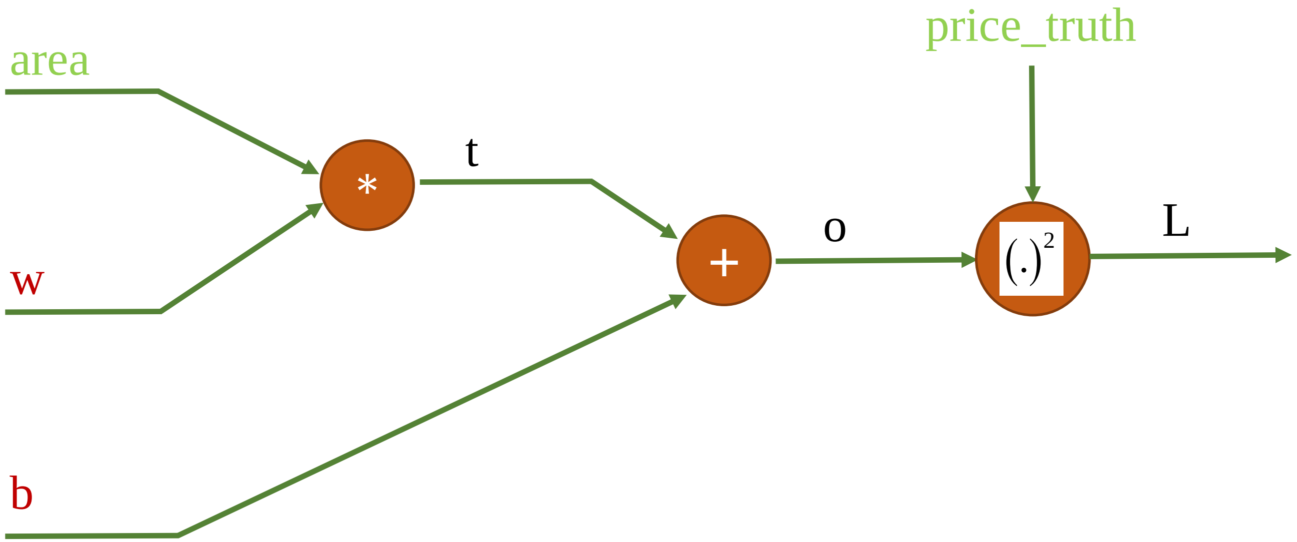
Computational graph

❖ House price prediction

❖ One-sample training

$$price = w * area + b$$

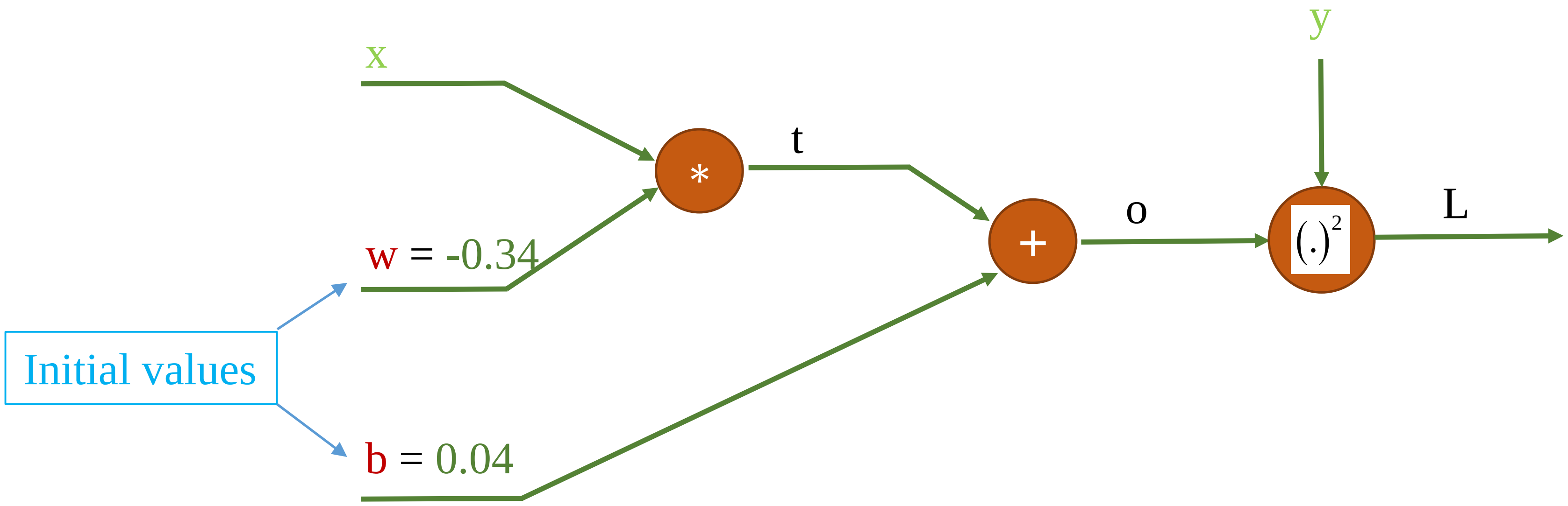
$$t = w * area$$



Computational graph

- **House price prediction**
  - **One-sample training**

$t = wx$ $\quad o = t + b$ $\quad L = (o - y)^2$



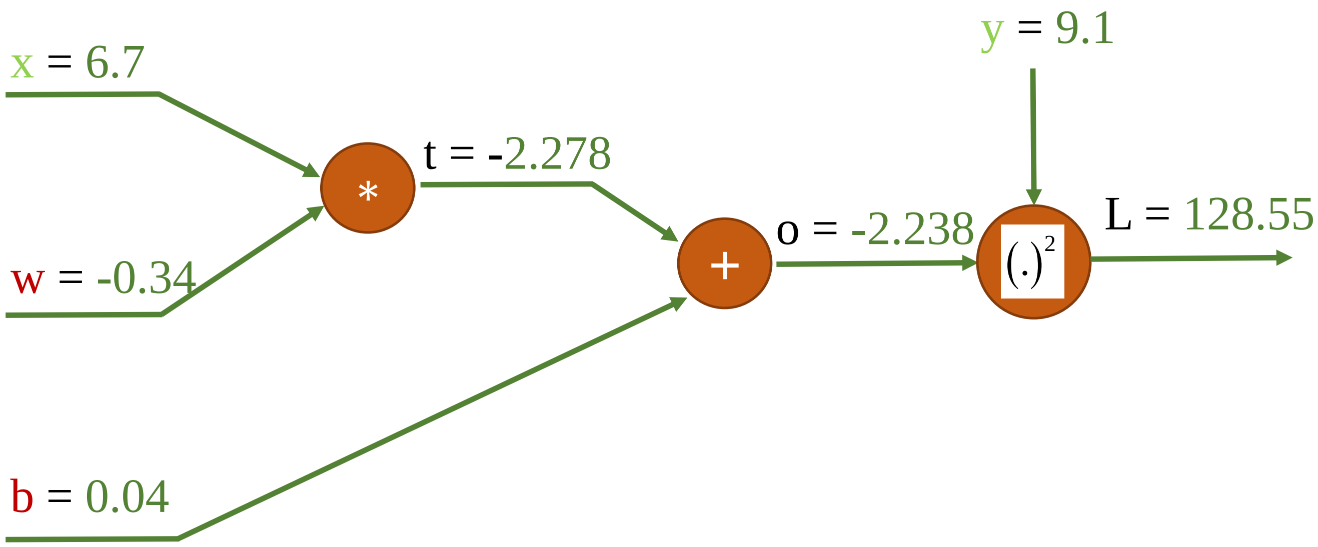
Computational graph

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

❖ House price prediction

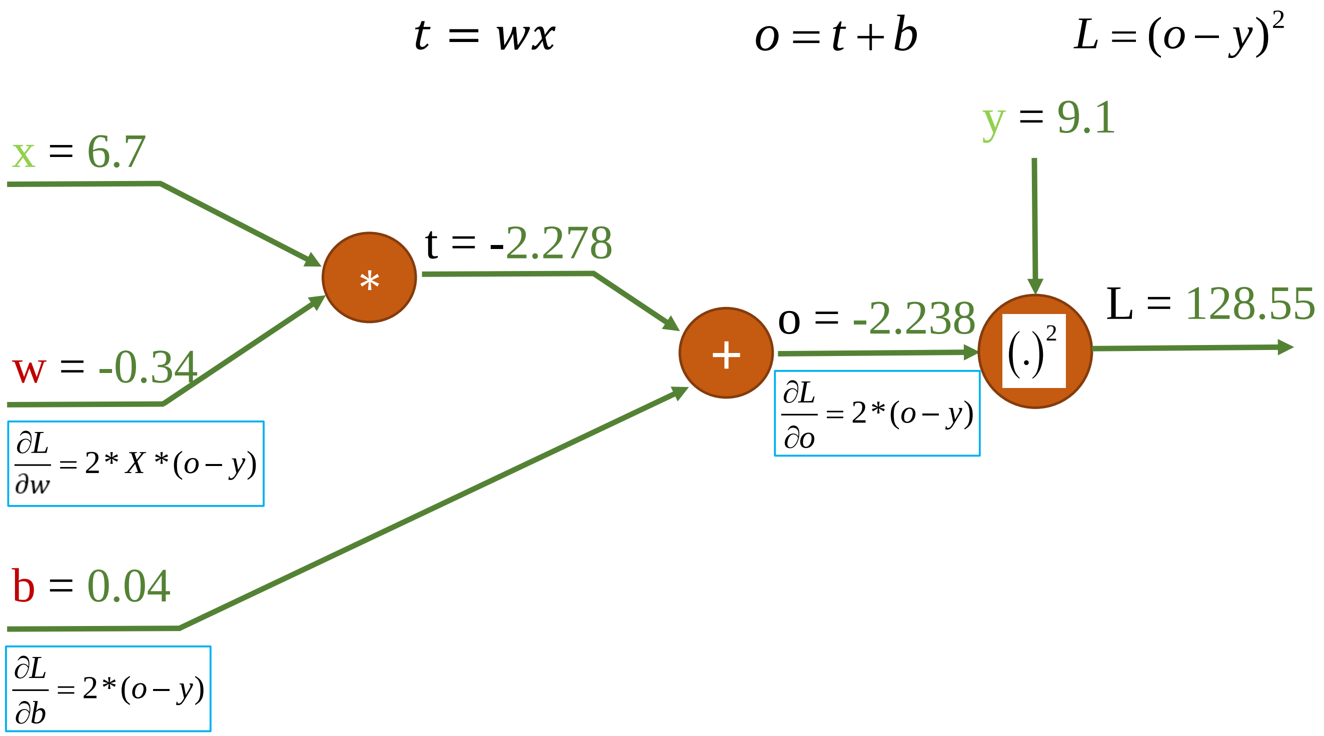
❖ One-sample training

$t = wx \qquad o = t + b \qquad L = (o - y)^2$



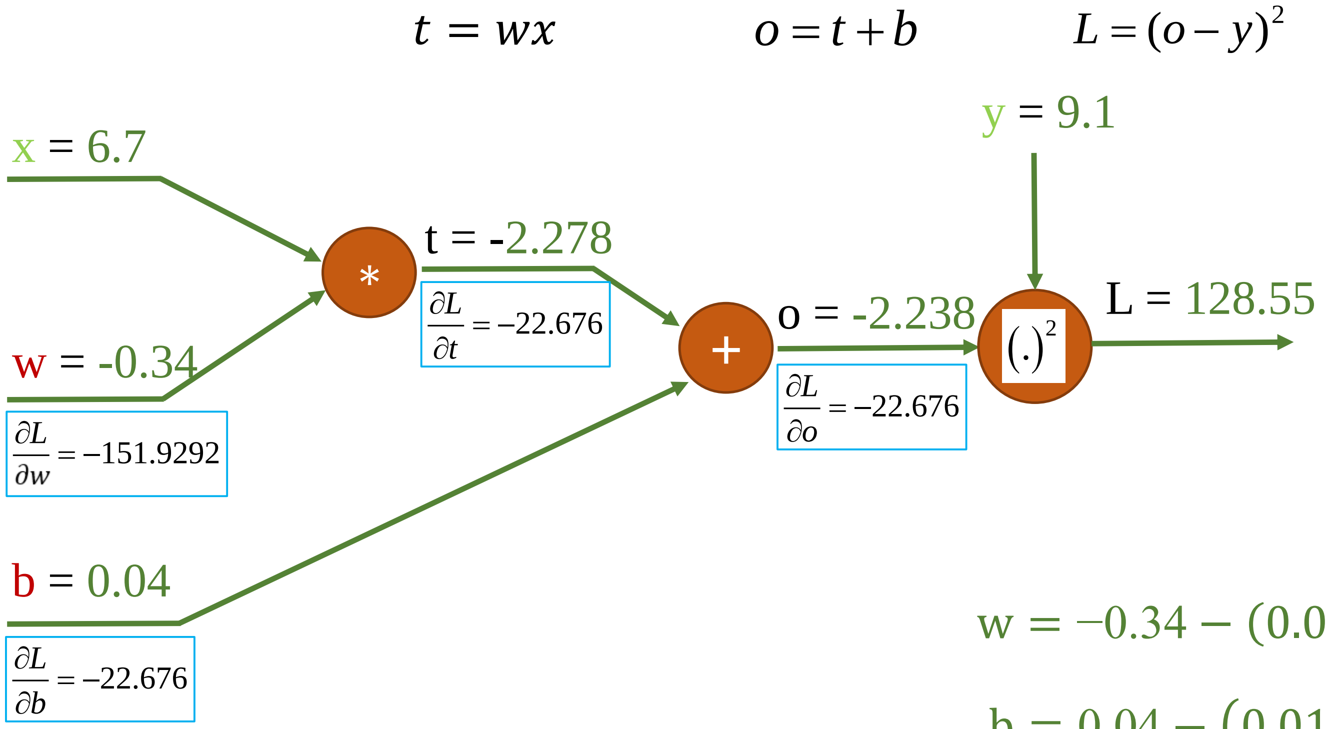
Computational graph

- **House price prediction**
  - **One-sample training**



Computational graph

- **House price prediction**
  - **One-sample training**



Cách cập nhật a và b

$$w = w - \eta * \frac{\partial L}{\partial w}$$

$$b = b - \eta * \frac{\partial L}{\partial b}$$

Learning rate $\eta = 0.01$

$$w = -0.34 - (0.01 * (-151.9)) = 1.179$$

$$b = 0.04 - \left(0.01 * (-22.67)\right) = 0.266$$

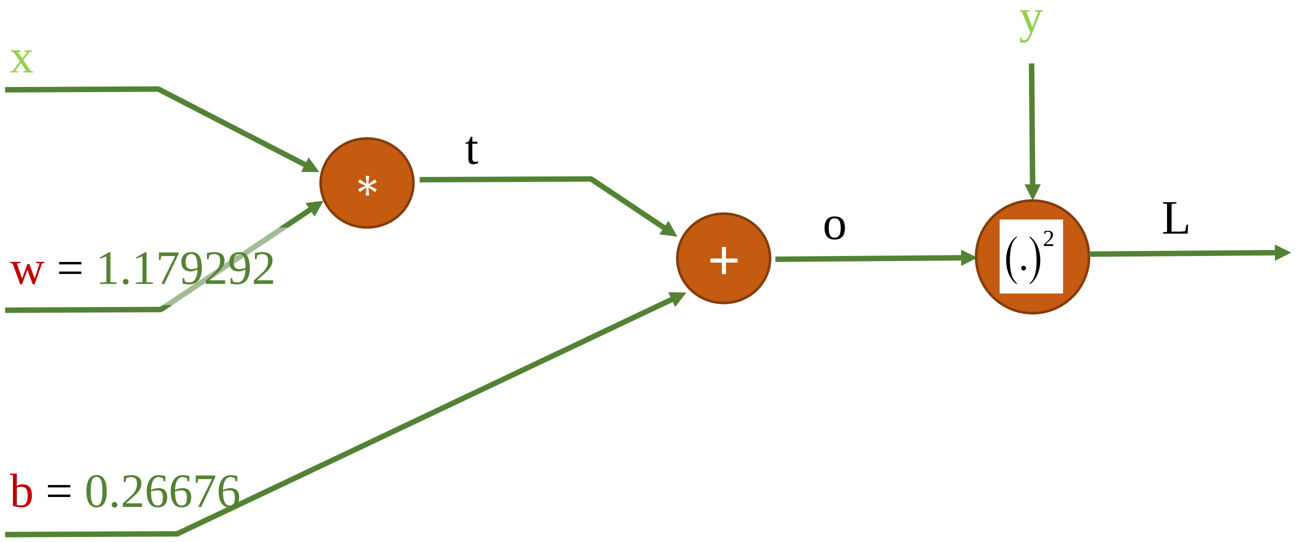
Computational graph

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

❖ House price prediction

❖ One-sample training

$t = wx$ $\qquad o = t + b$ $\qquad L = (o - y)^2$

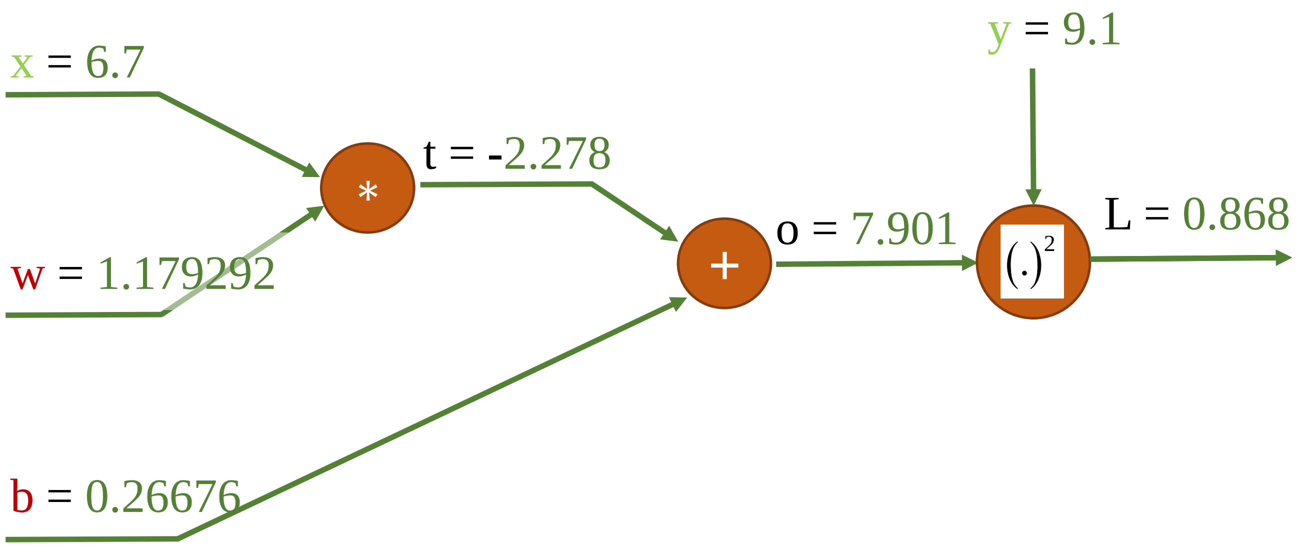


Computational graph

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

- House price prediction
  - One-sample training

$t = wx$ $o = t + b$ $L = (o - y)^2$



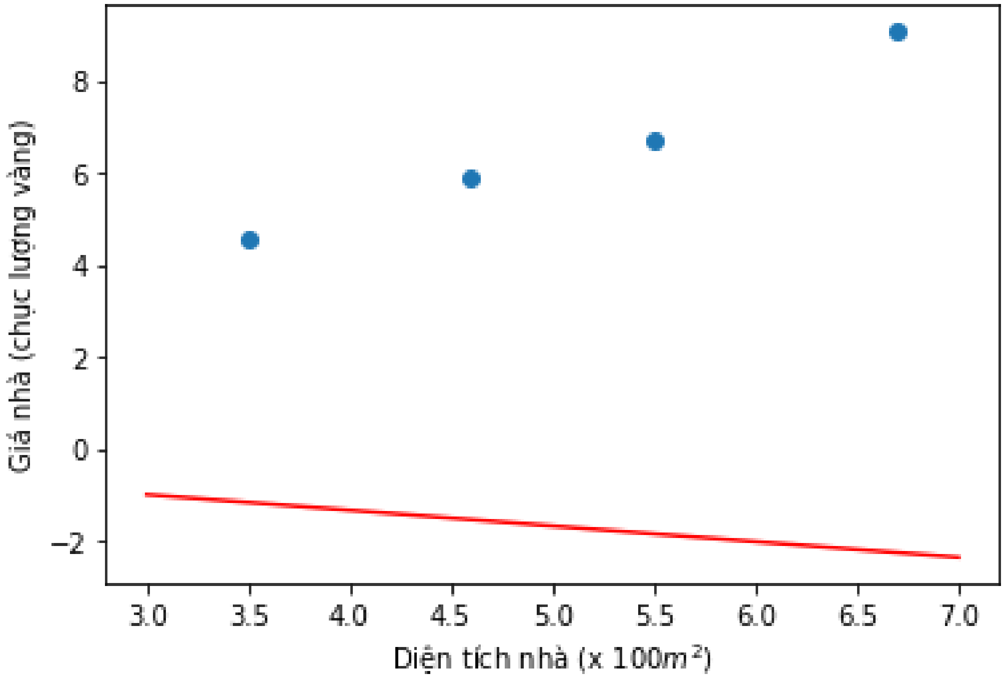
previous L = 128.55

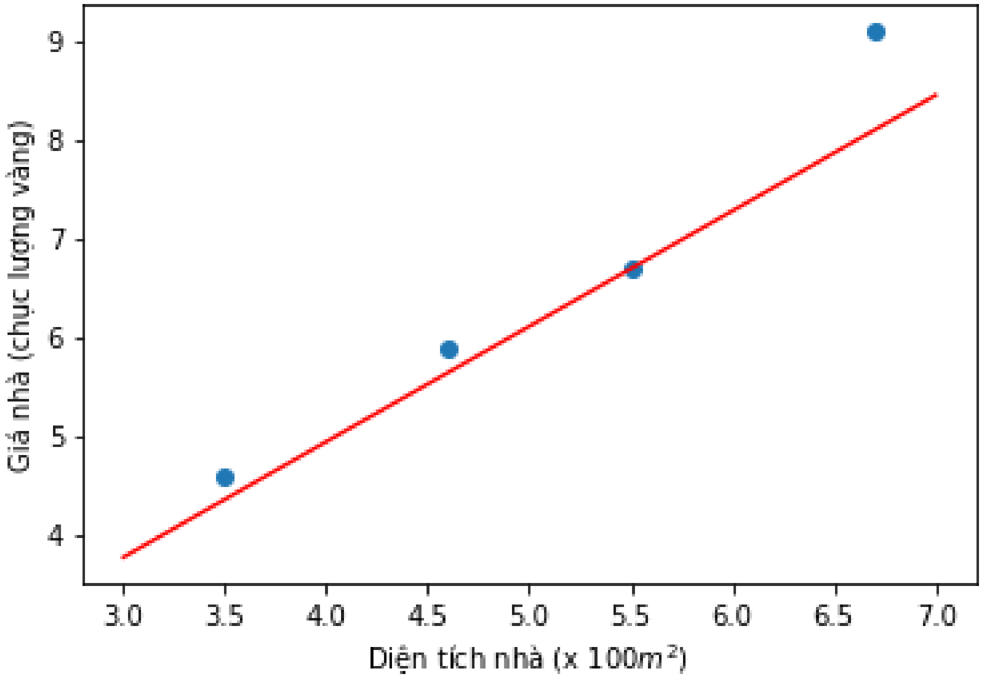
Updated a and b values help to reduce the L value

Computational graph

❖ **House price prediction**

❖ **One-sample training**



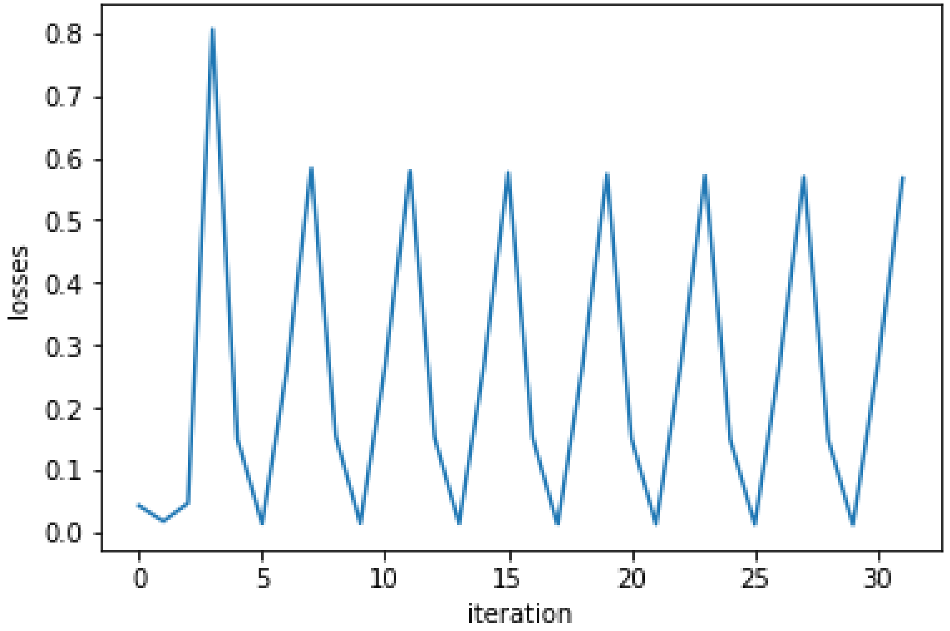


w = -0.34 b = 0.04 L = 128.55

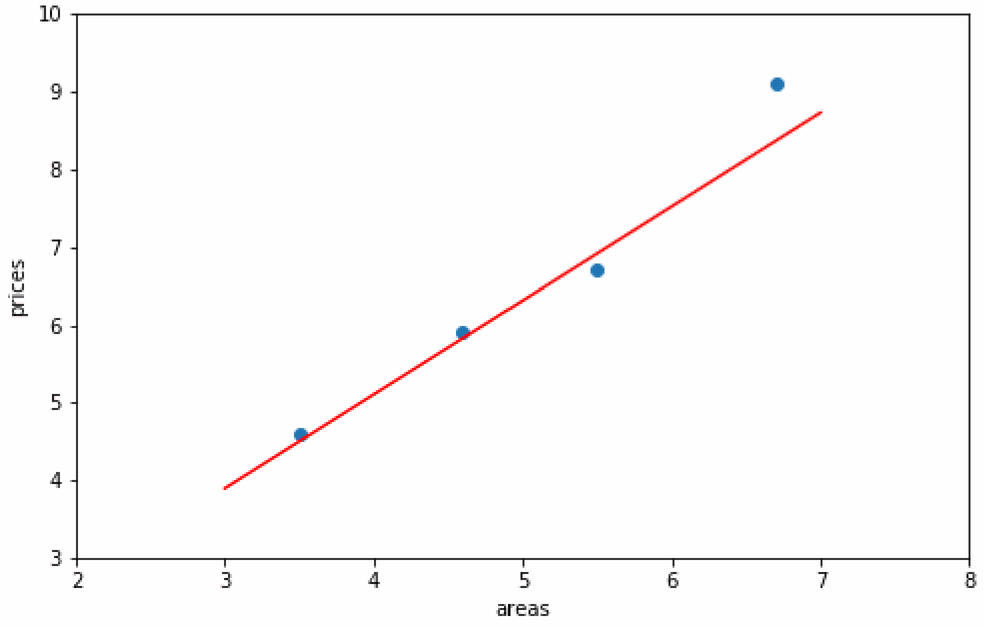
w = 1.179292 b = 0.26676 L = 0.868

Computational graph

- **House price prediction**
  - **One-sample training**



Losses for 30 iterations



Model updating for different iterations

Linear Regression (1-sample)

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

5) Cập nhật tham số

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

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

η is learning rate

Friendly version

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

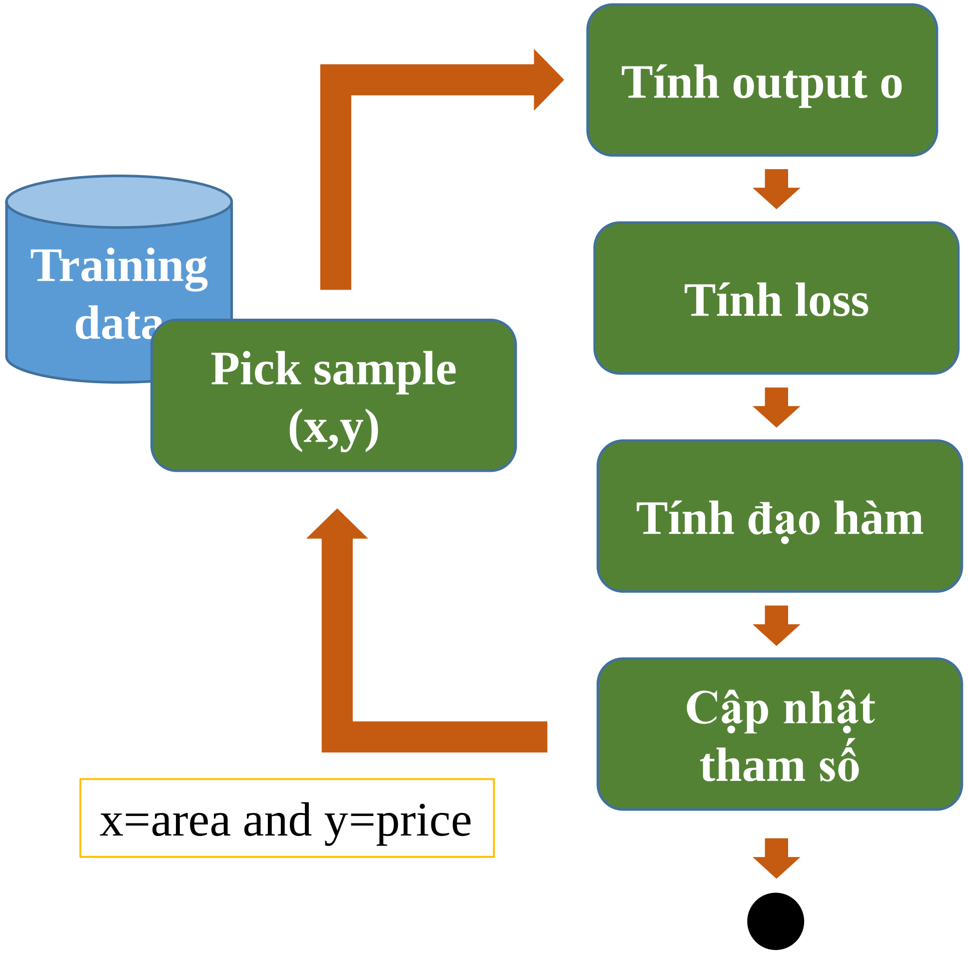
η is learning rate

Generalized formula

Computational graph

❖ House price prediction

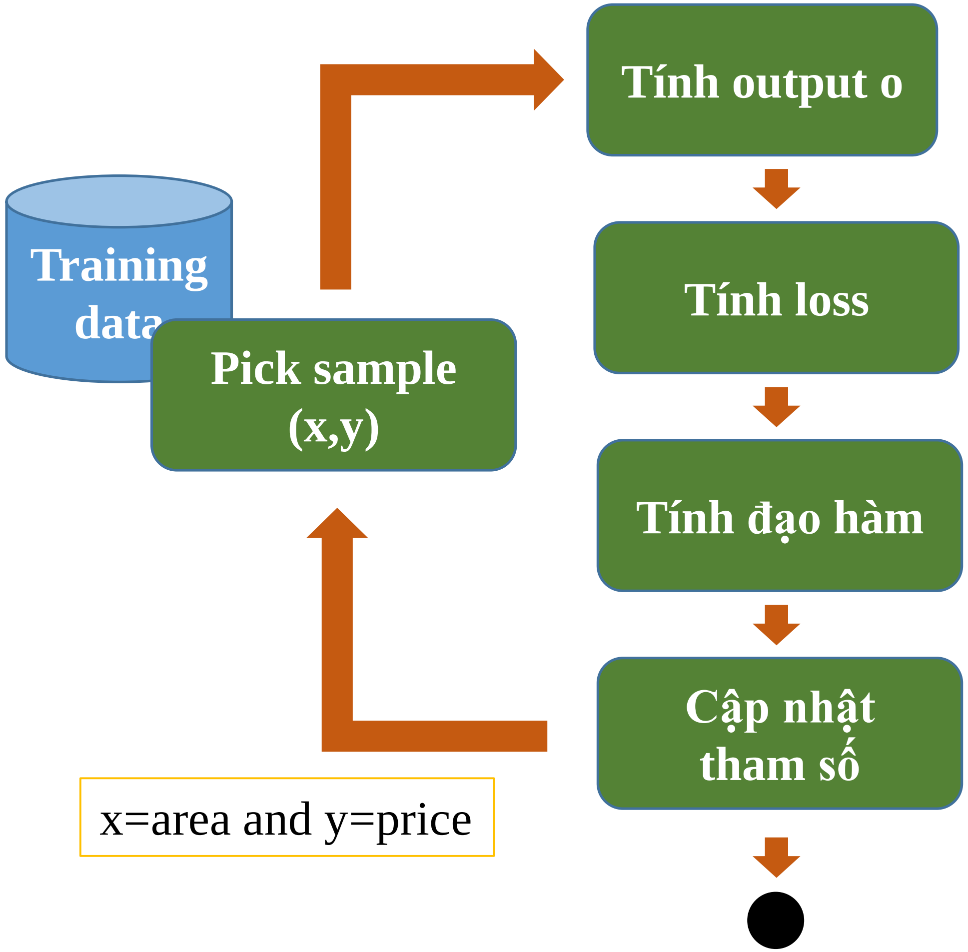
❖ One-sample training: implementation




```python
# Naive implementaion

# load data
from numpy import genfromtxt
import matplotlib.pyplot as plt

data = genfromtxt('data.csv', delimiter=',')
areas  = list(data[:,0])
prices = list(data[:,1])
data_size = len(areas)

print('areas: ', areas)
print('prices: ', prices)
print('data_size: ', data_size)
```

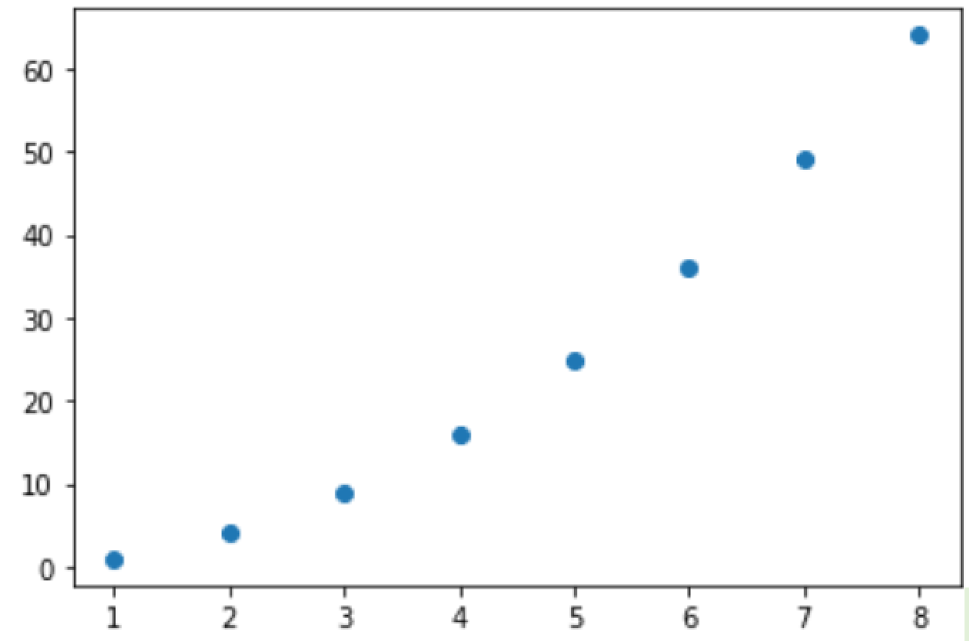
Computational graph

- House price prediction
  - One-sample training: implementation




```python
import matplotlib.pyplot as plt

x_values = [1, 2, 3, 4, 5, 6, 7, 8]
y_values = [x*x for x in x_values]

print('x_values: ', x_values)
print('y_values: ', y_values)

plt.scatter(x_values, y_values)
plt.show()
```

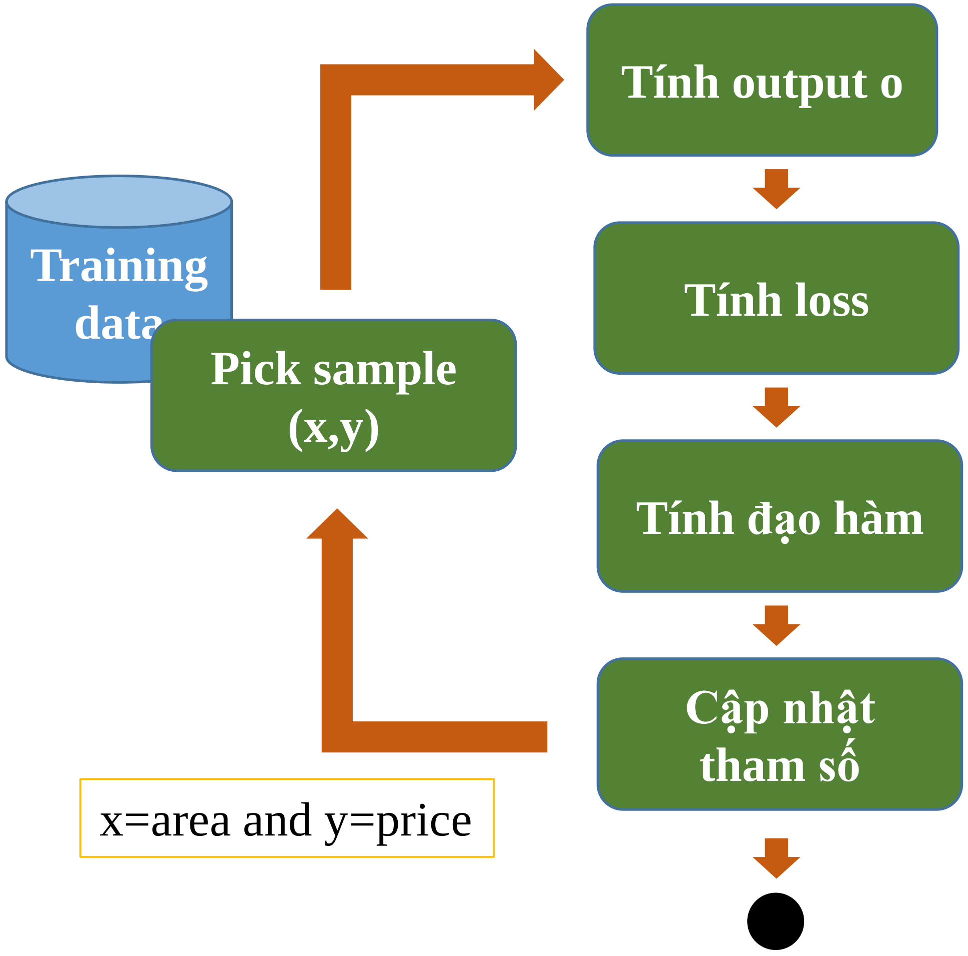
```
x_values:  [1, 2, 3, 4, 5, 6, 7, 8]
y_values:  [1, 4, 9, 16, 25, 36, 49, 64]
```



Computational graph

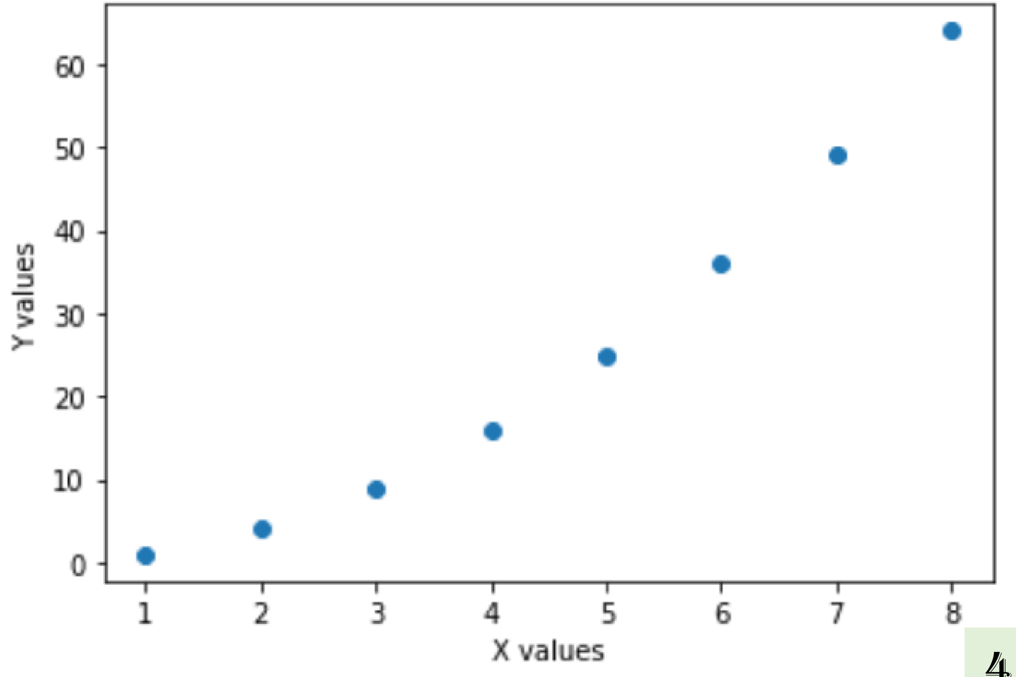
❖ House price prediction

❖ One-sample training: implementation




```python
import matplotlib.pyplot as plt

x_values = [1, 2, 3, 4, 5, 6, 7, 8]
y_values = [x*x for x in x_values]

print('x_values: ', x_values)
print('y_values: ', y_values)

plt.scatter(x_values, y_values)
plt.xlabel('X values')
plt.ylabel('Y values')
plt.show()
```

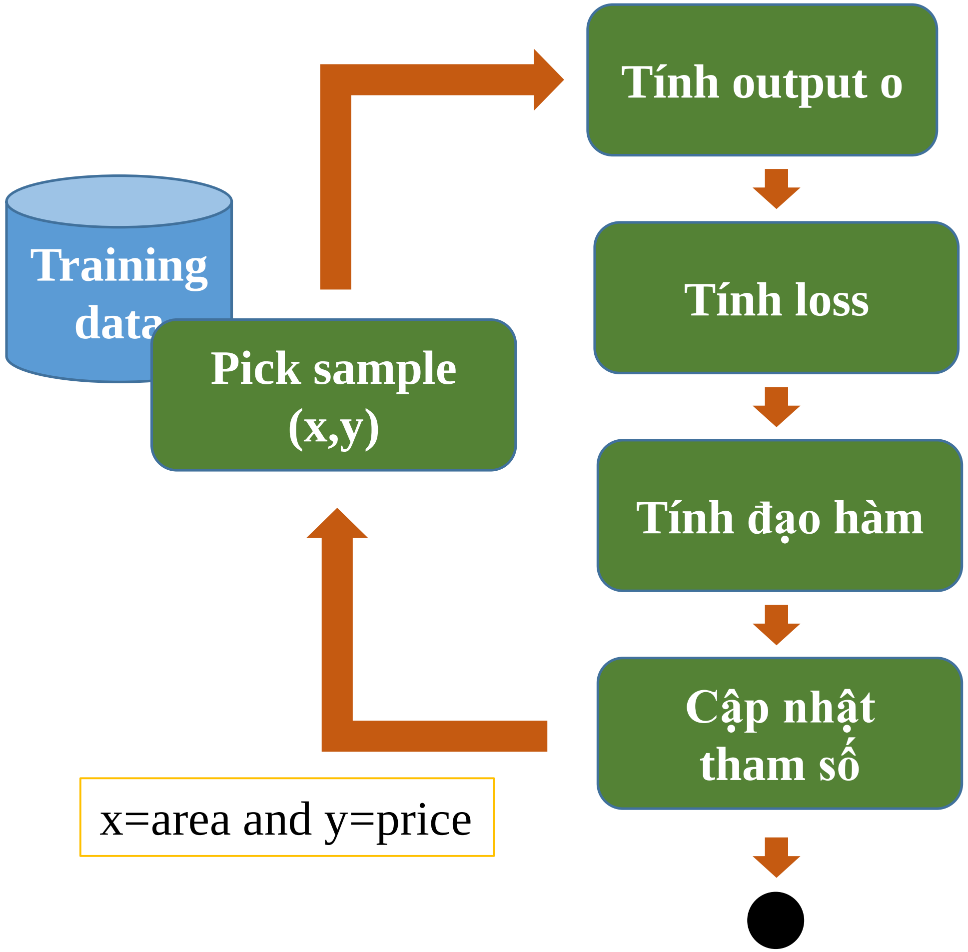
```
x_values:  [1, 2, 3, 4, 5, 6, 7, 8]
y_values:  [1, 4, 9, 16, 25, 36, 49, 64]
```



Computational graph

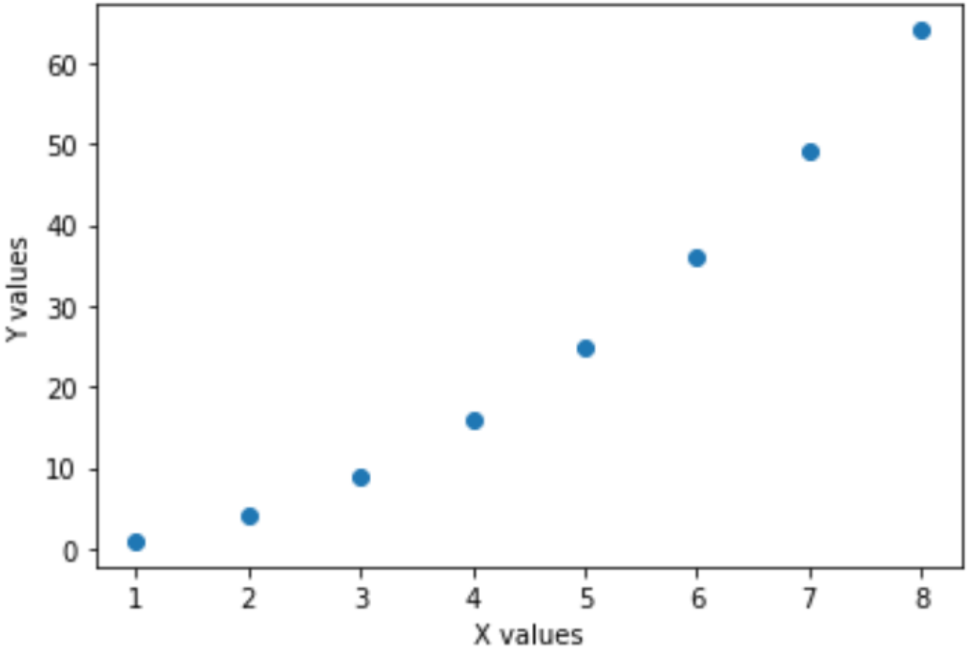
❖ **House price prediction**

❖ **One-sample training: implementation**



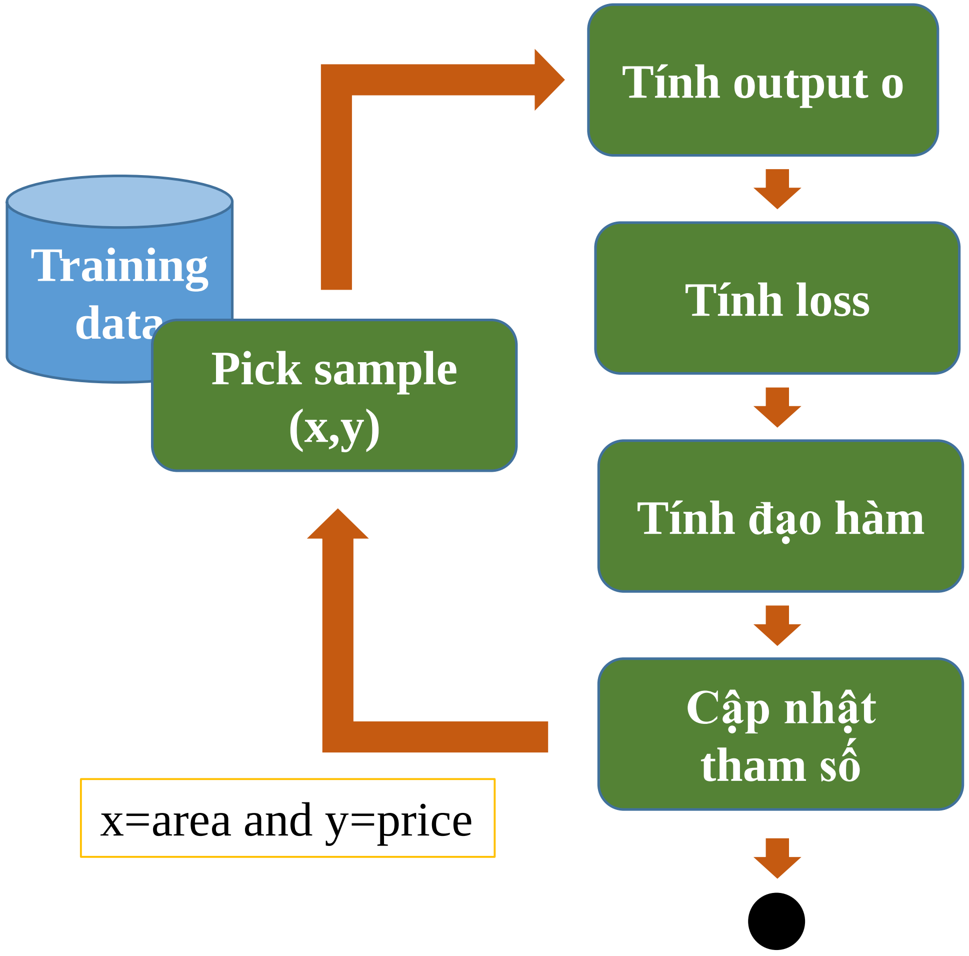

```python
import matplotlib.pyplot as plt
import numpy as np

x_values = np.array([1, 2, 3, 4, 5, 6, 7, 8])
y_values = x_values*x_values

print('x_values: ', x_values)
print('y_values: ', y_values)

plt.scatter(x_values, y_values)
plt.xlabel('X values')
plt.ylabel('Y values')
plt.show()
```

```
x_values:  [1 2 3 4 5 6 7 8]
y_values:  [ 1  4  9 16 25 36 49 64]
```



Computational graph

- **House price prediction**
  - **One-sample training: implementation**




```python
# Naive implementaion

# load data
from numpy import genfromtxt
import matplotlib.pyplot as plt

data = genfromtxt('data.csv', delimiter=',')
areas  = list(data[:,0])
prices = list(data[:,1])
data_size = len(areas)

print('areas: ', areas)
print('prices: ', prices)
print('data_size: ', data_size)

plt.scatter(areas, prices)
plt.xlabel('Diện tích nhà (x 100$m^2$)')
plt.ylabel('Giá nhà (chục lượng vàng)')
plt.xlim(3,7)
plt.ylim(4,10)
plt.show()
```

Computational graph

❖ House price prediction

❖ One-sample training: implementation

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

5) Cập nhật tham số

η is learning rate

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

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

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

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

    return (dw, db)

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

    return (w_new, b_new)
```

Computational graph

❖ House price prediction

❖ One-sample training: implementation

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

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
n = 0.01

# how long
epoch_max = 10

for epoch in range(epoch_max):
    for i in range(data_size):
        # get a sample

        # predict z

        # compute loss

        # compute gradient

        # update weights

```

Computational graph

❖ House price prediction

❖ One-sample training: implementation

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

5) Cập nhật tham số

η is learning rate

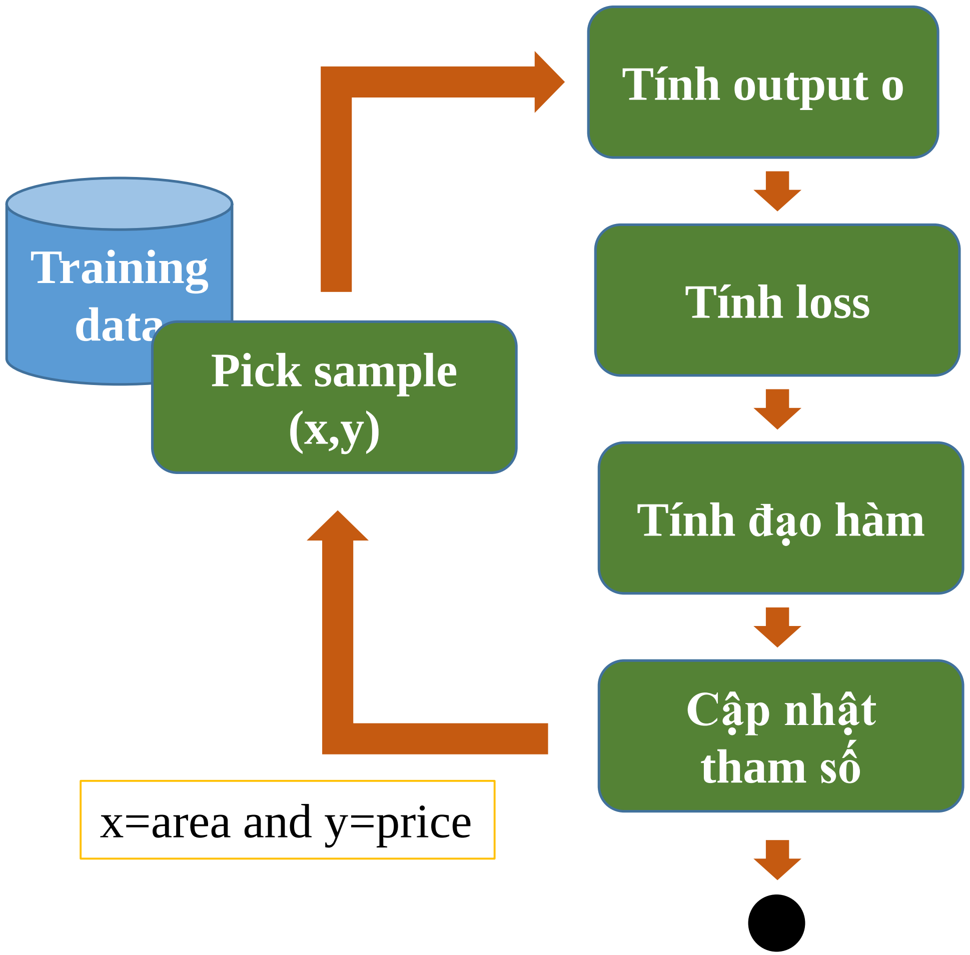
$$w = w - \eta L'_w$$

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

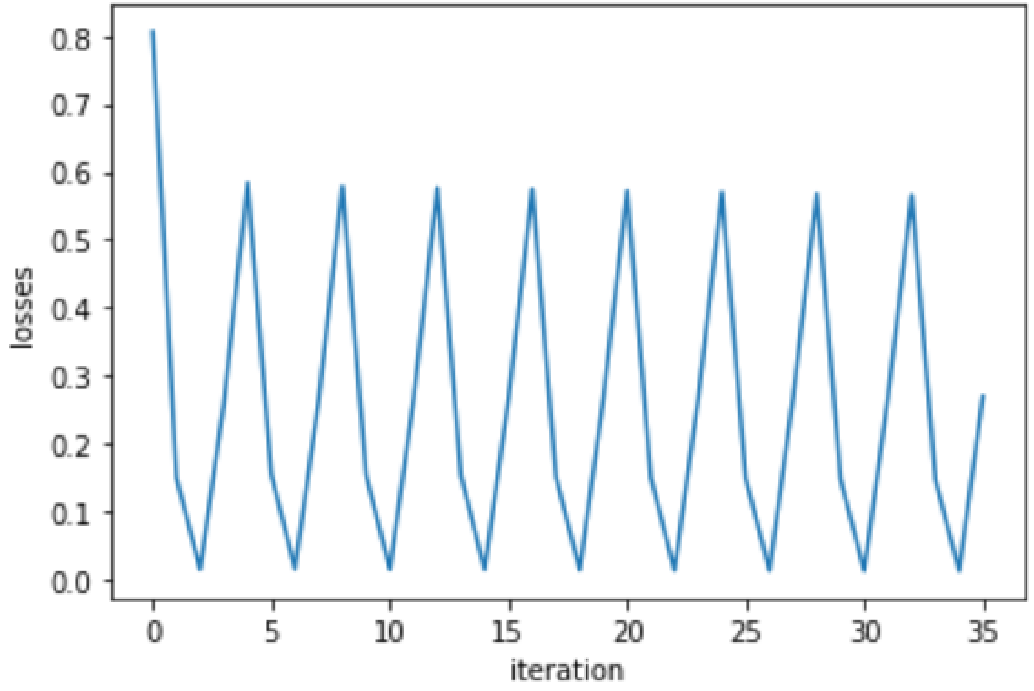
```python
# init weights
b = 0.04
w = -0.34
n = 0.01

# how long
epoch_max = 10

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

        # predict z
        z = predict(x,w,b)
        print('z: ', z)

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

        # compute gradient
        (dw, db) = gradient(z,y,x)
        print('dw: ', dw)
        print('db: ', db)

        # update weights
        (w, b) = update_weight(w,b,n,dw,db)
        print('w_new: ', w)
        print('b_new: ', b)
        print('\n\n')
```

Computational graph

- House price prediction
  - One-sample training: implementation




```python
import matplotlib.pyplot as plt

plt.plot(losses)
plt.xlabel('iteration')
plt.ylabel('losses')
plt.show()
```



Linear Regression

❖ House price prediction

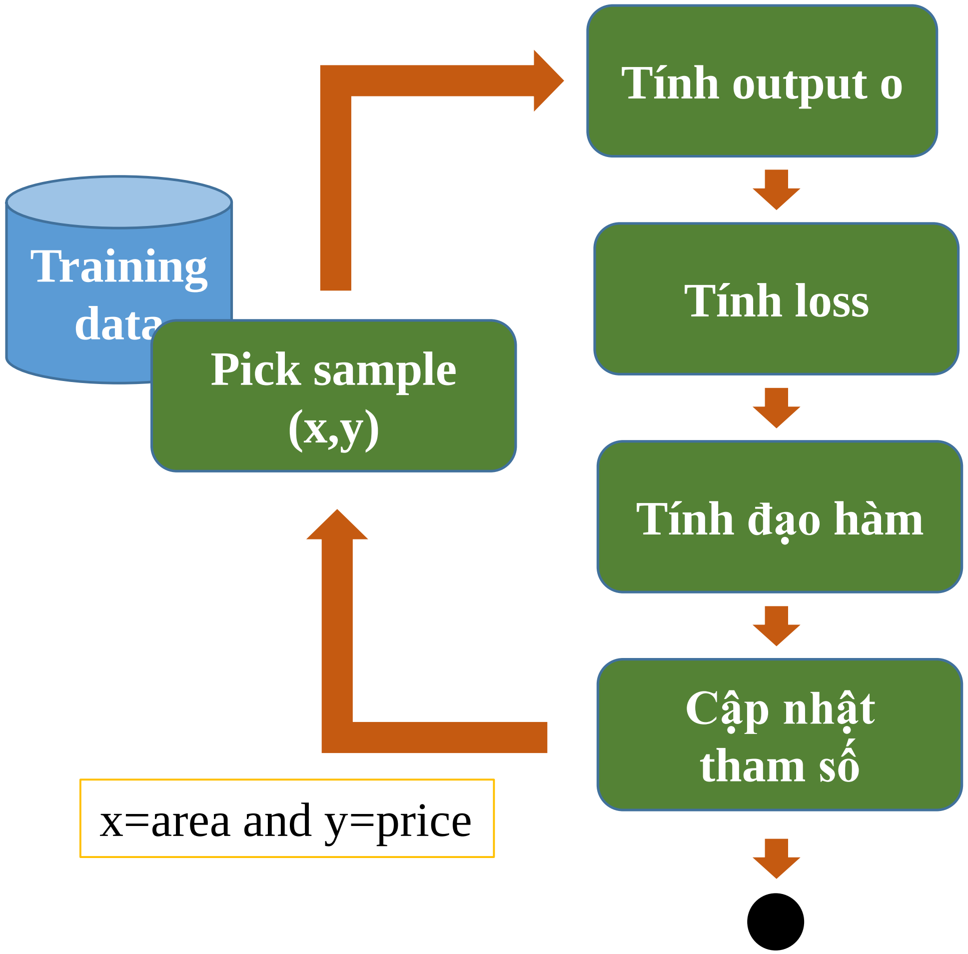
Demo

```
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)] ::
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> for epoch in range(n_epochs):
...     sum_of_losses = 0
...     gradients = np.zeros((2,1))
...
...     for index in range(4):
...         xi = X_b[index:index+1]
...         yi = y[index:index+1]
```

Computational graph

❖ **House price prediction**

❖ **Implementation: Vectorization**




```python
# Implementation - vectorization

# Load data
import numpy as np
from numpy import genfromtxt
import matplotlib.pyplot as plt

data = genfromtxt('data.csv', delimiter=',')
areas  = data[:,0]
prices = data[:,1]
data_size = areas.size

print(type(areas))
print('areas: ', areas)
print('prices: ', prices)
print('data_size: ', data_size)

plt.scatter(areas, prices)
plt.xlabel('Diện tích nhà (x 100$m^2$)')
plt.ylabel('Giá nhà (chục lượng vàng)')
plt.xlim(3,7)
plt.ylim(4,10)
plt.show()
```

Computational graph

❖ House price prediction

❖ Implementation: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

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

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

    return dtheta

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

    return dtheta_new
```

Computational graph

❖ House price prediction

❖ Implementation: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

```python
# vector [x, b]
data = np.c_[areas, np.ones((data_size, 1))]
print(data)

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

Computational graph

❖ House price prediction

❖ Implementation: Vectorization

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

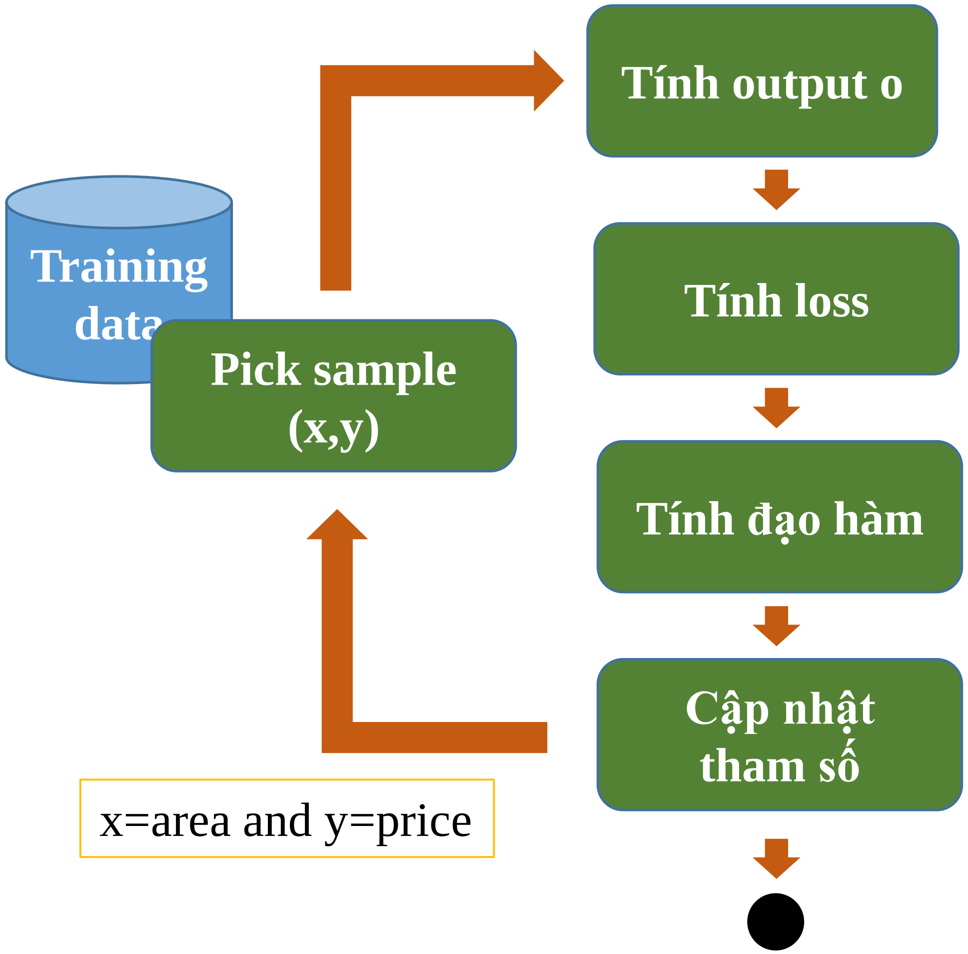
$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

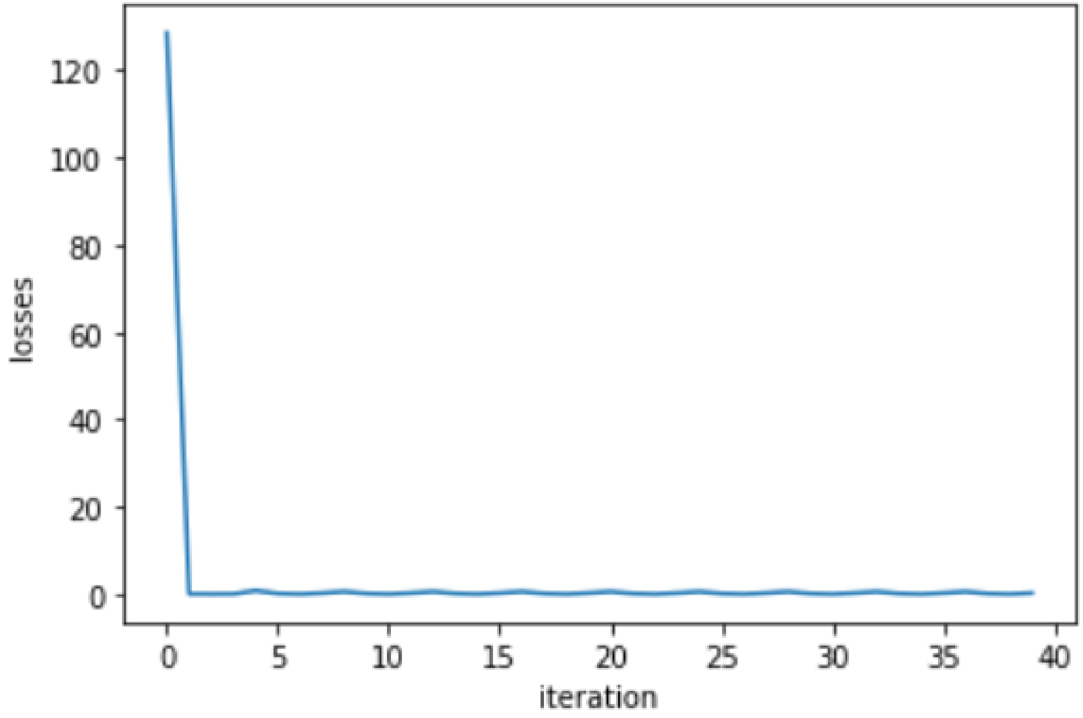
```python
# how long
epoch_max = 10

for epoch in range(epoch_max):
    for i in range(data_size):
        # get a sample
        x = data[i]
        y = prices[i:i+1]
        print('sample: ', x, y)

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

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

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

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

Computational graph

- ❖ **House price prediction**
  - ❖ **Implementation: Vectorization**




```python
import matplotlib.pyplot as plt

plt.plot(losses)
plt.xlabel('iteration')
plt.ylabel('losses')
plt.show()
```



Linear Regression

❖ House price prediction

Demo

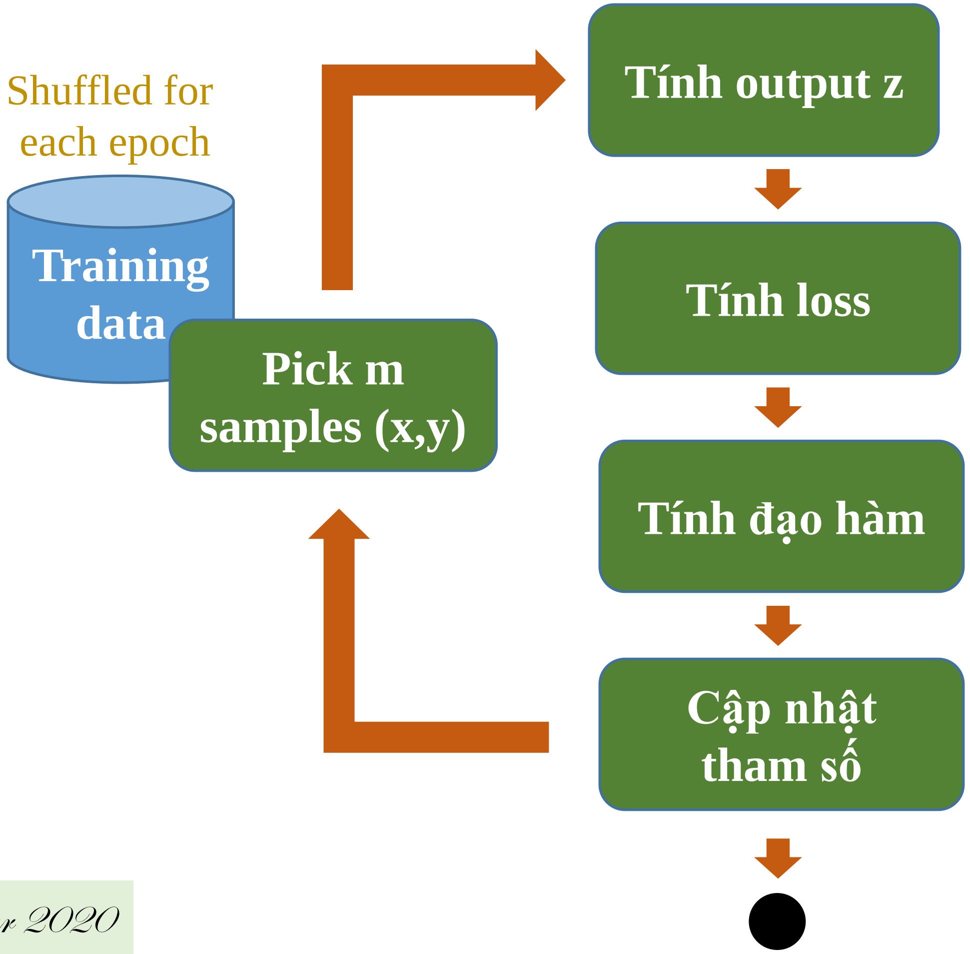
```
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)] ::
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> for epoch in range(n_epochs):
...     sum_of_losses = 0
...     gradients = np.zeros((2,1))
...
...     for index in range(4):
...         xi = X_b[index:index+1]
...         yi = y[index:index+1]
```

Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
  - 1-sample training
  - m-sample training
- Summary

Computational graph

- **House price prediction**
  - **m-sample training (1<m<N)**



1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

2) Tính output o_i

$$o^{(i)} = wx^{(i)} + b \quad \text{for } 0 \le i < m$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \le i < m$$

4) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \quad \text{for } 0 \le i < m$$

5) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{m}$$

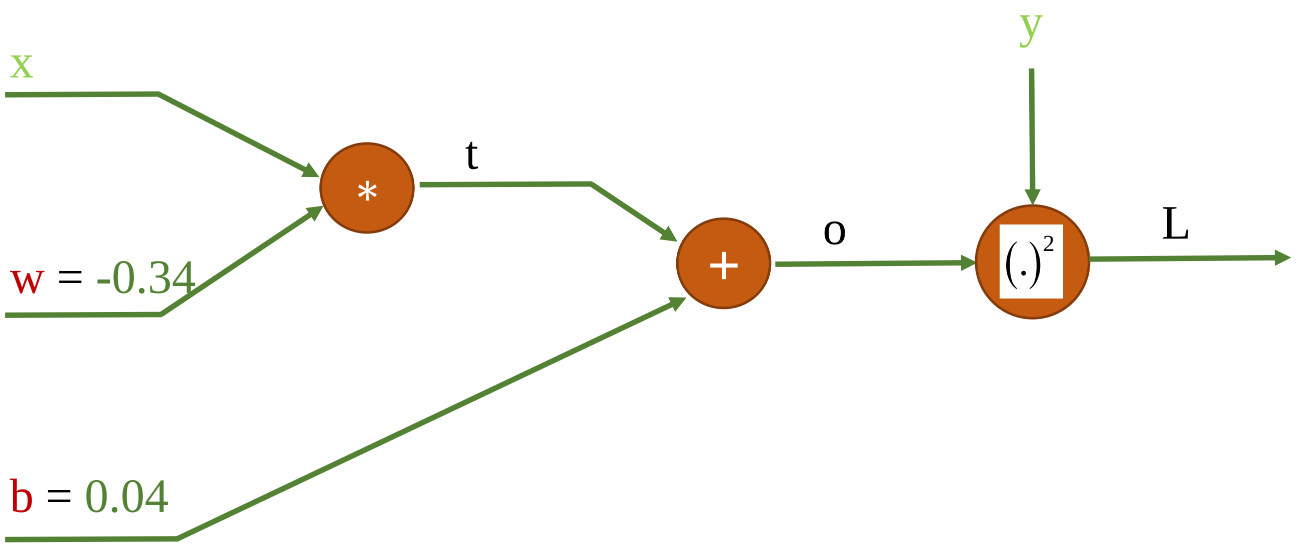
$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{m}$$

Learning rate η

Computational graph

- **House price prediction**
  - **m-sample training (1<m<N)**

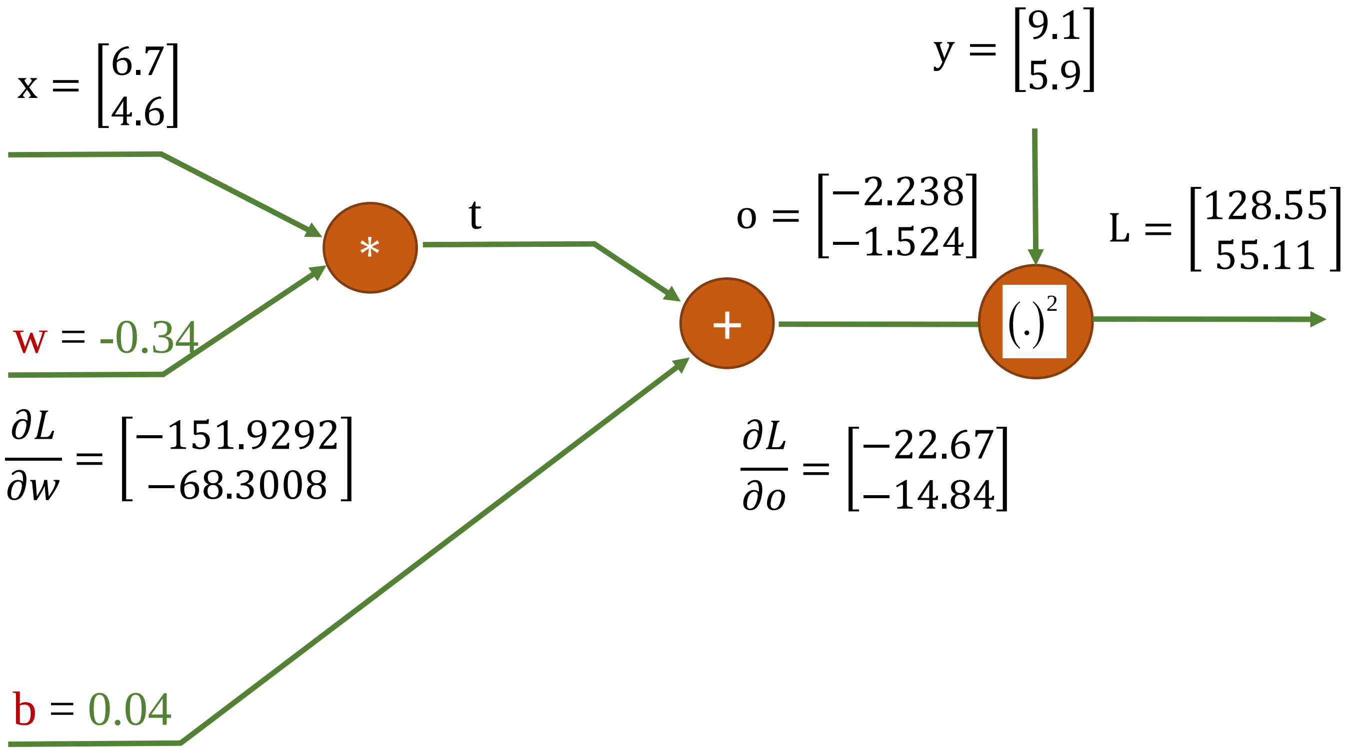
$t = wx$ $\qquad o = t + b$ $\qquad L = (o - y)^2$



Computational graph

- **House price prediction**
  - **m-sample training (1<m<N)**

m = 2



$$\frac{sum(\frac{\partial L}{\partial w})}{m} = -110.115 \qquad \frac{\partial L}{\partial w} = \begin{bmatrix} -151.9292 \\ -68.3008 \end{bmatrix} \qquad \frac{\partial L}{\partial o} = \begin{bmatrix} -22.67 \\ -14.84 \end{bmatrix}$$

$$\frac{sum(\frac{\partial L}{\partial b})}{m} = -18.762 \qquad \frac{\partial L}{\partial b} = \begin{bmatrix} -22.676 \\ -14.848 \end{bmatrix}$$

$$t = wx \qquad o = t + b \qquad L = (o - y)^2$$

Computational graph

❖ House price prediction

❖ m-sample training (1<m<N)

Cách cập nhật a và b

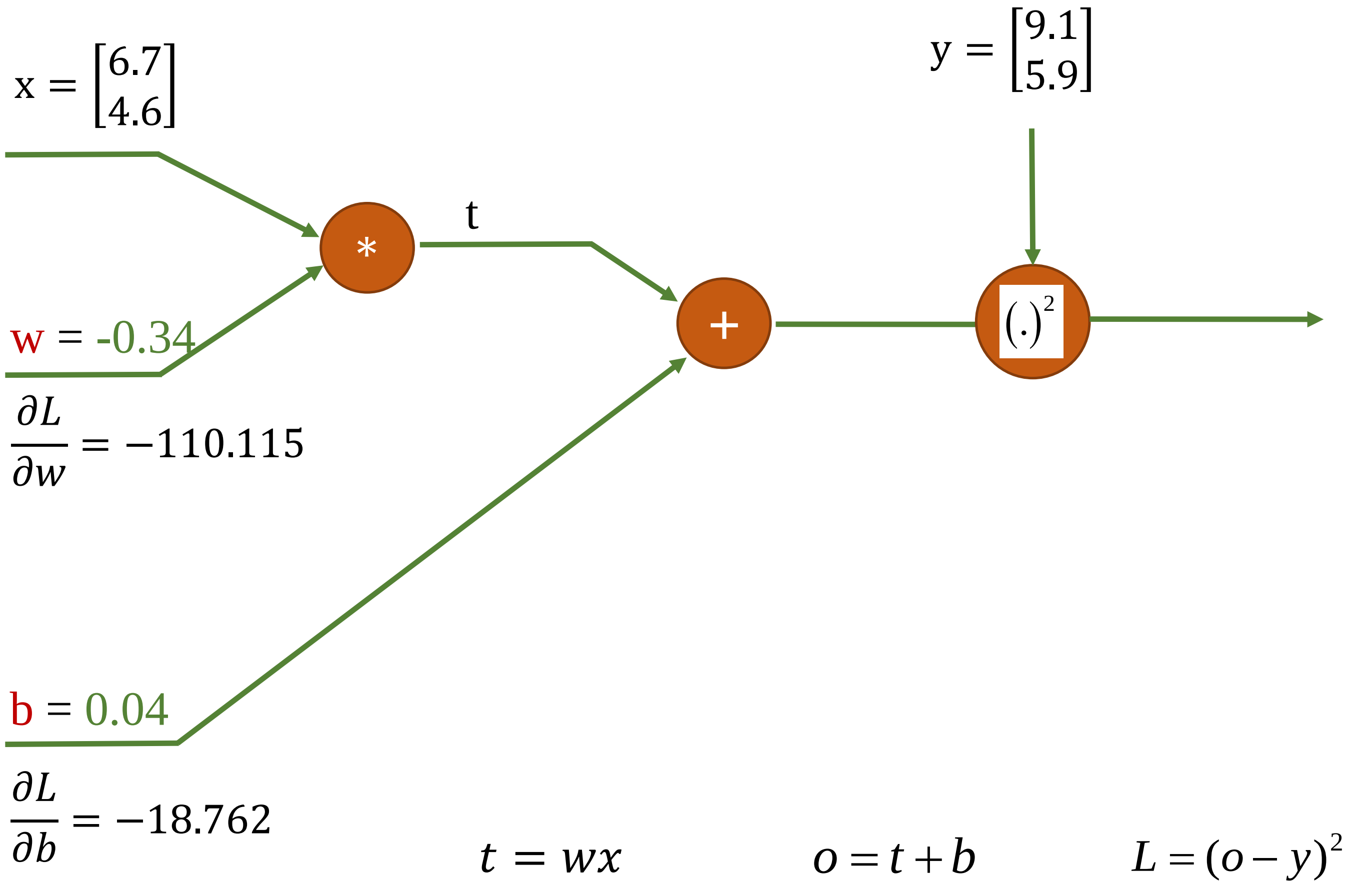
$$w = w - \eta * \frac{\partial L}{\partial w}$$

$$b = b - \eta * \frac{\partial L}{\partial b}$$

Learning rate $\eta = 0.01$

$$\mathrm{w} = -0.34 - (0.01 * (-110.115)) = 0.761$$

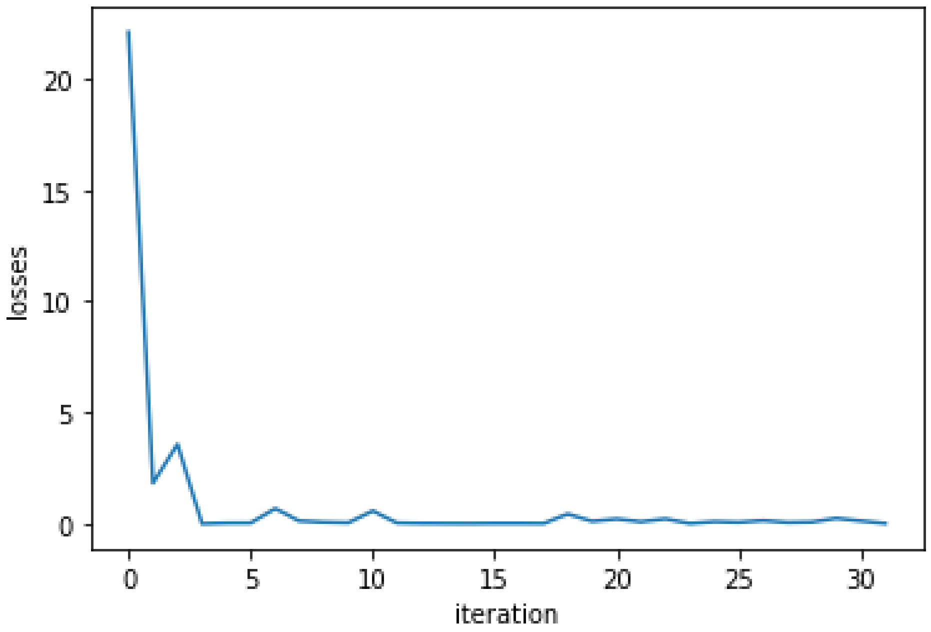
$$\mathrm{b} = 0.04 - \big(0.01 * (-18.762)\big) = 0.227$$



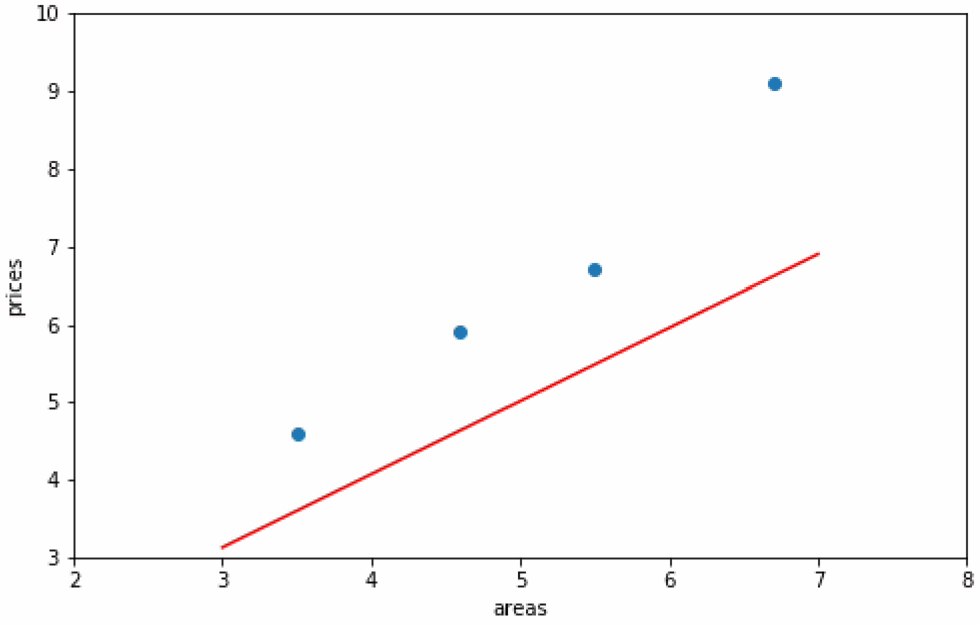
$t = wx$ $o = t + b$ $L = (o - y)^2$

Computational graph

- ❖ **House price prediction**
  - ❖ **m-sample training (1<m<N)**



Losses for 30 iterations



Model updating for different iterations

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = wx^{(i)} + b \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{m}$$
$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{m}$$

η is learning rate

Friendly version

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{m}$$

η is learning rate

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

```python
# load data

import numpy as np
from numpy import genfromtxt
import matplotlib.pyplot as plt

data = genfromtxt('data.csv', delimiter=',')
areas  = data[:,0]
prices = data[:,1]
data_size = areas.size

print(type(areas))
print('areas: ', areas)
print('prices: ', prices)
print('data_size: ', data_size)

plt.scatter(areas, prices)
plt.xlabel('areas')
plt.ylabel('prices')
plt.xlim(3,7)
plt.ylim(4,10)
plt.show()
```

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}^{\prime(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}^{\prime(i)}}{m}$$

η is learning rate

```python
# vector [x, b]
data = np.c_[areas, np.ones((data_size, 1))]
print(data)

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

# how long
epoch_max = 1

# mini-batch size
m = 2
```

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \quad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}^{\prime(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}^{\prime(i)}}{m}$$

η is learning rate

```
for epoch in range(epoch_max):
    sum_of_losses = 0
    gradients = np.zeros((2,))

    for j in range(0, m, m):
        for index in range(j, j+m):
            xi = data[index]
            yi = prices[index]
            print('\ndata: ', xi, yi)

            # predict z/o
            oi = xi.dot(theta)
            print('z: ', oi)

            # compute loss
            li = (oi - yi)*(oi - yi)
            print('loss: ', index, li)

            # compute gradient
            g_li = 2*(oi - yi)
            print('g_li: ', g_li)
            gradient_i = xi*g_li
            print('gradient_i: ', index, gradient_i)

            gradients = gradients + gradient_i
            sum_of_losses = sum_of_losses + li

        sum_of_losses = sum_of_losses/2
        gradients     = gradients/2
        print('\ngradients: ', gradients)

        theta = theta - eta*gradients
        print('new params: ', theta)
```


Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

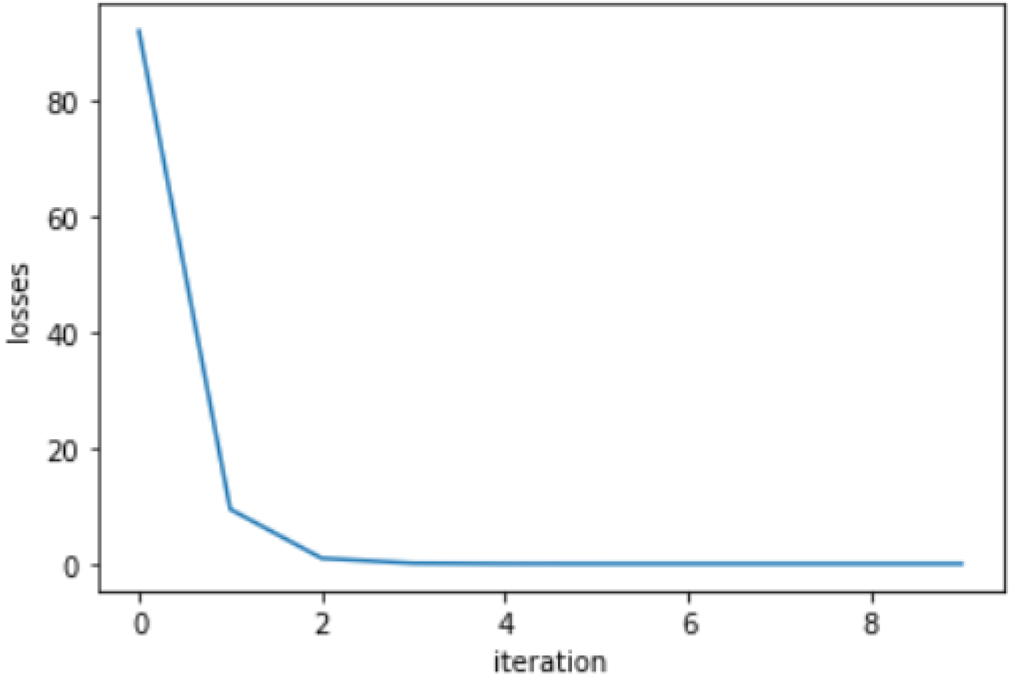
$$L_{\boldsymbol{\theta}}^{\prime(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}^{\prime(i)}}{m}$$

η is learning rate

```python
import matplotlib.pyplot as plt

plt.plot(losses)
plt.xlabel('iteration')
plt.ylabel('losses')
plt.show()
```



Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

More vectorization

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

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$= \begin{bmatrix} -0.34 & 0.049 \end{bmatrix} \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} -2.238 & -1.524 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y})^2$$
$$= ([-2.238 \quad -1.524] - [9.1 \quad 5.9])^2$$
$$= [128.5 \quad 55.11]$$

Generalized formula

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2$$
$$= [128.5 \quad 55.11]$$

$$\boldsymbol{b} = 2(\boldsymbol{o} - \boldsymbol{y}) = 2(\boldsymbol{o} - \boldsymbol{y})$$
$$= [-22.676 \quad -14.848]$$

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}^{\prime(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}^{\prime(i)}}{m}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2$$
$$= [128.5 \quad 55.11]$$

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y}^T) = 2(\boldsymbol{o} - \boldsymbol{y}^T)$$
$$= [-22.676 \quad -14.848]$$

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} = \begin{bmatrix} -22.676 & -14.848 \\ -22.676 & -14.848 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

Generalized formula

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

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2$$
$$= [128.5 \quad 55.11]$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y}^T) = 2(\boldsymbol{o} - \boldsymbol{y}^T)$$
$$= [-22.676 \quad -14.848]$$

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} = \begin{bmatrix} -22.676 & -14.848 \\ -22.676 & -14.848 \end{bmatrix}$$

Gradient for w from $x^{(i)}$

$$\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} = \begin{bmatrix} -151.92 & -68.301 \\ -22.676 & -14.848 \end{bmatrix}$$

Gradient for b from $x^{(i)}$

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m}$$

η is learning rate

Generalized formula

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2 = [128.5 \quad 55.11]$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y}^T) = 2(\boldsymbol{o} - \boldsymbol{y}^T) = [-22.676 \quad -14.848]$$

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} = \begin{bmatrix} -22.676 & -14.848 \\ -22.676 & -14.848 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} = \begin{bmatrix} -151.92 & -68.301 \\ -22.676 & -14.848 \end{bmatrix}$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -151.92 & -68.301 \\ -22.676 & -14.848 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -110.11 \\ -18.762 \end{bmatrix}$$

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{m} \qquad \eta \text{ is learning rate}$$

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2 = [128.5 \quad 55.11]$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y}^T)$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -110.11 \\ -18.762 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

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

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{m}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} 6.7 & 4.6 \\ 1 & 1 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 9.1 \\ 5.9 \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix}$$

$$\boldsymbol{L} = (\boldsymbol{o} - \boldsymbol{y}^T)^2 = [128.5 \quad 55.11]$$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x} = [-2.238 \quad -1.524]$$

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y}^T)$$

$$L_{\boldsymbol{\theta}}' = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -110.11 \\ -18.762 \end{bmatrix}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{L_{\boldsymbol{\theta}}'}{m}$$

$$= \begin{bmatrix} -0.34 \\ 0.049 \end{bmatrix} - 0.005 \begin{bmatrix} -110.11 \\ -18.762 \end{bmatrix} = \begin{bmatrix} 0.761 \\ 0.227 \end{bmatrix}$$

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{m}$$

η is learning rate

Generalized formula

1) Pick m samples $(\boldsymbol{x}, \boldsymbol{y})$ from training data

1.1) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

1.2) Tính loss

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

1.3) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L_{\boldsymbol{\theta}}' = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

1.4) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{L_{\boldsymbol{\theta}}'}{m}$$

η is learning rate

More generalized formula

Linear Regression (m-samples)

1) Pick m samples $(\boldsymbol{x}, \boldsymbol{y})$ from training data

1.1) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

1.2) Tính loss

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

1.3) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

1.4) Cập nhật tham số

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

η is learning rate

```python
import numpy as np
from numpy import genfromtxt

data = genfromtxt('data.csv', delimiter=',')
areas  = data[:,0]
prices = data[:,1:]
data_size = areas.size

# vector [x, b]
data = np.c_[areas, np.ones((data_size, 1))]
data = data.T

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

# how long
epoch_max = 10

# mini-batch size
m = 2
```

More generalized formula

Linear Regression (m-samples)

1) Pick m samples $(\boldsymbol{x}, \boldsymbol{y})$ from training data

1.1) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

1.2) Tính loss

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

1.3) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

1.4) Cập nhật tham số

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

η is learning rate

```
losses = [] # for debug
for epoch in range(epoch_max):
    gradients = np.zeros((2,1))

    for i in range(0, data_size, m):
        # get m samples
        x = data[:, i:i+m]
        y = prices[i:i+m, :]

        # predict z/o
        z = theta.T.dot(x)

        # compute loss
        loss = np.multiply((z-y.T), (z-y.T))
        losses.append(np.mean(loss))

        # compute gradient
        b = 2*(z-y.T)
        gradient = np.multiply(x, np.vstack((b, b)))
        gradient = gradient.dot(np.ones((m, 1)))/m

        # update weights
        theta = theta - eta*gradient
```

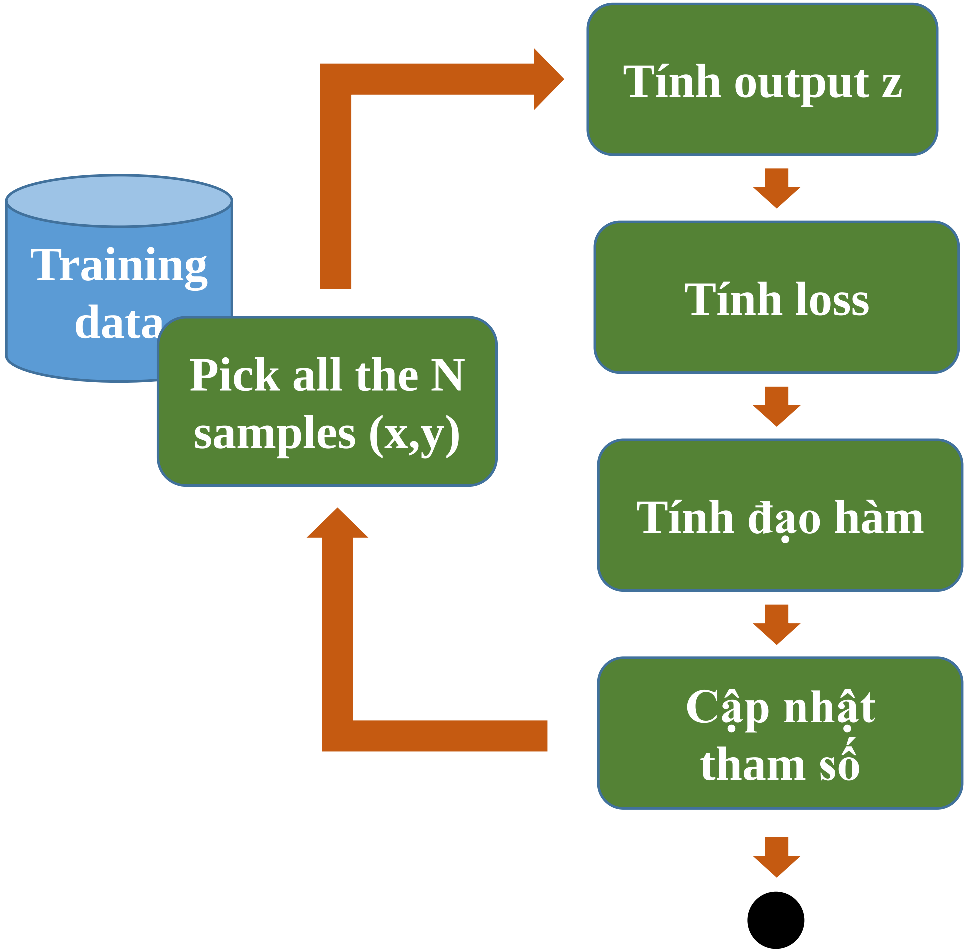

More generalized formula

Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
  - 1-sample training
  - m-sample training
  - N-sample training
- Summary

Computational graph

- **House price prediction**
  - **N-sample training**



1) Pick all the N samples $(x^{(i)}, y^{(i)})$ from training data

2) Tính output o_i

$$o^{(i)} = wx^{(i)} + b \qquad \text{for } 0 \leq i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < N$$

4) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \leq i < N$$

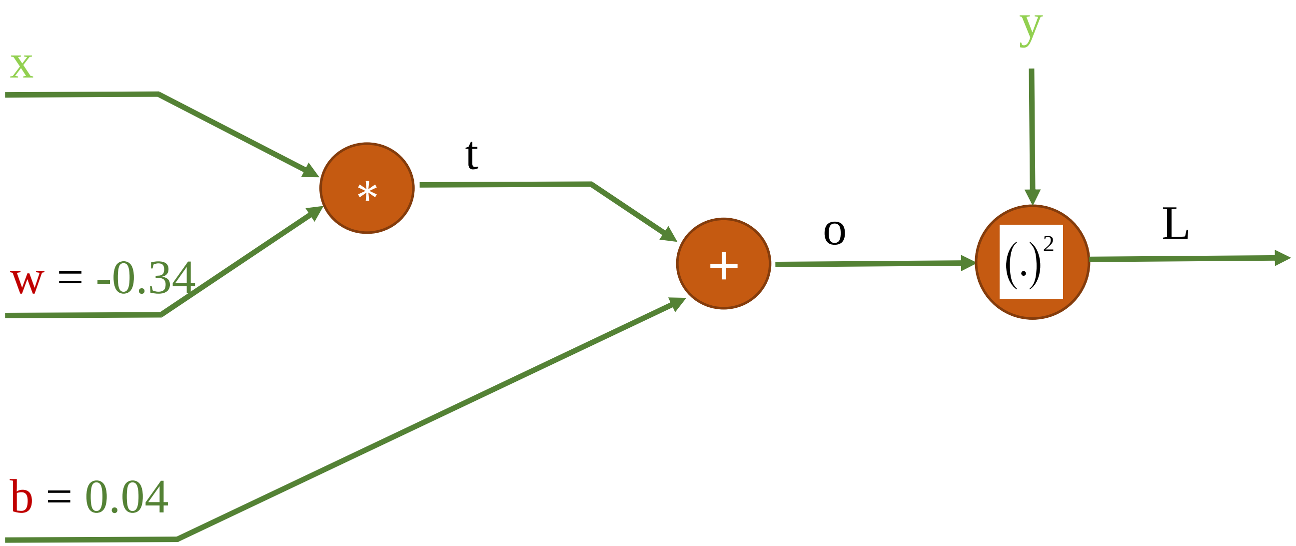
5) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{N}$$
$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{N} \qquad \text{Learning rate } \eta$$

Computational graph

- **House price prediction**
  - **N-sample training**

$$t = wx \qquad o = t + b \qquad L = (o - y)^2$$

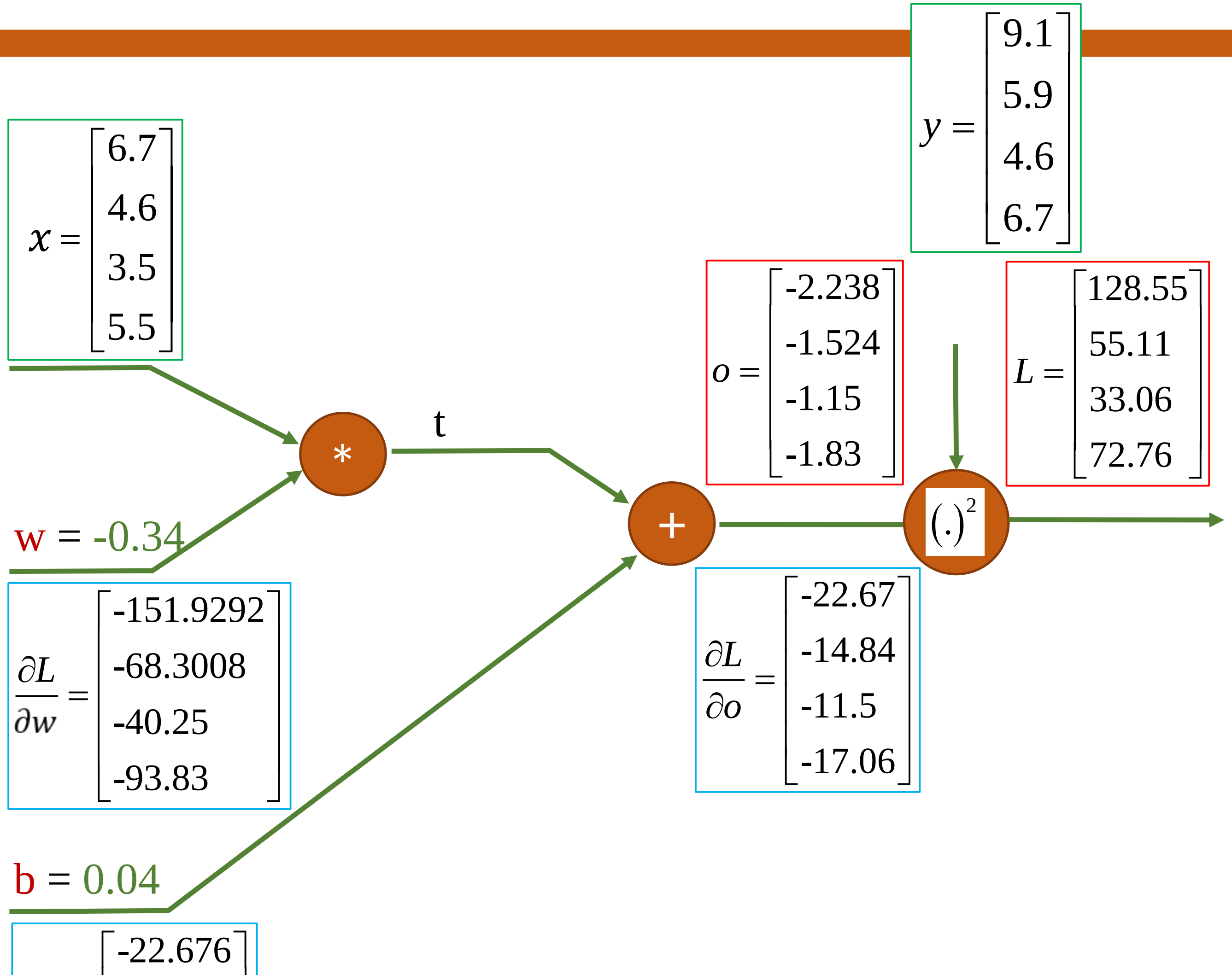


Computational graph

- **House price prediction**
  - **N-sample training**

$$\frac{sum(\frac{\partial L}{\partial w})}{4} = -88.5775$$

$$\frac{sum(\frac{\partial L}{\partial b})}{4} = -16.521$$



$t = wx \qquad o = t + b \qquad L = (o - y)^2$

Computational graph

- **House price prediction**
  - **N-sample training**

Cách cập nhật a và b

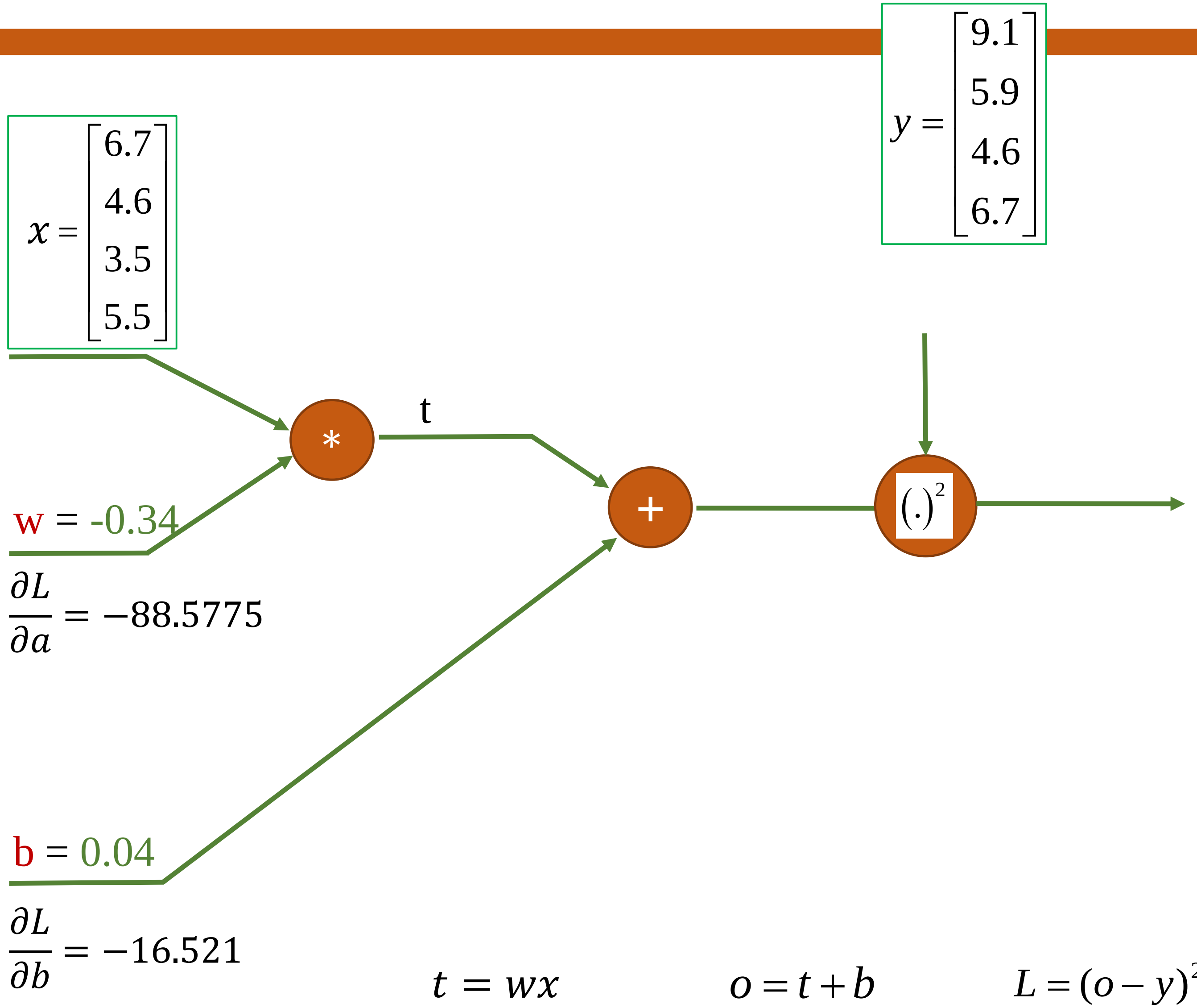
$$w = w - \eta * \frac{\partial L}{\partial w}$$

$$b = b - \eta * \frac{\partial L}{\partial b}$$

Learning rate $\eta = 0.01$

$$\text{w} = -0.34 - (0.01 * (-88.5775)) = 0.54$$

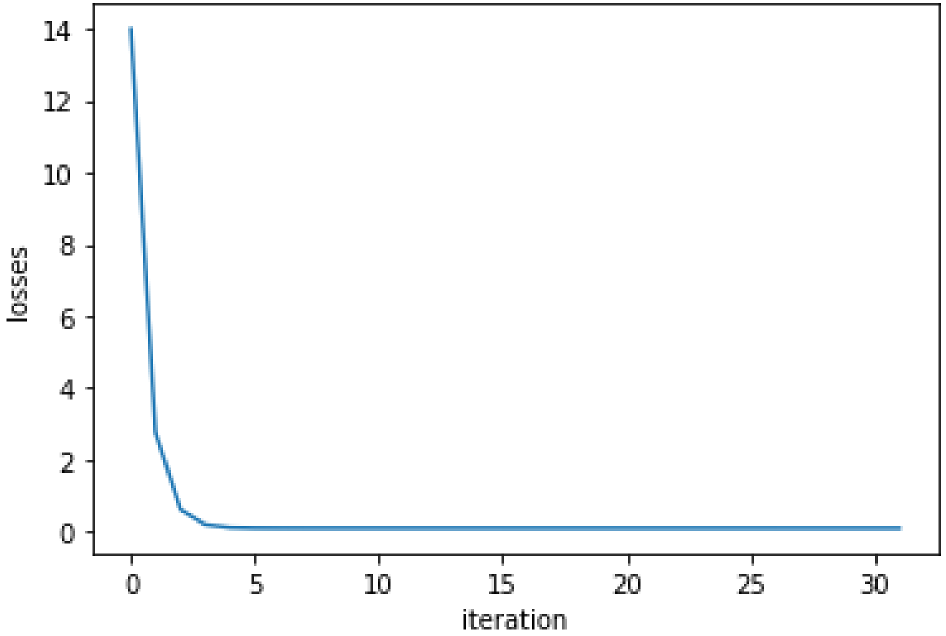
$$\text{b} = 0.04 - \left(0.01 * (-16.521)\right) = 0.205$$



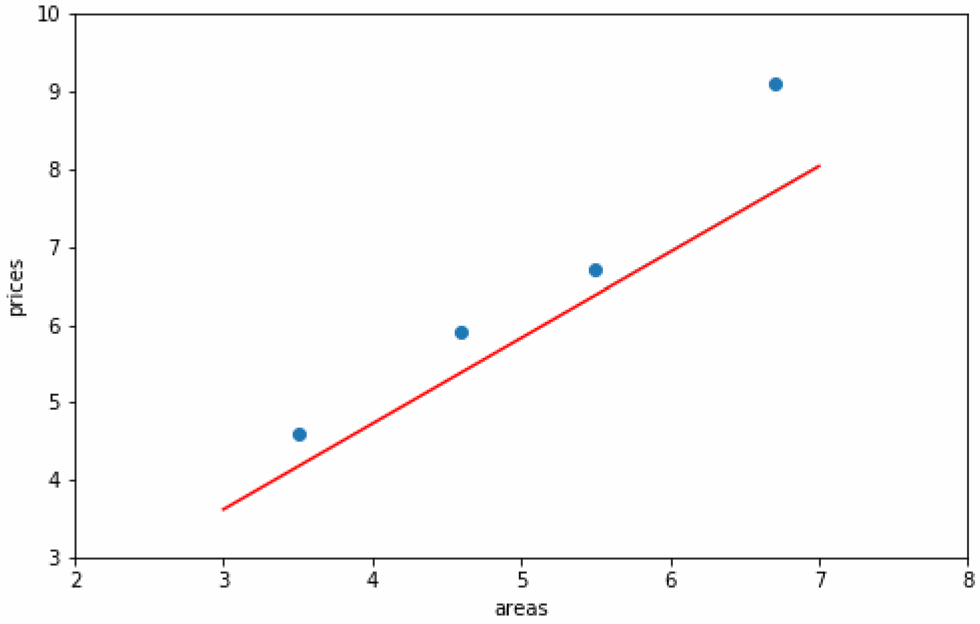
Computational graph

❖ House price prediction

❖ N-sample training



Losses for 30 iterations



Model updating for different iterations

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = wx^{(i)} + b \qquad \text{for } 0 \leq i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < N$$

4) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < N$$

5) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{N}$$

$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{N}$$

η is learning rate

Friendly version

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \leq i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \leq i < N$$

4) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{N}$$

η is learning rate

Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{N}$$

η is learning rate

```python
# load data

import numpy as np
from numpy import genfromtxt
import matplotlib.pyplot as plt

data = genfromtxt('data.csv', delimiter=',')
areas  = data[:,0]
prices = data[:,1]
data_size = areas.size

print(type(areas))
print('areas: ', areas)
print('prices: ', prices)
print('data_size: ', data_size)

plt.scatter(areas, prices)
plt.xlabel('areas')
plt.ylabel('prices')
plt.xlim(3,7)
plt.ylim(4,10)
plt.show()
```

Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{N}$$

η is learning rate

```
# vector [x, b]
data = np.c_[areas, np.ones((data_size, 1))]
print(data)

n_epochs = 1
eta = 0.01

theta = np.array([[-0.34],[0.04]])
print('theta', theta)
```


Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{N}$$

η is learning rate

```python
for epoch in range(n_epochs):
    sum_of_losses = 0
    gradients = np.zeros((2,1))

    for index in range(data_size):
        xi = data[index:index+1]
        yi = prices[index:index+1]
        print('\ndata: ', xi, yi)

        oi = xi.dot(theta)
        li = (oi - yi)*(oi - yi)
        g_li = 2*(oi - yi)

        print('z: ', oi)
        print('loss: ', index, li)
        print('gradient_loss: ', index, g_li)

        cg = xi.T.dot(g_li)
        print('variable gradient: ', index, cg)

        gradients = gradients + cg
        sum_of_losses = sum_of_losses + li

    sum_of_losses = sum_of_losses/data_size
    print('\nsum_of_losses: ', sum_of_losses)

    gradients     = gradients/data_size
    print('\ngradients: ', gradients)

    theta = theta - eta*gradients
    print('new params: ', theta)
```

Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

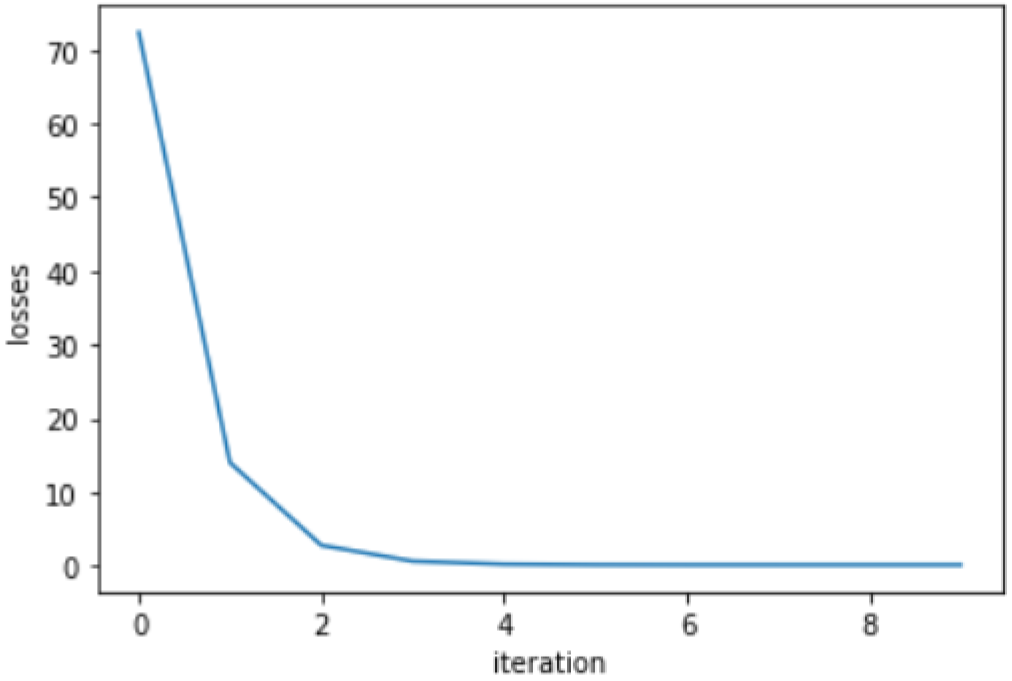
$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{N}$$

η is learning rate

```
import matplotlib.pyplot as plt

plt.plot(losses)
plt.xlabel('iteration')
plt.ylabel('losses')
plt.show()
```



Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L'^{(i)}_{\boldsymbol{\theta}} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \qquad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L'^{(i)}_{\boldsymbol{\theta}}}{N}$$

η is learning rate

Generalized formula

1) Pick all the N samples from training data

2) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

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

4) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

5) Cập nhật tham số

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

η is learning rate

More generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

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

4) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

5) Cập nhật tham số

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

η is learning rate

```python
# full code
import numpy as np
from numpy import genfromtxt

data = genfromtxt('data.csv', delimiter=',')
areas  = data[:,0]
prices = data[:,1:]
data_size = areas.size

# vector [x, b]
data = np.c_[areas, np.ones((data_size, 1))]
data = data.T

n_epochs = 10
eta = 0.01

# init weight
theta = np.array([[-0.34],[0.04]])
```

More generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $\boldsymbol{o}$

$$\boldsymbol{o} = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

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

4) Tính đạo hàm

$$\boldsymbol{k} = 2(\boldsymbol{o} - \boldsymbol{y})$$

$$L'_{\boldsymbol{\theta}} = \left(\begin{bmatrix} \boldsymbol{k} \\ \boldsymbol{k} \end{bmatrix} \odot \boldsymbol{x} \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\odot$ is element-wise multiplication

5) Cập nhật tham số

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

η is learning rate

More generalized formula

```python
losses = [] # for debug
for epoch in range(n_epochs):
    # compute output
    z = theta.T.dot(data)

    # compute loss
    loss = np.multiply((z-prices.T), (z-prices.T))
    losses.append(np.mean(loss))

    # compute gradient
    b = 2*(z-prices.T)
    gradient = np.multiply(data, np.vstack((b, b)))
    gradient = gradient.dot(np.ones((data_size, 1)))/data_size

    # update weights
    theta = theta - eta*gradient
```

Linear Regression

	Advantages	Disadvantages
1 sample	Simple to understand and implement Faster learning on some problems Noisy update is beneficial sometime	Computationally expensive Noisy gradient signal Convergence problem
m sample	A balance between the robustness of 1-sample and the efficiency of N-sample	
N sample	Computationally efficient More stable error gradient parallel processing	Premature convergence Memory problem Training speed is slower

Outline

- Machine Learning
- Derivative/Gradient
- Linear Regression
- Computational Graph
- Summary

Linear Regression

❖ Generalized formula

House price data

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

Advertising data

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

Model

$$\text{price} = \text{w} * area + b$$

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

Model (vectorization)

$$\text{y} = \boldsymbol{\theta}^T \boldsymbol{x} \quad \text{where} \quad \boldsymbol{\theta}^T = [b \quad w]^T$$
$$\boldsymbol{x} = [x_0 \ \ area]^T$$
$$x_0 = 1$$

Model

$$\text{Sale} = w_1 * TV + w_2 * Radio + w_3 * Newspaper + b$$

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

Model (vectorization)

$$\text{y} = \boldsymbol{\theta}^T \boldsymbol{x} \quad \text{where} \quad \boldsymbol{\theta}^T = [b \quad w_1 \quad w_2 \quad w_3]^T$$
$$\boldsymbol{x} = [x_0 \ \ TV \ \ Radio \ \ Newspaper]^T$$
$$x_0 = 1$$

Linear Regression

❖ Generalized formula

Boston House Price Data

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

Model

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

Model (vectorization)

$$y = \boldsymbol{\theta}^T \boldsymbol{x} \quad \text{where} \quad \begin{aligned} \boldsymbol{\theta}^T &= [b \quad w_1 \quad \ldots \quad w_{13}]^T \\ \boldsymbol{x} &= [x_0 \quad x_1 \quad \ldots \quad x_{13}]^T \\ x_0 &= 1 \end{aligned}$$

Linear Regression (1-sample)

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

2) Tính output o

$$o = wx + b$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

$$L'_w = 2x(o - y)$$

$$L'_b = 2(o - y)$$

5) Cập nhật tham số

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

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

η is learning rate

Friendly version

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

2) Tính output o

$$o = \boldsymbol{\theta}^T \boldsymbol{x}$$

3) Tính loss

$$L = (o - y)^2$$

4) Tính đạo hàm

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

5) Cập nhật tham số

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

η is learning rate

Generalized formula

Linear Regression (m-samples)

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = wx^{(i)} + b \quad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{m}$$

$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{m}$$

η is learning rate

Friendly version

1) Pick m samples $(x^{(i)}, y^{(i)})$ from training data

1.1) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \quad \text{for } 0 \leq i < m$$

1.2) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \leq i < m$$

1.3) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \quad \text{for } 0 \leq i < m$$

2) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{m}$$

η is learning rate

Generalized formula

Linear Regression (N-samples)

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = wx^{(i)} + b \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \quad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L_w'^{(i)} = 2x(o^{(i)} - y^{(i)})$$
$$L_b'^{(i)} = 2(o^{(i)} - y^{(i)}) \quad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$w = w - \eta \frac{\sum_i L_w'^{(i)}}{N}$$

$$b = b - \eta \frac{\sum_i L_b'^{(i)}}{N}$$

η is learning rate

Friendly version

1) Pick all the N samples from training data

2) Tính output $o^{(i)}$

$$o^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \qquad \text{for } 0 \le i < N$$

3) Tính loss

$$L^{(i)} = (o^{(i)} - y^{(i)})^2 \qquad \text{for } 0 \le i < N$$

4) Tính đạo hàm

$$L_{\boldsymbol{\theta}}'^{(i)} = 2\boldsymbol{x}(o^{(i)} - y^{(i)}) \quad \text{for } 0 \le i < N$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\sum_i L_{\boldsymbol{\theta}}'^{(i)}}{N}$$

η is learning rate

Generalized formula


Thank you!