Stanford
ONLINE

DeepLearning.AI

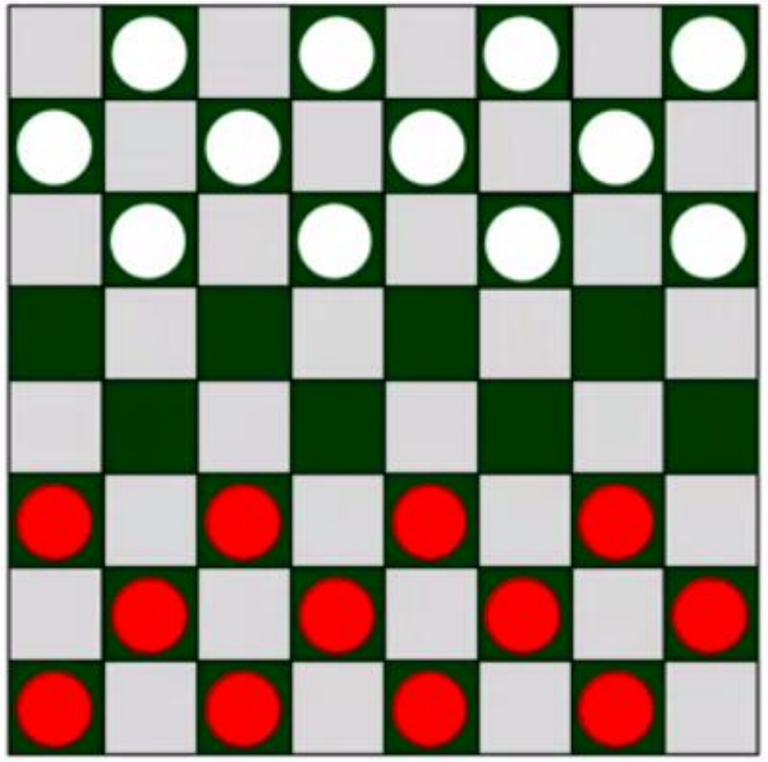
Machine Learning Overview

What is Machine Learning?

Machine learning

"Field of study that gives computers the ability to learn without being explicitly programmed."

Arthur Samuel (1959)



Machine learning algorithms

rapid advancements

used most in real-world applications

- Supervised learning ← course 1, 2

- Unsupervised learning ←
- Recommender systems
- Reinforcement learning

course 3

Practical advice for applying learning algorithms

Stanford
ONLINE

DeepLearning.AI

Machine Learning Overview

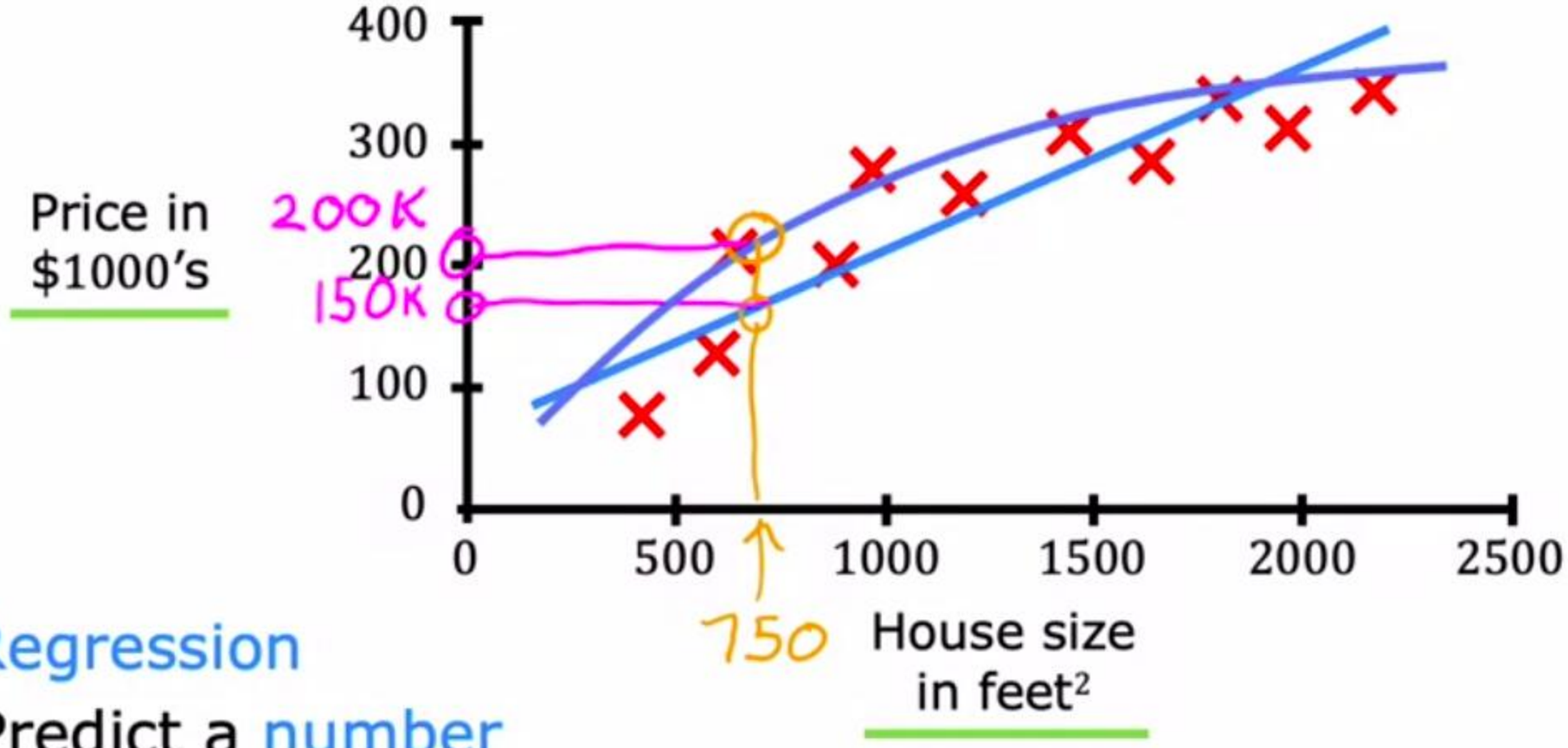
Supervised Learning Part 1

Supervised learning

Learns from being given "right answers"

Input (X)	Output (Y)	Application
email →	spam? (0/1)	spam filtering
audio →	text transcripts	speech recognition
English →	Spanish	machine translation
ad, user info →	click? (0/1)	online advertising
image, radar info →	position of other cars	self-driving car
image of phone →	defect? (0/1)	visual inspection

Regression: Housing price prediction



Regression

Predict a number

infinitely many possible outputs

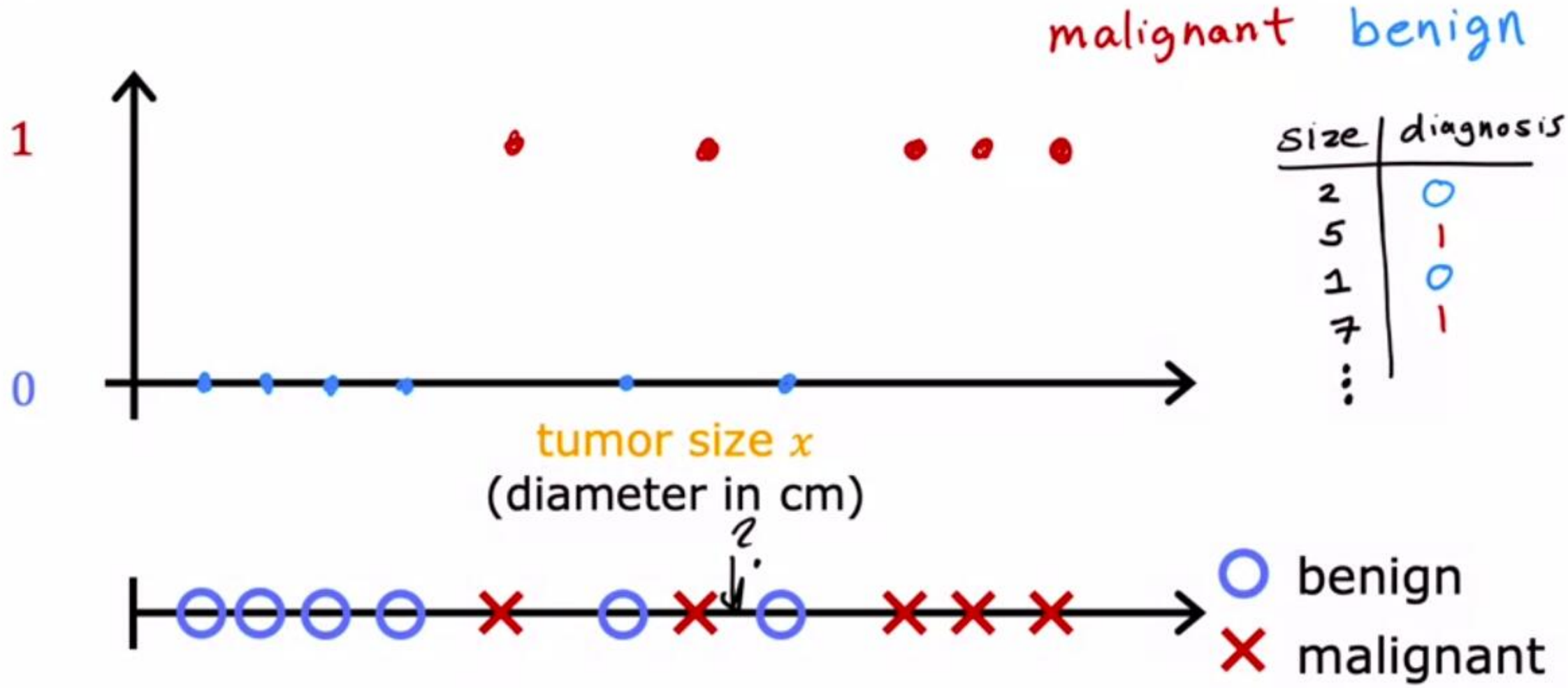
Stanford
ONLINE

DeepLearning.AI

Machine Learning Overview

Supervised Learning Part 2

Classification: Breast cancer detection

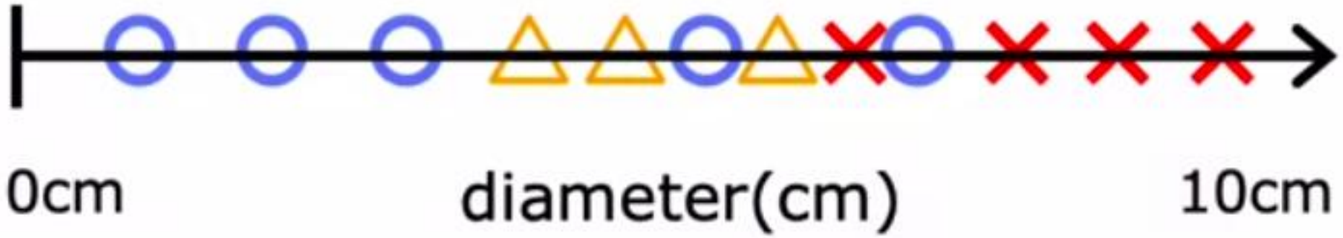


Classification: Breast cancer detection

benign

malignant type1

malignant type 2



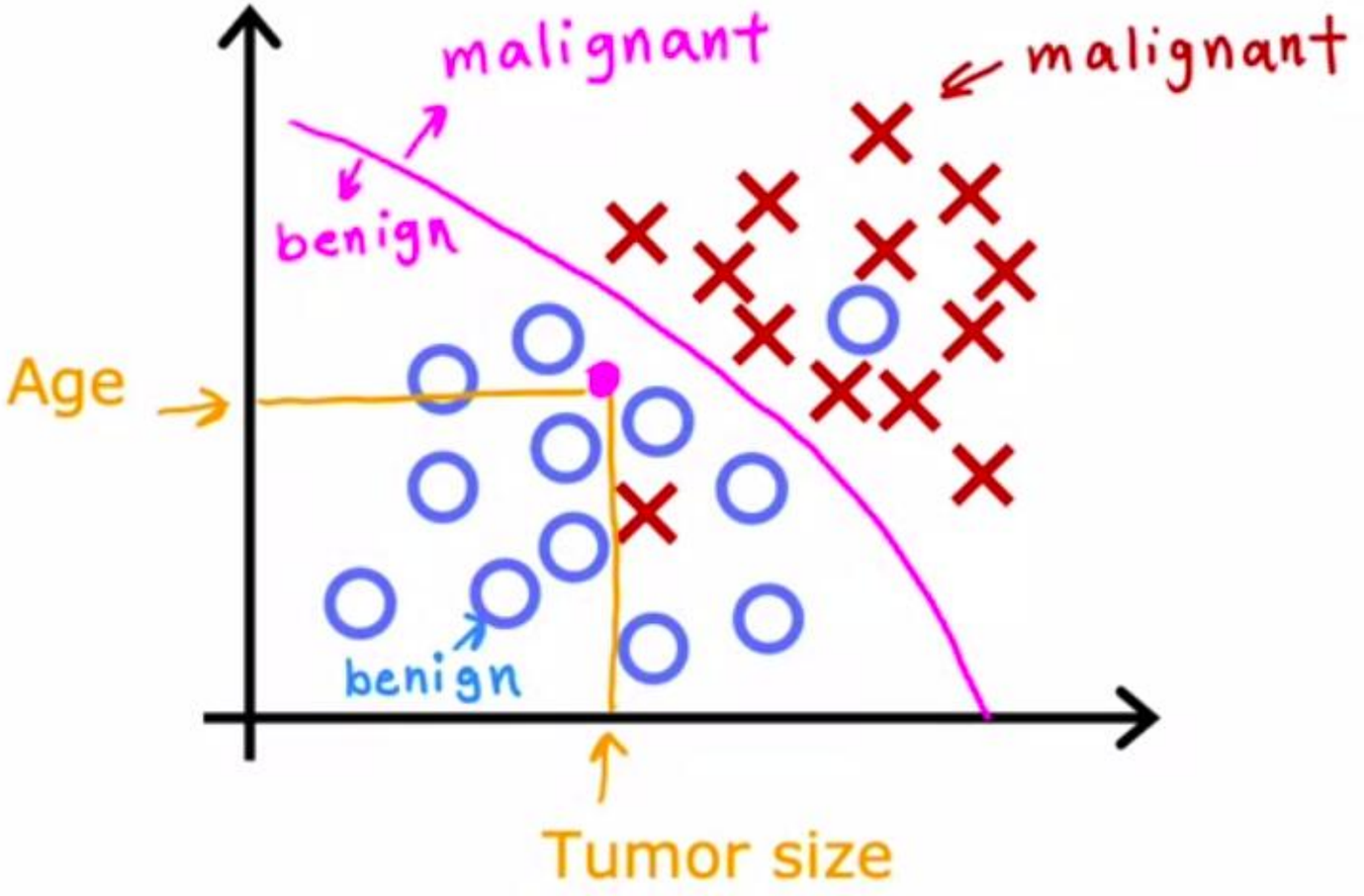
class category

Classification

predict categories cat dog benign malignant 0, 1, 2

small number of possible outputs

Two or more inputs



Supervised learning

Learns from being given "right answers"

Regression

Predict a number

infinitely many possible outputs

Classification

predict categories

Stanford
ONLINE

DeepLearning.AI

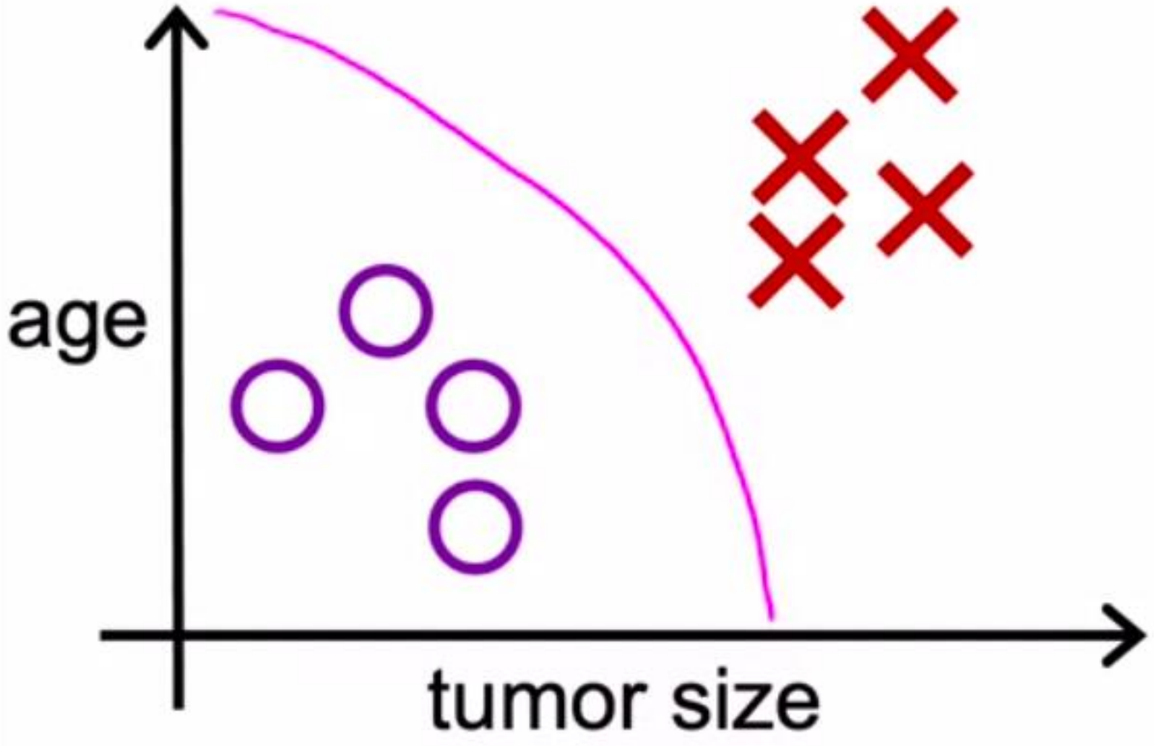
Machine Learning Overview

Unsupervised Learning Part 1

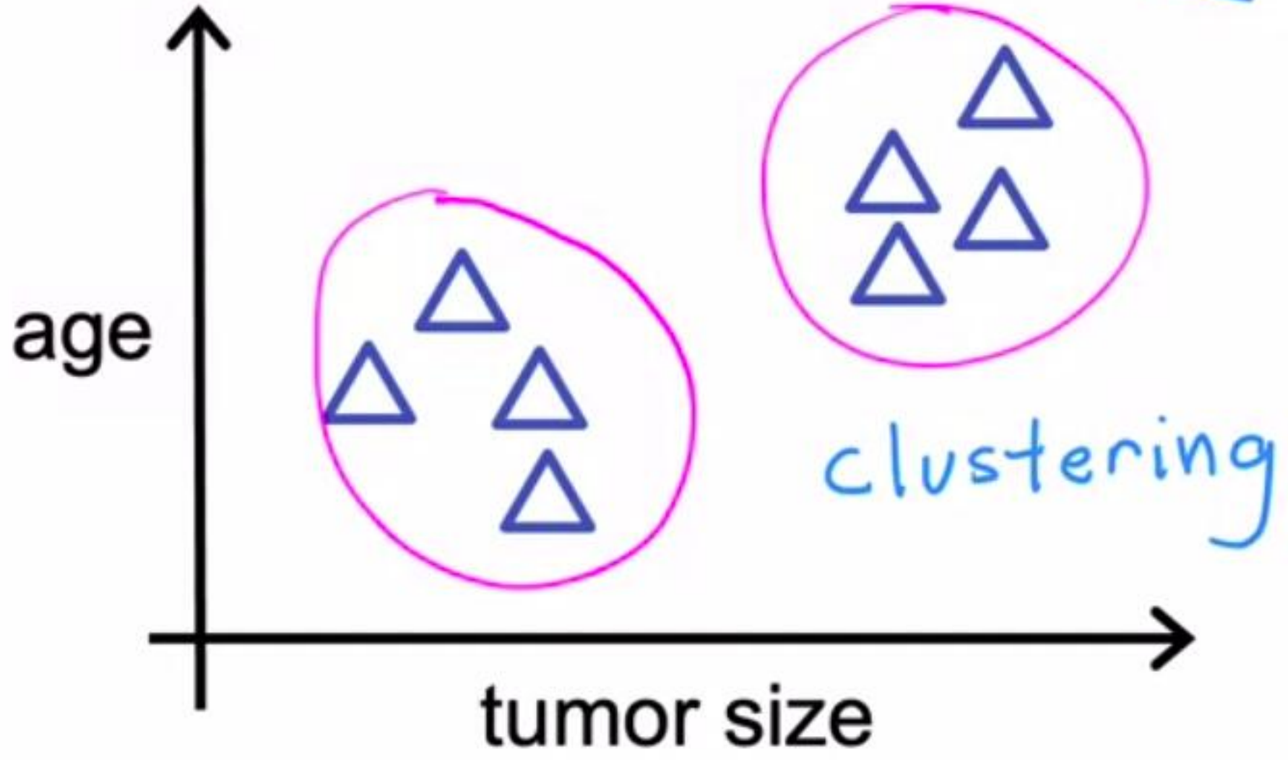
Previous: Supervised learning

Now: Unsupervised learning

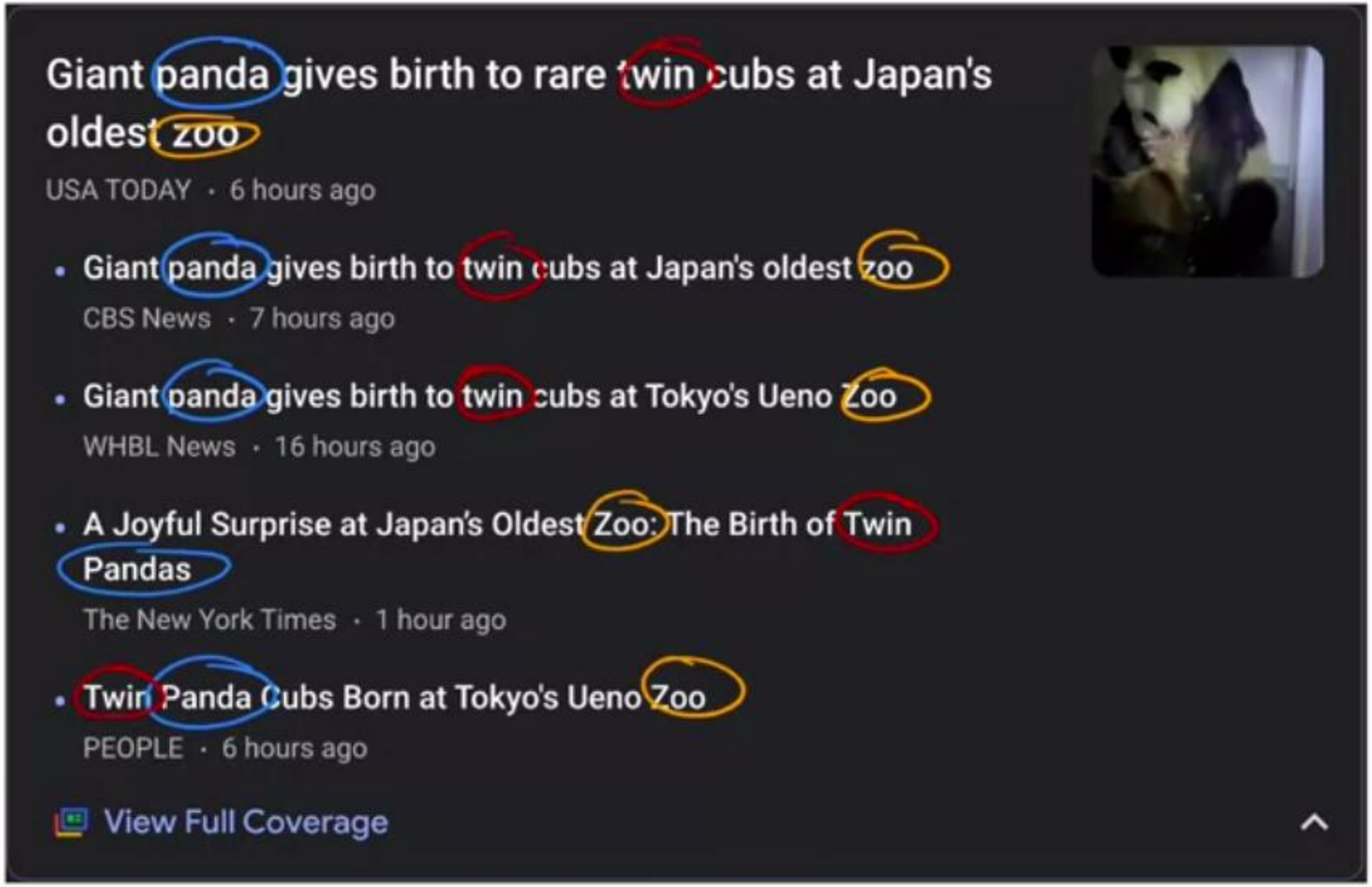
Supervised learning
Learn from data labeled with the "right answers"



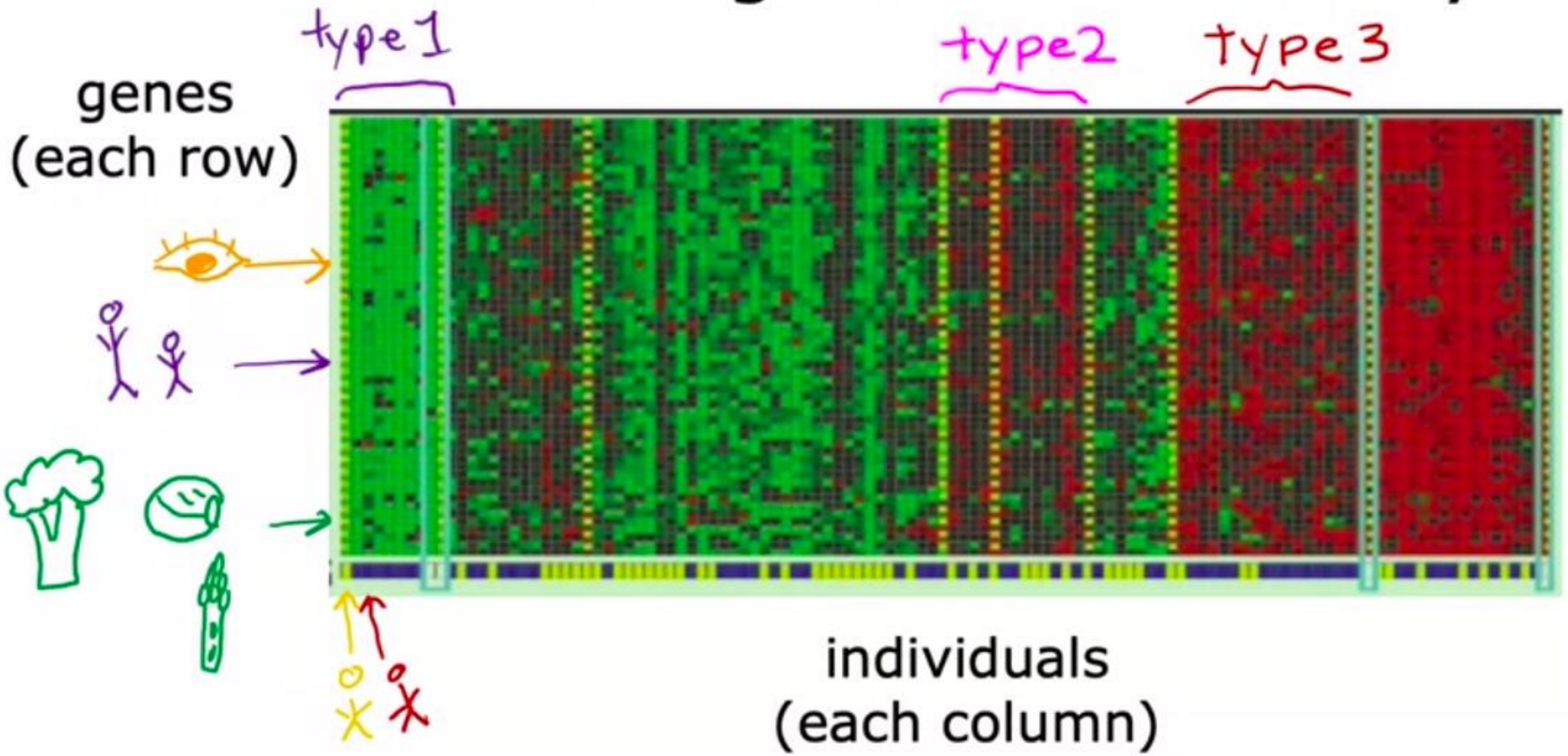
Unsupervised learning
Find something interesting in unlabeled data.



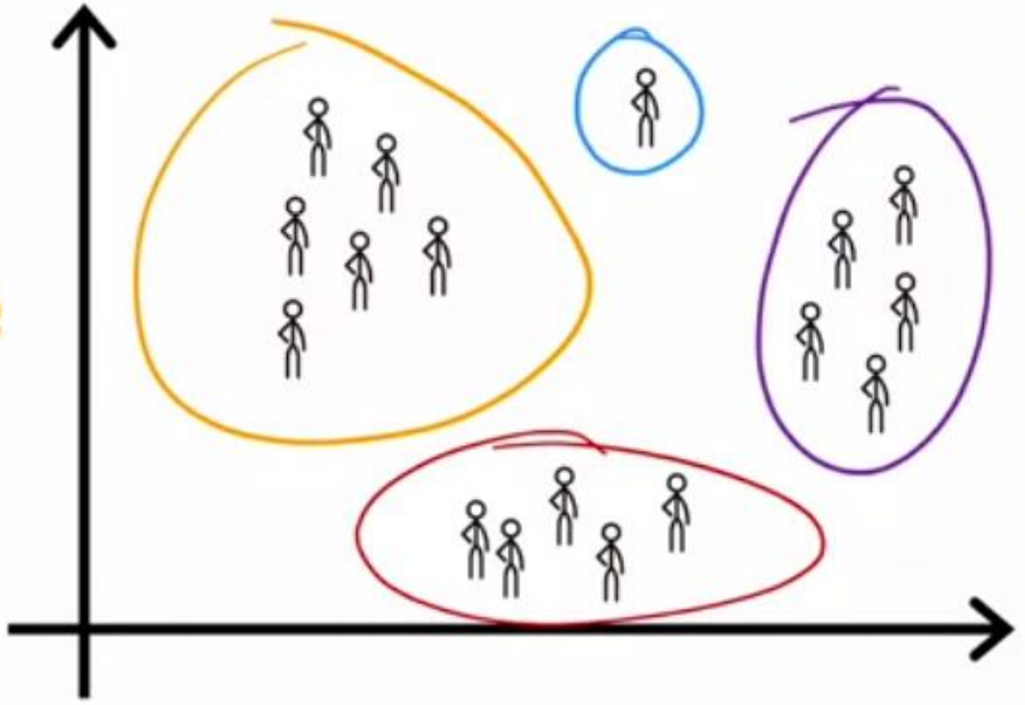
Clustering: Google news



Clustering: DNA microarray



Clustering: Grouping customers



growing skills and knowledge

develop career

stay updated with AI

Stanford
ONLINE

DeepLearning.AI

Machine Learning Overview

Unsupervised Learning Part 2

Unsupervised learning

Data only comes with inputs x, but not output labels y.
Algorithm has to find structure in the data.

Clustering
Group similar data points together.

Dimensionality reduction
Compress data using fewer numbers.

Anomaly detection
Find unusual data points.

Question

Of the following examples, which would you address using an unsupervised learning algorithm?

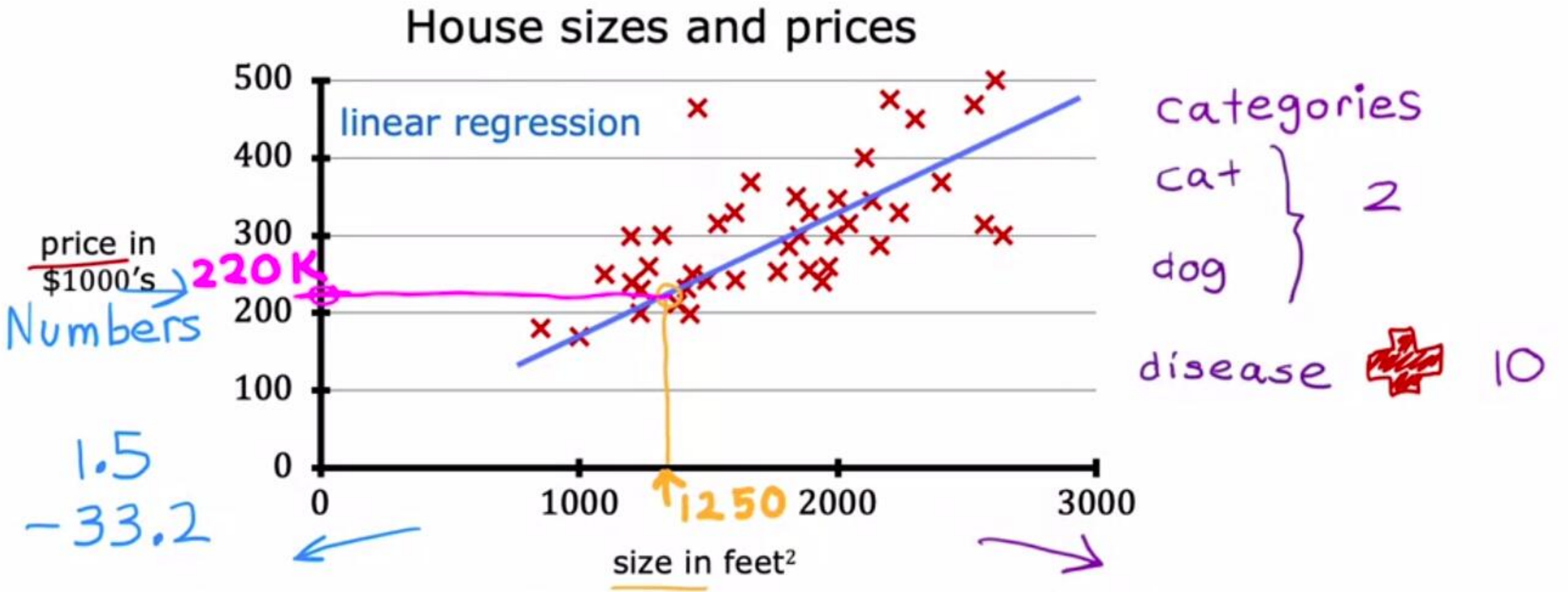
- [ ] Given email labeled as spam/not spam, learn a spam filter.
- [x] Given a set of news articles found on the web, group them into sets of articles about the same story.
- [x] Given a database of customer data, automatically discover market segments and group customers into different market segments.
- [ ] Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

Linear Regression Model Part 1



Regression model
Predicts numbers

Supervised learning model
Data has "right answers"

Classification model
Predicts categories
Small number of possible outputs

House sizes and prices

Data table

size in feet²	price in $1000's
2104	400
1416	232
1534	315
852	178
...	...
3210	870

Terminology

Training set: Data used to train the model

	x size in feet^2	y price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

$m = 47$

$x^{(1)} = 2104$ $y^{(1)} = 400$

$(x^{(1)}, y^{(1)}) = (2104, 400)$

$x^{(2)} = 1416$ $x^{(2)} \neq x^2$ not exponent

Notation:

x = "input" variable
feature

y = "output" variable
"target" variable

m = number of training examples

(x, y) = single training example

$(x^{(i)}, y^{(i)})$

$(x^{(i)}, y^{(i)})$ = i^{th} training example

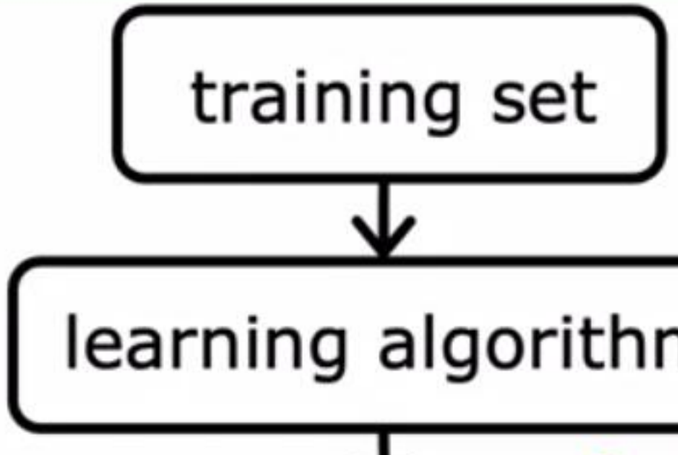
index (1^{st}, 2^{nd}, 3^{rd} ...)

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

Linear Regression Model Part 2



features
targets

"y-hat"

$\hat{y}$

~~hypothesis~~ function

feature model prediction

(estimated y)

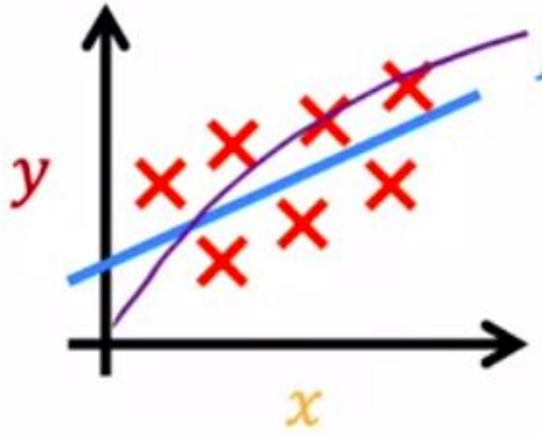
target

size → f → price (estimated)

How to represent f?

$$f_{w,b}(x) = wx + b$$

$$f(x)$$



$f_{w,b}(x) = wx + b$

$f(x) = wx + b$

linear

single feature x

Linear regression with one variable.

size

Univariate linear regression.

one variable

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

Cost Function

Training set

features → size in feet² (x)

targets → price \$1000's (y)

size in feet² (x)	price \$1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

Model: $f_{w,b}(x) = wx + b$

w, b: parameters

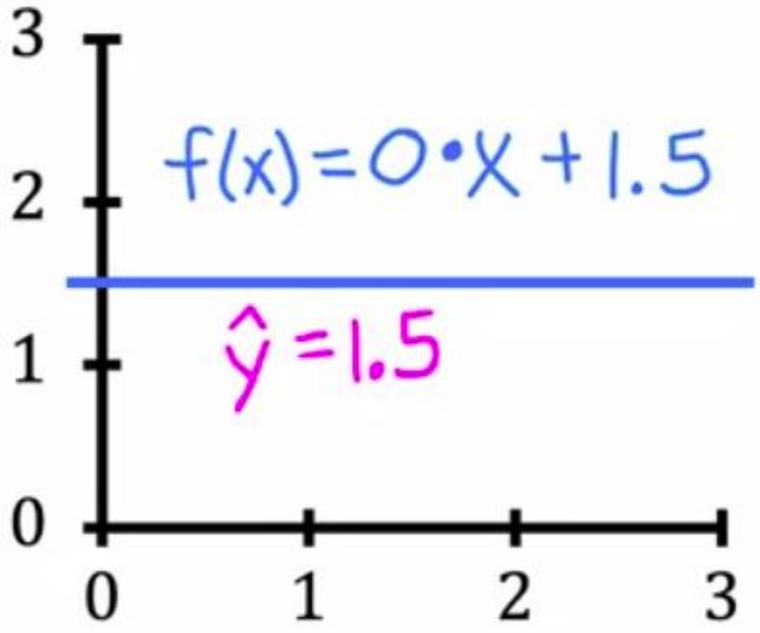
coefficients

weights

What do w, b do?

$$f_{w,b}(x) = wx + b$$

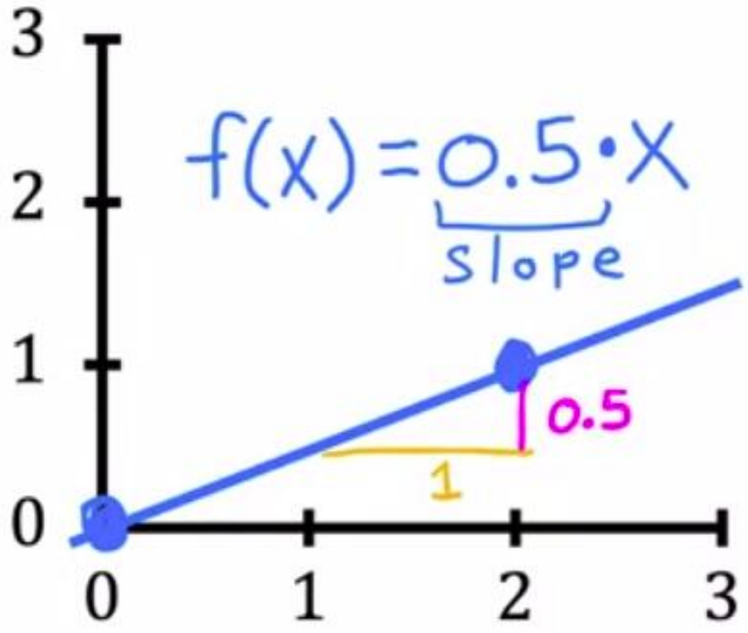
$f(x)$



→ $w = 0$

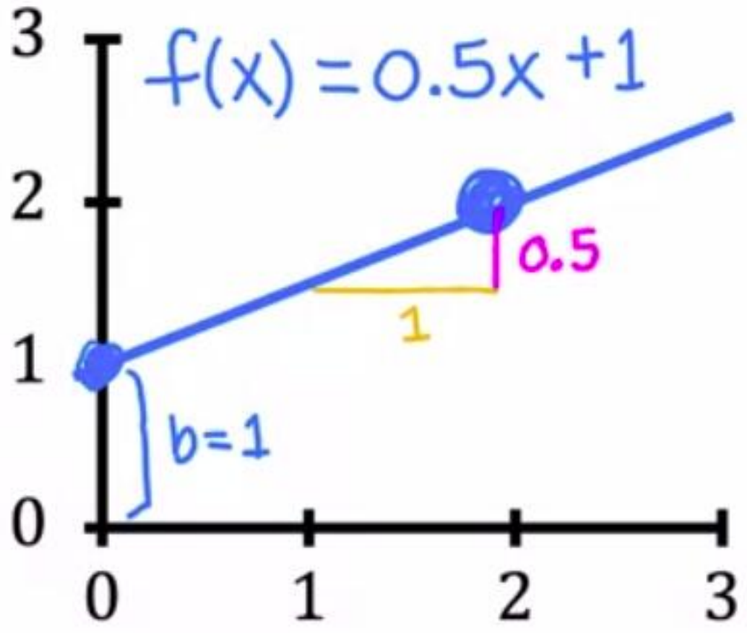
→ $b = 1.5$

y-intercept



→ $w = 0.5$

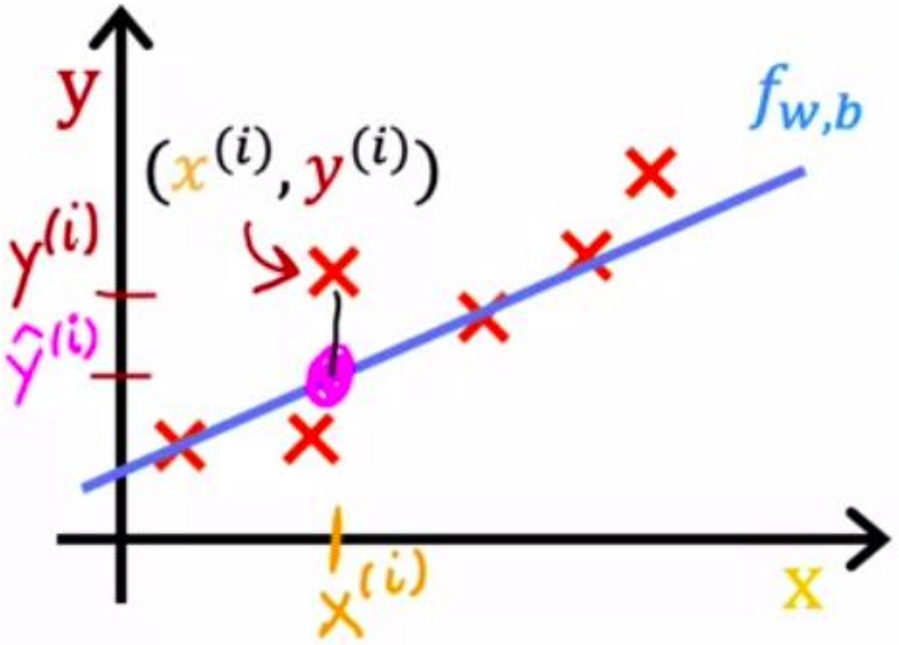
→ $b = 0$



→ $w = 0.5$

→ $b = 1$

Cost function: Squared error cost function



$$\hat{y}^{(i)} = f_{w,b}(x^{(i)})$$

$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

$\hat{y}^{(i)}$: error

m = number of training examples

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2$$

$J(w,b)$: intuition

Find w, b:

$\hat{y}^{(i)}$ is close to $y^{(i)}$ for all $(x^{(i)}, y^{(i)})$.

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

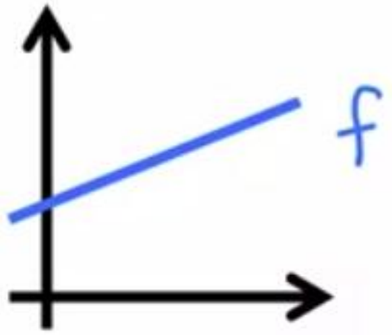
Cost Function Intuition

model:

$$f_{w,b}(x) = wx + b$$

parameters:

w, b



cost function:

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

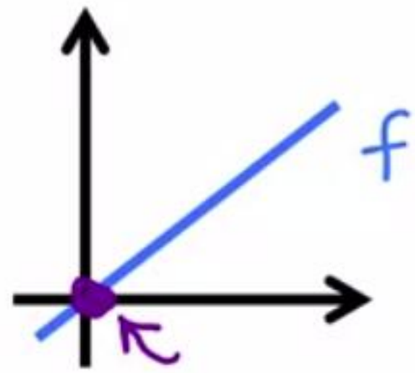
goal:

$$\underset{w,b}{\text{minimize}}\, J(w,b)$$

simplified

$$f_w(x) = wx$$

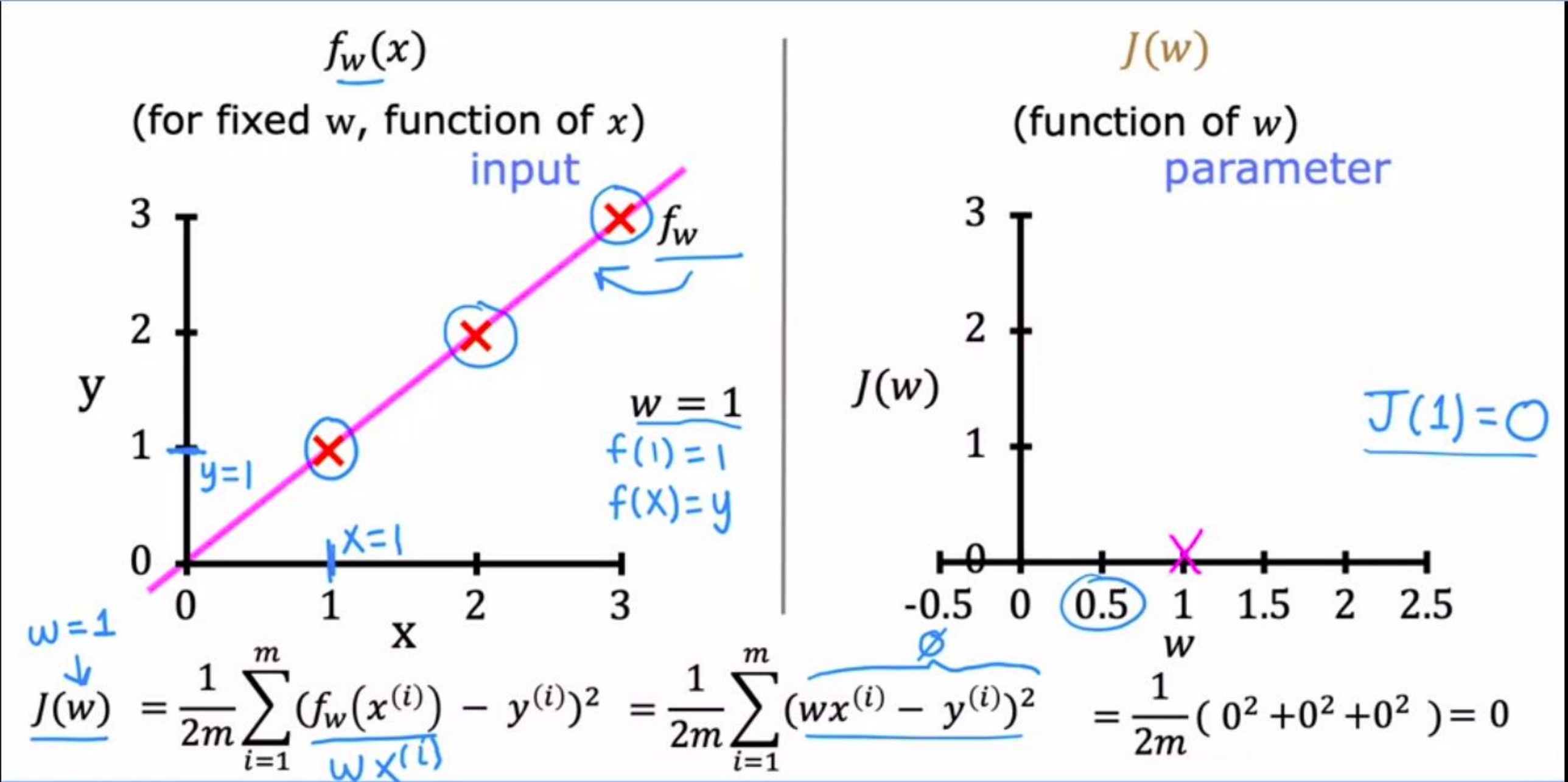
$b = \emptyset$

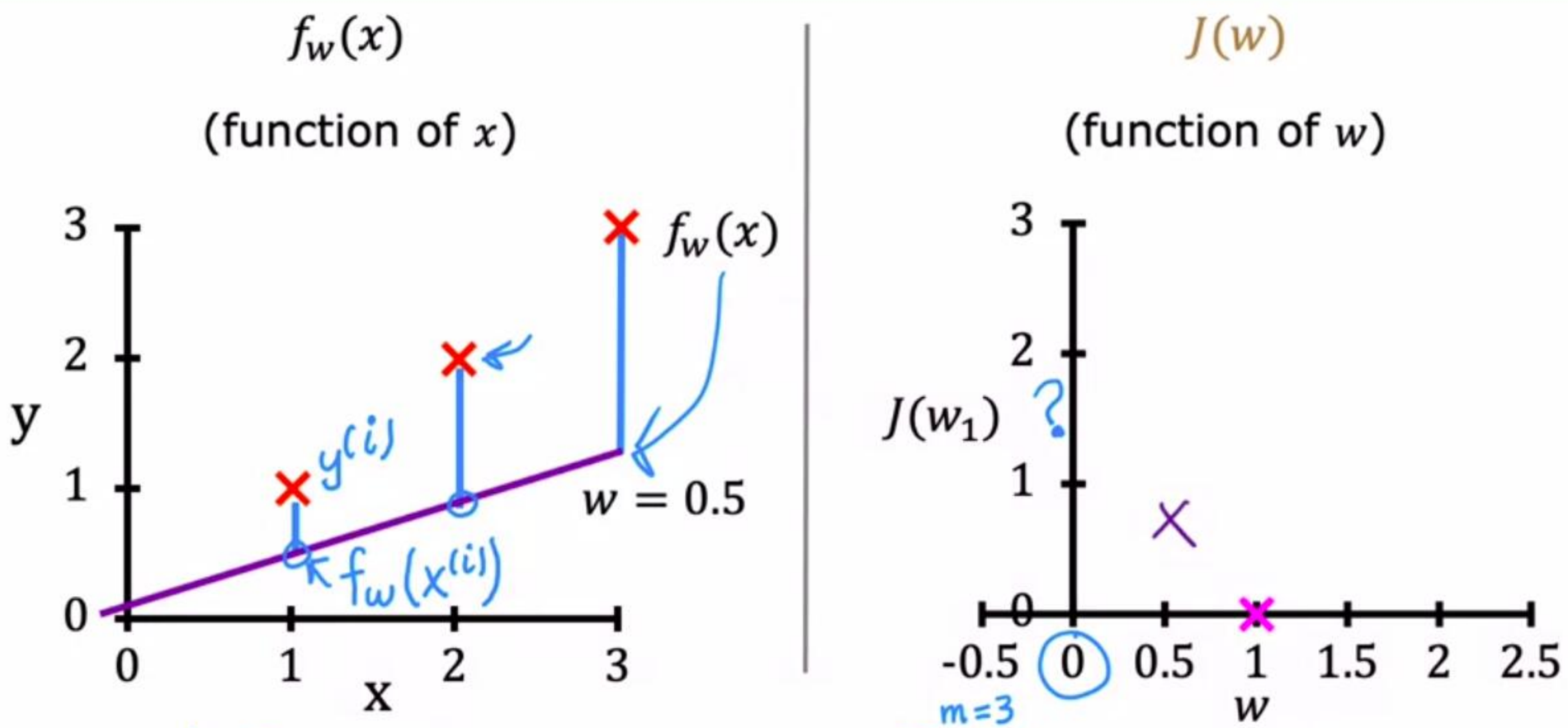


$$J(w) = \frac{1}{2m}\sum_{i=1}^{m}(f_w(x^{(i)}) - y^{(i)})^2$$

$wx^{(i)}$

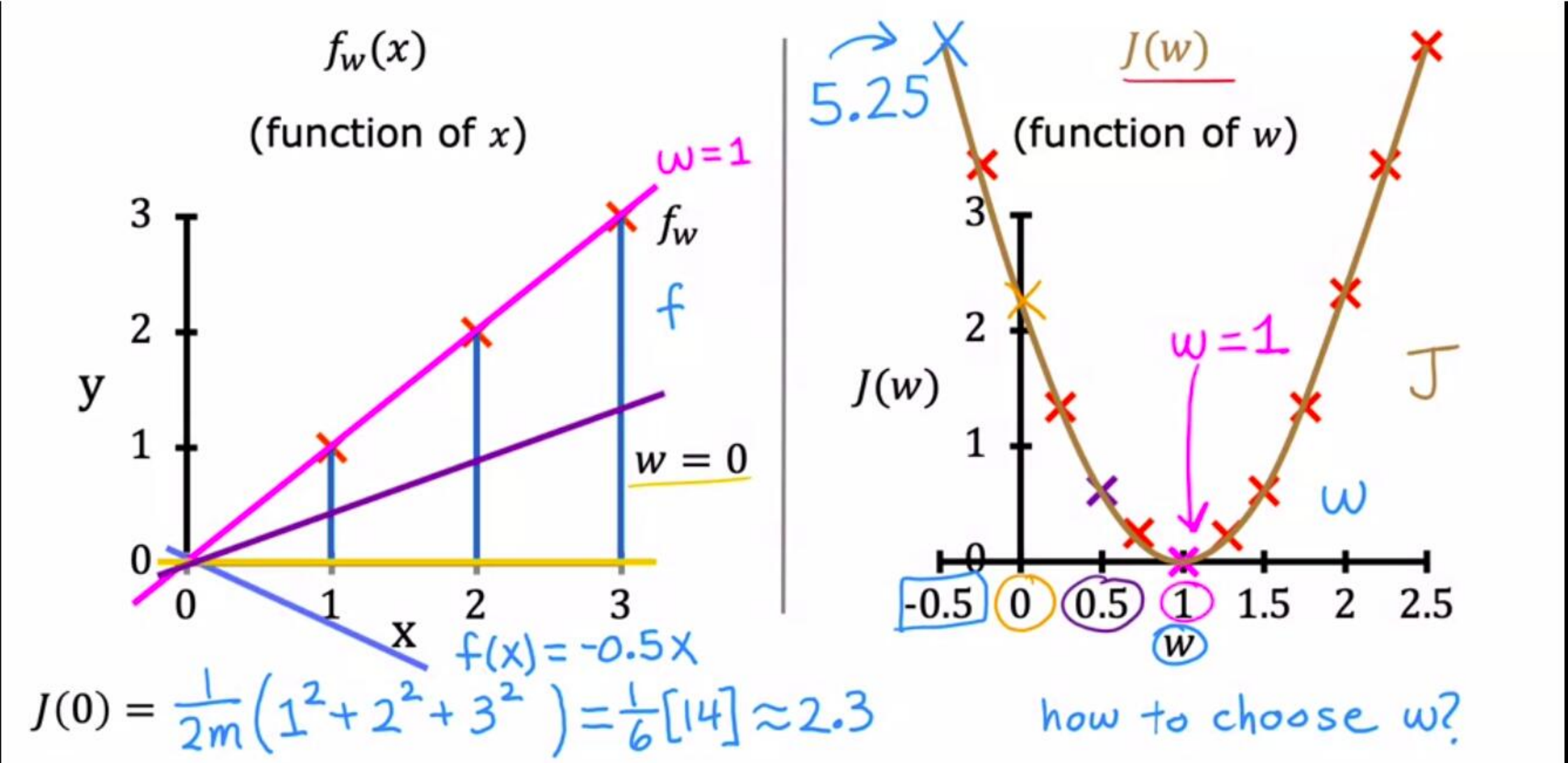
$$\underset{w}{\text{minimize}}\, J(w)$$


$f_w(x)$
(for fixed w, function of x)
input
f_w
3
2
1
0
y
$w = 1$
$f(1) = 1$
$f(x) = y$
$y=1$
$x=1$
0
1
2
3
x
$J(w)$
(function of w)
parameter
3
2
1
0
$J(w)$
$J(1) = 0$
-0.5
0
0.5
1
1.5
2
2.5
w
$w=1$
$J(w) = \frac{1}{2m}\sum_{i=1}^{m}(f_w(x^{(i)}) - y^{(i)})^2 = \frac{1}{2m}\sum_{i=1}^{m}(wx^{(i)} - y^{(i)})^2 = \frac{1}{2m}(0^2+0^2+0^2) = 0$
$wx^{(i)}$



$$J(0.5) = \frac{1}{2m}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right] = \frac{1}{2\times 3}[3.5] = \frac{3.5}{6} \approx 0.58$$

$m = 3$

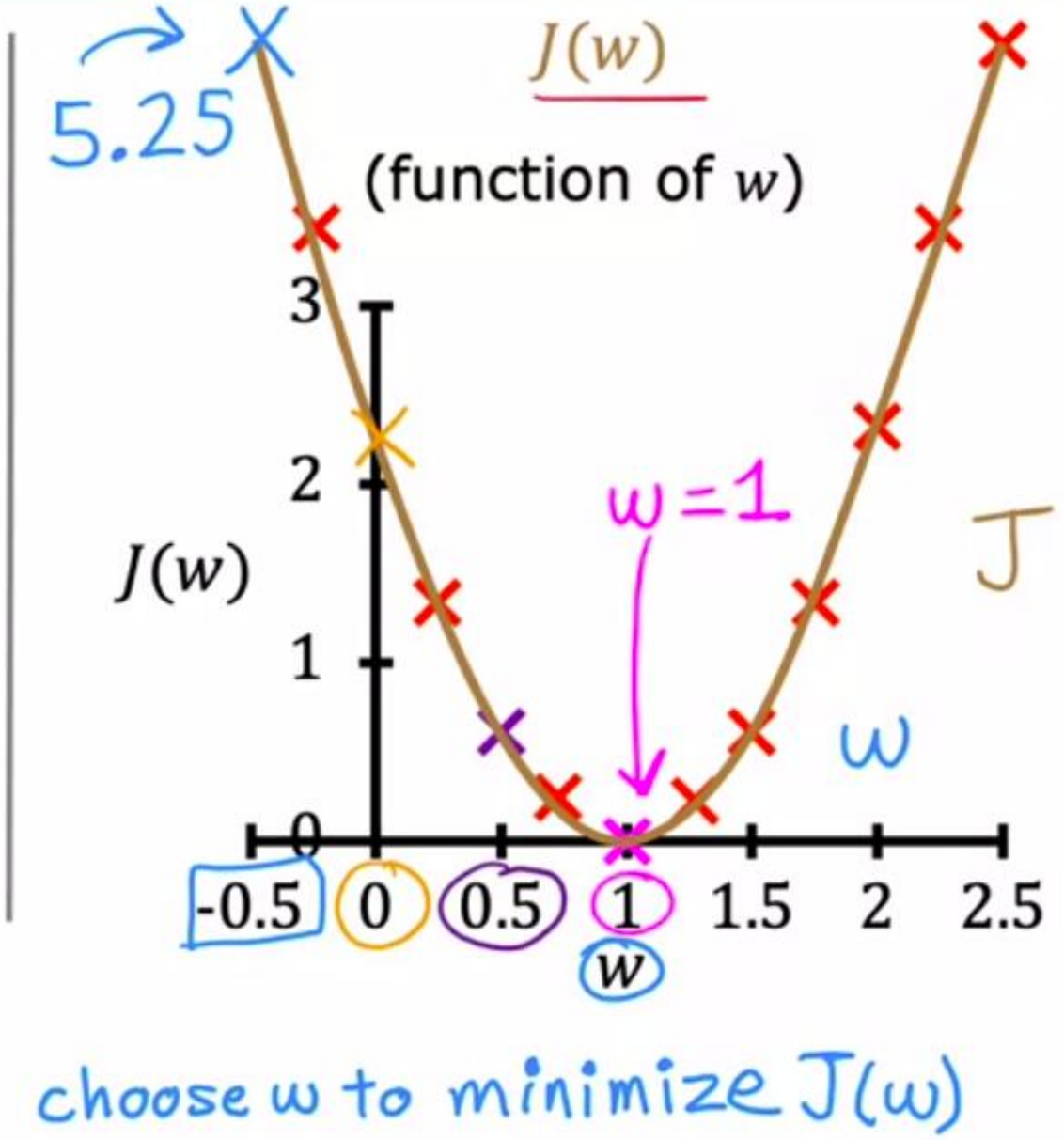

$f_w(x)$
(function of x)
$w=1$
f_w
f
$w=0$
y
x
0
1
2
3
$f(x) = -0.5x$
$J(0) = \frac{1}{2m}(1^2+2^2+3^2) = \frac{1}{6}[14] \approx 2.3$
$J(w)$
(function of w)
5.25
J
w
-0.5
0
0.5
1
1.5
2
2.5
how to choose w?

goal of linear regression:

$$\underset{w}{\text{minimize}}\, J(w)$$

general case:

$$\underset{w,b}{\text{minimize}}\, J(w,b)$$



choose w to minimize J(w)

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

Visualizing the Cost Function

Model $f_{w,b}(x) = wx + b$

~~before: b=0~~

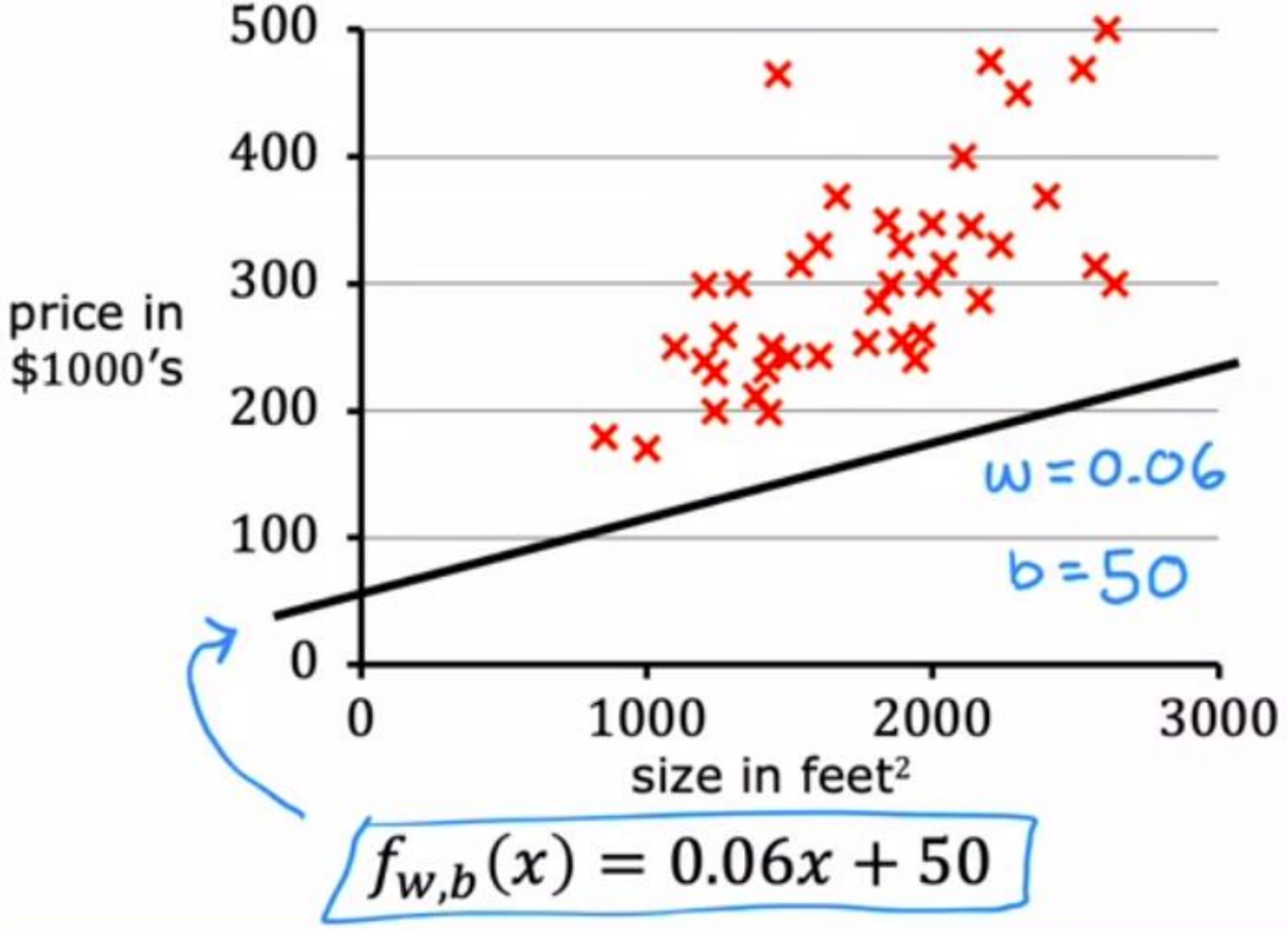
Parameters w, b

Cost Function $J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$

Objective $\underset{w,b}{\text{minimize}}\, J(w,b)$

$f_{w,b}$

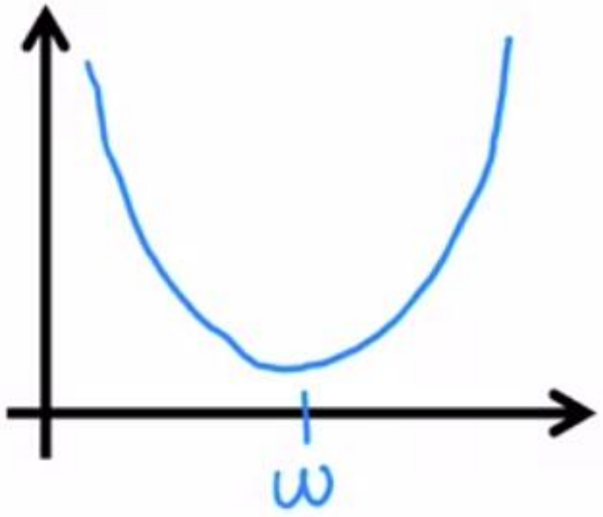
(function of x)



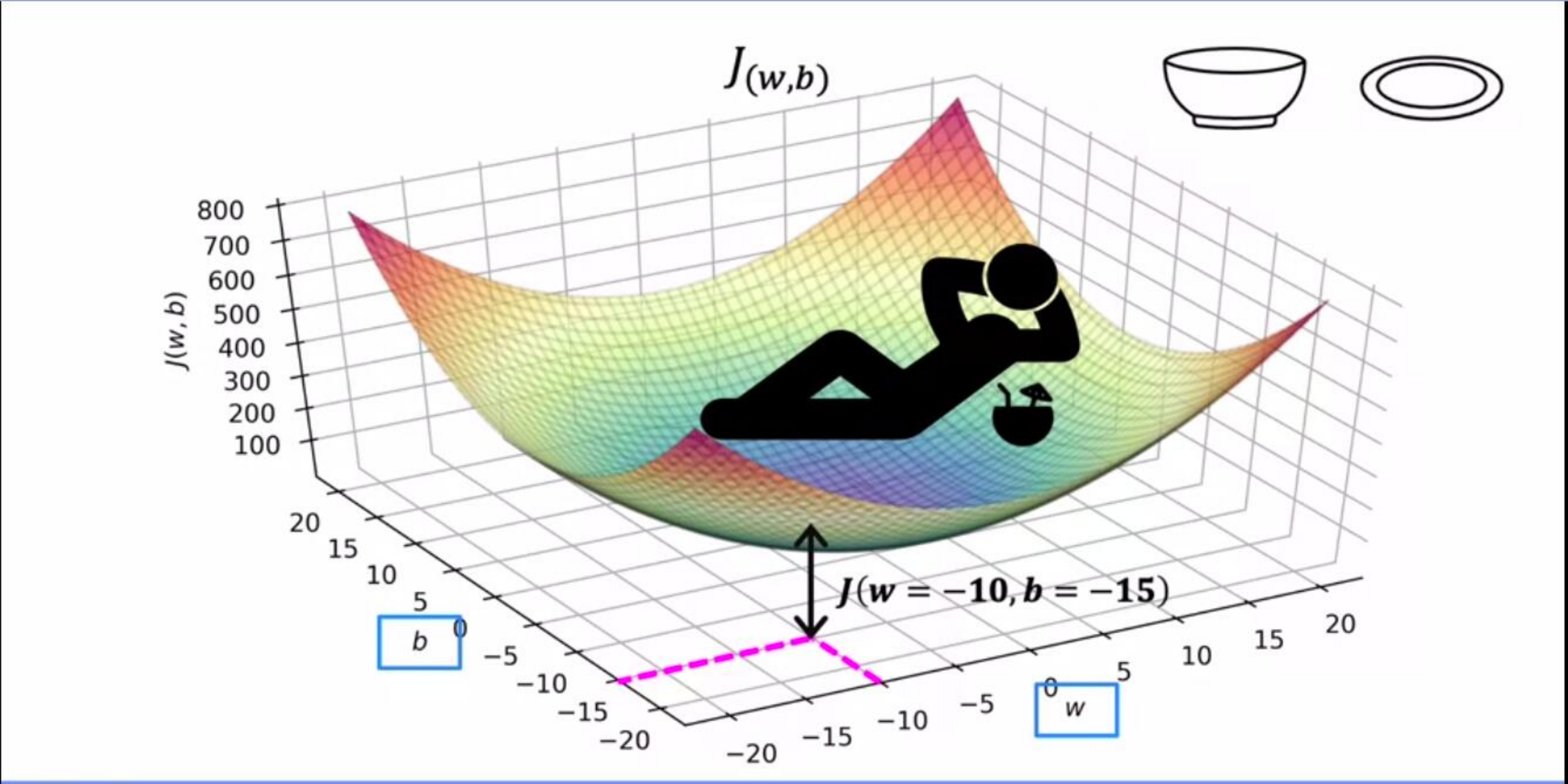
$f_{w,b}(x) = 0.06x + 50$

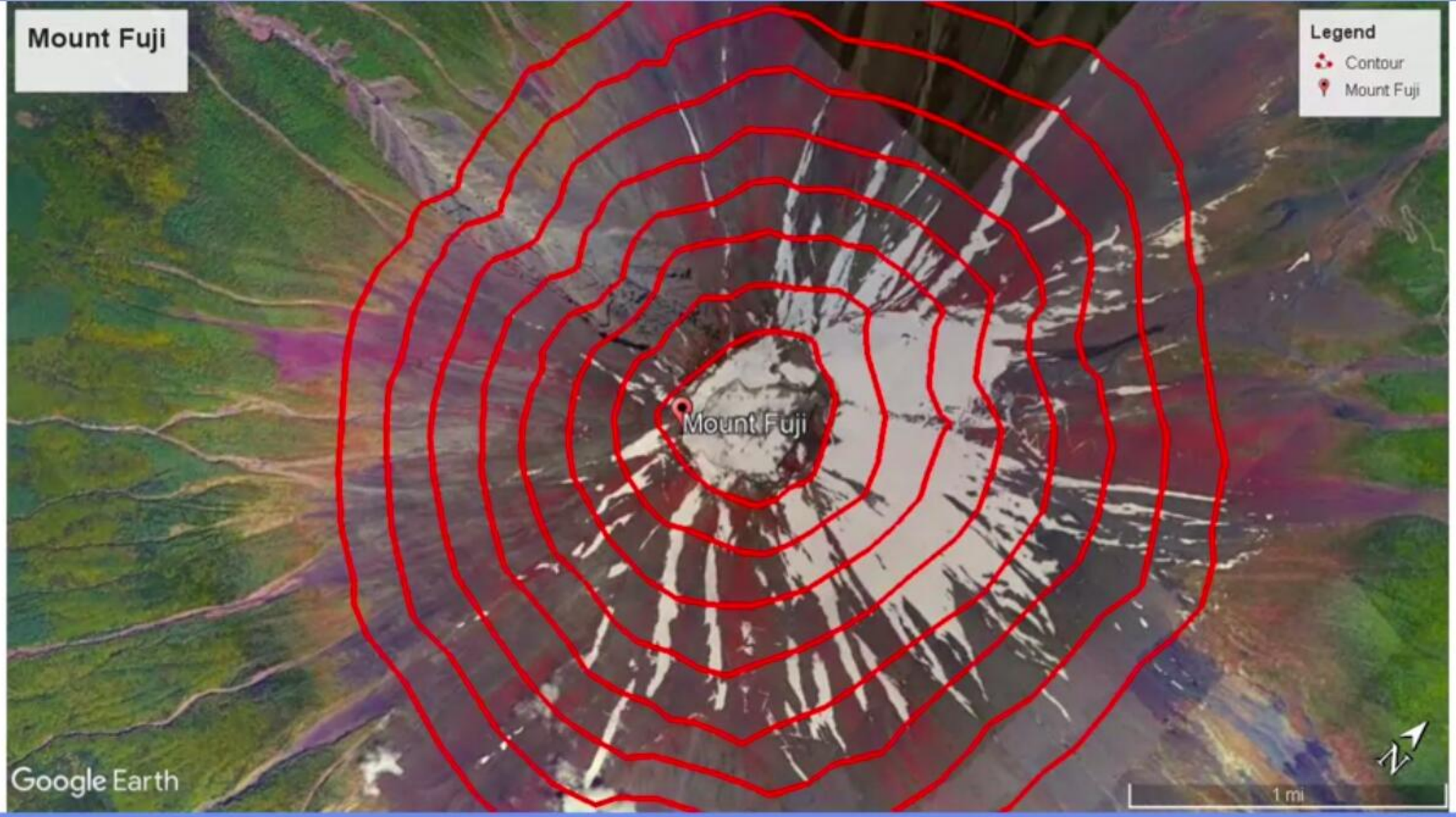
J

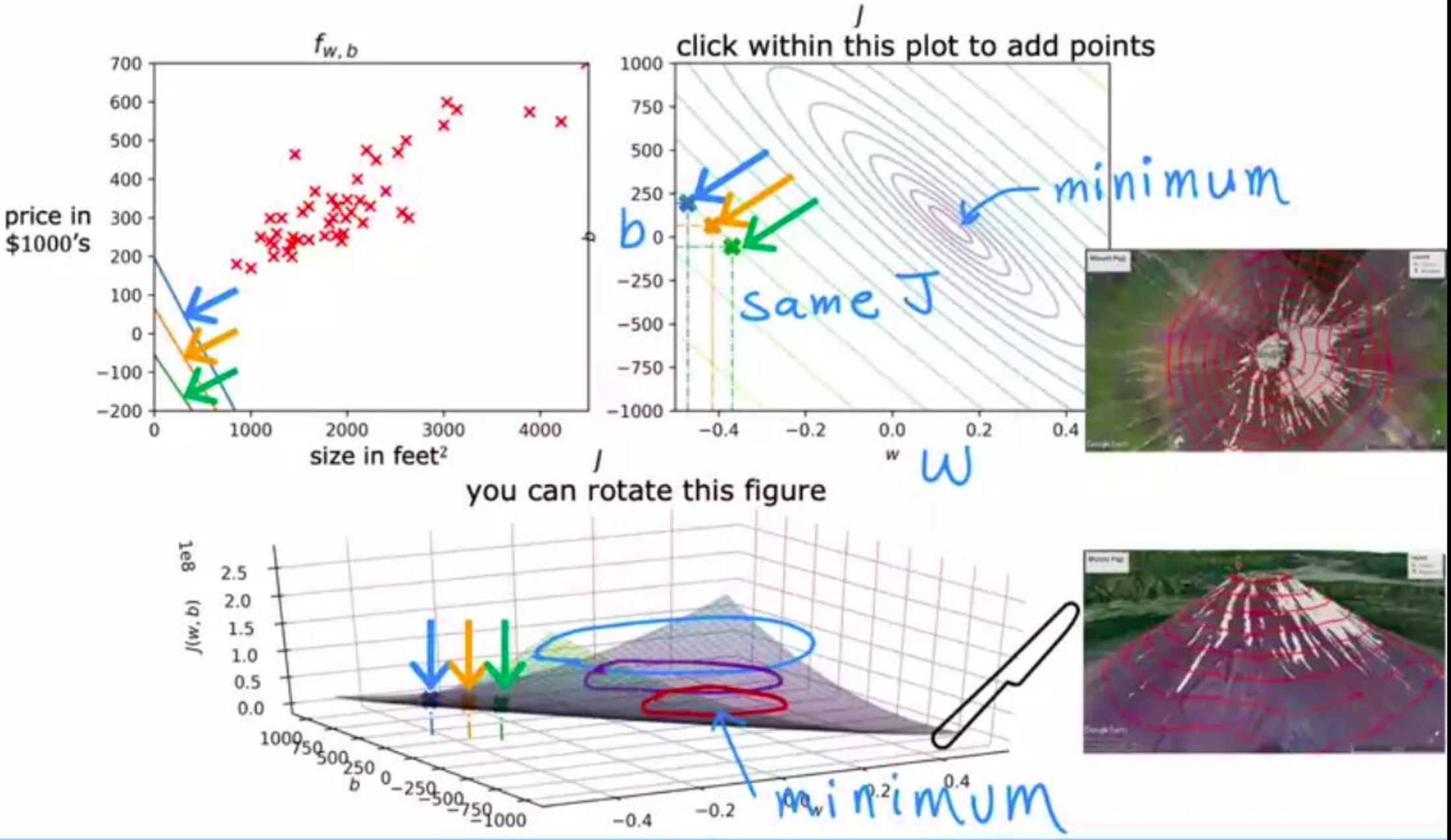
(function of w, b)



w, b


$J_{(w,b)}$
$J(w,b)$
800
700
600
500
400
300
200
100
20
15
10
5
0
−5
−10
−15
−20
b
w
$J(w=-10, b=-15)$


Mount Fuji
Legend
Contour
Mount Fuji
Mount Fuji
Google Earth
1 mi

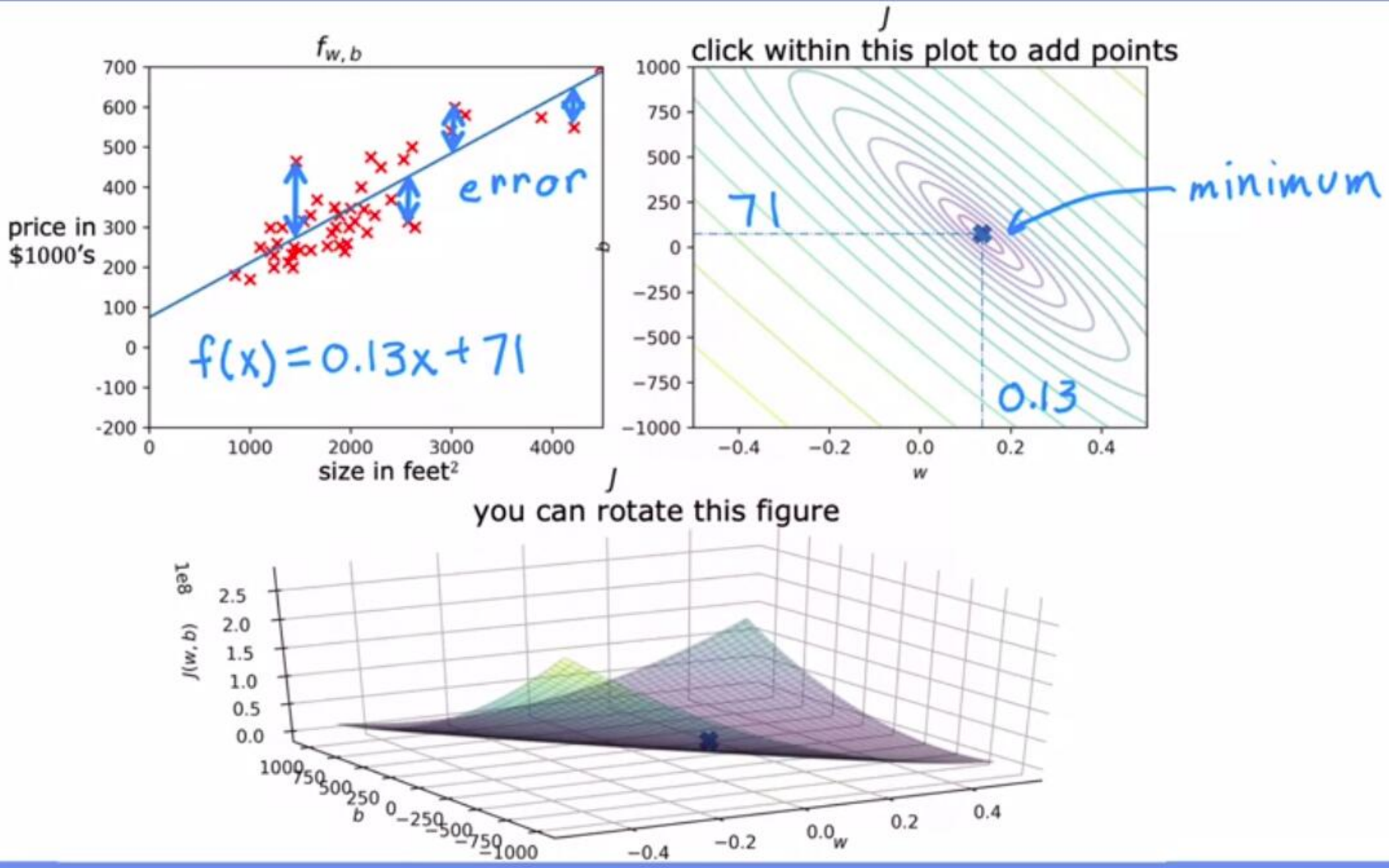

$f_{w,b}$
price in $1000's
size in feet²
J
click within this plot to add points
b
w
minimum
same J
J
you can rotate this figure
1e8
J(w, b)
minimum

Stanford
ONLINE

DeepLearning.AI

Linear Regression with One Variable

Visualization examples


$f_{w,b}$
price in $1000's
size in feet²
error
f(x)=0.13x+71
J
click within this plot to add points
minimum
71
0.13
w
b
J
you can rotate this figure
1e8
J(w, b)
b
w

Stanford
ONLINE

DeepLearning.AI

Training Linear Regression

Gradient Descent

Have some function $J(w,b)$ for linear regression or any function

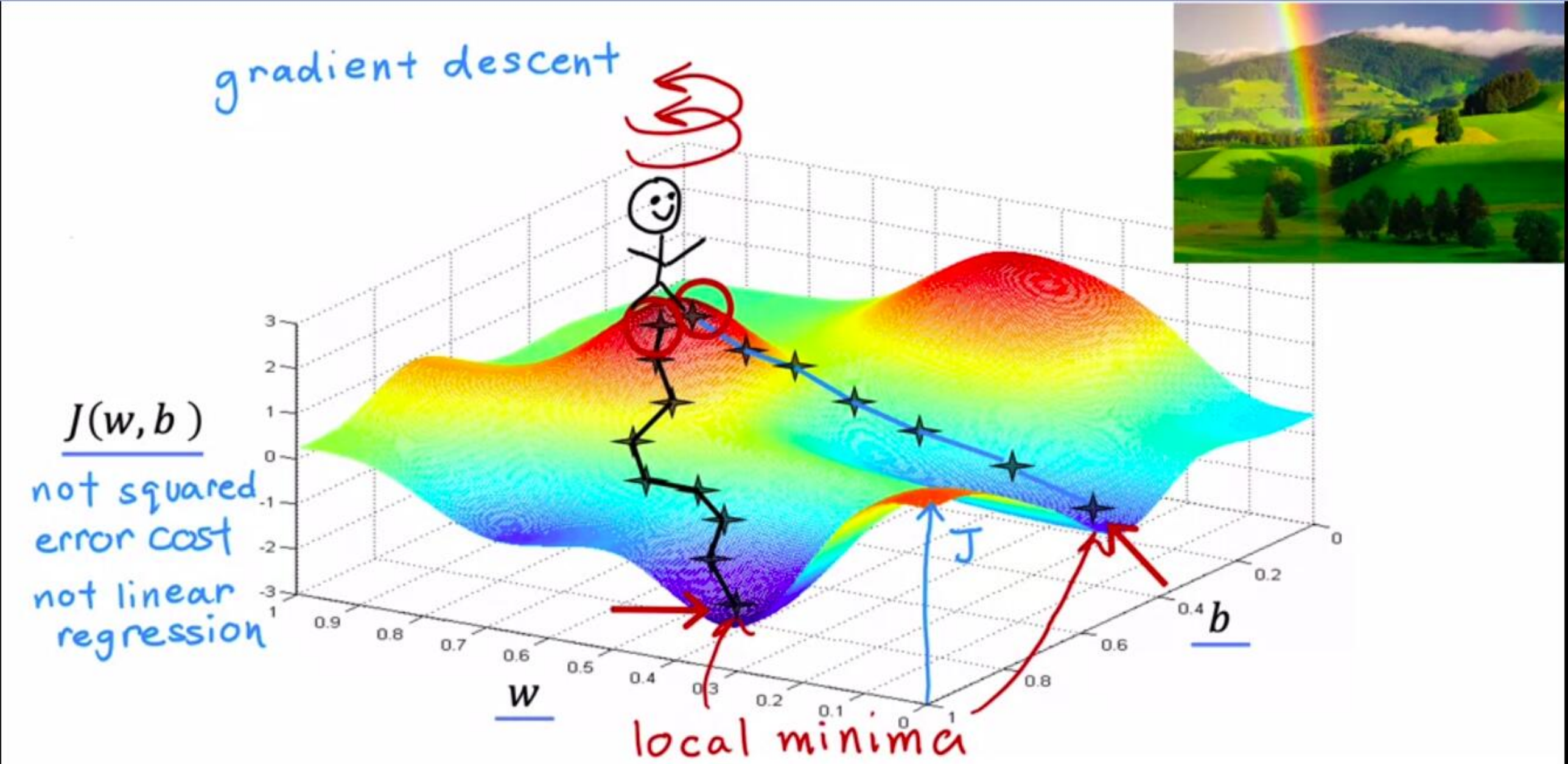
Want $\min_{w,b} J(w,b)$ $\min_{w_1,\ldots,w_n,b} J(w_1,w_2,\ldots,w_n,b)$

Outline:

- Start with some w,b (set $w=0$, $b=0$)
- Keep changing w,b to reduce $J(w,b)$
- Until we settle at or near a minimum

J not always

may have >1 minimum


gradient descent
$J(w,b)$
not squared error cost
not linear regression
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
w
J
0
0.2
0.4
0.6
0.8
1
b
local minima

Stanford
ONLINE

DeepLearning.AI

Training Linear Regression

Implementing Gradient Descent

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{d}{dw} J(w,b)$$

$$b = b - \alpha \frac{d}{db} J(w,b)$$

Learning rate

Derivative

Simultaneously update w and b

Assignment

$a = c$

$a = a + 1$

Code

Truth assertion

$a = c$

$a = a + 1$ ✗

Math

a==c

Correct: Simultaneous update

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$$w = tmp\_w$$

$$b = tmp\_b$$

Incorrect

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$w = tmp\_w$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$$b = tmp\_b$$

Stanford
ONLINE

DeepLearning.AI

Training Linear Regression

Gradient Descent Intuition

Gradient descent algorithm

repeat until convergence {

learning rate

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

derivative

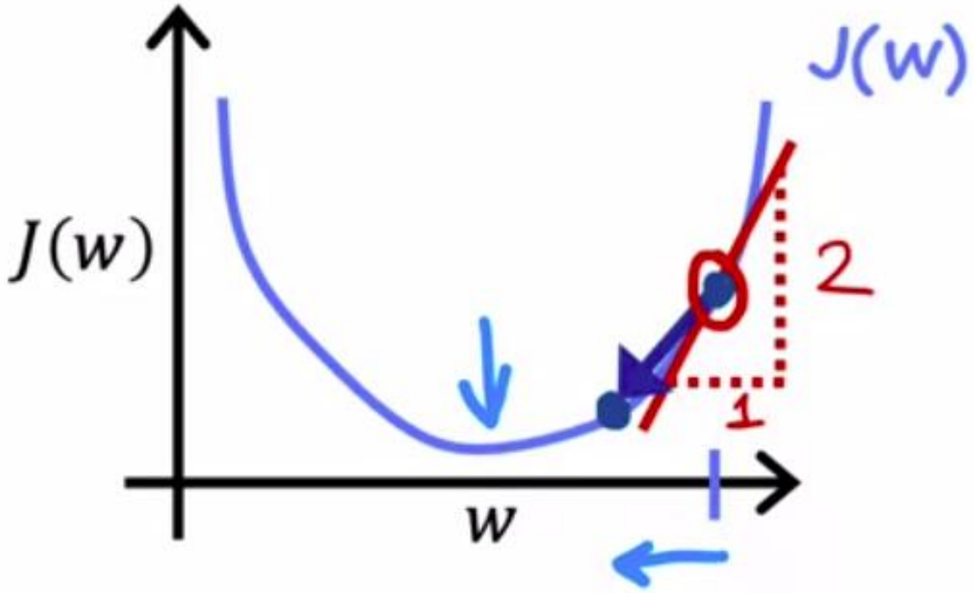
$$b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

}

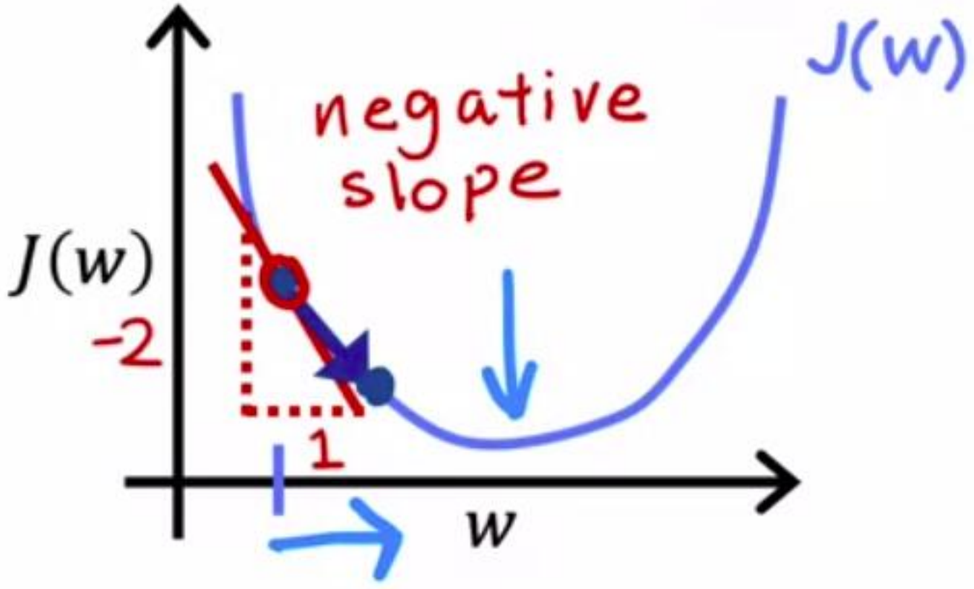
$J(w)$

$$w = w - \alpha \frac{\partial}{\partial w} J(w)$$

$$\min_w J(w)$$



$$w = w - \alpha \underbrace{\frac{d}{dw} J(w)}_{>0}$$



$$w = w - \alpha \cdot (positive\ number)$$

$$\underbrace{\frac{d}{dw} J(w)}_{<0}$$

$$w = w - \alpha \cdot (negative\ number)$$

Stanford
ONLINE

DeepLearning.AI

Training Linear Regression

Learning Rate

$$w = w - \alpha \frac{d}{dw} J(w)$$

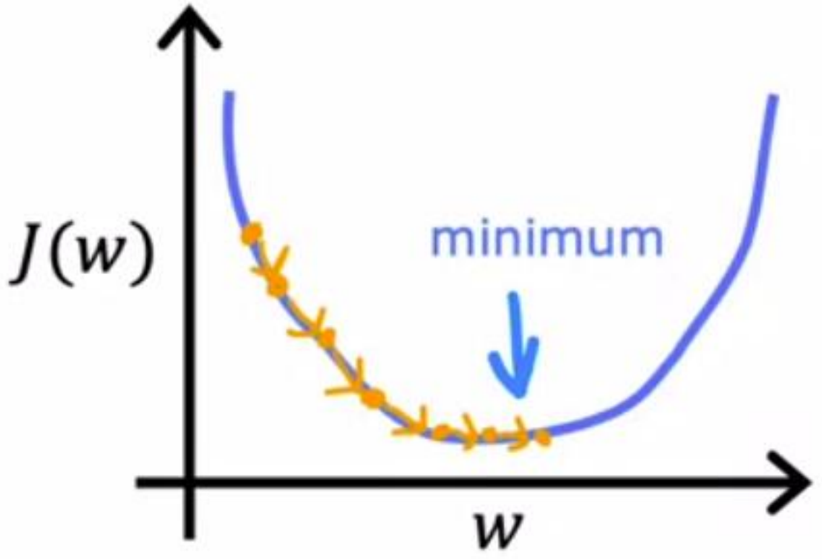
If α is too small...

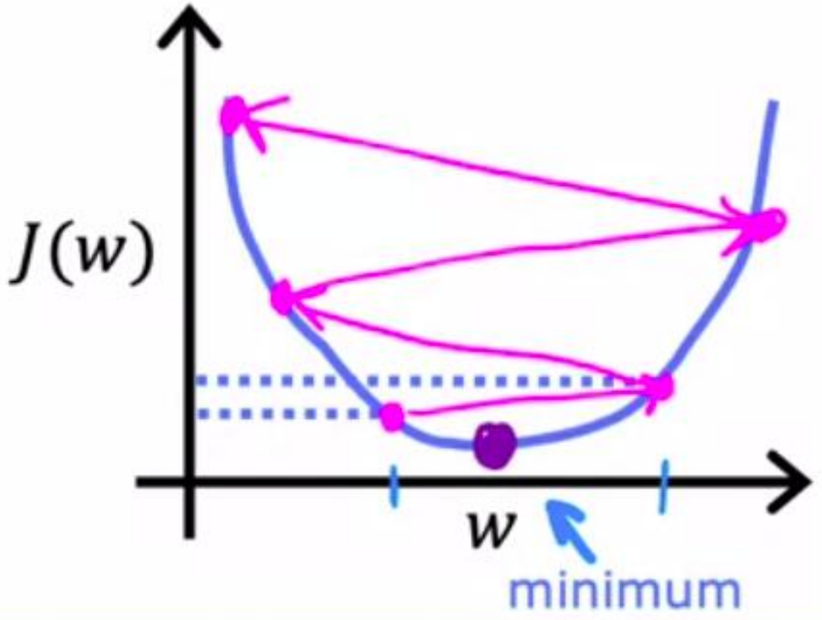
Gradient descent may be slow.

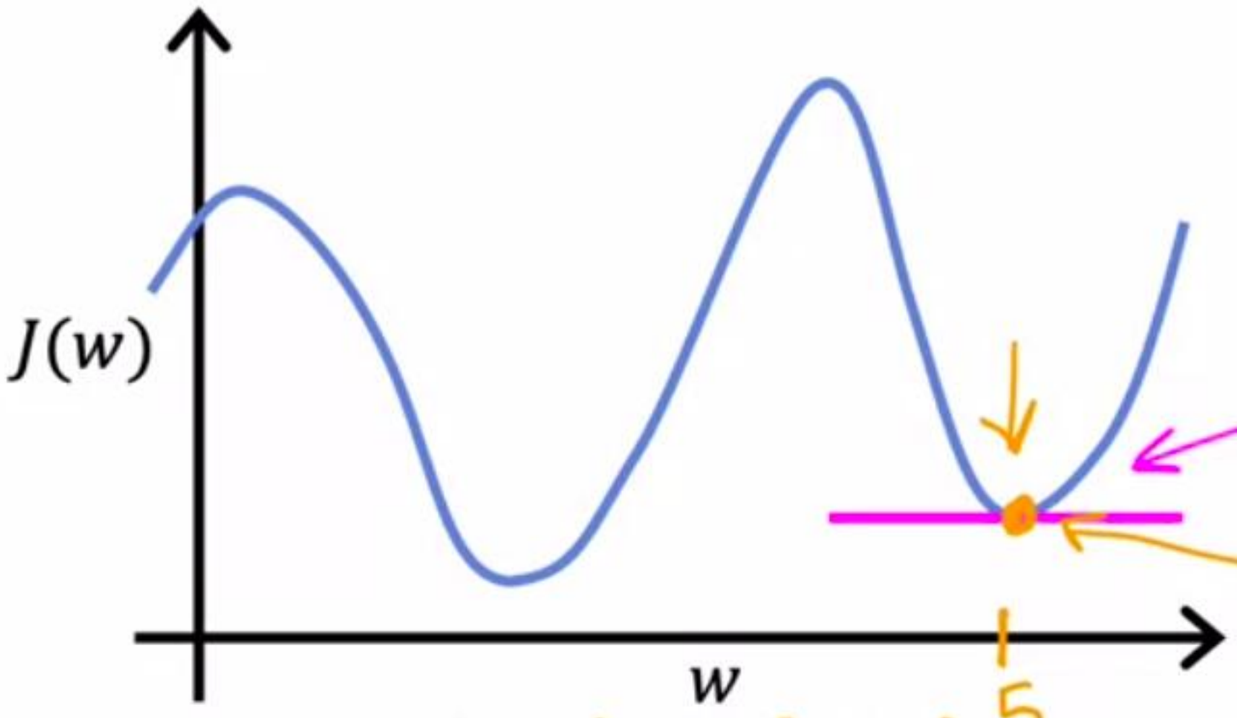
If α is too large...

Gradient descent may:

- Overshoot, never reach minimum
- Fail to converge, diverge







current value of w → 5

$$w = w - \alpha \frac{d}{dw} J(w)$$

0.1

$=0$

$5 - 0.1 \cdot 0$

$w = w - \alpha \cdot 0$

5 5

$w = w$

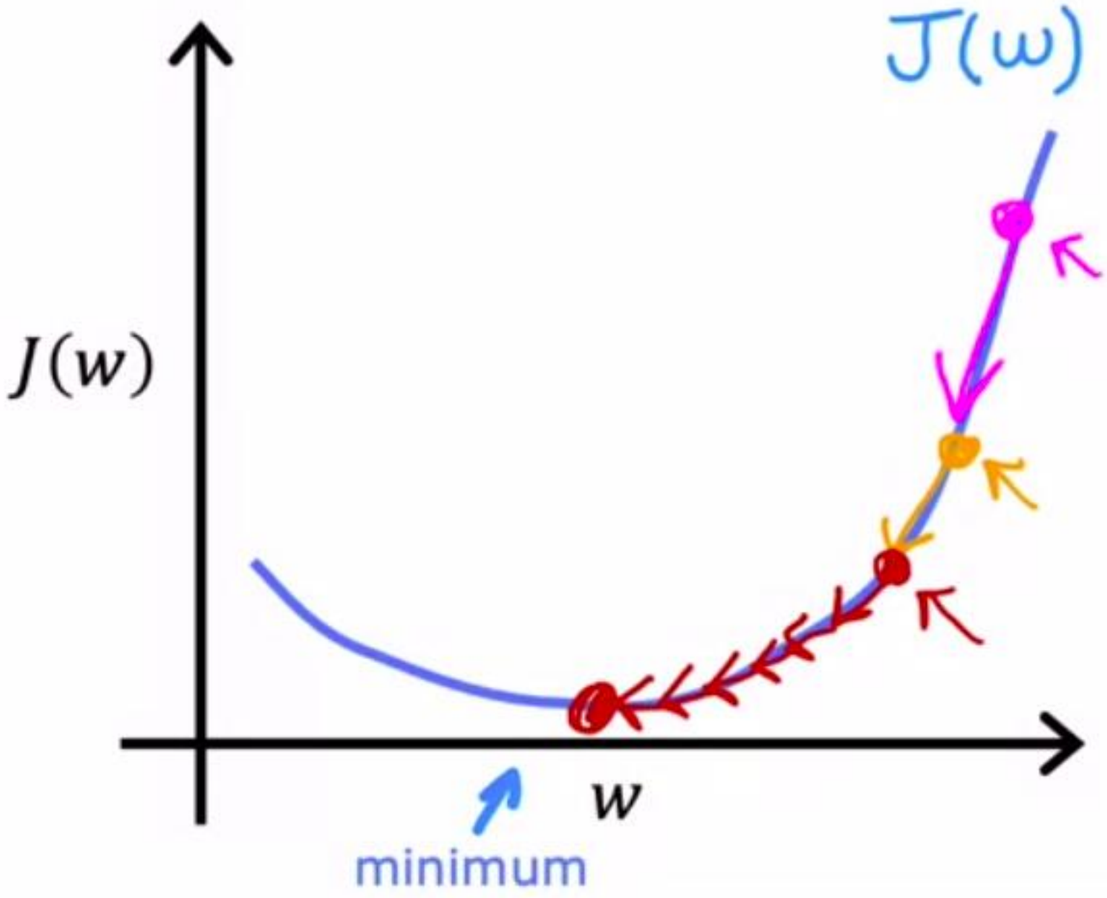
Can reach local minimum with fixed learning rate

$$w = w - \alpha \frac{d}{dw} J(w)$$

smaller

not as large

large



Near a local minimum,
- Derivative becomes smaller
- Update steps become smaller

Can reach minimum without decreasing learning rate α

Linear regression model

$$f_{w,b}(x) = wx + b$$

Cost function

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

Gradient descent algorithm

repeat until convergence {

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b) \rightarrow \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})x^{(i)}$$

$$b = b - \alpha \frac{\partial}{\partial b} J(w,b) \rightarrow \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})$$

}

(Optional)

$$\frac{\partial}{\partial w} J(w,b) = \frac{d}{dw} \frac{1}{2m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2 = \frac{d}{dw} \frac{1}{2m} \sum_{i=1}^{m} \left(wx^{(i)} + b - y^{(i)}\right)^2$$

$$= \frac{1}{2m} \sum_{i=1}^{m} \left(wx^{(i)} + b - y^{(i)}\right) 2x^{(i)} = \boxed{\frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)}\right) x^{(i)}}$$

$$\frac{\partial}{\partial b} J(w,b) = \frac{d}{db} \frac{1}{2m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2 = \frac{d}{db} \frac{1}{2m} \sum_{i=1}^{m} \left(wx^{(i)} + b - y^{(i)}\right)^2$$

$$= \frac{1}{2m} \sum_{i=1}^{m} \left(wx^{(i)} + b - y^{(i)}\right) 2 = \boxed{\frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(x^{(i)}) - y^{(i)}\right)}$$

no $x^{(i)}$

Gradient descent algorithm

$\frac{\partial}{\partial w} J(w,b)$

repeat until convergence {

$$w = w - \alpha \frac{1}{m} \sum_{i=1}^{m} (f_{w,b}(x^{(i)}) - y^{(i)}) \quad x^{(i)}$$

$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} (f_{w,b}(x^{(i)}) - y^{(i)})$$

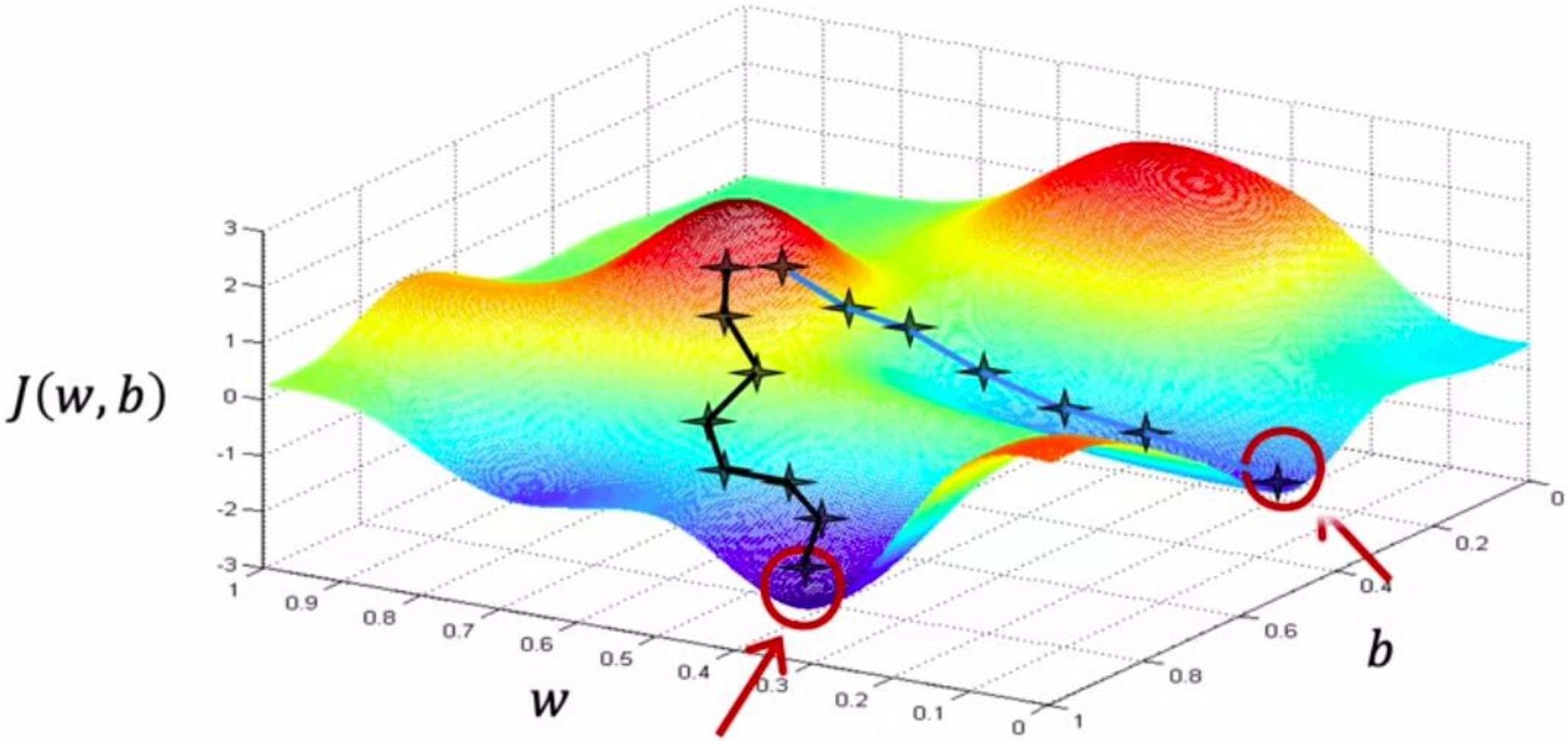
}

Update w and b simultaneously

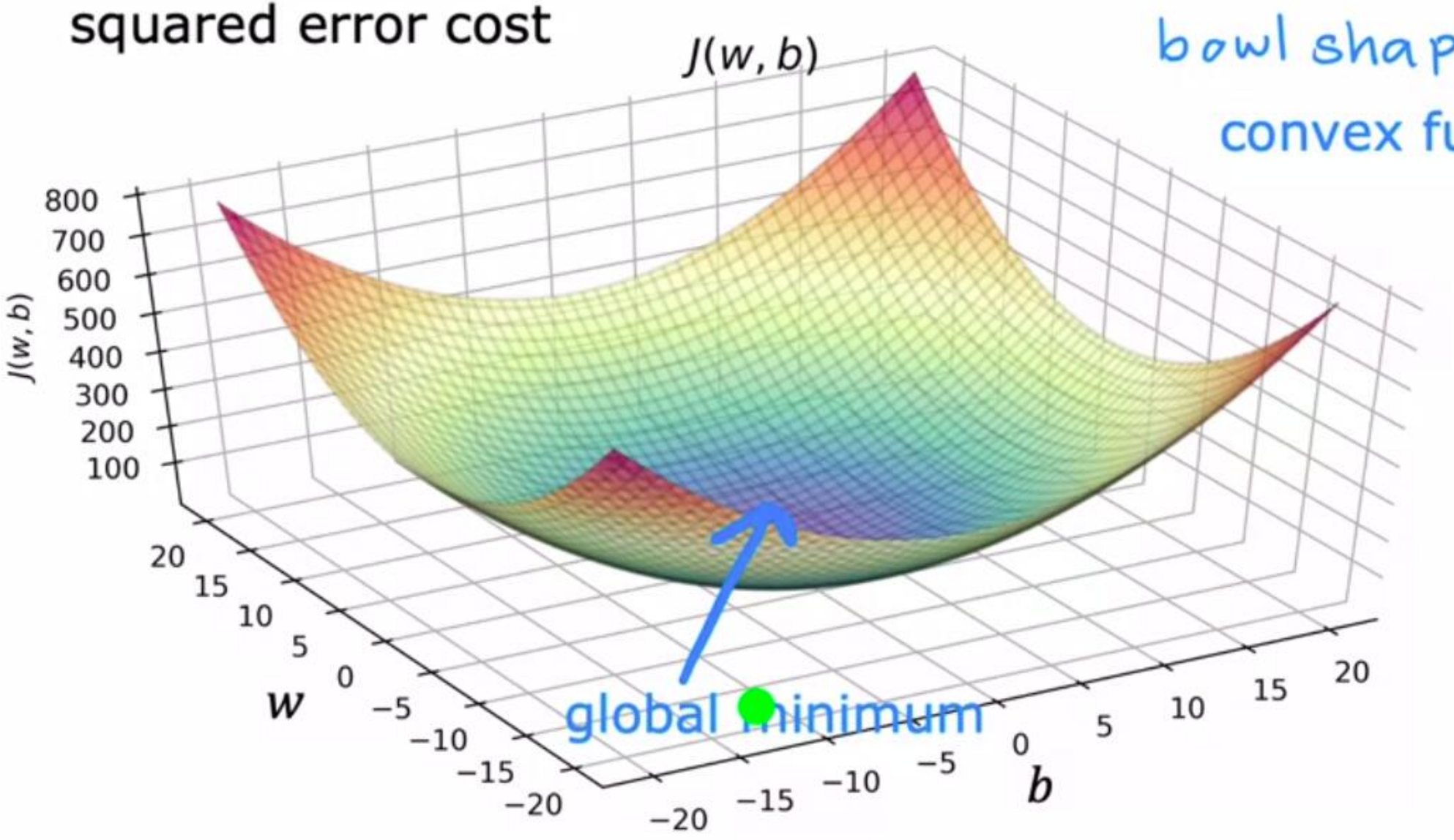
$\frac{\partial}{\partial b} J(w,b)$

$f_{w,b}(x^{(i)}) = wx^{(i)} + b$

More than one local minimum



squared error cost

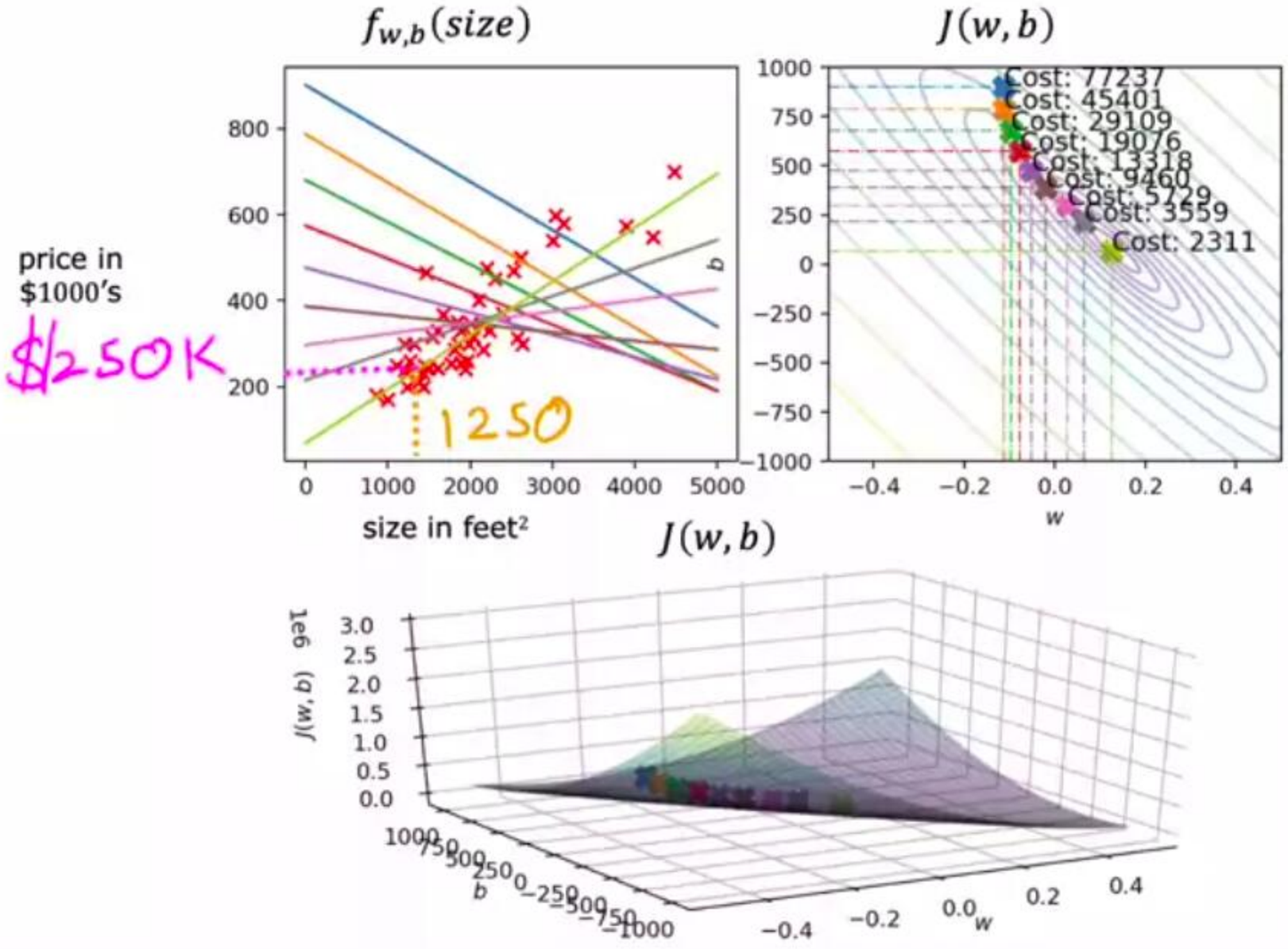


Stanford
ONLINE

DeepLearning.AI

Training Linear Regression

Running Gradient Descent


$f_{w,b}(size)$
price in $1000's
$250K
800
600
400
200
1250
0
1000
2000
3000
4000
5000
size in feet²
b
$J(w,b)$
1000
750
500
250
0
−250
−500
−750
−1000
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
Cost: 5729
Cost: 3559
Cost: 2311
−0.4
−0.2
0.0
0.2
0.4
w
$J(w,b)$
1e6
$J(w,b)$
3.0
2.5
2.0
1.5
1.0
0.5
0.0
1000
750
500
250
0
−250
−500
−750
−1000
b
−0.4
−0.2
0.0
w
0.2
0.4

"Batch" gradient descent

"Batch": Each step of gradient descent uses all the training examples.

other gradient descent: subsets

	x size in feet^2	y price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

$m = 47$

$$\sum_{i=1}^{m} \left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)^2$$