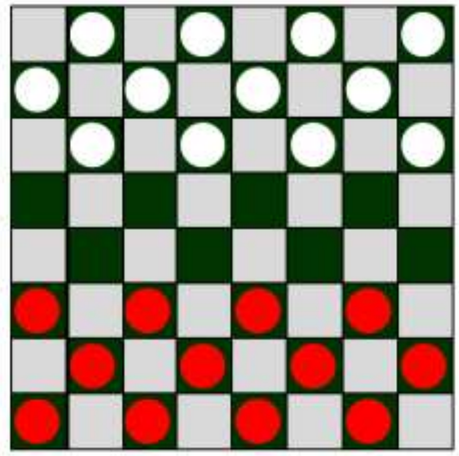
Machine learning

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

Arthur Samuel (1959)



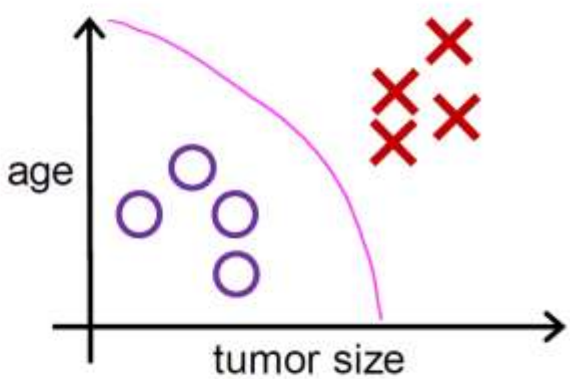
Supervised learning

Learns from being given "right answers"

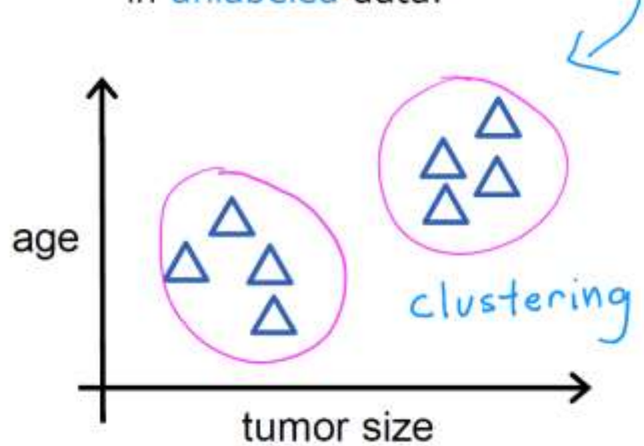
Regression
Predict a number
infinitely many possible outputs

Classification
predict categories
small number of possible outputs

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



Unsupervised learning
Find something interesting in unlabeled data.



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.

---

training set — features, targets

↓

learning algorithm

↓

$x \rightarrow \boxed{f} \rightarrow \hat{y}$ "y-hat" $\hat{y}$

feature ~~hypothesis~~ function

model

prediction (estimated y)

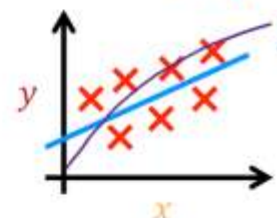
target

size → $\boxed{f}$ → 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

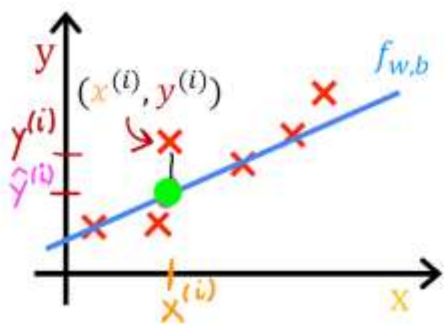
Training set

features size in feet² (x)	targets 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?



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

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

Cost function: Squared error cost function

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

m = number of training examples

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

intuition (next!)

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

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

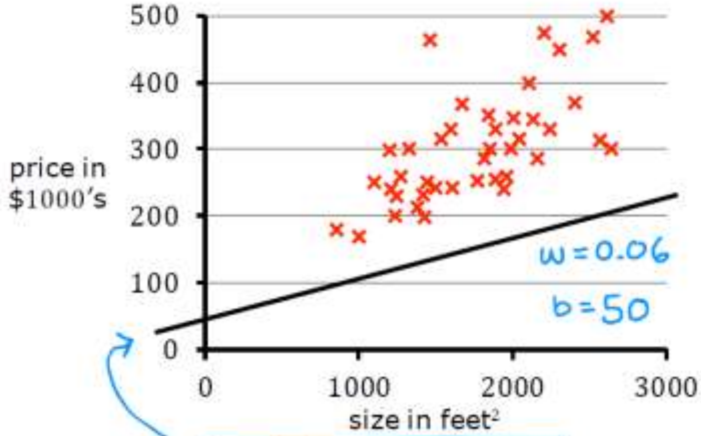
Parameters $\quad w, b \quad$ ~~before: b=0~~

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

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

$f_{w,b}$

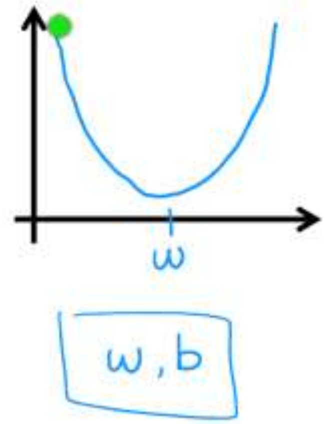
(function of x)



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

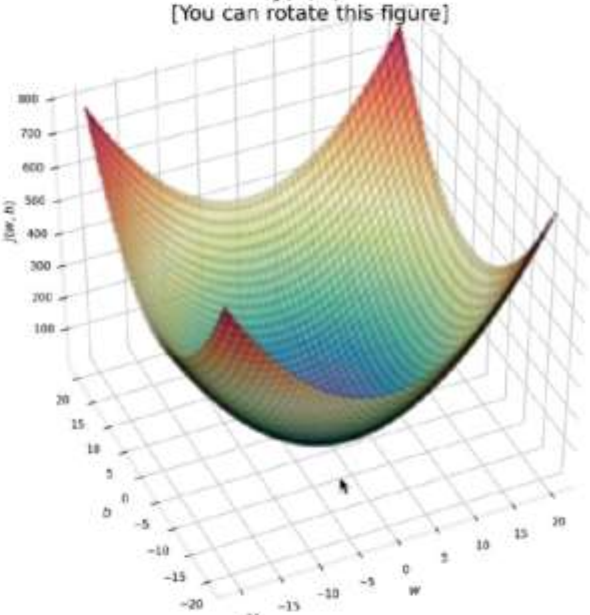
J

(function of w, b)



3D surface plot

$J(w, b)$
[You can rotate this figure]



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

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

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

may have >1 minimum

J not always

Gradient descent algorithm

Repeat until convergence

$$\left\{\begin{array}{l} w = w - \alpha \boxed{\frac{\partial}{\partial w} J(w,b)} \\ b = b - \alpha \frac{\partial}{\partial b} J(w,b) \end{array}\right.$$

Learning rate

Derivative

Simultaneously update w and b

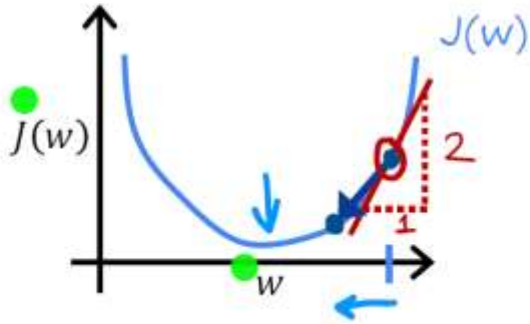
Assignment	Truth assertion
$a = c$	$a = c$
$a = a + 1$	~~$a = a + 1$~~
Code	Math
	a==c

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

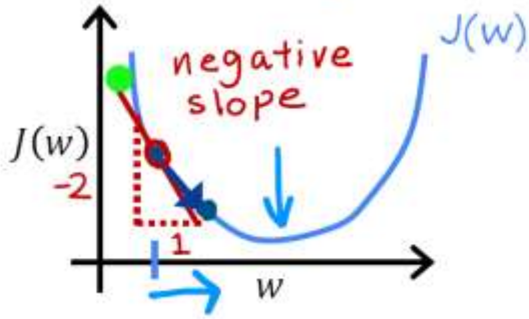
$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$
$w = tmp\_w$
$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$
$b = tmp\_b$



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

> 0

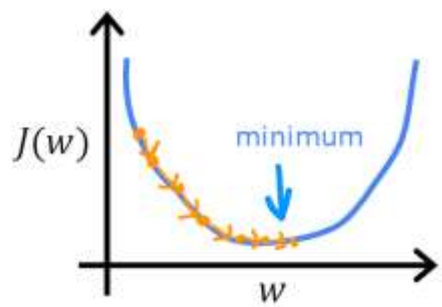
$$w = w - \alpha \cdot (positive\ number)$$



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

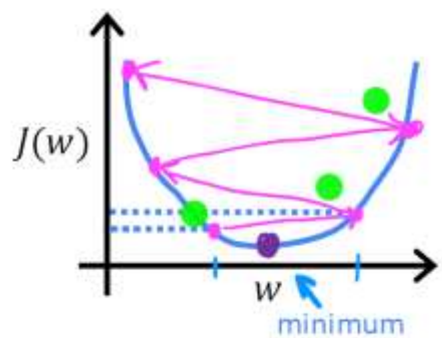
$$w = w - \alpha \cdot (negative\ number)$$

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



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

Can reach local minimum with fixed learning rate α

smaller

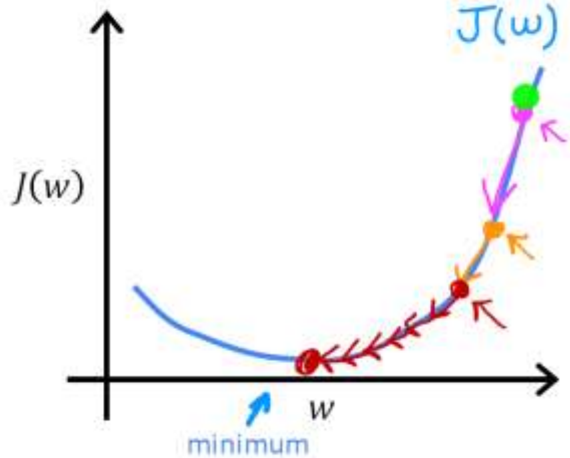
not as large

large

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

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 {

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

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

}

next slide is optional!

(Optional)

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

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

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

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

no $x^{(i)}$

Gradient descent algorithm

repeat until convergence {

$$w = w - \alpha \underbrace{\boxed{\frac{1}{m}\sum_{i=1}^{m}\left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right) x^{(i)}}}_{\frac{\partial}{\partial w} J(w,b)}$$

$$b = b - \alpha \underbrace{\boxed{\frac{1}{m}\sum_{i=1}^{m}\left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)}}_{\frac{\partial}{\partial b} J(w,b)}$$

}

Update w and b simultaneously

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

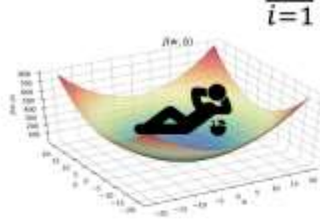
"Batch" gradient descent

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

other gradient descent: subsets

	x size in feet²	y price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

$m = 47 \longrightarrow \sum_{i=1}^{m}\left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)^2$



$f_{\vec{w},b}(\vec{x}) = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$

$\vec{w} = [w_1 \ w_2 \ w_3 \ \dots \ w_n]$ parameters of the model

b is a number

vector $\vec{x} = [x_1 \ x_2 \ x_3 \ \dots \ x_n]$

$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b = w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n + b$

dot product

multiple linear regression

(not multivariate regression)

	Previous notation	Vector notation
Parameters	$w_1, \cdots, w_n$	$\vec{w} = [w_1 \quad \cdots \quad w_n]$ vector of length n
	b	b still a number
Model	$f_{\vec{w},b}(\vec{x}) = w_1 x_1 + \cdots + w_n x_n + b$	$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$ dot product
Cost function	$J(\underbrace{w_1, \cdots, w_n}, b)$	$J(\vec{w}, b)$

Gradient descent

Previous notation	Vector notation
repeat {	repeat {
$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\underbrace{w_1, \cdots, w_n}, b)$	$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w}, b)$
$b = b - \alpha \frac{\partial}{\partial b} J(\underbrace{w_1, \cdots, w_n}, b)$	$b = b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b)$
}	}

Gradient descent

One feature

repeat {

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

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

simultaneously update w, b

}

n features ($n \geq 2$)

repeat {

$j=1$

$$\underbrace{w_1} = w_1 - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right) - y^{(i)}\right)\underbrace{x_1^{(i)}}} \rightarrow \frac{\partial}{\partial w_1}J(\vec{w},b)$$

$\vdots$

$j=n$

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

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

simultaneously update w_j (for $j = 1, \cdots, n$) and b

}

An alternative to gradient descent

→ Normal equation
- Only for linear regression
- Solve for w, b without iterations

Disadvantages
- Doesn't generalize to other learning algorithms.
- Slow when number of features is large (> 10,000)

What you need to know
- Normal equation method may be used in machine learning libraries that implement linear regression.
- Gradient descent is the recommended method for finding parameters w,b