Copyright Notice

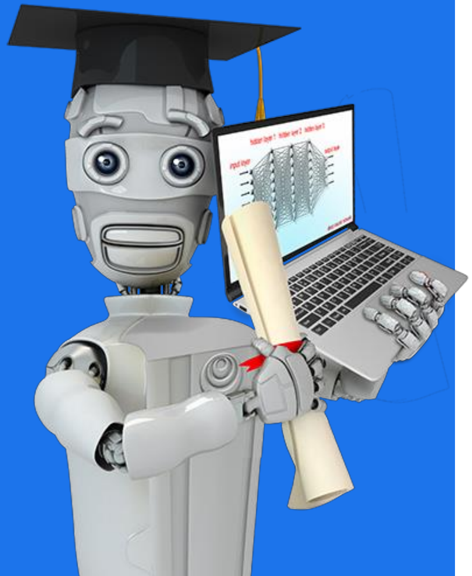
These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see https://creativecommons.org/licenses/by-sa/2.0/legalcode

Stanford ONLINE

DeepLearning.AI



Classification

Motivations

Classification

Question	Answer "y"	
Is this email spam?	no	yes
Is the transaction fraudulent?	no	yes
Is the tumor malignant?	no	yes

y can only be one of two values

"binary classification"

class = category

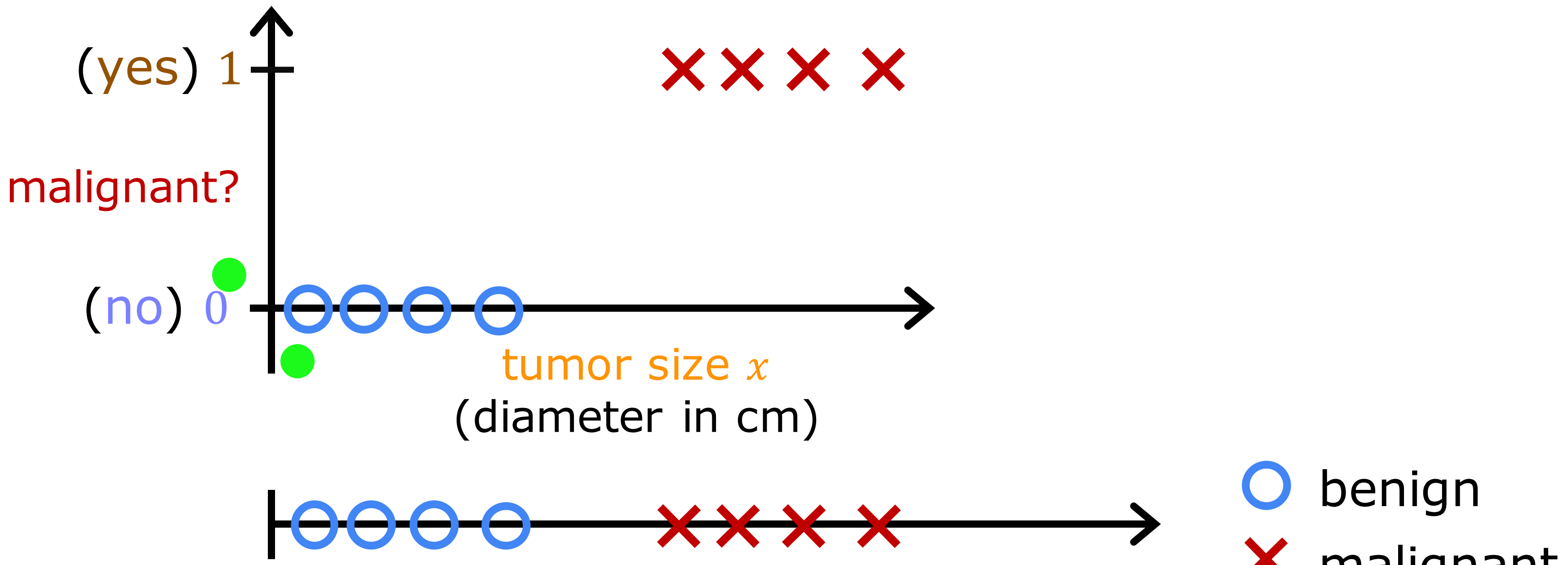
false true

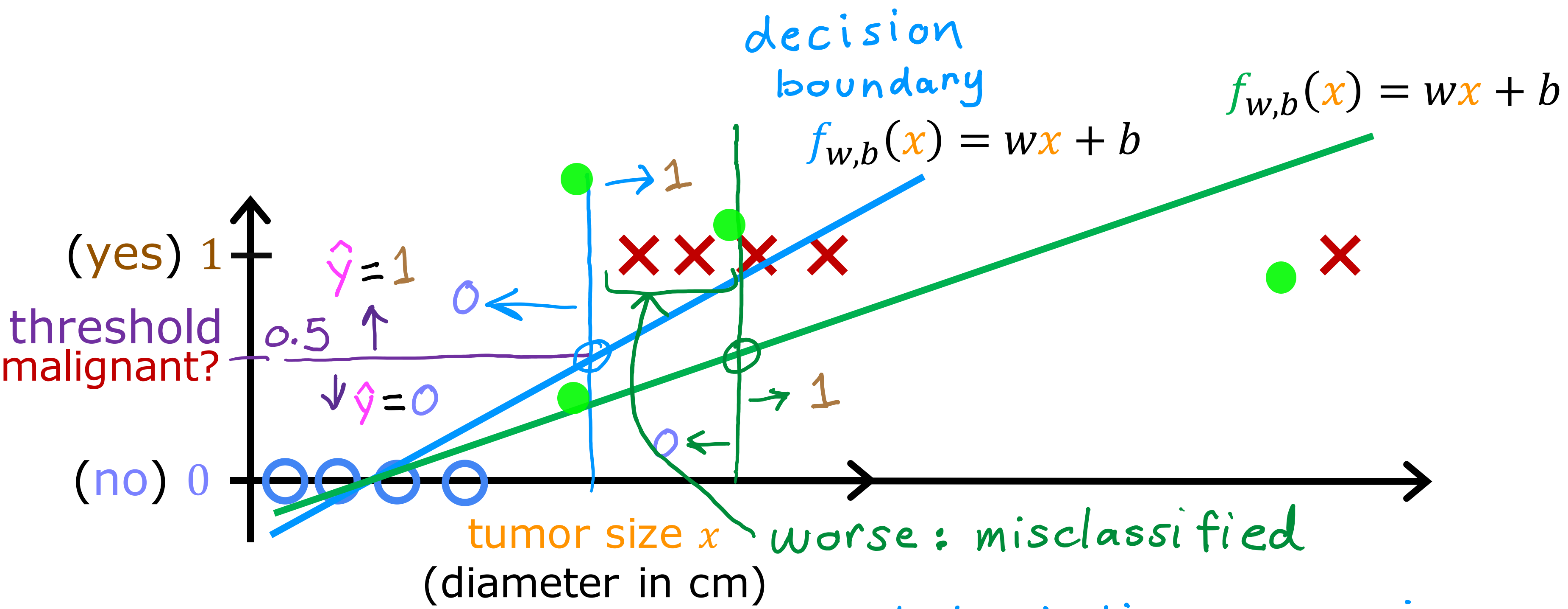
0 1

useful for classification

"negative class" ≠ "bad"
absence

"positive class" ≠ "good"
presence


(yes) 1
malignant?
(no) 0
tumor size x
(diameter in cm)
benign
malignant



if $f_{w,b}(x) < 0.5 \rightarrow \hat{y} = 0$

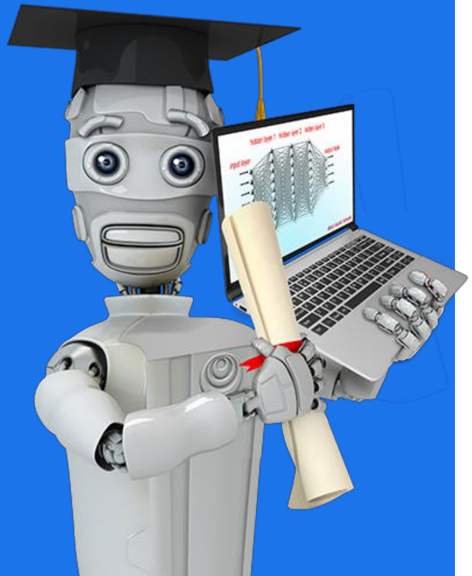
if $f_{w,b}(x) \geq 0.5 \rightarrow \hat{y} = 1$

next: logistic regression

$\underbrace{\text{logistic regression}}_{\text{classification}}$

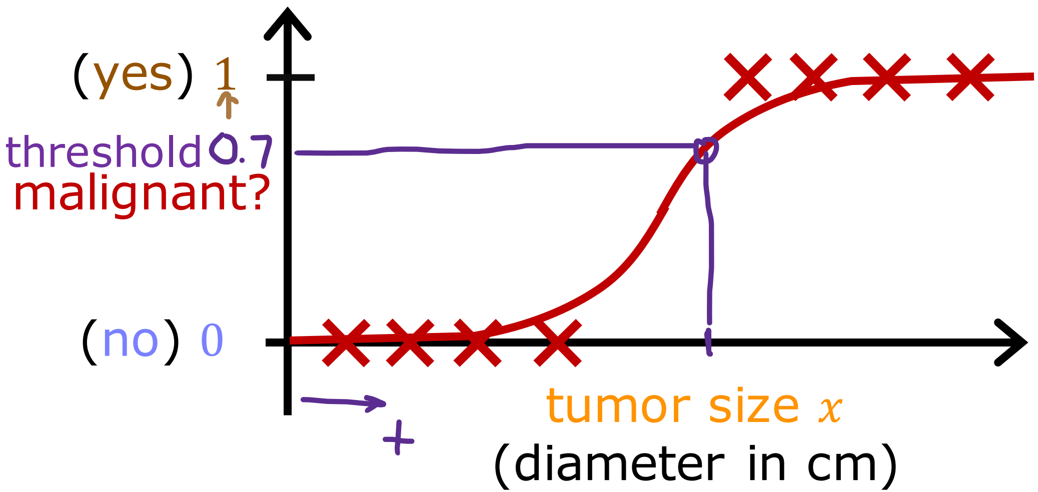
Stanford
ONLINE

DeepLearning.AI

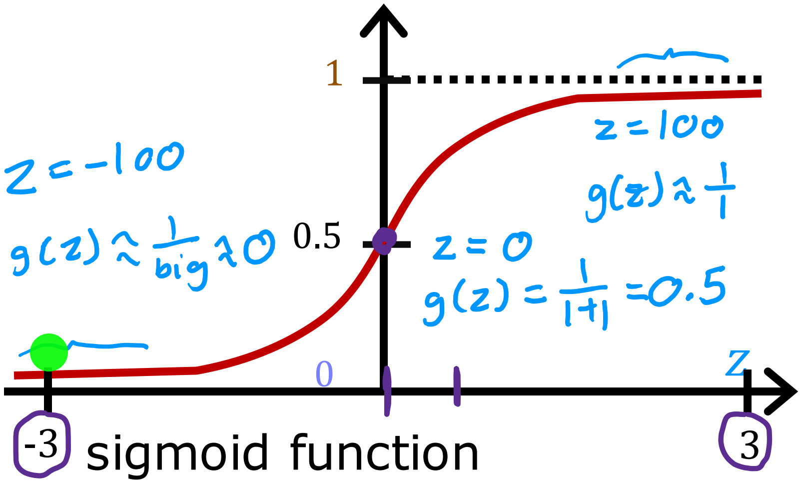


Classification

Logistic Regression



Want outputs between 0 and 1



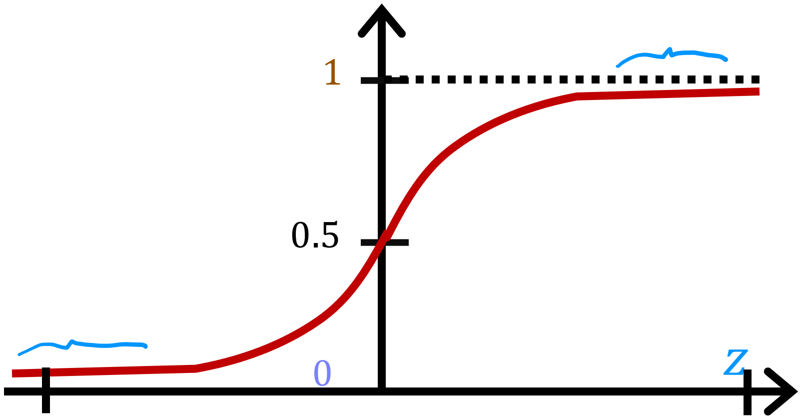
sigmoid function

logistic function

outputs between 0 and 1

$$g(z) = \frac{1}{1+e^{-z}} \qquad 0 < g(z) < 1$$

Want outputs between 0 and 1



sigmoid function

logistic function

$\boxed{\text{outputs between } 0 \text{ and } 1 \qquad g(z) = \frac{1}{1+e^{-z}} \qquad 0 < g(z) < 1}$

$f_{\vec{w},b}(\vec{x})$

$\boxed{\vec{w} \cdot \vec{x} + b}$

$\downarrow$

z

$\downarrow$

$\boxed{g(z) = \frac{1}{1+e^{-z}}}$

$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w} \cdot \vec{x} + b}_{z}) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}+b)}}$$

"logistic regression"

$e \approx 2.7$

Interpretation of logistic regression output

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}+b)}}$$

"probability" that class is 1

Example:

x is "tumor size"

y is 0 (not malignant)

or 1 (malignant)

$$f_{\vec{w},b}(\vec{x}) = 0.7$$

70% chance that y is 1

P(1)

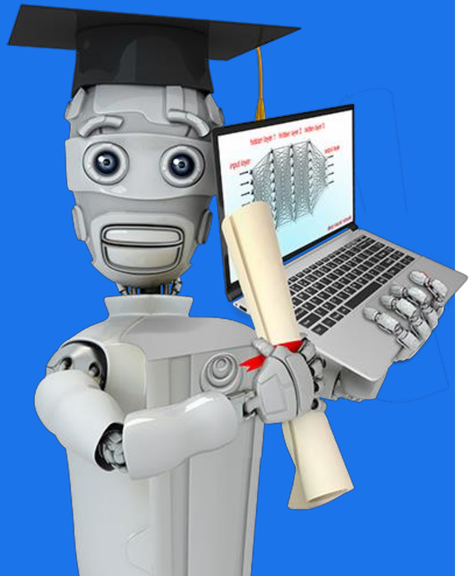
$$f_{\vec{w},b}(\vec{x}) = P(y = 1|\vec{x};\vec{w},b)$$

Probability that y is 1,
given input $\vec{x}$, parameters $\vec{w}$,b

$$P(y = 0) + P(y = 1) = 1$$

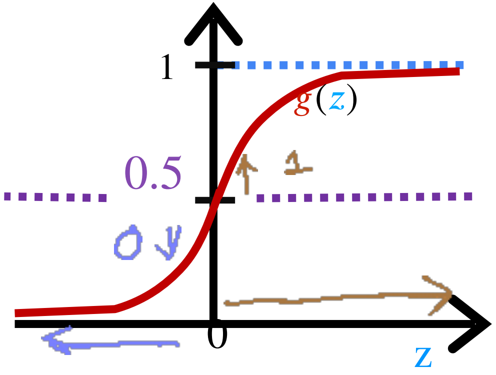
Stanford
ONLINE

DeepLearning.AI



Classification

Decision Boundary



$f_{\vec{w},b}(\vec{x})$

$z = \vec{w} \cdot \vec{x} + b$

↓

z

↓

$g(z) = \frac{1}{1+e^{-z}}$

$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w} \cdot \vec{x}}_{z} + b) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$$

$$= P(y = 1 \mid x; \vec{w}, b)$$ 0.7 0.3

0 or 1? threshold

Is $f_{\vec{w},b}(\vec{x}) \geq 0.5$?

Yes: $\hat{y} = 1$ No: $\hat{y} = 0$

When is $f_{\vec{w},b}(\vec{x}) \geq 0.5$?

$g(z) \geq 0.5$

$z \geq 0$	$z < 0$
$\vec{w} \cdot \vec{x} + b \geq 0$	$\vec{w} \cdot \vec{x} + b < 0$
$\hat{y} = 1$	$\hat{y} = 0$

Decision boundary

$$f_{\vec{w},b}(\vec{x}) = g(z) = g(\underbrace{\underset{1}{w_1}x_1 + \underset{1}{w_2}x_2 + \underset{-3}{b}})$$

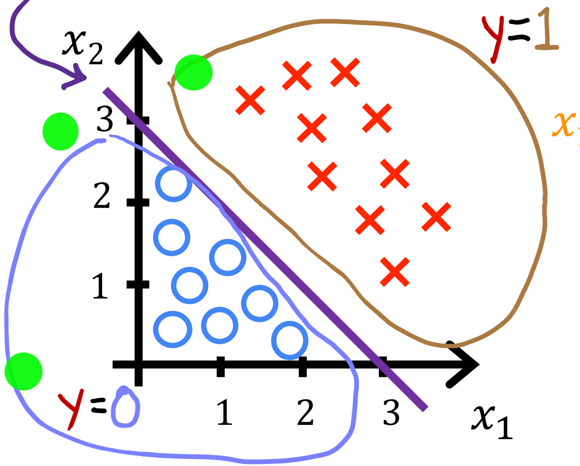
Decision boundary $z = \vec{w} \cdot \vec{x} + b = 0$

$z = x_1 + x_2 - 3 = 0$

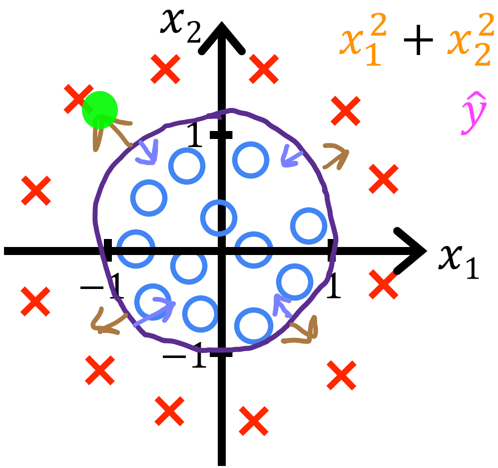
$x_1 + x_2 = 3$

$x_1 + x_2 \geq 3$

$x_1 + x_2 < 3$



Non-linear decision boundaries



$x_1^2 + x_2^2 \geq 1$

$\hat{y} = 1$

$x_1^2 + x_2^2 < 1$

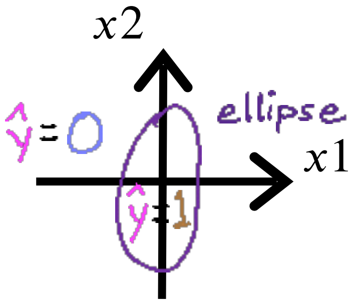
$\hat{y} = 0$

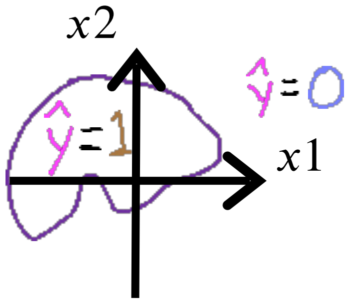
$$\overbrace{\underset{1}{w_1} x_1^2 + \underset{1}{w_2} x_2^2 + \underset{-1}{b}}^{z}$$

decision boundary $z = x_1^2 + x_2^2 - 1 = 0$

$x_1^2 + x_2^2 = 1$

Non-linear decision boundaries

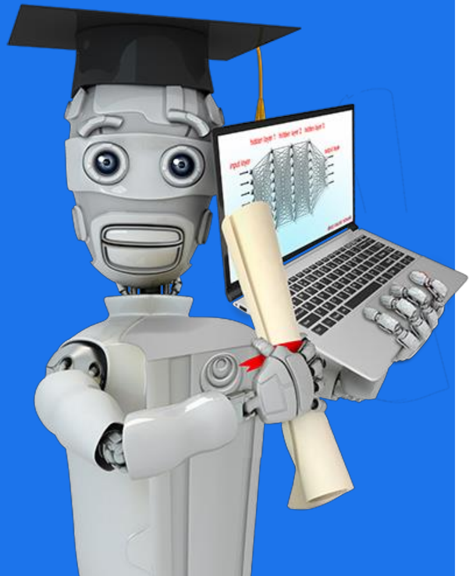




$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1x_1 + w_2x_2$$
$$+ w_3x_1^2 + w_4x_1x_2 + w_5x_2^2$$
$$+ w_6x_1^3 + \cdots + b)$$

Stanford ONLINE

DeepLearning.AI



Cost Function

Cost Function for Logistic Regression

Training set

	tumor size (cm) x_1	...	patient's age x_n	malignant? y
$i=1$	10		52	1
⋮	2		73	0
	5		55	0
	12		49	1
$i=m$	...		...	...

$i = 1, \ldots, m$ ← training examples

$j = 1, \ldots, n$ ← features

$\boxed{\text{target } y \text{ is } 0 \text{ or } 1}$

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}+b)}}$$

How to choose $\vec{w} = [w_1 \quad w_2 \quad \cdots \quad w_n]$ and b?

Squared error cost

cost

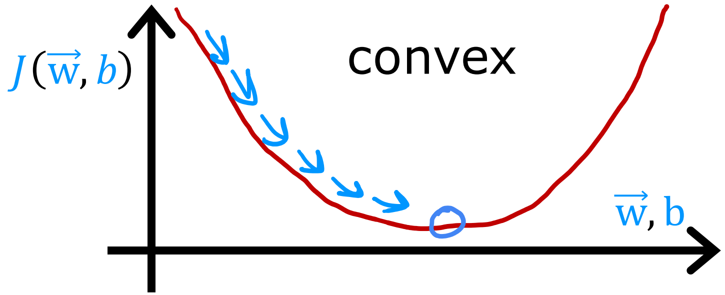
$$J(\vec{w},b) = \frac{1}{m}\sum_{i=1}^{m} \boxed{\frac{1}{2}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2}$$

average of training set

loss $\boxed{L\left(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right)}$

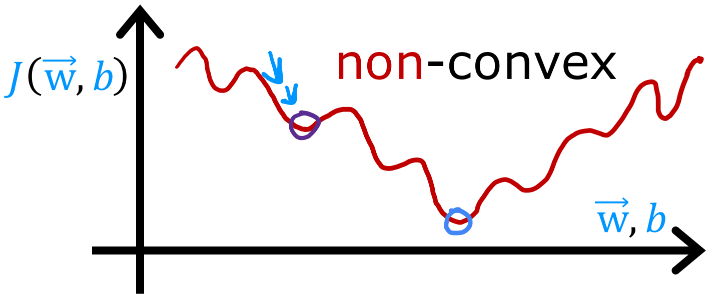
linear regression

$$f_{\vec{w},b}(\vec{x}) = \vec{w}\cdot\vec{x} + b$$



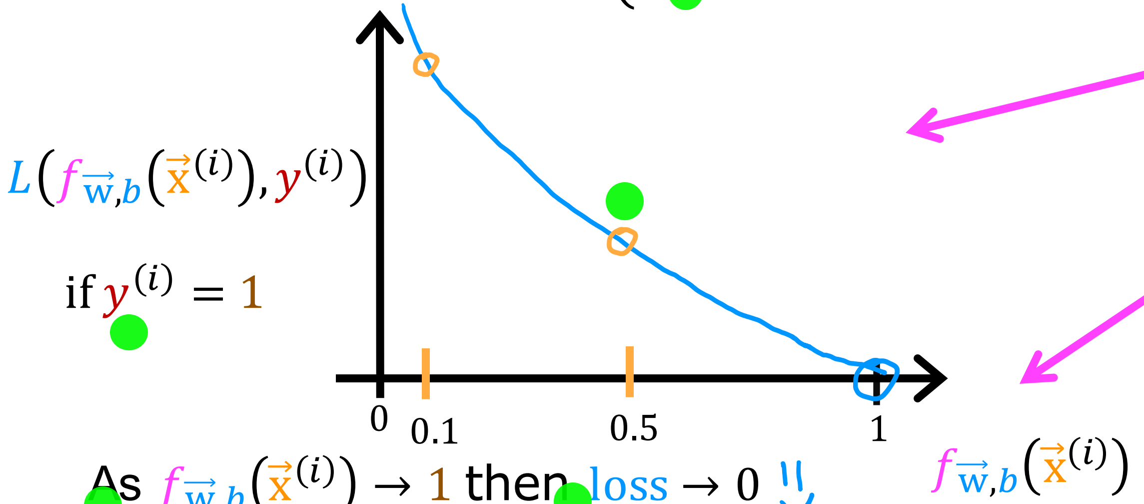
logistic regression

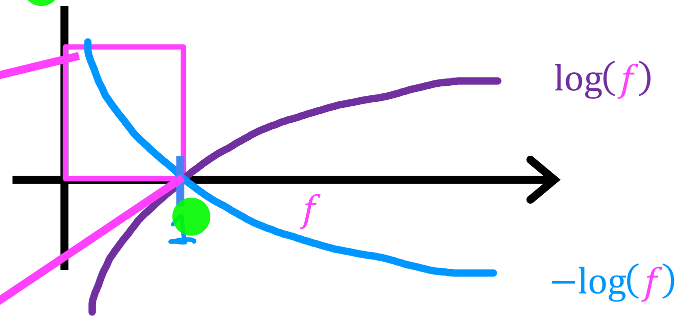
$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$$



Logistic loss function

$$L\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right), y^{(i)}\right) = \begin{cases} -\log\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$





As $f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) \to 1$ then loss $\to 0$

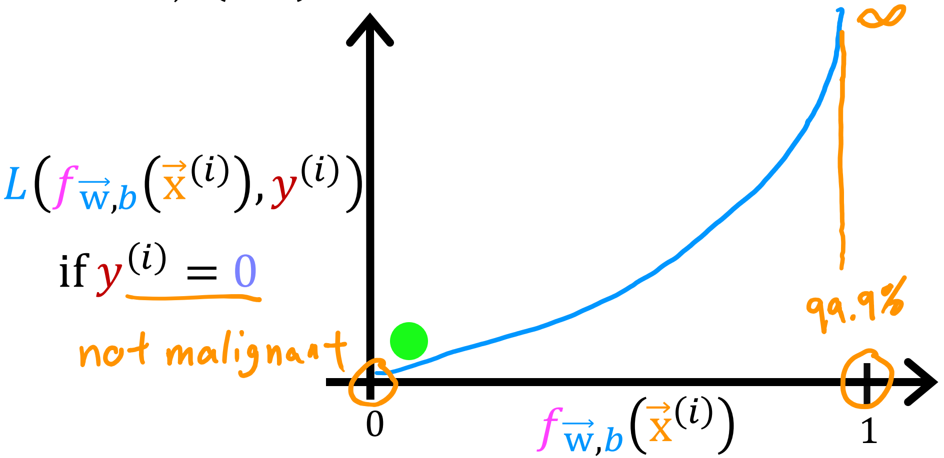
As $f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) \to 0$ then loss $\to \infty$

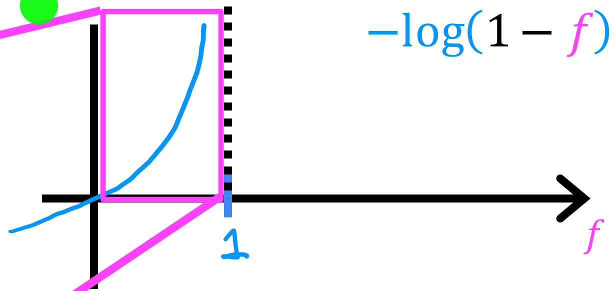
Loss is lowest when $f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right)$ predicts close to true label $y^{(i)}$.

Logistic loss function

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \begin{cases} -\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$

As $f_{\vec{w},b}\left(\vec{x}^{(i)}\right) \rightarrow 0$ then loss $\rightarrow 0$





As $f_{\vec{w},b}\left(\vec{x}^{(i)}\right) \rightarrow 1$ then loss $\rightarrow \infty$

The further prediction $f_{\vec{w},b}\left(\vec{x}^{(i)}\right)$ is from target $y^{(i)}$, the higher the loss.

Cost

cost

$$J(\vec{w}, b) = \frac{1}{m} \sum_{i=1}^{m} \underbrace{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right)}_{\text{loss}}$$

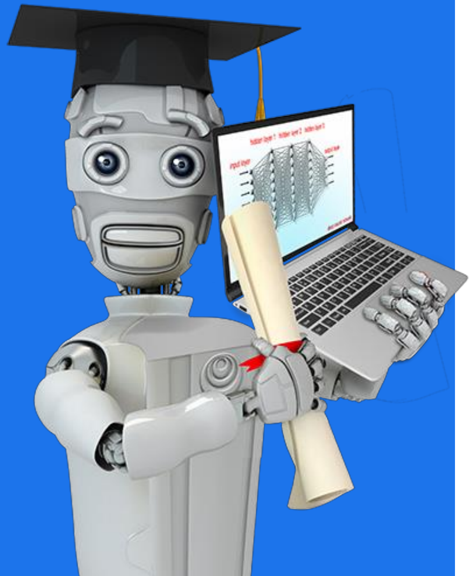
$$= \begin{cases} -\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$

convex
↳ can reach a global minimum

find w, b that minimize cost J

Stanford ONLINE

DeepLearning.AI



Cost Function

Simplified Cost Function for Logistic Regression

Simplified loss function

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \begin{cases} \boxed{-\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)} & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$

$$\boxed{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = -\underbrace{y^{(i)}}_{1}\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) \cancel{- \left(1 - y^{(i)}\right)\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)}}$$

$\underbrace{(1-1)}_{0}$

if $y^{(i)} = 1$:

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \underbrace{-1 \log\left(f(\vec{x})\right)}$$

Simplified loss function

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \begin{cases} -\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ \boxed{-\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)} & \text{if } y^{(i)} = 0 \end{cases}$$

$$\boxed{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = -\underbrace{\cancel{y^{(i)}\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)}}_{0} - \underbrace{\left(1 - y^{(i)}\right)}_{(1-0)}\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)}$$

if $y^{(i)} = 1$:

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = -1 \log\left(f(\vec{x})\right)$$

if $y^{(i)} = 0$:

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \underbrace{-(1-0)\log\left(1 - f(\vec{x})\right)}$$

Simplified cost function

loss

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = -\, y^{(i)}\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) - \left(1-y^{(i)}\right)\log\left(1-f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)$$

cost

$$J(\vec{w},b) = \frac{1}{m}\sum_{i=1}^{m}\left[\underbrace{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right)}\right]$$

convex

(single global minimum)

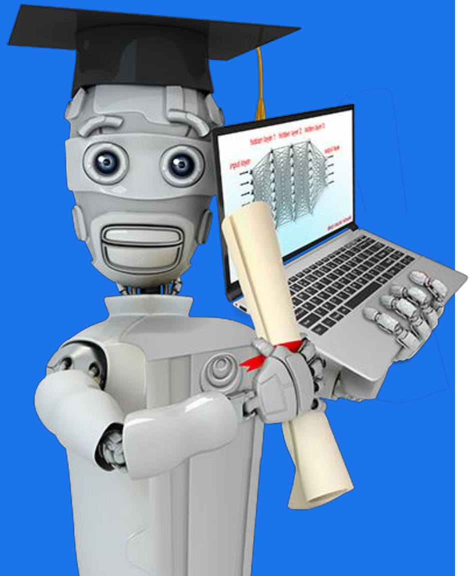
$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) + \left(1-y^{(i)}\right)\log\left(1-f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)\right]$$

maximum likelihood

(don't worry about it!)

Stanford ONLINE

DeepLearning.AI



Gradient Descent

Gradient Descent Implementation

Training logistic regression

Find $\vec{w}, b$

Given new $\vec{x}$, output $f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$

$$P(y = 1|\vec{x}; \vec{w}, b)$$

Gradient descent

cost

$$J(\vec{w},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right]$$

repeat {

$j = 1 \ldots n$

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

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

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

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

} simultaneous updates

Gradient descent for logistic regression

repeat {

looks like linear regression!

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

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

} simultaneous updates

Same concepts:

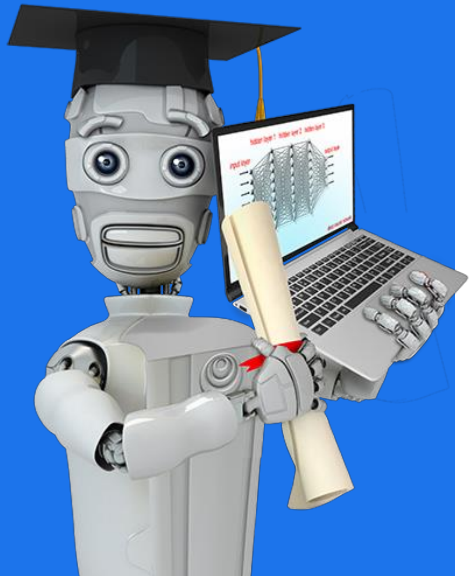
- Monitor gradient descent (learning curve)
- Vectorized implementation
- Feature scaling

Linear regression $f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$

Logistic regression $f_{\vec{w},b}(\vec{x}) = \dfrac{1}{1 + e^{(-\vec{w} \cdot \vec{x} + b)}}$

Stanford
ONLINE

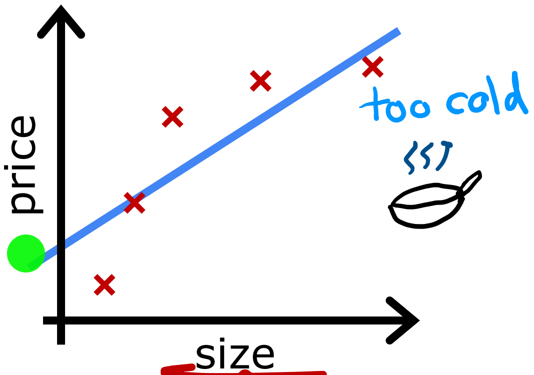
DeepLearning.AI



Regularization to Reduce Overfitting

The Problem of Overfitting

Regression example

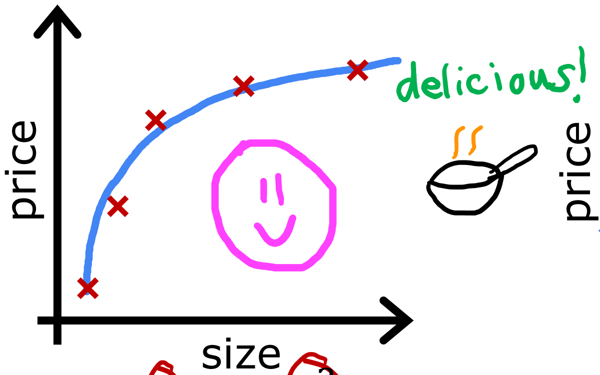


$w_1 x + b$

underfit

- Does not fit the training set well

high bias

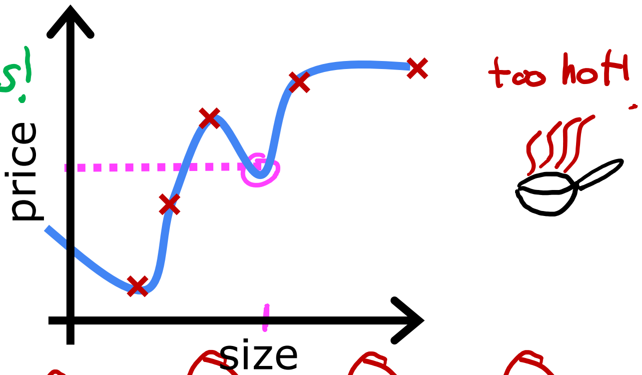


$w_1 x + w_2 x^2 + b$

just right

- Fits training set pretty well

generalization



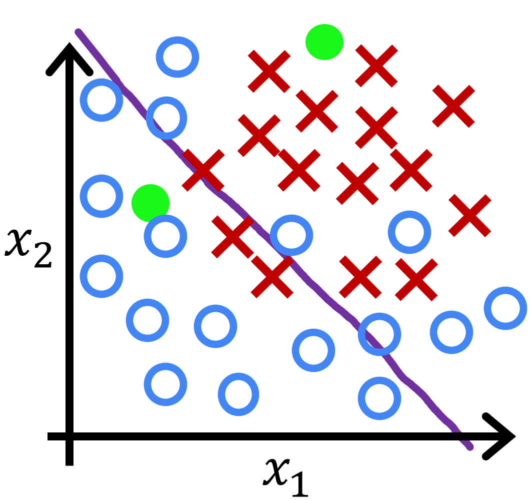
$w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + b$

overfit

- Fits the training set extremely well

high variance

Classification

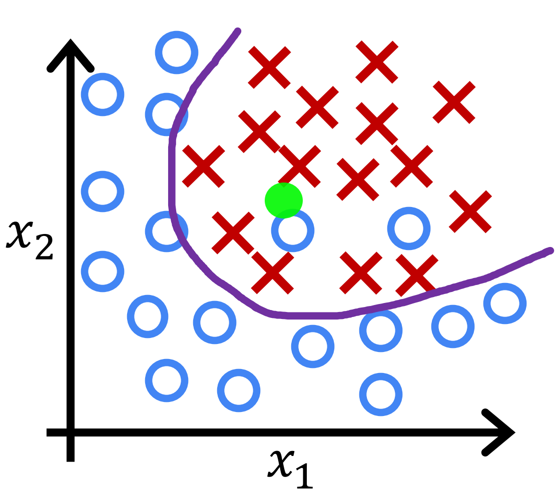


$$z = w_1x_1 + w_2x_2 + b$$

$$f_{\vec{w},b}(\vec{x}) = g(z)$$

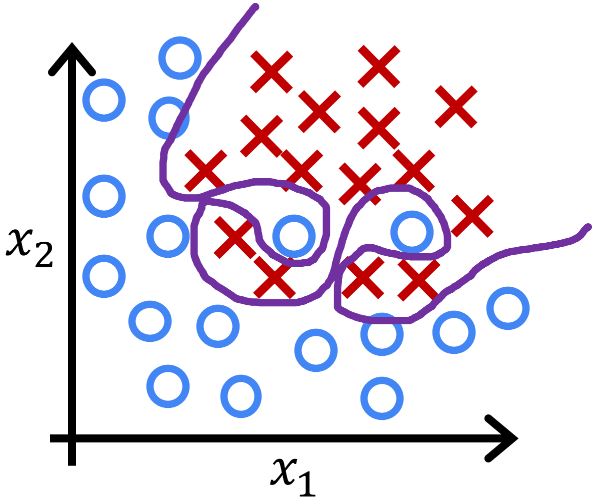
g is the sigmoid function

underfit high bias



$$z = w_1x_1 + w_2x_2$$
$$+w_3x_1^2 + w_4x_2^2$$
$$+w_5x_1x_2 + b$$

just right

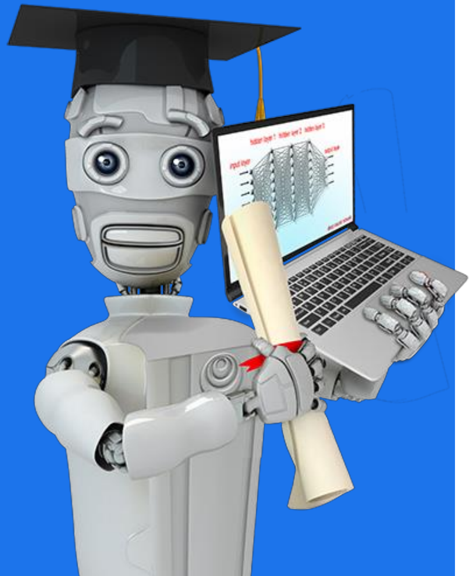


$$z = w_1x_1 + w_2x_2$$
$$+w_3x_1^2x_2 + w_4x_1^2x_2^2$$
$$+w_5x_1^2x_2^3 + w_6x_1^3x_2$$
$$+ \cdots + b$$

overfit

Stanford
ONLINE

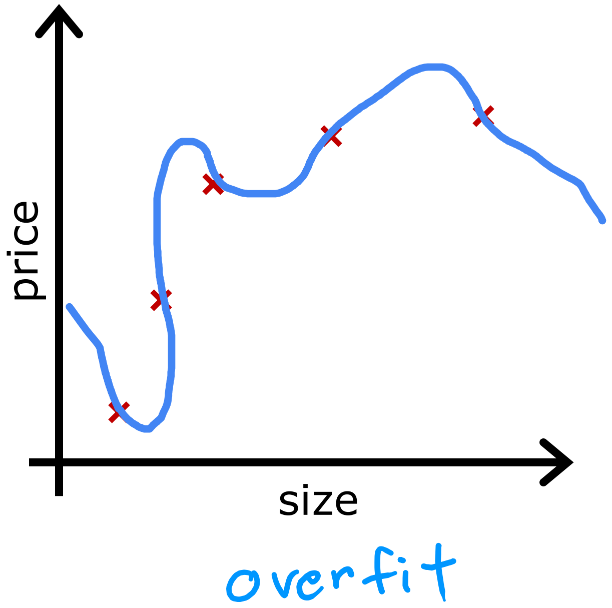
DeepLearning.AI

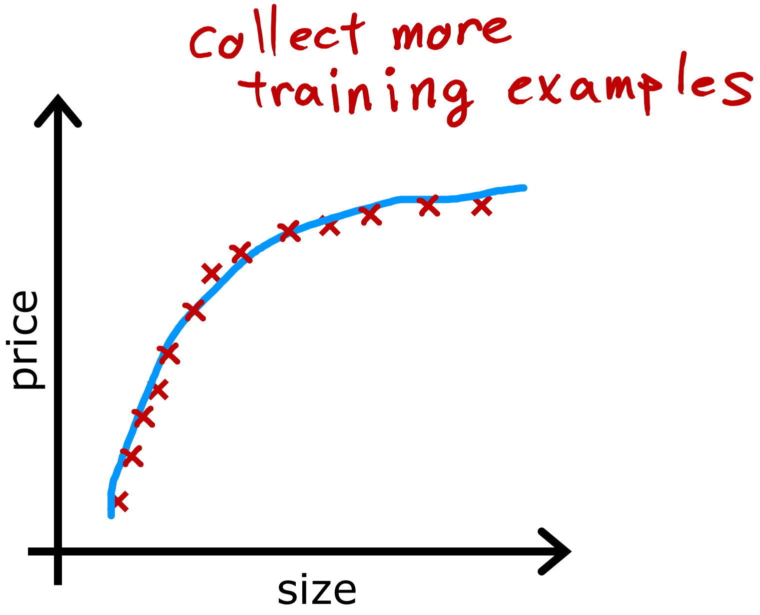


Regularization to Reduce Overfitting

Addressing Overfitting

Collect more training examples





Select features to include/exclude

size	bedrooms	floors	age	avg income	...	distance to coffee shop	price
x_1	x_2	x_3	x_4	x_5		x_{100}	y

all features

+

insufficient data

↓

overfit

selected features

size
bedrooms
age

just right

feature selection

course 2

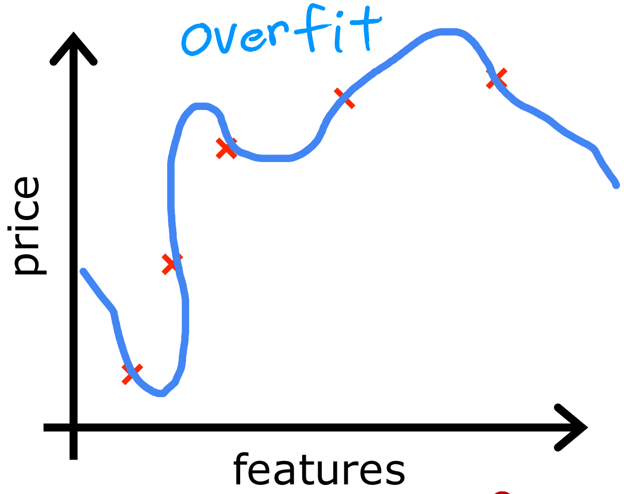
disadvantage

↓

useful features could be lost

Regularization

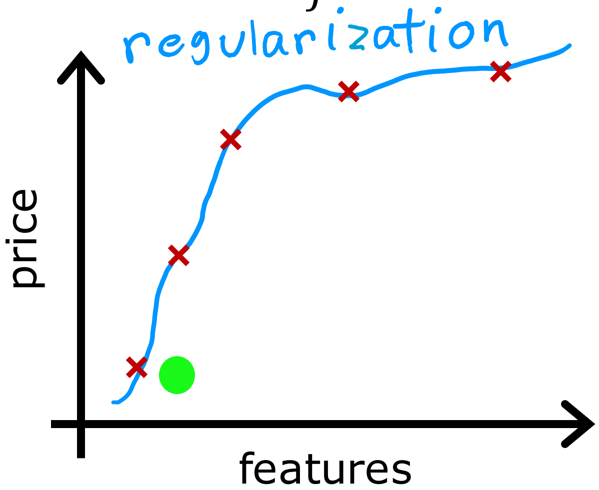
Reduce the size of parameters w_j



$$f(x) = 28x - 385x^2 + 39x^3 - \underbrace{\cancel{174}x^4}_{0} + 100$$

eliminate feature

large values for w_j



$$f(x) = 13x - 0.23x^2 + \underbrace{0.000014}x^3 - \underbrace{0.0001}\,x^4 + 10$$

small values for w_j

Addressing overfitting

Options

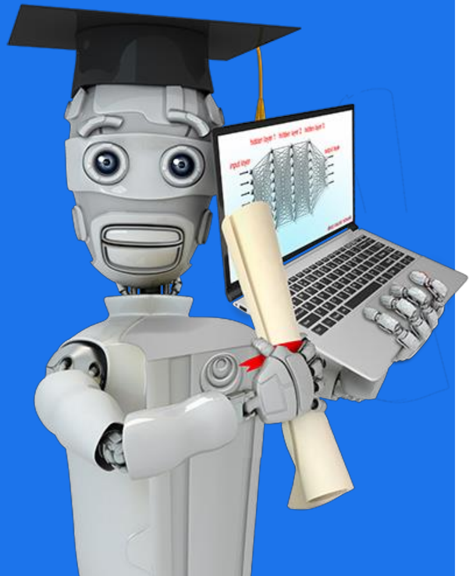
1. Collect more data

2. Select features
   – Feature selection   in course 2

3. Reduce size of parameters
   – "Regularization"   next videos!

Stanford
ONLINE

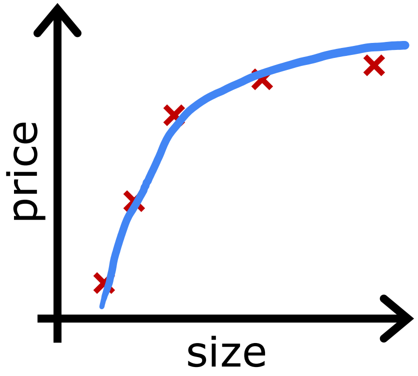
DeepLearning.AI



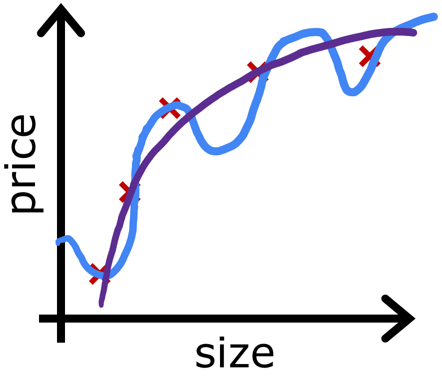
Regularization to Reduce Overfitting

Cost Function with Regularization

Intuition



$w_1x + w_2x^2 + b$



$w_1x + w_2x^2 + \cancel{\underbrace{w_3}_{\approx 0}x^3} + \cancel{\underbrace{w_4}_{\approx 0}x^4} + b$

make w_3, w_4 really small (≈ 0)

$$\min_{\vec{w},b} \frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2 + \cancel{1000\underbrace{w_3^2}_{0.001}} + \cancel{1000\underbrace{w_4^2}_{0.002}}$$

Regularization

small values $w_1, w_2, \cdots, w_n, b$

simpler model $w_3 \approx 0$

less likely to overfit $w_4 \approx 0$

size	bedrooms	floors	age	avg income	...	distance to coffee shop	price
x_1	x_2	x_3	x_4	x_5		x_{100}	y

n features

$n = 100$

$w_1, w_1, w_2, \cdots, w_{100}, b$

$$J(\vec{w}, b) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2 + \underbrace{\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2}_{\text{regularization term}} + \frac{\lambda}{2m}b^2\right]$$

"lambda"

regularization parameter

$\lambda > 0$

can include or exclude b

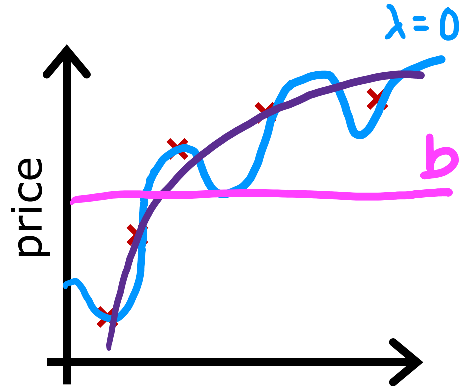
Regularization

$$\min_{\vec{w},b} J(\vec{w},b) = \min_{\vec{w},b}\left[\overbrace{\frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2}^{\text{mean squared error}} + \overbrace{\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2}^{\text{regularization term}}\right]$$

fit data

Keep w_j small

λ balances both goals



choose $\lambda = 10^{10}$

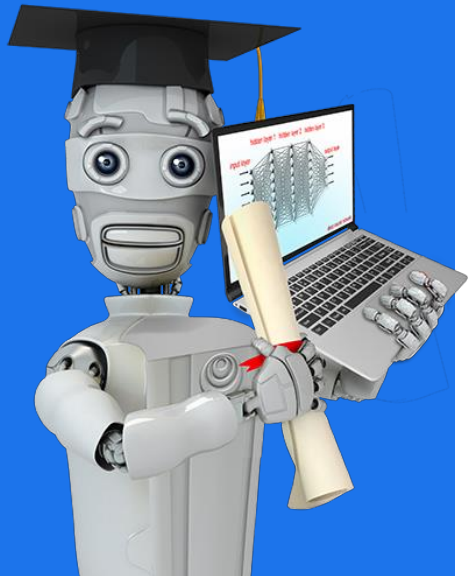
$$f_{\vec{w},b}(\vec{x}) = \underset{\approx 0}{\cancel{w_1 x}} + \underset{\approx 0}{\cancel{w_2 x^2}} + \underset{\approx 0}{\cancel{w_3 x^3}} + \underset{\approx 0}{\cancel{w_4 x^4}} + b$$

$f(x) = b$

Choose λ

Stanford
ONLINE

DeepLearning.AI



Regularization to Reduce Overfitting

Regularized Linear Regression

Regularized linear regression

$$\min_{\vec{w},b} J(\vec{w},b) = \min_{\vec{w},b}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2\right]$$

Gradient descent

repeat {

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w},b) \qquad = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m} w_j$$

$j = 1, \ldots, n$

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

don't have to regularize b

} simultaneous update

Implementing gradient descent

repeat {

- $$w_j = w_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left[\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) - y^{(i)} \right) x_j^{(i)} \right] + \frac{\lambda}{m} w_j \right]$$

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

} simultaneous update $j = 1 \ldots n$

Implementing gradient descent

repeat {

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

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

} simultaneous update $j = 1 \ldots n$

$$w_j = \underbrace{1 w_j - \alpha \frac{\lambda}{m} w_j}_{w_j \underbrace{\left(1 - \alpha \frac{\lambda}{m}\right)}_{\text{shrink } w_j}} \underbrace{- \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)}}_{\text{usual update}}$$

$\alpha \frac{\lambda}{m}$

$0.01 \frac{1}{50} = 0.0002$

$w_j \underbrace{(1 - 0.0002)}_{0.9998}$

How we get the derivative term (optional)

$$\frac{\partial}{\partial w_j} J(\vec{w}, b) = \frac{d}{d w_j}\left[\frac{1}{2m}\sum_{i=1}^{m}\Big(\underbrace{f(\vec{x}^{(i)})}_{\vec{w}\cdot\vec{x}^{(i)}+b} - y^{(i)}\Big)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2\right]$$

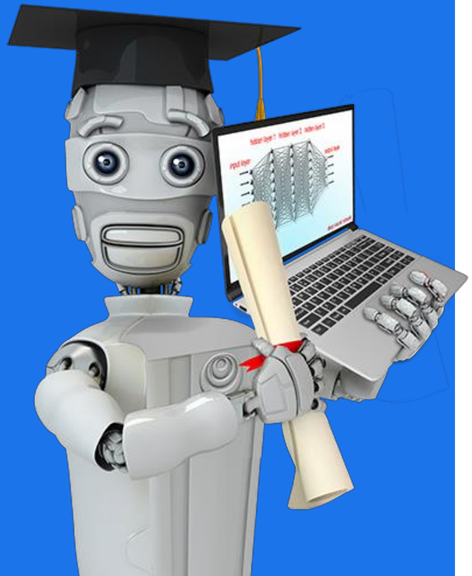
$$= \frac{1}{\cancel{2}m}\sum_{i=1}^{m}\left[\left(\vec{w}\cdot\vec{x}^{(i)} + b - y^{(i)}\right)\cancel{2}\, x_j^{(i)}\right] + \frac{\lambda}{\cancel{2}m}\cancel{2}\, w_j \qquad \text{No } \sum_{j=1}^{n}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left[\Big(\underbrace{\vec{w}\cdot\vec{x}^{(i)} + b}_{f(\vec{x})} - y^{(i)}\Big) x_j^{(i)}\right] + \frac{\lambda}{m} w_j$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left[\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}\right] + \frac{\lambda}{m} w_j$$

Stanford
ONLINE

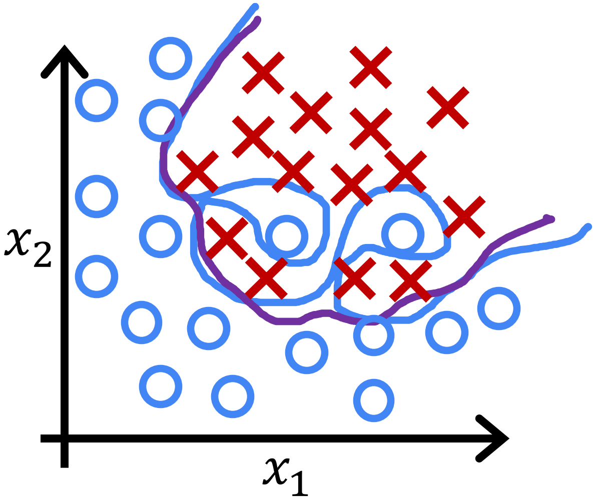
DeepLearning.AI



Regularization to Reduce Overfitting

Regularized Logistic Regression

Regularized logistic regression



$$z = w_1x_1 + w_2x_2 + w_3x_1^2x_2 + w_4x_1^2x_2^2 + w_5x_1^2x_2^3 + \cdots + b$$

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-z}}$$

Cost function

$$J(\vec{w},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

$$\min_{\vec{w},b} J(\vec{w},b) \rightarrow w_j \downarrow$$

Regularized logistic regression

$$J(\vec{w},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

$\min_{\vec{w},b}$

Gradient descent

Looks same as for linear regression!

repeat {

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w},b) \qquad = \frac{1}{m}\sum_{i=1}^{m}\left[\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right) x_j^{(i)}\right] + \frac{\lambda}{m} w_j$$

$j = 1 \ldots n$

logistic regression

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

}

don't have to regularize b