

Linear Regression with multiple variables

Prof. Dr. Christina Bauer

christina.bauer@th-deg.de

Faculty of Computer Science

USEFUL LINEAR ALGEBRA

Data set

2104

1416

1534

852

→ Matrix 4 x 2

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix}$$

$$h_{\theta}(x) = -40 + 0.25x$$

→ 2 x 1 Vector

*

$$\begin{bmatrix} -40 \\ 0.25 \end{bmatrix}$$

=

Result: 4 x 1 matrix
containing the predictions

$$\begin{bmatrix} -40 * 1 + 0,25 * 2104 \\ -40 * 1 + 0,25 * 1416 \\ -40 * 1 + 0,25 * 1534 \\ -40 * 1 + 0,25 * 853 \end{bmatrix}$$

USEFUL LINEAR ALGEBRA #2

Data set


2104

1416

1534

852

→ Matrix 4 x 2


$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix}$$

More than one hypothesis:


$$h_{\theta}(x) = -40 + 0.25x$$

$$h_{\theta}(x) = 200 + 0.1x$$


$$h_{\theta}(x) = -150 + 0.4x$$

*

→ 2 x 3 Matrix


$$\begin{bmatrix} -40 & 200 & -150 \\ 0.25 & 0.1 & 0.4 \end{bmatrix}$$

Result: 4 x 3 matrix
containing the predictions
for each hypothesis in a
row


$$\begin{bmatrix} 486 & 410 & 692 \\ 314 & 342 & 416 \\ 344 & 353 & 464 \\ 173 & 285 & 191 \end{bmatrix}$$

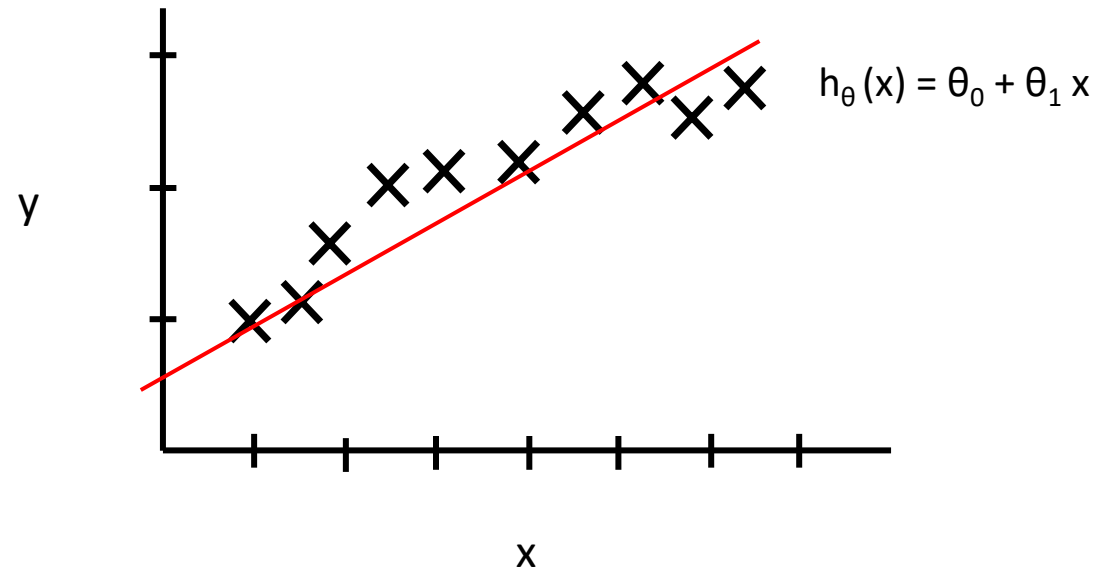
ONE FEATURE

Input feature

Size in m ² x	Price in K € y
51	203
65	240
72	334
...	...

Output variable

Hypothesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$



MULTIPLE FEATURES

x_1	x_2	x_3	x_4	y
Size in m ²	#bedrooms	#floors	Age of house	Price in K €
51	2	1	10	203
65	2	1	15	240
72	3	2	5	334
...

Notation:

m = number of training examples

n = number of features

$x^{(i)}$ = input (feature) of i^{th} training example

$x_j^{(i)}$ = value of feature j in i^{th} training example

Example:

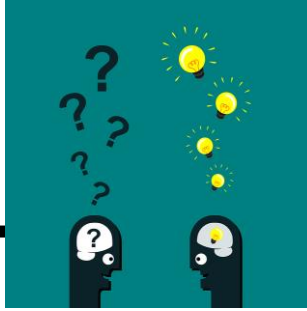
m = e.g. 50

$n = 4$

$$x^{(2)} = \begin{bmatrix} 65 \\ 2 \\ 1 \\ 15 \end{bmatrix} \in \mathbb{R}^4$$

$$x_3^{(2)} = 1$$

QUESTION



In this training set, what is $x_1^{(4)}$?

A: The size of the 1st home in the training set

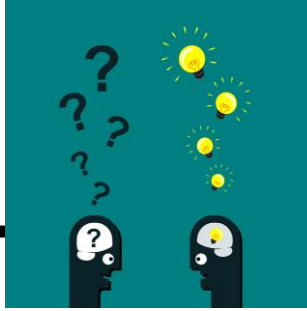
B: The age of the 1st home in the training set

C: The size of the 4th home in the training set

D: The age of the 4th home in the training set

Size in m ²	#bedrooms	#floors	Age of house (years)	Price in K €
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...

QUESTION



In this training set, what is $x_1^{(4)}$?

A: The size of the 1st home in the training set

B: The age of the 1st home in the training set

~~C~~: The size of the 4th home in the training set

D: The age of the 4th home in the training set

Size in feet ²	#bedrooms	#floors	Age of house (years)	Price in K €
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...

HYPOTHESIS

One feature:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Multiple features:

$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

Example: $h_{\theta}(x) = 90 + 0.2x_1 + 0.01x_2 + 4x_3 - 1.5x_4$

size**floors**

“basic price”**bedrooms****age**

HYPOTHESIS

$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots + \theta_n x_n$$

For convenience of notation we define an „additional x_0 feature” that equals 1 $\rightarrow x_0^i = 1$

Have the same number of elements

Transpose θ

Multiply the two vectors θ^T and x

Result is our hypothesis

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$$
$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$
$$\theta^T = [\theta_0 \ \theta_1 \ \dots \ \theta_n]$$

1 x (n+1) matrix

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix}$$


(n+1) x 1 matrix

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots + \theta_n x_n = \theta^T x$$

\rightarrow Multivariate linear regression


GRADIENT DESCENT FOR MULTIPLE VARIABLES

Hypothesis: $h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots + \theta_n x_n = \theta^T x$

 $X_0 = 1$

Parameters: $\theta_0, \theta_1, \theta_2, \dots, \theta_n$  $\theta \rightarrow n+1$ -dimensional vector

Cost function: $J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

 $J(\theta)$

Gradient descent:

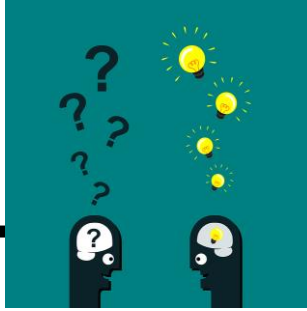
Repeat until convergence {

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1, \dots, \theta_n)$ (simultaneously update $j = 0, \dots, n$)

}

 $J(\theta)$

QUESTION



When there are n features, we define the cost function as

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

For linear regression, which of the following are also equivalent and correct definitions of $J(\theta)$? (more than one may apply)

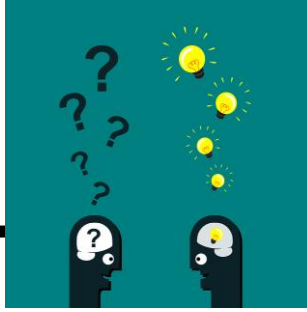
A: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m (\theta^T x^{(i)} - y^{(i)})^2$

B: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=0}^n \theta_j x_j^{(i)} \right) - y^{(i)} \right)^2$ (inner sum starts at 0)

C: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=1}^n \theta_j x_j^{(i)} \right) - y^{(i)} \right)^2$ (inner sum starts at 1)

D: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=1}^n \theta_j x_j^{(i)} \right) - \left(\sum_{j=0}^n y_j^{(i)} \right) \right)^2$

QUESTION



When there are n features, we define the cost function as

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

For linear regression, which of the following are also equivalent and correct definitions of $J(\theta)$?

~~A:~~ $J(\theta) = \frac{1}{2m} \sum_{i=1}^m (\theta^T x^{(i)} - y^{(i)})^2$

~~B:~~ $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=0}^n \theta_j x_j^{(i)} \right) - y^{(i)} \right)^2$ (inner sum starts at 0)

C: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=1}^n \theta_j x_j^{(i)} \right) - y^{(i)} \right)^2$ (inner sum starts at 1)

D: $J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=1}^n \theta_j x_j^{(i)} \right) - \left(\sum_{j=0}^n y_j^{(i)} \right) \right)^2$

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \dots + \theta_n x_n$$

GRADIENT DESCENT FOR MULTIPLE VARIABLES

$n=1$

Repeat until convergence {

$$\frac{\partial}{\partial \theta_0} J(\theta)$$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x^i$$

(simultaneously update θ_1 and θ_0)

}

$n \geq 1$

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_j^i$$

(simultaneously update θ_j for $j = 0, \dots, n$)
}

→

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_0^i$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_1^i$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_2^i$$

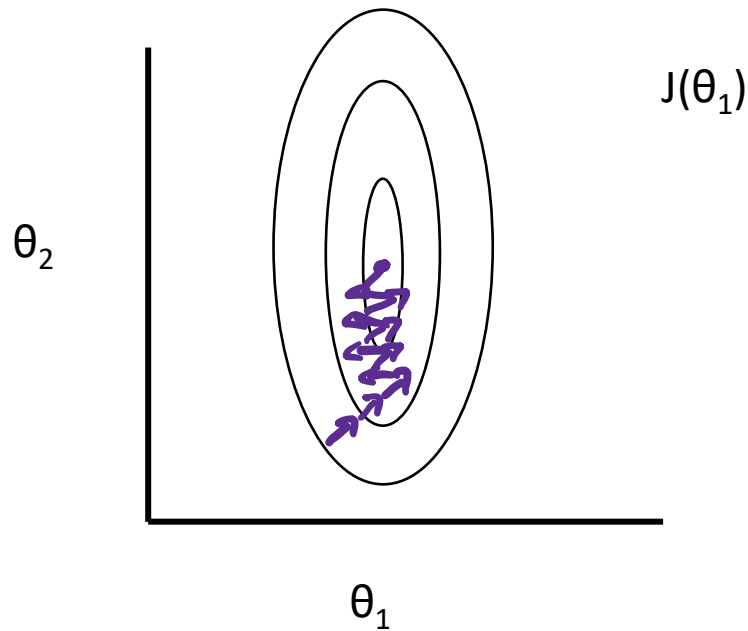
...

FEATURE SCALING

Make sure features are on a similar scale

e. g. $x_1 = \text{size (10-200 m}^2\text{)}$

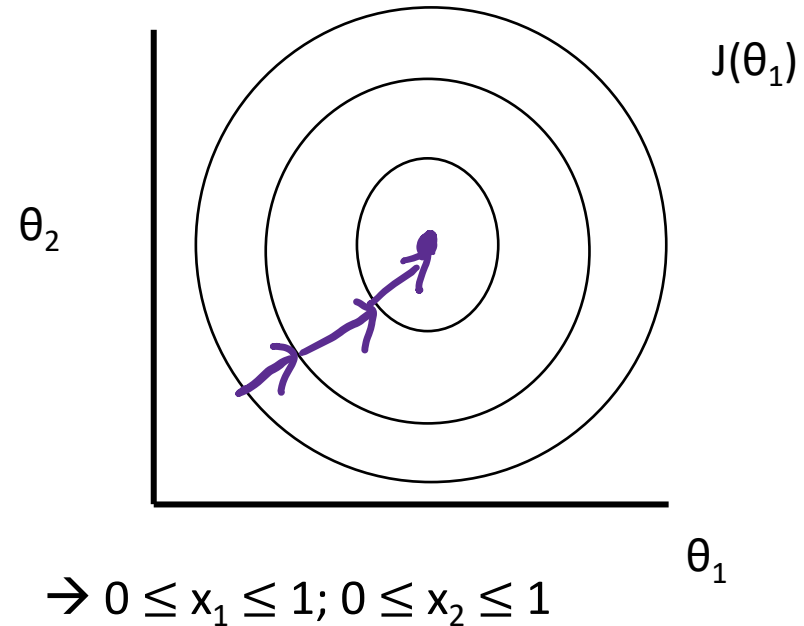
$x_2 = \text{number of bedrooms (1-8)}$



Make sure features are on a similar scale

e. g. $x_1 = \text{size (m}^2\text{)} / 200$

$x_2 = \text{number of bedrooms} / 8$



$\rightarrow 0 \leq x_1 \leq 1; 0 \leq x_2 \leq 1$

\rightarrow Gradient descent will converge much faster

FEATURE SCALING

Get every feature into approximately a $-1 \leq x_i \leq 1$ range

$$x_0 = 1$$

OK:

$$0 \leq x_1 \leq 3$$

$$-2 \leq x_2 \leq 0.5$$

Not OK:

$$-100 \leq x_3 \leq 100$$

$$-0.0001 \leq x_4 \leq 0.0001$$

Rule of thumb:

$$-3 \leq x_i \leq 3$$

$$-\frac{1}{3} \leq x_i \leq \frac{1}{3}$$

FEATURE SCALING – MEAN NORMALIZATION

Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean
(do not apply to $x_0 = 1$)

E. g.

$$x_1 = \frac{\text{size} - 100}{190} \rightarrow \text{average size} = 100$$

$$x_2 = \frac{\#bedrooms - 4}{7} \rightarrow 1\text{-}8 \text{ bedrooms}$$

$$\rightarrow -0.5 \leq x_1 \leq 0.5; -0.5 \leq x_2 \leq 0.5$$

General rule:

$$x_1 := \frac{x_1 - \mu_1}{s_1}$$

s_1 = range (max – min) or
standard deviation

μ_1 = average value of x in
training set

FEATURE SCALING – MEAN NORMALIZATION

General rule:

$$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$$

s_1 = range (max – min) or standard deviation

μ_1 = average value of x in training set

s_1 = standard deviation

→ Example:

Data set: 2,4,4,4,5,5,7,9

Mean: sum/number of examples = $40/8 = 5$

Variance:

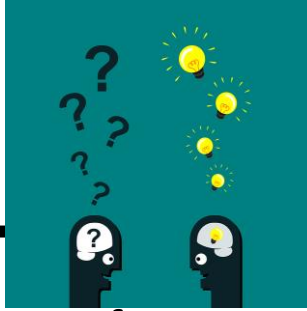
e. g. data point 1: $(2-5)^2 = 9$

$$\sigma^2 = (9+1+1+1+0+0+4+16)/8 = 4$$

Standard deviation:

$$\sigma = \sqrt{\sigma^2} = \sqrt{4} = 2$$

QUESTION



Suppose you are using a learning algorithm to estimate the price of houses in a city. You want one of your features x_i to capture the age of the house. In your training set, all of your houses have an age between 30 and 50 years, with an average age of 38 years. Which of the following would you use as features, assuming you use feature scaling and mean normalization?

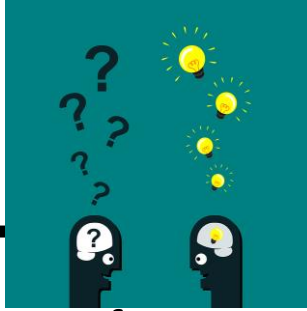
A: $x_i = \text{age of house}$

B: $x_i = \frac{\text{age of house}}{50}$

C: $x_i = \frac{\text{age of house} - 38}{50}$

D: $x_i = \frac{\text{age of house} - 38}{20}$

QUESTION



Suppose you are using a learning algorithm to estimate the price of houses in a city. You want one of your features x_i to capture the age of the house. In your training set, all of your houses have an age between 30 and 50 years, with an average age of 38 years. Which of the following would you use as features, assuming you use feature scaling and mean normalization?

A: $x_i = \text{age of house}$

B: $x_i = \frac{\text{age of house}}{50}$

C: $x_i = \frac{\text{age of house} - 38}{50}$

~~D: $x_i = \frac{\text{age of house} - 38}{20}$~~

LEARNING RATE α

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

- Make sure gradient descent is working correctly
- Choose an appropriate learning rate α

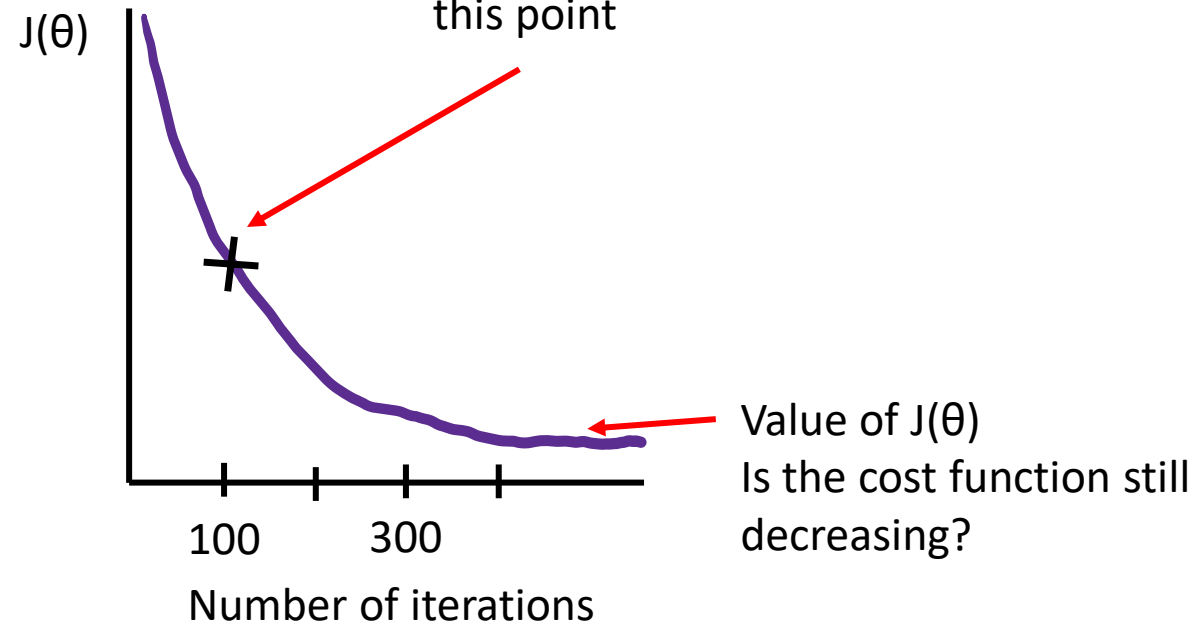
MAKE SURE GRADIENT DESCENT IS WORKING CORRECTLY

Goal: Min $J(\theta)$

θ

Value of $J(\theta)$

After 100 iterations $\rightarrow \theta$
has some specific values at
this point



$J(\theta)$ should decrease after every iteration

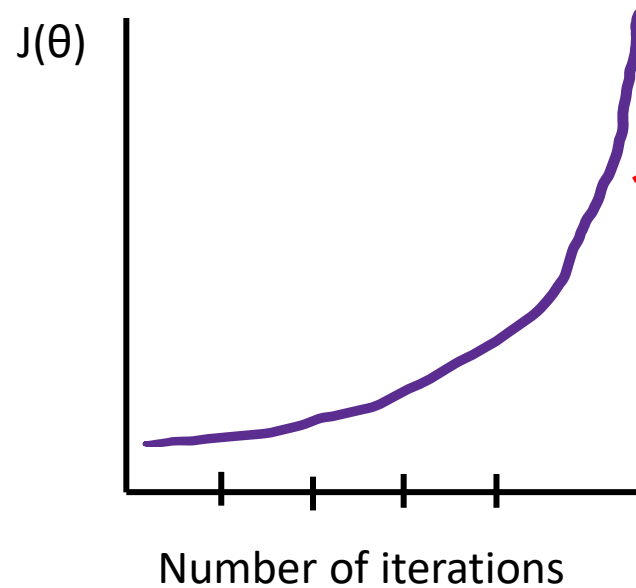
It is hard to tell in advance how many iterations
are needed to converge

Example automatic convergence
test:

Declare convergence if $J(\theta)$
decreases by less than $\varepsilon = 10^{-3}$ in
one iteration

\rightarrow Choosing ε can be difficult

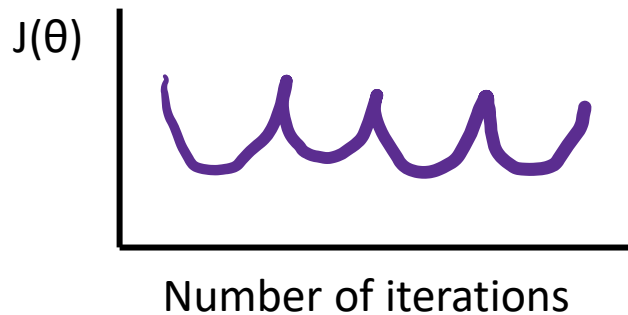
MAKE SURE GRADIENT DESCENT IS WORKING CORRECTLY



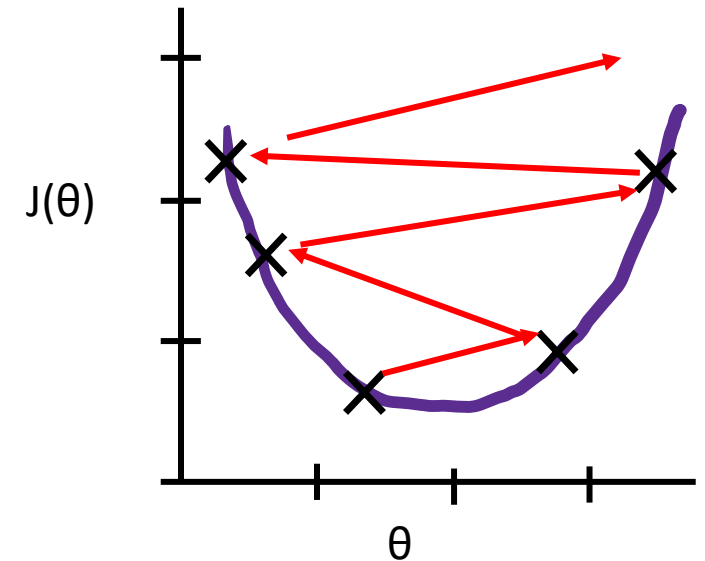
Gradient descent is not working
→ Choose a smaller rate α

→ For sufficiently small α , $J(\theta)$ should decrease on every iteration

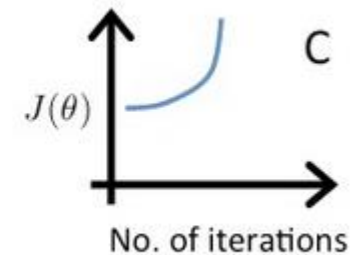
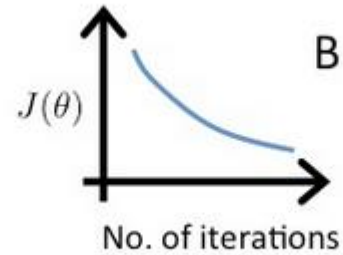
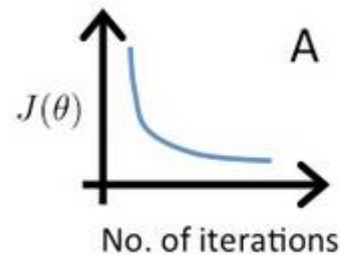
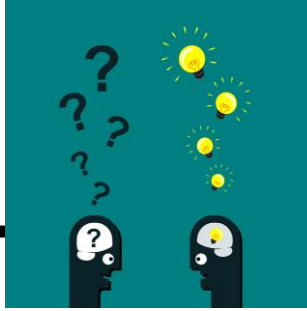
→ If α is too small, gradient descent can be slow to converge



common cause:



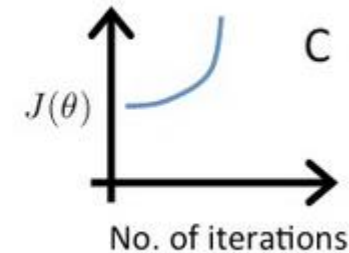
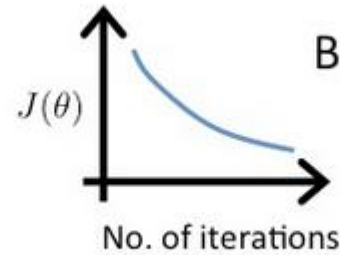
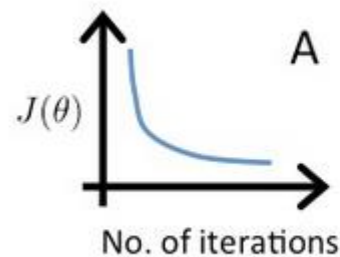
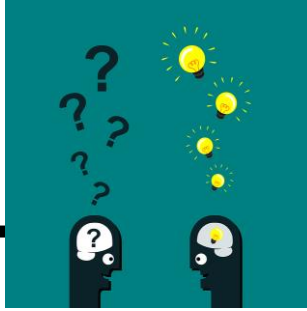
QUESTION



Suppose a friend ran gradient descent three times, with $\alpha=0.01$, $\alpha=0.1$, $\alpha=1$, and got the three plots (labeled A, B, and C). Which plots corresponds to which values of α ?

- A: A is $\alpha=0.01$ B is $\alpha=0.1$, C is $\alpha=1$
- B: A is $\alpha=0.1$ B is $\alpha=0.01$, C is $\alpha=1$
- C: A is $\alpha=1$ B is $\alpha=0.01$, C is $\alpha=0.1$
- D: A is $\alpha=1$ B is $\alpha=0.1$, C is $\alpha=0.01$

QUESTION


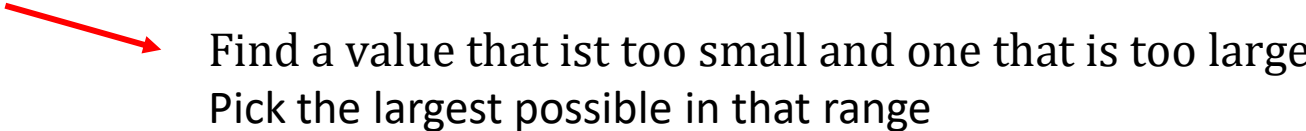


Suppose a friend ran gradient descent three times, with $\alpha=0.01$, $\alpha=0.1$, $\alpha=1$, and got the three plots (labeled A, B, and C). Which plots corresponds to which values of α ?

- A: A is $\alpha=0.01$ B is $\alpha=0.1$, C is $\alpha=1$
- ~~B: A is $\alpha=0.1$ B is $\alpha=0.01$, C is $\alpha=1$~~
- C: A is $\alpha=1$ B is $\alpha=0.01$, C is $\alpha=0.1$
- D: A is $\alpha=1$ B is $\alpha=0.1$, C is $\alpha=0.01$

In graph C, the cost function is increasing, so the learning rate is set too high. Both graphs A and B converge to an optimum of the cost function, but graph B does so very slowly, so its learning rate is set too low. Graph A lies between the two.

MAKE SURE GRADIENT DESCENT IS WORKING CORRECTLY

- If α is too small: slow to converge
- If α is too large: $J(\theta)$ may not decrease on every iteration; may not converge /slowly converge
- To choose α , try to run gradient descent with a range of values and look at the plots
- E.g. 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1,
 $\approx 3 \cdot x$
 Find a value that is too small and one that is too large
Pick the largest possible in that range

NEW FEATURES

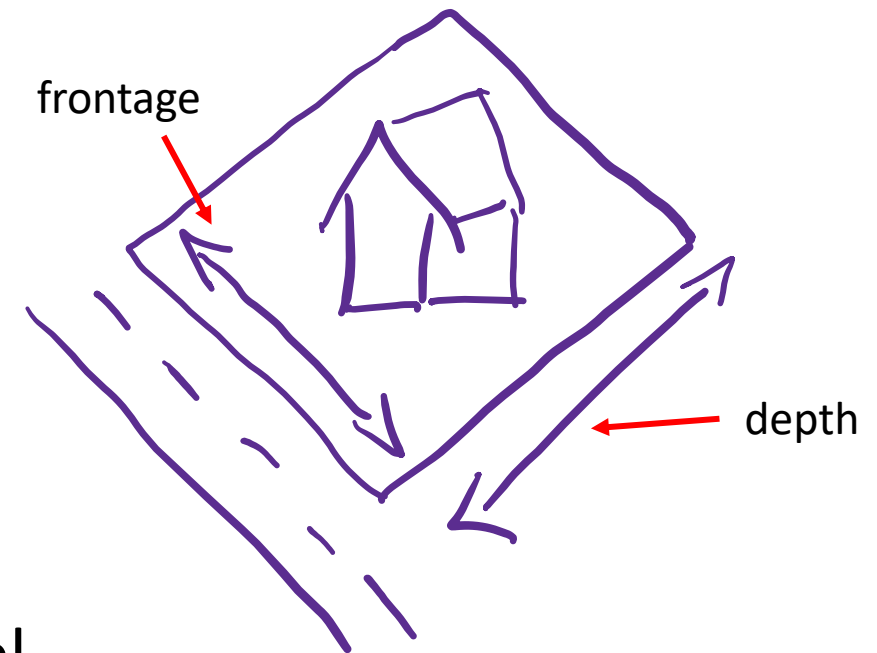
$$h_{\theta}(x) = \theta_0 + \theta_1 \text{ frontage} + \theta_2 \text{ depth}$$

Create new feature: area

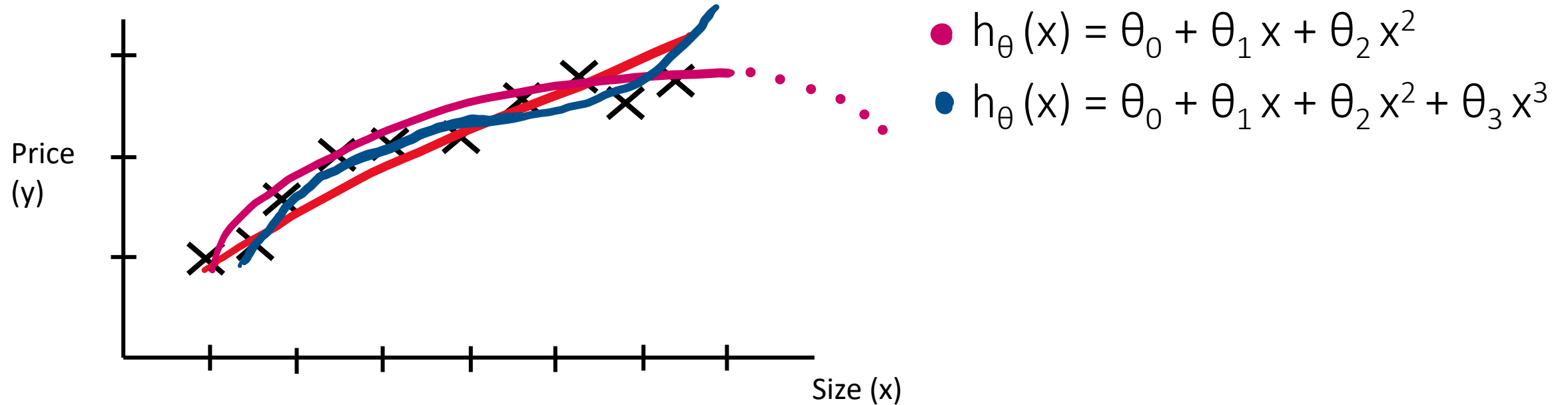
$$X = \text{frontage} * \text{depth}$$

$$\rightarrow h_{\theta}(x) = \theta_0 + \theta_1 \text{ area}$$

→ Find new features to get a better model



POLYNOMIAL REGRESSION



$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$h_{\theta}(x) = \theta_0 + \theta_1 (\text{size}) + \theta_2 (\text{size})^2 + \theta_3 (\text{size})^3$$

→ set: $x_1 = \text{size}$; $x_2 = (\text{size})^2$; $x_3 = (\text{size})^3$

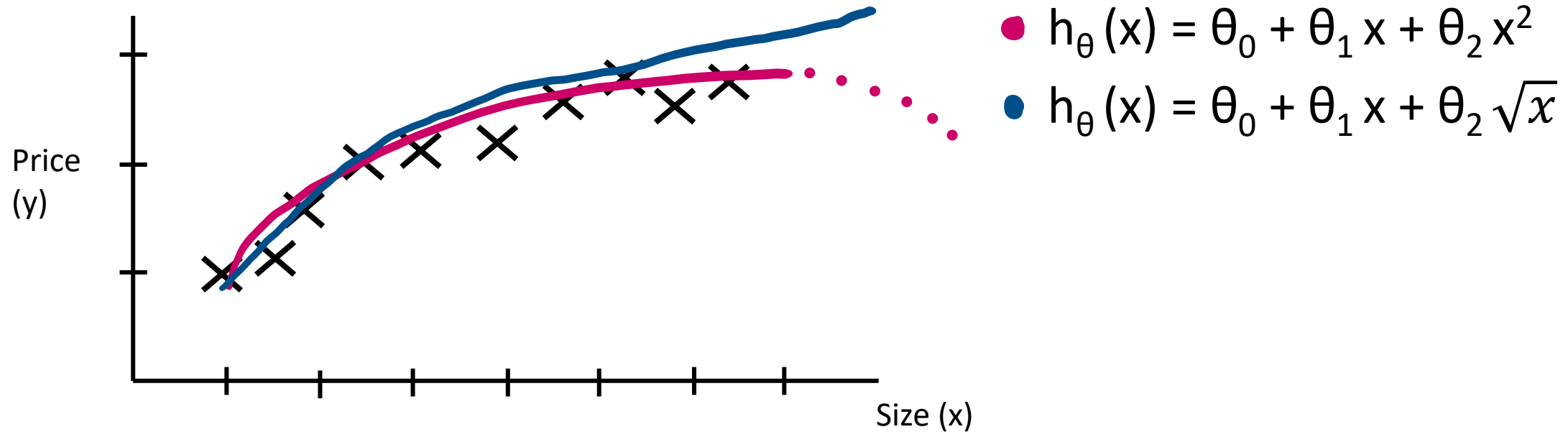
→ Apply multivariate linear regression

Importance of feature scaling increases, e. g.:

range size: 1-100 → range size²: 1-10000

→ Range size³: 1- 1000000

POLYNOMIAL REGRESSION — FEATURE CHOICE



$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$h_{\theta}(x) = \theta_0 + \theta_1 (\text{size}) + \theta_2 (\text{size})^2$$

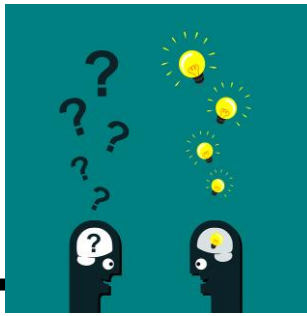
$$\text{set: } x_1 = \text{size}; \quad x_2 = (\text{size})^2$$

$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$h_{\theta}(x) = \theta_0 + \theta_1 (\text{size}) + \theta_2 \sqrt{\text{size}}$$

$$\text{set: } x_1 = \text{size}; \quad x_2 = \sqrt{\text{size}}$$

QUESTION



Suppose you want to predict a house's price as a function of its size. Your model is
$$h_{\theta}(x) = \theta_0 + \theta_1(\text{size}) + \theta_2\sqrt{\text{size}}$$

Suppose size ranges from 1 to 1000 (feet²). You will implement this by fitting a model:
$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Finally, suppose you want to use feature scaling (without mean normalization). Which of the following choices for x_1 and x_2 should you use? (Note: $\sqrt{1000} \approx 32$)

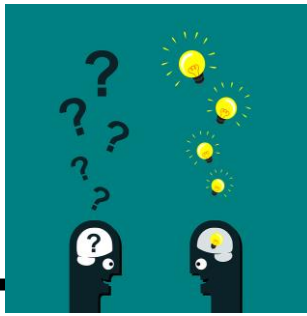
A: $x_1 = \text{size}; x_2 = 32\sqrt{\text{size}}$

B: $x_1 = 32(\text{size}); x_2 = \sqrt{\text{size}}$

C: $x_1 = \frac{\text{size}}{1000}; x_2 = \frac{\sqrt{\text{size}}}{32}$

D: $x_1 = \frac{\text{size}}{32}; x_2 = \sqrt{\text{size}}$

QUESTION



Suppose you want to predict a house's price as a function of its size. Your model is
$$h_{\theta}(x) = \theta_0 + \theta_1(\text{size}) + \theta_2\sqrt{\text{size}}$$

Suppose size ranges from 1 to 1000 (feet²). You will implement this by fitting a model:
$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Finally, suppose you want to use feature scaling (without mean normalization). Which of the following choices for x_1 and x_2 should you use? (Note: $\sqrt{1000} \approx 32$)

A: $x_1 = \text{size}; x_2 = 32\sqrt{\text{size}}$

B: $x_1 = 32(\text{size}); x_2 = \sqrt{\text{size}}$

~~C: $x_1 = \frac{\text{size}}{1000}; x_2 = \frac{\sqrt{\text{size}}}{32}$~~

D: $x_1 = \frac{\text{size}}{32}; x_2 = \sqrt{\text{size}}$

Step 1:

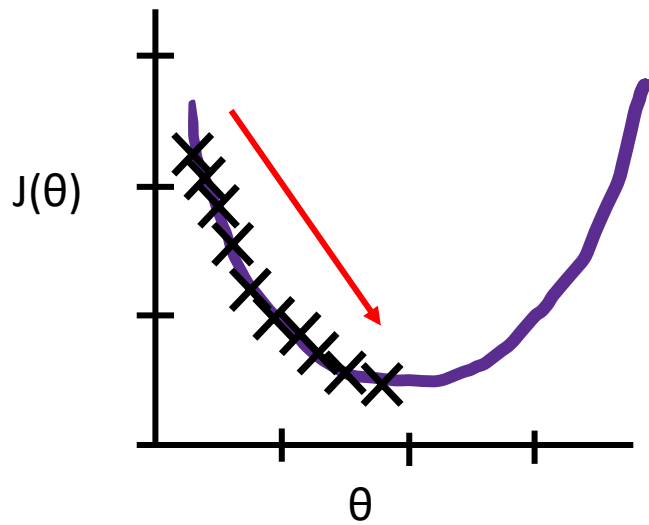
$$h_{\theta}(x) = \theta_0 + \theta_1(\text{size}) + \theta_2\sqrt{\text{size}}$$

$$\text{set: } x_1 = \text{size}; x_2 = \sqrt{\text{size}}$$

Step 2: Feature scaling

$$x_i := \frac{x_i - \mu_i}{s_i} \rightarrow x_1 := \frac{\text{size} - \mu_1}{s_1} \rightarrow x_2 := \frac{\sqrt{\text{size}} - \mu_2}{s_2}$$

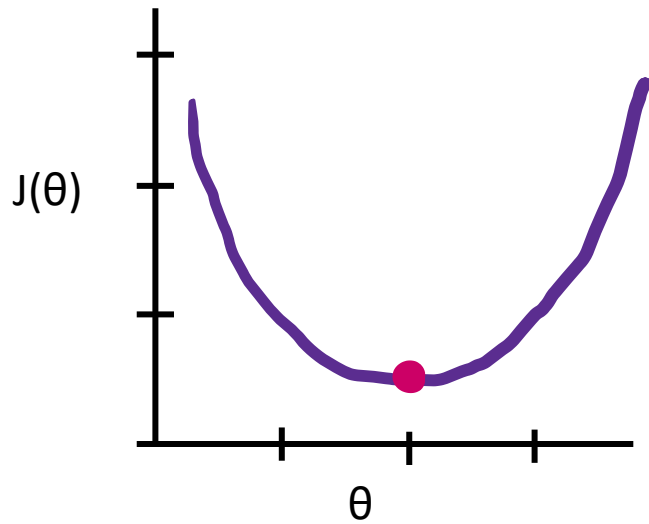
NORMAL EQUATION



Normal equation:
Method to solve for θ analytically

NORMAL EQUATION

→ $\theta \in \mathbb{R}$ (not a vector)



$$J(\theta) = a\theta^2 + b\theta + c$$

Find the minimum: $\frac{d}{d\theta} J(\theta) = 0$

→ $\theta \in \mathbb{R}^{n+1}$

$$J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Find the minimum:

$$\frac{\partial}{\partial \theta_j} J(\theta) = 0 \text{ (for every } j\text{)}$$

solve for $\theta_0, \theta_1, \dots, \theta_n$

NORMAL EQUATION

x_0	Size in m ² x_1	#bedrooms x_2	#floors x_3	Age of house in years x_4	Price in K € y
1	51	2	1	10	203
1	65	2	1	15	240
1	72	3	2	5	334
1	69	3	1	20	320

$m = 4$

$$X = \begin{bmatrix} 1 & 51 & 2 & 1 & 10 \\ 1 & 65 & 2 & 1 & 15 \\ 1 & 72 & 3 & 2 & 5 \\ 1 & 69 & 3 & 1 & 20 \end{bmatrix}$$

$m \times (n+1)$

$$y = \begin{bmatrix} 203 \\ 240 \\ 334 \\ 320 \end{bmatrix}$$

$m \times 1$

$\rightarrow \theta = (X^T X)^{-1} X^T y$

NORMAL EQUATION

m examples $((x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)}))$; n features

$x^{(i)} = \begin{bmatrix} x^{(i)}_0 \\ x^{(i)}_1 \\ x^{(i)}_2 \\ \dots \\ x^{(i)}_n \end{bmatrix} \in \mathbb{R}^{n+1}$

\rightarrow design matrix

$X = \begin{bmatrix} x^{(1)T} \\ x^{(2)T} \\ \dots \\ x^{(m)T} \end{bmatrix}$

$m \times (n+1)$

e.g. $x^{(i)} = \begin{bmatrix} 1 \\ x^{(i)}_1 \end{bmatrix}$

$X = \begin{bmatrix} 1 & x^{(1)}_1 \\ 1 & x^{(2)}_1 \\ \dots & \dots \\ 1 & x^{(m)}_1 \end{bmatrix}$

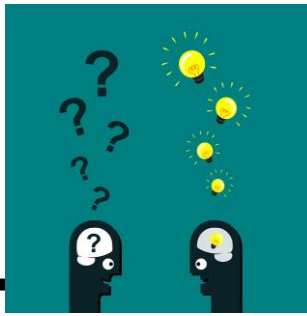
$m \times 2$

$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \dots \\ y^{(m)} \end{bmatrix}$

each of the training examples $x^{(i)}$ looks like this vector

$$\rightarrow \theta = (X^T X)^{-1} X^T y$$

QUESTION



Suppose you have the training in the table. You would like to predict a child's weight as a function of his/her age and height with the model.

$$\text{Weight} = \theta_0 + \theta_1 \text{age} + \theta_2 \text{height}$$

What are X and y ?

age (x_1)	height in cm (x_2)	weight in kg (y)
4	89	16
9	124	28
5	103	20

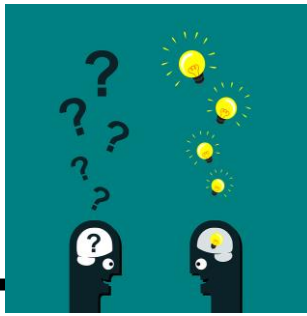
A: $X = \begin{bmatrix} 4 & 89 \\ 9 & 124 \\ 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

C: $X = \begin{bmatrix} 4 & 89 & 1 \\ 9 & 124 & 1 \\ 5 & 103 & 1 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

B: $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

D: $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

QUESTION



Suppose you have the training in the table. You would like to predict a child's weight as a function of his/her age and height with the model.

$$\text{Weight} = \theta_0 + \theta_1 \text{age} + \theta_2 \text{height}$$

What are X and y ?

age (x_1)	height in cm (x_2)	weight in kg (y)
4	89	16
9	124	28
5	103	20

A: $X = \begin{bmatrix} 4 & 89 \\ 9 & 124 \\ 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

C: $X = \begin{bmatrix} 4 & 89 & 1 \\ 9 & 124 & 1 \\ 5 & 103 & 1 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

B: $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

~~**D:**~~ $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}$ $y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

NORMAL EQUATION

$$\theta = (X^T X)^{-1} X^T y \rightarrow \min_{\theta} J(\theta)$$



Example implemtation:

Octave: `pinv(X'*X)*X'*y`

Python (e. g.):

```
x_transpose = np.transpose(x)
x_transpose_dot_x = x_transpose.dot(x)
temp1= np.linalg.inv(x_transpose_dot_x)
temp2 = X_transpose.dot(y)
theta = temp1.dot(temp2)
```

$(X^T X)^{-1}$ inverse of matrix $X^T X$

$$\rightarrow X^T X = A$$

$$\rightarrow (X^T X)^{-1} = A^{-1}$$

→ Normal equation method: no need to do feature scaling

GRADIENT DESCENT AND NORMAL EQUATION

m training examples, n features

Gradient Descent

- Need to choose α
- Needs many iterations
- Works well when n is large (e. g. $n = 10^6$)

Normal equation

- No need to choose α
- No iterations
- Need to compute $(X^T X)^{-1}$
- Slow if n is large
- Does not work for all algorithms (e. g. classification problems)

$$X \rightarrow m \times (n+1)$$

$$X^T \rightarrow (n+1) \times m$$

$$X^T X \rightarrow (n+1) \times (n+1)$$

→ Cost of inverting a matrix $\approx O(n^3)$

NORMAL EQUATION AND NON-INVERTIBILITY

$$\theta = (X^T X)^{-1} X^T y$$

What if $X^T X$ is non-invertible? (singular/degenerative)

Most common causes:

- Redundant features (linearly dependent):
 - e. g. $x_1 = \text{size in feet}^2$; $x_2 = \text{size in m}^2 \rightarrow x_1 = (3.28)^2 x_2$
- Too many features: (e. g. $m \leq n$)
 - e. g. $m = 10$; $n = 100 \rightarrow \theta \in \mathbb{R}^{100+1}$
 - \rightarrow Delete some feature or use regularization

WRAP-UP

Gradient Descent for Multiple Variables

The gradient descent equation itself is generally the same form; we just have to repeat it for our 'n' features:

Repeat until convergence {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_0^i$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_1^i$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_2^i$$

...

}

WRAP-UP

Gradient Descent for Multiple Variables

In Summary:

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x_j^i \text{ for } j := 0, \dots, n$$

(simultaneously update θ_j for $j = 0, \dots, n$)

}

WRAP-UP

Feature Scaling

We can speed up gradient descent by having each of our input values in roughly the same range. This is because θ will descend quickly on small ranges and slowly on large ranges, and so will oscillate inefficiently down to the optimum when the variables are very uneven.

The way to prevent this is to modify the ranges of our input variables so that they are all roughly the same. Ideally:

$$-1 \leq x_{(i)} \leq 1$$

or

$$-0.5 \leq x_{(i)} \leq 0.5$$

These aren't exact requirements; we are only trying to speed things up. The goal is to get all input variables into roughly one of these ranges.

WRAP-UP

Feature Scaling

Two techniques to help with this are **feature scaling** and **mean normalization**. Feature scaling involves dividing the input values by the range (i. e. the maximum value minus the minimum value) of the input variable, resulting in a new range of just 1. Mean normalization involves subtracting the average value for an input variable from the values for that input variable resulting in a new average value for the input variable of just zero. To implement both of these techniques, adjust your input values as shown in this formula:

$$x_i := \frac{x_i - \mu_i}{s_i}$$

Where μ_i is the **average** of all the values for feature (i) and s_i is the range of values (max - min), or s_i is the standard deviation.

WRAP-UP

Learning Rate

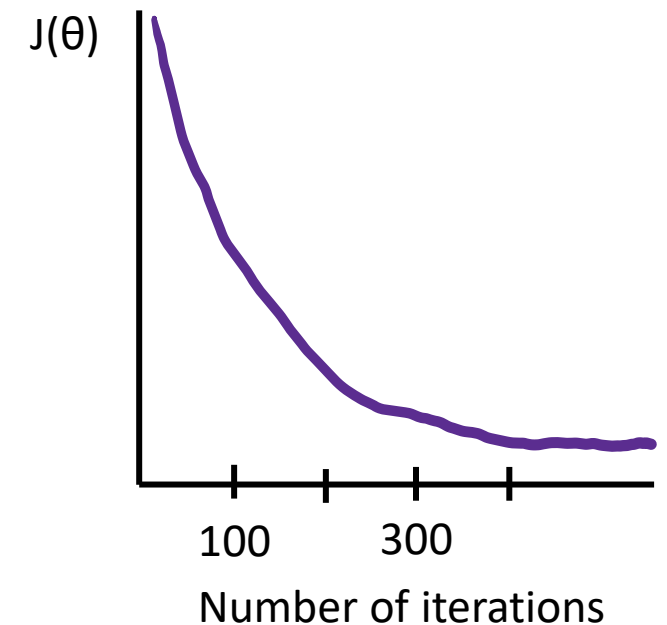
Debugging gradient descent: Make a plot with number of iterations on the x-axis. Now plot the cost function, $J(\theta)$ over the number of iterations of gradient descent. If $J(\theta)$ ever increases, then you probably need to decrease α .

Automatic convergence test: Declare convergence if $J(\theta)$ decreases by less than ϵ in one iteration, where ϵ is some small value such as 10^{-3} . However, in practice it is difficult to choose this threshold value.

To summarize:

If α is too small: slow convergence.

If α is too large: $J(\theta)$ may not decrease on every iteration and thus may not converge.



WRAP-UP

Features and Polynomial Regression

We can improve our features and the form of our hypothesis function in a couple different ways.

We can combine multiple features into one. For example, we can combine x_1 and x_2 into a new feature x_3 by taking $x_1 * x_2$. Our hypothesis function need not be linear (a straight line) if that does not fit the data well.

We can change the behaviour or curve of our hypothesis function by making it a quadratic, cubic or square root function (or any other form).

For example, if our hypothesis function is $h_{\theta}(x) = \theta_0 + \theta_1 x_1$ then we can create additional features based on x_1 , to get the quadratic function $h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2$ or the cubic function $h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^3$. In the cubic version, we have created new features $x_2 = x_1^2$ and $x_3 = x_1^3$. To make it a square root function, we could do: $h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 \sqrt{x_1}$. One important thing to keep in mind is, if you choose your features this way then feature scaling becomes very important.

WRAP-UP

Normal Equation

Normal equation is an alternative to Gradient descent performing the minimization explicitly and without resorting to an iterative algorithm. $J(\theta)$ is minimized by explicitly taking its derivatives with respect to the θ_j 's, and setting them to zero. This allows us to find the optimum theta without iteration. The normal equation formula is:

$$\theta = (X^T X)^{-1} X^T y$$

Gradient Descent

- Need to choose α
- Needs many iterations
- Works well when n is large (e. g. $n = 10^6$)

Normal equation

- No need to choose α
- No iterations
- Need to compute $(X^T X)^{-1}$
- Slow if n is large

There is no need to do feature scaling with the normal equation. With the normal equation, computing the inversion has complexity $O(n^3)$.

WRAP-UP

Normal Equation

If $\mathbf{X}^T\mathbf{X}$ is noninvertible, the common causes are:

- Redundant features, where two features are linearly dependent
- Too many features (e. g. $m \leq n$).

Solutions to the above problems include deleting a feature that is linearly dependent with another or deleting one or more features when there are too many features.

QUIZ – QUESTION 1

Suppose $m = 4$ students have taken some class, and the class had a midterm exam and a final exam. You have collected a dataset of their scores on the two exams. You'd like to use polynomial regression to predict a student's final exam score from their midterm exam score. Concretely, suppose you want to fit a model of the form $h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$, where x_1 is the midterm score and x_2 is $(\text{midterm score})^2$. Further, you plan to use both feature scaling (dividing by the "max-min", or range, of a feature) and mean normalization.

- What is the normalized feature $x_2^{(4)}$? (Hint: midterm = 69, final = 78 is training example 4.) Please round off your answer to two decimal places.

midterm exam	(midterm exam) ²	final exam
89	7921	96
72	5184	74
94	8836	87
69	4761	78

QUIZ – QUESTION 1

Suppose $m = 4$ students have taken some class, and the class had a midterm exam and a final exam. You have collected a dataset of their scores on the two exams. You'd like to use polynomial regression to predict a student's final exam score from their midterm exam score. Concretely, suppose you want to fit a model of the form $h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$, where x_1 is the midterm score and x_2 is (midterm score)². Further, you plan to use both feature scaling (dividing by the "max-min", or range, of a feature) and mean normalization.

- What is the normalized feature $x_2^{(4)}$? (Hint: midterm = 69, final = 78 is training example 4.) Please round off your answer to two decimal places.

midterm exam	(midterm exam) ²	final exam
89	7921	96
72	5184	74
94	8836	87
69	4761	78

$$x_i := \frac{x_i - \mu_i}{s_i}$$

$$x_2 :=$$

$$\frac{4761 - ((7921 + 5184 + 8836 + 4761)/4)}{8836 - 4761}$$

$$\frac{-1914.5}{4075} = \underline{\underline{-0.47}}$$

QUIZ – QUESTION 2

You run gradient descent for 15 iterations with $\alpha=0.3$ and compute $J(\theta)$ after each iteration. You find that the value of $J(\theta)$ **increases** over time. Based on this, which of the following conclusions seems most plausible?

A: $\alpha=0.3$ is an effective choice of learning rate.

B: Rather than use the current value of α , it'd be more promising to try a larger value of α (say $\alpha=1.0$).

C: Rather than use the current value of α , it'd be more promising to try a smaller value of α (say $\alpha=0.1$).

QUIZ – QUESTION 2

You run gradient descent for 15 iterations with $\alpha=0.3$ and compute $J(\theta)$ after each iteration. You find that the value of $J(\theta)$ **increases** over time. Based on this, which of the following conclusions seems most plausible?

A: $\alpha=0.3$ is an effective choice of learning rate.

B: Rather than use the current value of α , it'd be more promising to try a larger value of α (say $\alpha=1.0$).

~~C~~: Rather than use the current value of α , it'd be more promising to try a smaller value of α (say $\alpha=0.1$).

QUIZ – QUESTION 3

Suppose you have $m = 14$ training examples with $n = 3$ features (excluding the additional all-ones feature for the intercept term, which you should add). The normal equation is $\theta = (X^T X)^{-1} X^T y$.

For the given values of m and n , what are the dimensions of θ , X , and y in this equation?

- A: X is 14×3 , y is 14×1 , θ is 3×1
- B: X is 14×4 , y is 14×1 , θ is 4×1
- C: X is 14×3 , y is 14×1 , θ is 3×3
- D: X is 14×4 , y is 14×4 , θ is 4×4

QUIZ – QUESTION 3

Suppose you have $m = 14$ training examples with $n = 3$ features (excluding the additional all-ones feature for the intercept term, which you should add). The normal equation is $\theta = (X^T X)^{-1} X^T y$.

For the given values of m and n , what are the dimensions of θ , X , and y in this equation?

A: ~~X is 14×3~~ , y is 14×1 , ~~θ is 3×1~~

~~B~~: X is 14×4 , y is 14×1 , θ is 4×1

C: ~~X is 14×3~~ , y is 14×1 , ~~θ is 3×3~~

D: X is 14×4 , ~~y is 14×4~~ , ~~θ is 4×4~~

X has m rows and $n + 1$ columns (+1 because of the $x_0=1$ term). y is an m -vector. θ is an $(n+1)$ -vector.

QUIZ – QUESTION 4

Suppose you have a dataset with $m = 50$ examples and $n = 15$ features for each example. You want to use multivariate linear regression to fit the parameters θ to our data. Should you prefer gradient descent or the normal equation?

- A: The normal equation, since gradient descent might be unable to find the optimal θ .
- B: Gradient descent, since it will always converge to the optimal θ .
- C: Gradient descent, since $(X^T X)^{-1}$ will be very slow to compute in the normal equation.
- D: The normal equation, since it provides an efficient way to directly find the solution.

QUIZ – QUESTION 4

Suppose you have a dataset with $m = 50$ examples and $n = 15$ features for each example. You want to use multivariate linear regression to fit the parameters θ to our data. Should you prefer gradient descent or the normal equation?

A: The normal equation, since gradient descent might be unable to find the optimal θ .

B: Gradient descent, since it will always converge to the optimal θ .

C: Gradient descent, since $(X^T X)^{-1}$ will be very slow to compute in the normal equation.

~~D: The normal equation, since it provides an efficient way to directly find the solution.~~

QUIZ – QUESTION 5

Which of the following are reasons for using feature scaling? (more than one may apply)

A: It is necessary to prevent gradient descent from getting stuck in local optima.

B: It prevents the matrix $X^T X$ (used in the normal equation) from being non-invertible (singular/degenerate).

C: It speeds up gradient descent by making it require fewer iterations to get to a good solution.

D: It speeds up solving for θ using the normal equation.

QUIZ – QUESTION 5

Which of the following are reasons for using feature scaling? (more than one may apply)

A: It is necessary to prevent gradient descent from getting stuck in local optima.

B: It prevents the matrix $X^T X$ (used in the normal equation) from being non-invertible (singular/degenerate).

~~C~~: It speeds up gradient descent by making it require fewer iterations to get to a good solution.

D: It speeds up solving for θ using the normal equation.