

Linear Regression with one variable

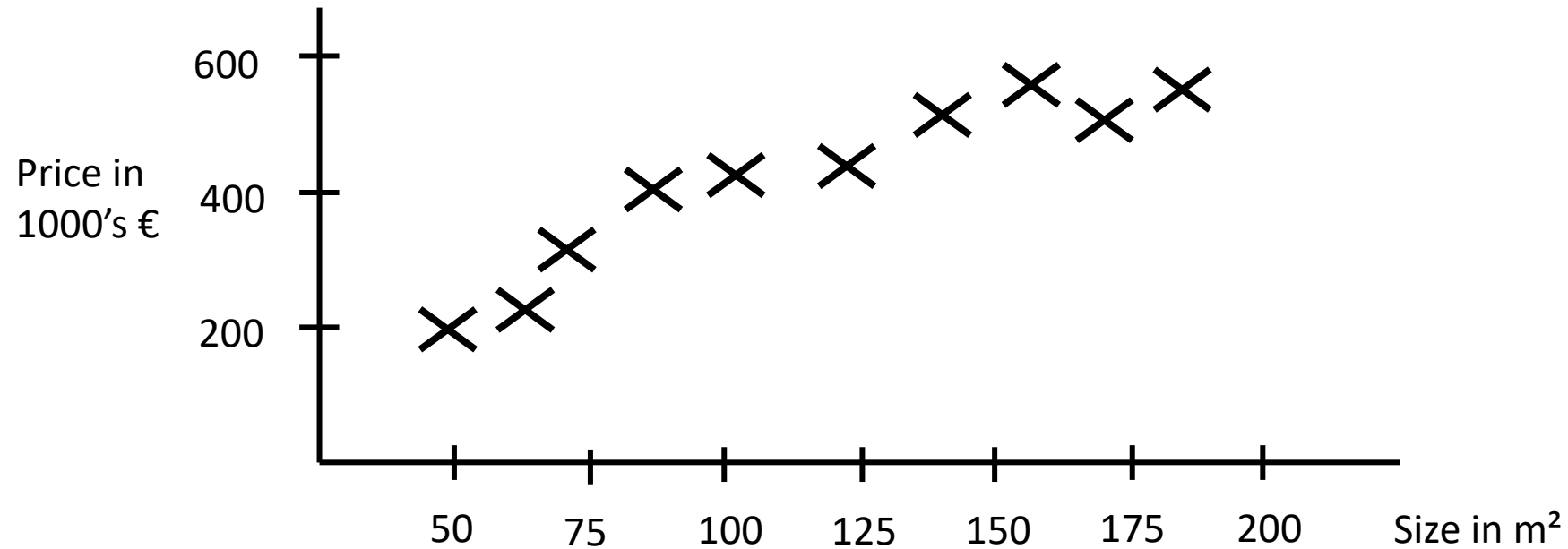
Prof. Dr. Christina Bauer

christina.bauer@th-deg.de

Faculty of Computer Science

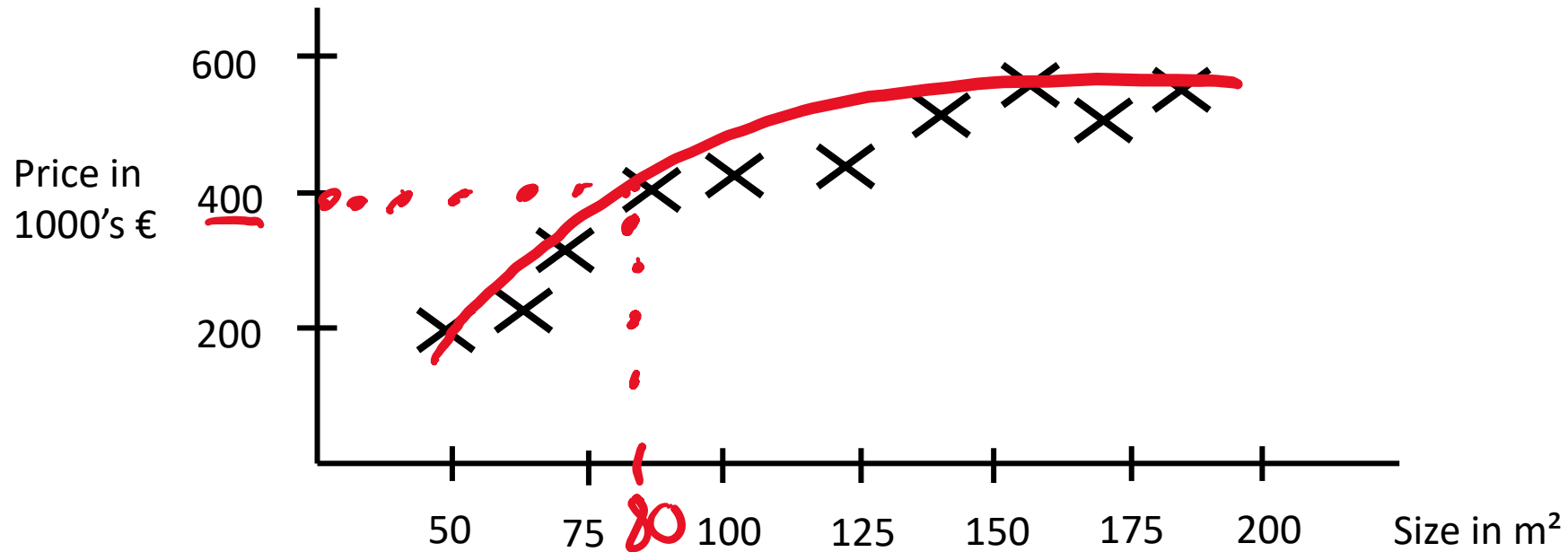
SUPERVISED LEARNING

House price prediction



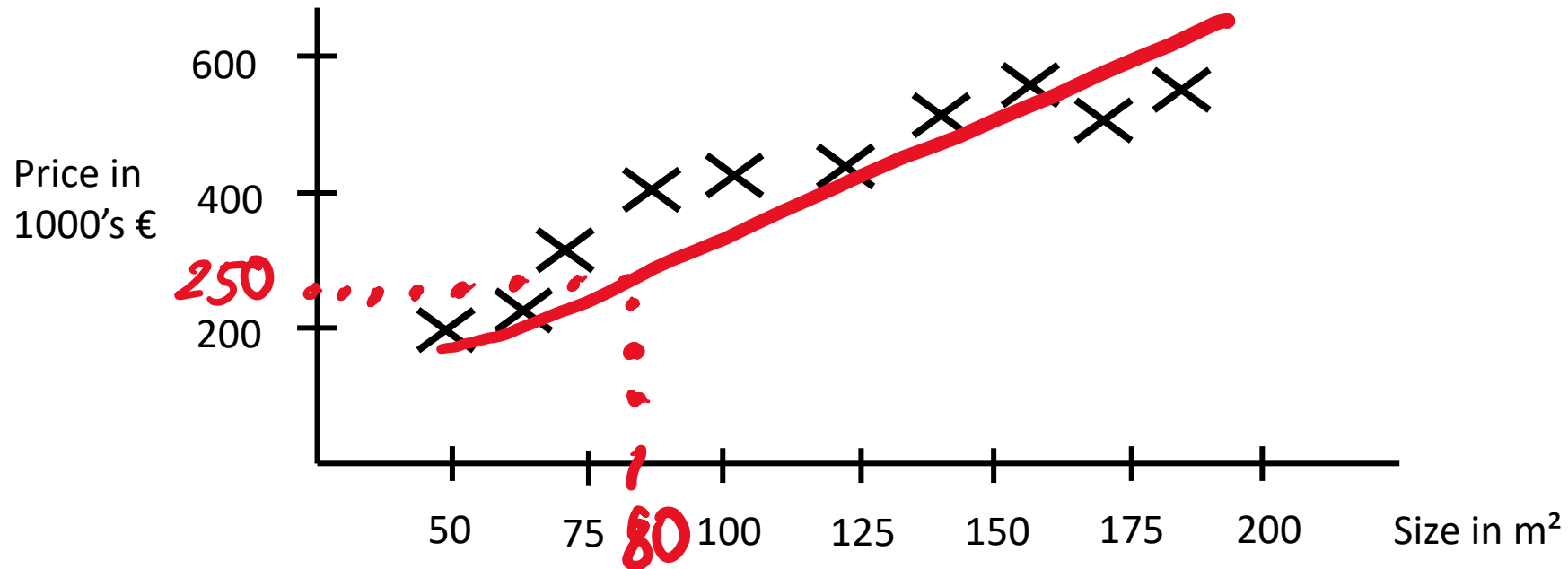
SUPERVISED LEARNING — SECOND-ORDER POLYNOMIAL

House price prediction

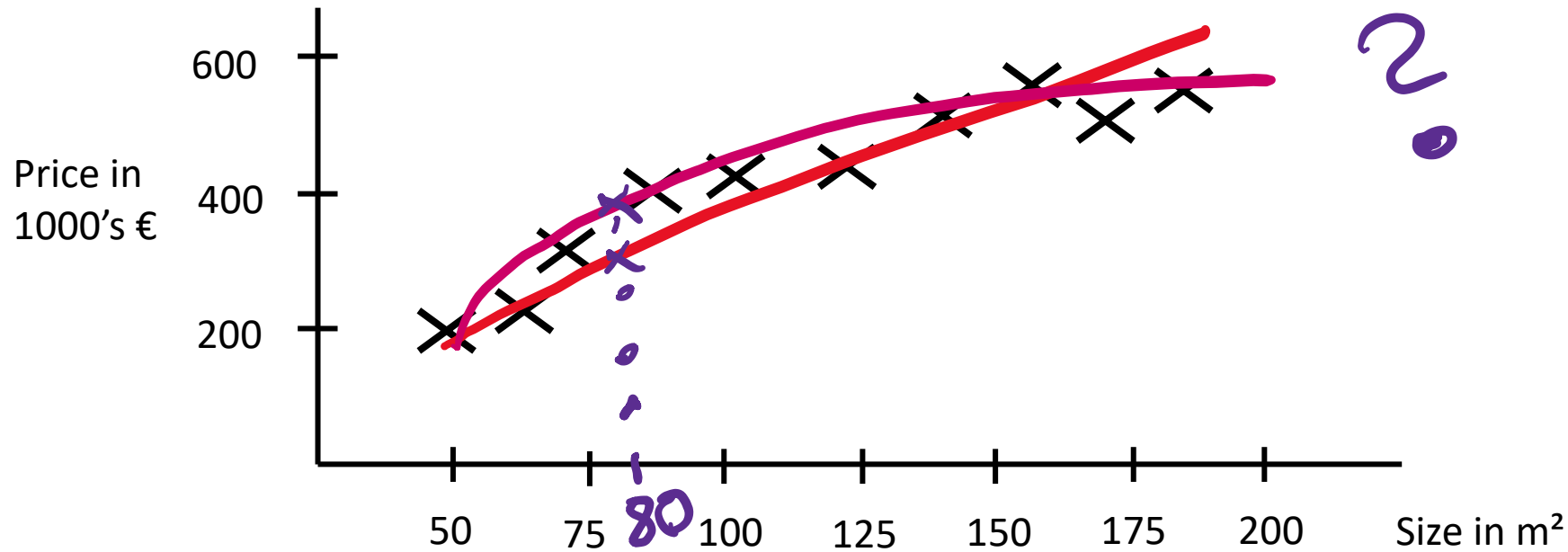


SUPERVISED LEARNING — LINEAR

House price prediction



SUPERVISED LEARNING - HOUSE PRICE PREDICTION



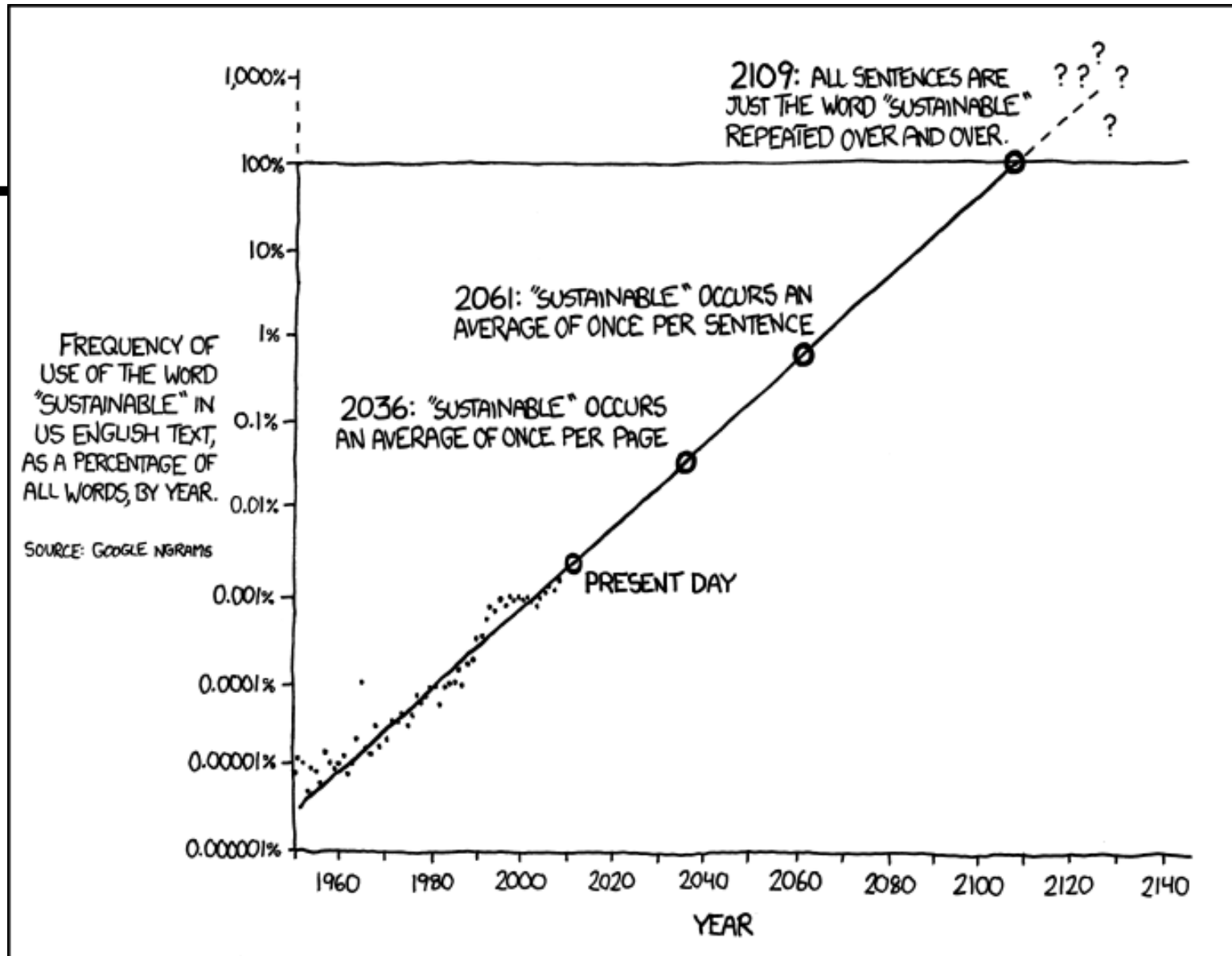
Supervised learning

“Right answers” are given

- Data set of prices, e. g. (51,203)
- Algorithm produces more of the right answers

Regression problem

Predict continuous value
output



THE WORD "SUSTAINABLE" IS UNSUSTAINABLE.

LINEAR REGRESSION

R

```
lm([target variable] ~ [predictor variables],  
data = [data source])
```

NymPy (Python)

```
model = LinearRegression().fit(x, y)
```

Matlab

```
mdl = fitlm(X, y)
```

TRAINING SET

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

Notation

m = number of training examples

x 's = input variable / features

y 's = output variables / target variable

(x, y) = is single training example

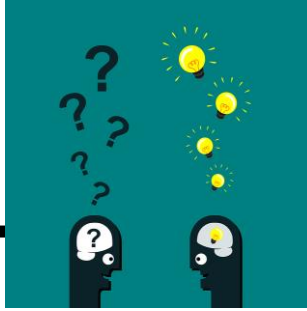
$(x^{(i)}, y^{(i)})$ = i^{th} training example

$$x^{(1)} = 51$$

$$x^{(2)} = 65$$

$$y^{(1)} = 203$$

QUESTION



Consider the training set shown below. What is $y^{(3)}$?

A: 1416

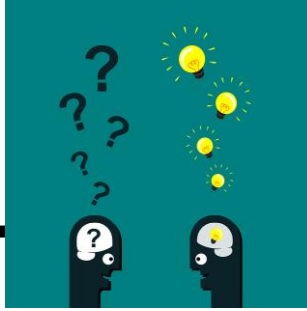
B: 1534

C: 315

D: 0

Size in feet ²	Price in K \$
2104	460
1416	232
1534	315
852	178
...	...

QUESTION



Consider the training set shown below. What is $y^{(3)}$?

A: 1416

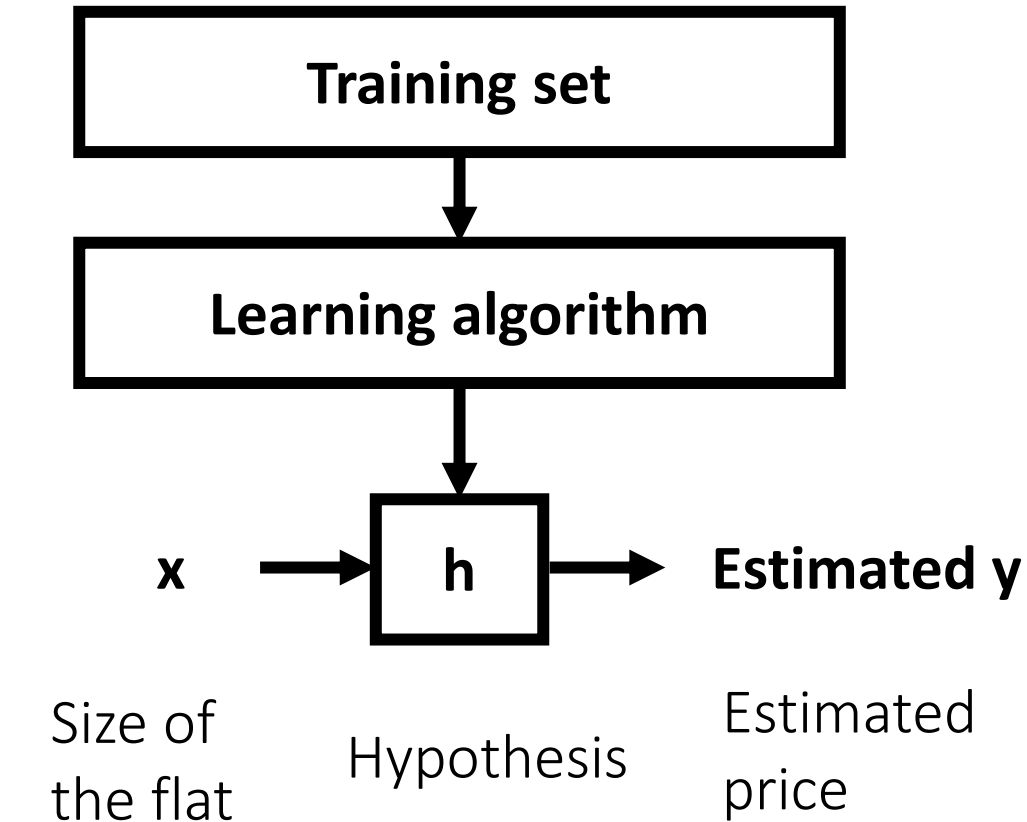
B: 1534

C: 315

D: 0

Size in feet ²	Price in K \$
2104	460
1416	232
1534	315
852	178
...	...

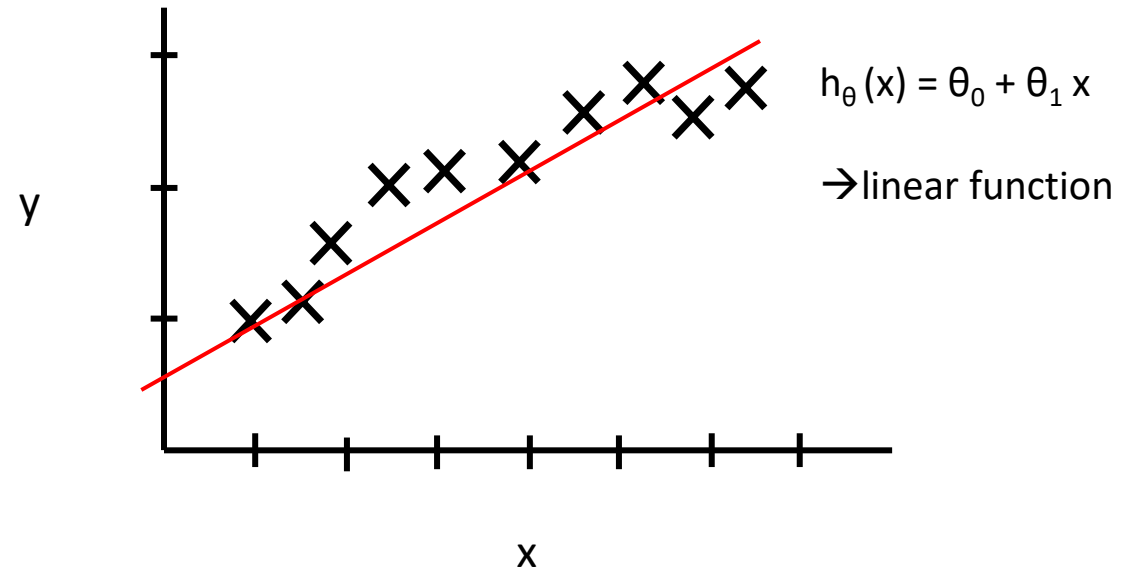
SUPERVISED LEARNING ALGORITHM - MODEL



h is a function that maps from x 's to y 's

How do we represent h ?

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



Linear regression with one variable
Univariate linear regression

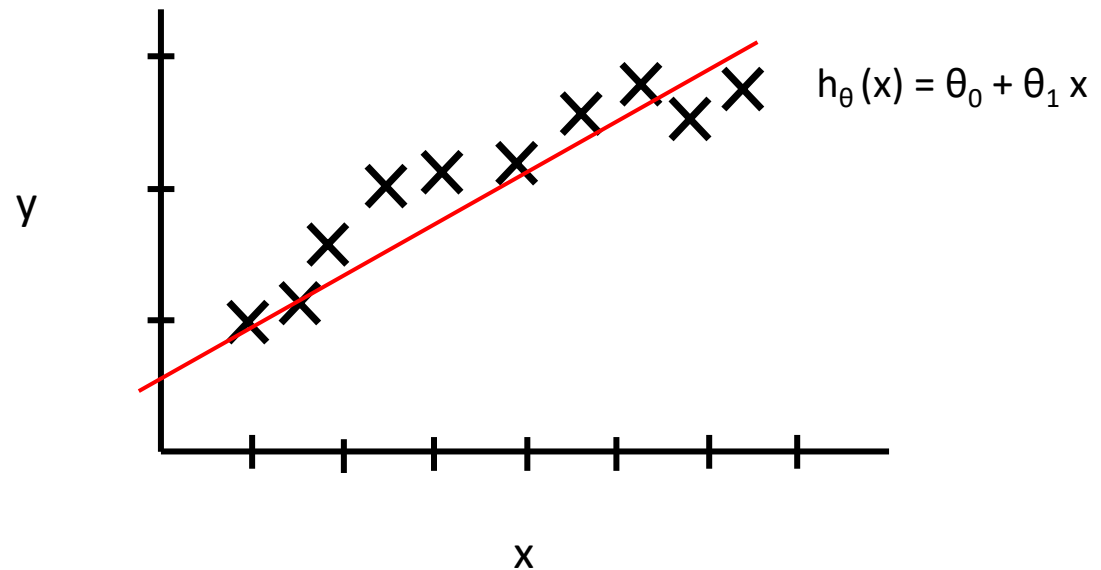
COST FUNCTION

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

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

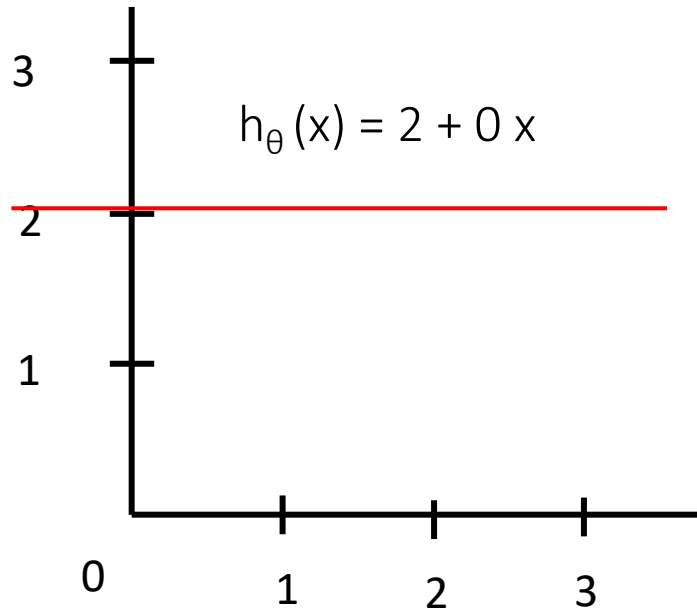
θ_i 's: Parameters

How to choose these parameters?

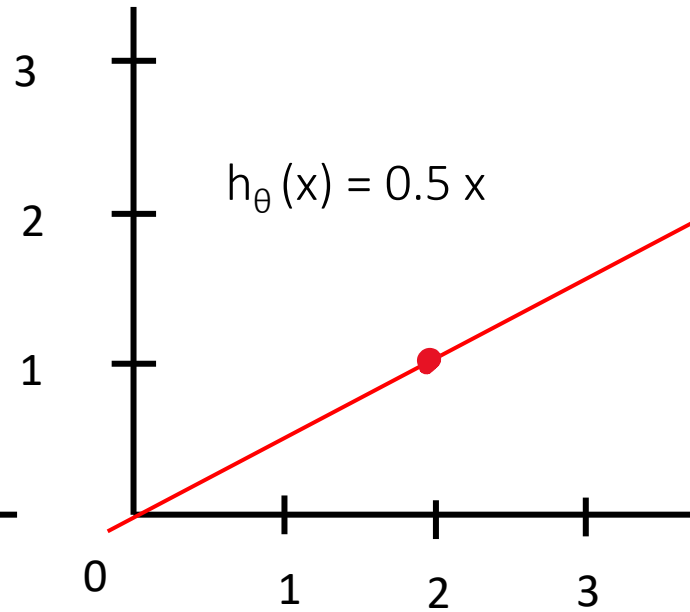


COST FUNCTION

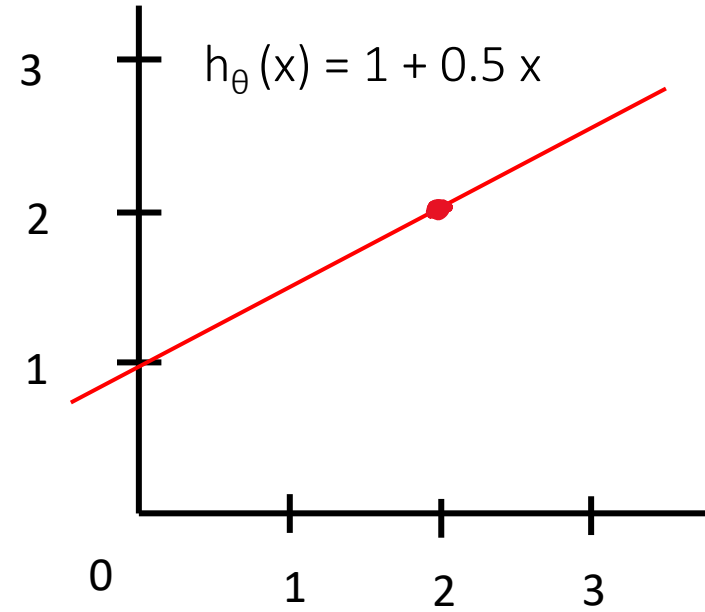
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



$$\theta_0 = 2$$
$$\theta_1 = 0$$

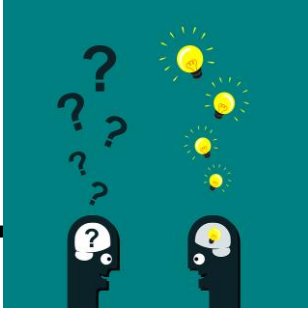


$$\theta_0 = 0$$
$$\theta_1 = 0.5$$



$$\theta_0 = 1$$
$$\theta_1 = 0.5$$

QUESTION



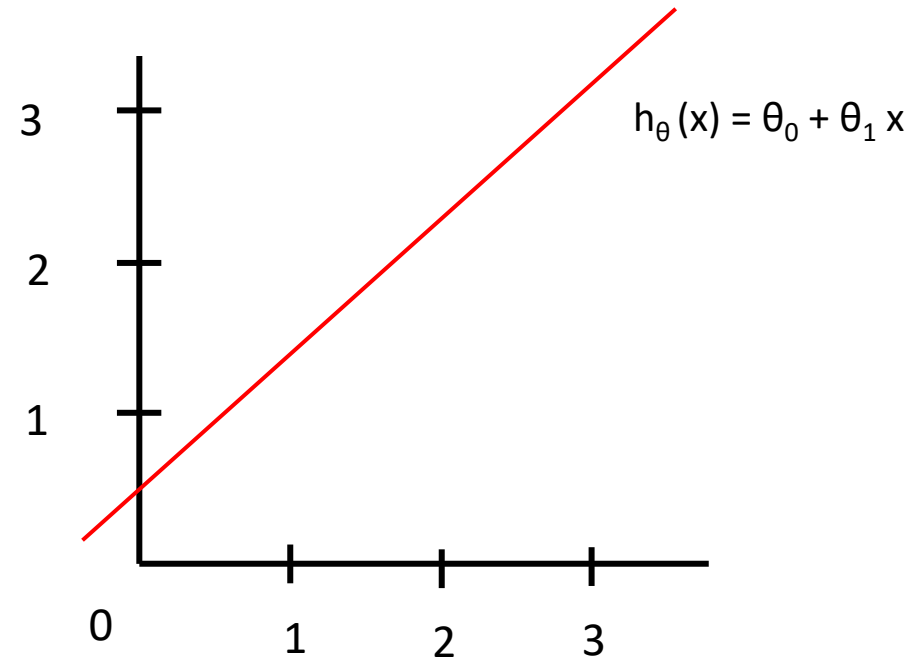
What are θ_0 and θ_1 ?

A: $\theta_0 = 0, \theta_1 = 1$

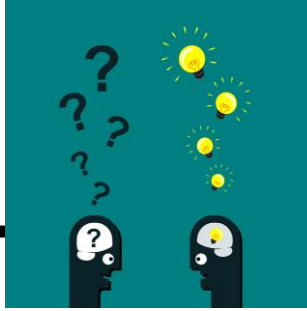
B: $\theta_0 = 0.5, \theta_1 = 1$

C: $\theta_0 = 1, \theta_1 = 0.5$

D: $\theta_0 = 1, \theta_1 = 1$



QUESTION



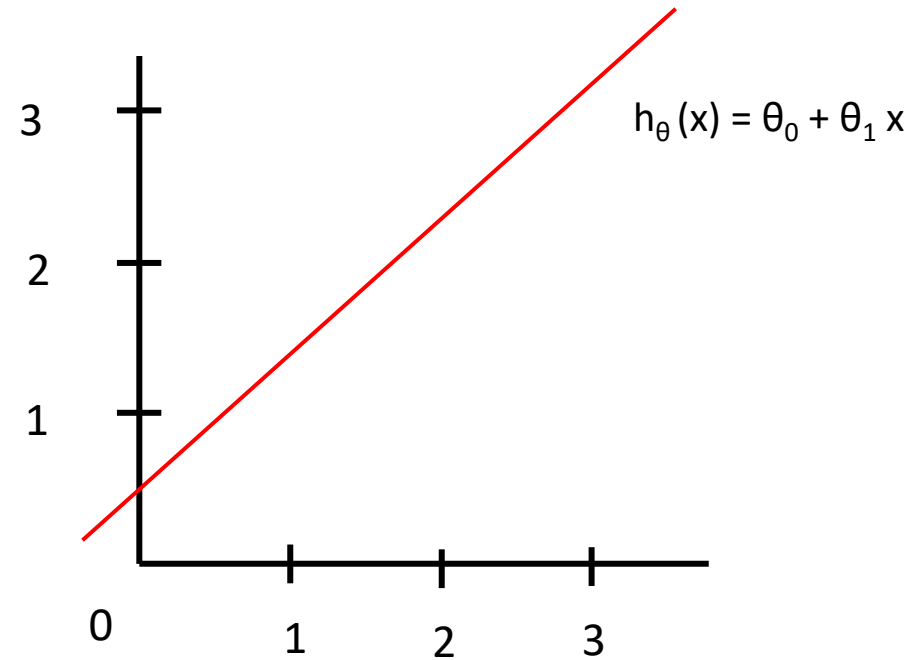
What are θ_0 and θ_1 ?

A: $\theta_0 = 0, \theta_1 = 1$

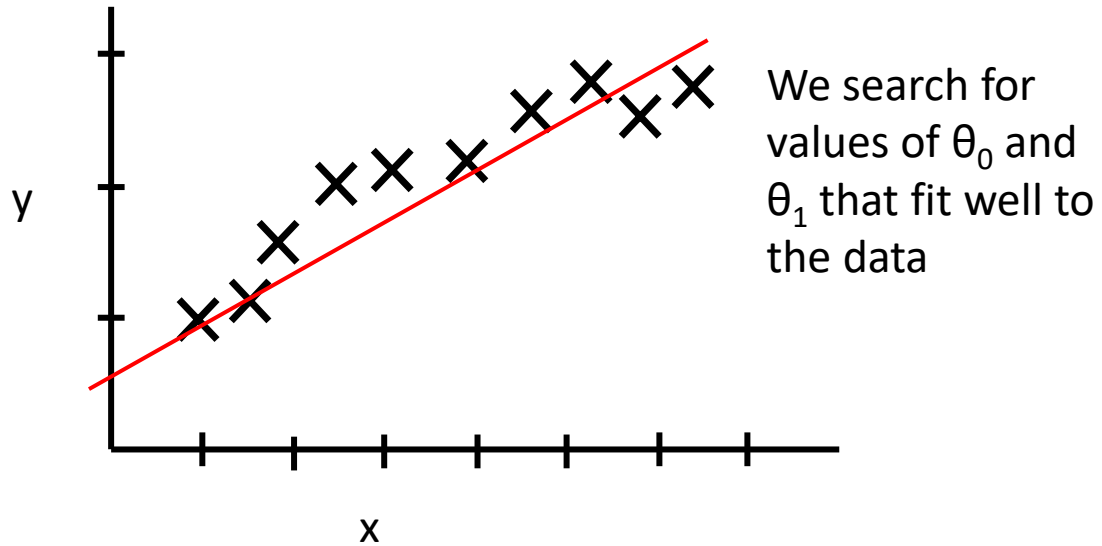
B: $\theta_0 = 0.5, \theta_1 = 1$

C: $\theta_0 = 1, \theta_1 = 0.5$

D: $\theta_0 = 1, \theta_1 = 1$



COST FUNCTION



→ We choose θ_0 and θ_1 so that $h_\theta(x)$ is close to y for our training examples (x,y)

Minimize θ_0, θ_1

Minimize $(h_\theta(x) - y)^2 \rightarrow$ “output – actual value”

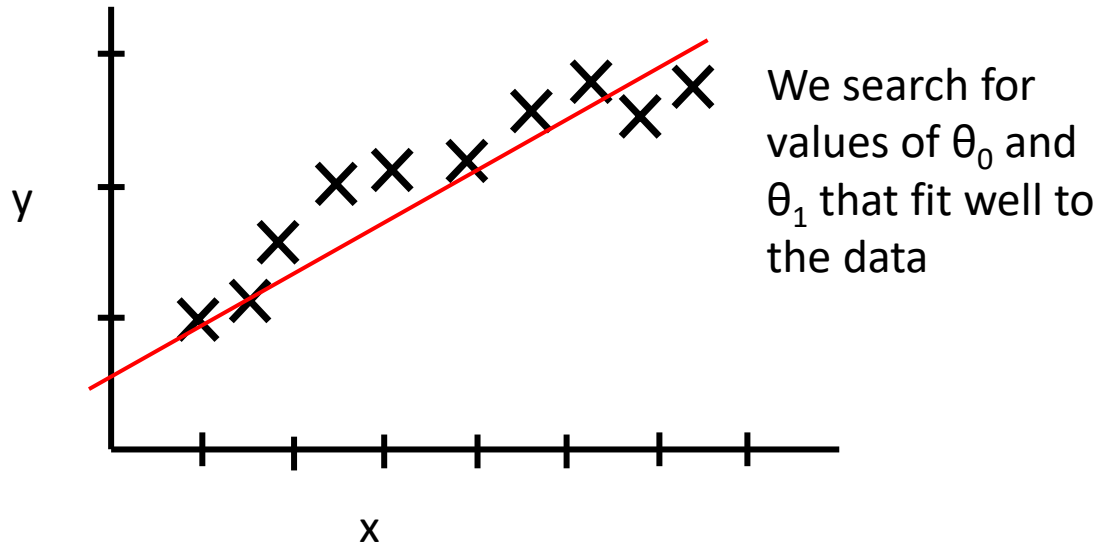
For all training data examples $(x^{(i)}, y^{(i)}) \rightarrow$

Minimize $\sum_{i=1}^m (h_\theta(x^i) - y^i)^2 \rightarrow$ average error \rightarrow

Minimize $\frac{1}{2m} \sum_{i=1}^m (h_\theta(x^i) - y^i)^2 \rightarrow$ Cost function
 \rightarrow

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

COST FUNCTION



→ We choose θ_0 and θ_1 so that $h_{\theta}(x)$ is close to y for our training examples (x, y)

→ Minimize over θ_0, θ_1 my cost function $J(\theta_0, \theta_1)$

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

minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1

→ **Squared error function**

→ There are other cost functions

→ The squared error cost function is most used for regression problems

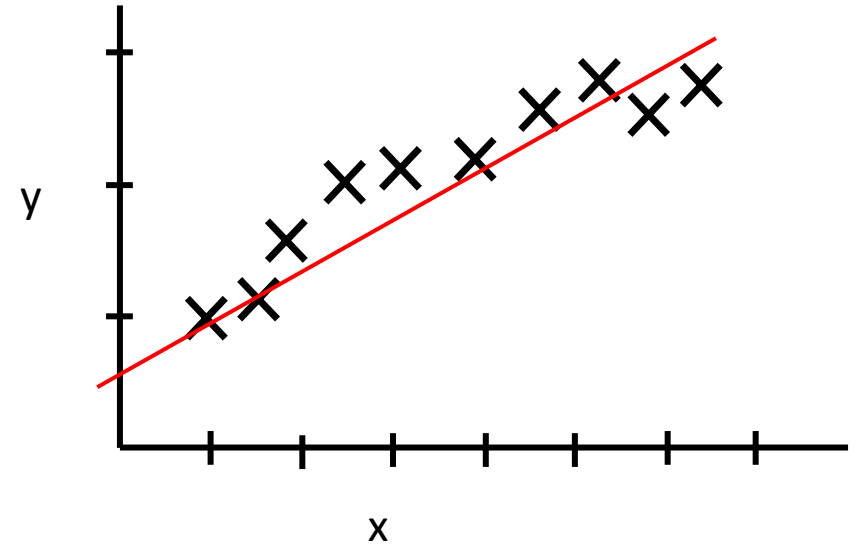
COST FUNCTION

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

Parameters: θ_0, θ_1

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

Goal: minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1



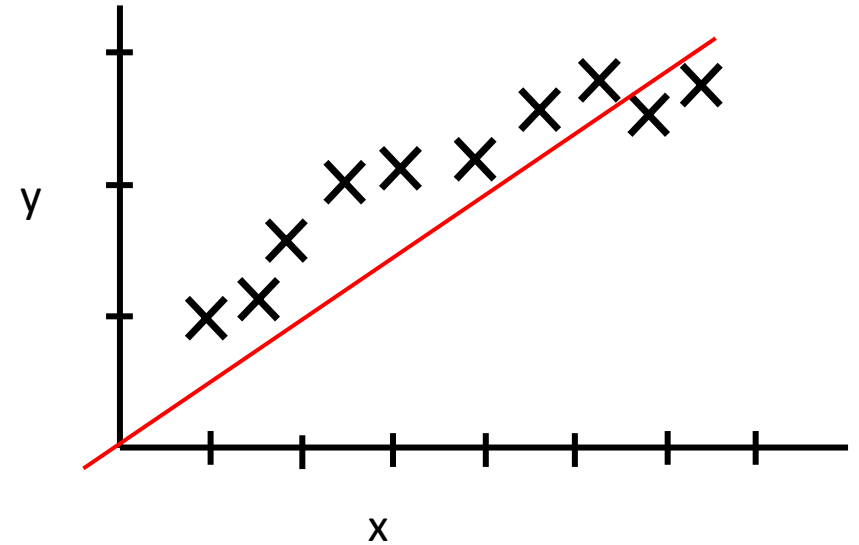
COST FUNCTION - SIMPLIFIED

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

Parameters: θ_1

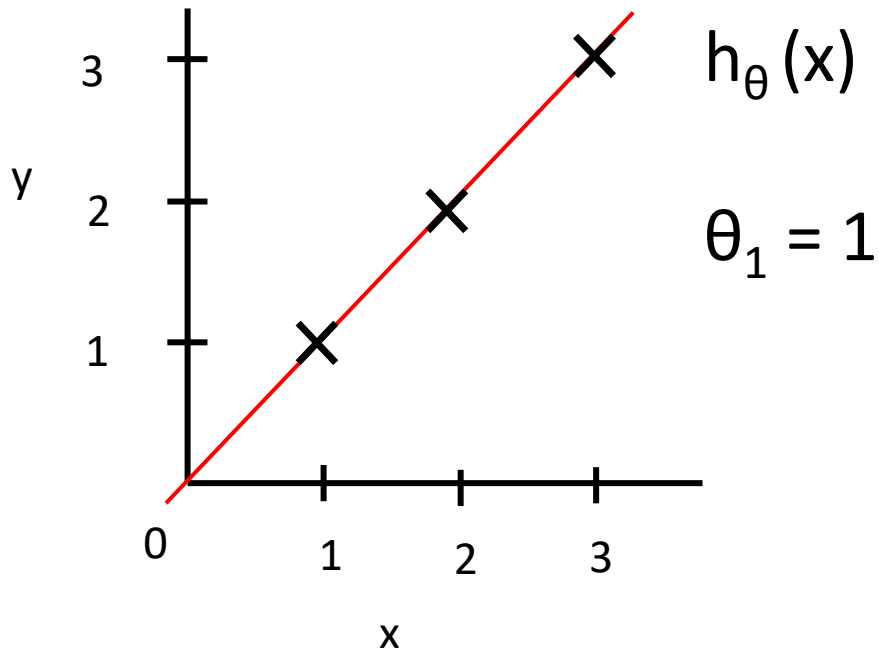
$$\text{Cost Function: } J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$$

Goal: minimize $J(\theta_1)$
 θ_1

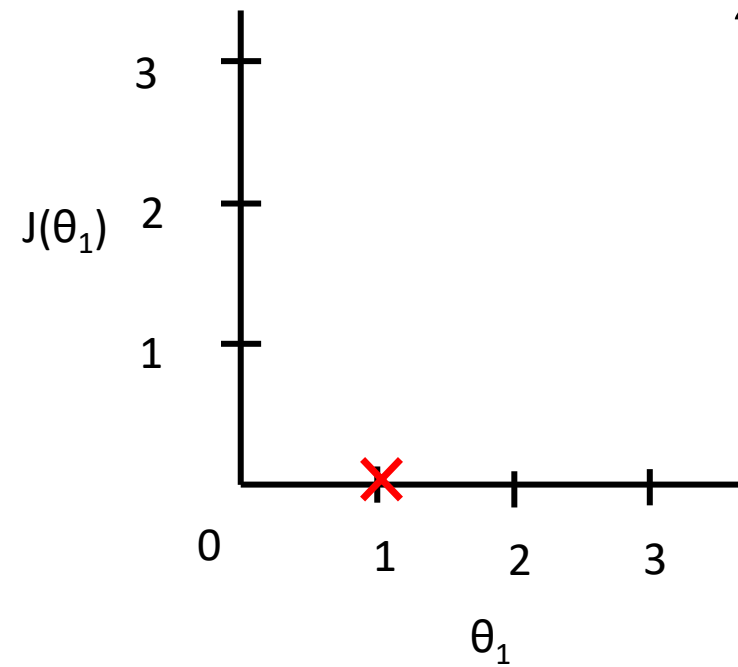


COST FUNCTION - SIMPLIFIED

Hypothesis: $h_{\theta}(x) = \theta_1 x$
→ Function of x



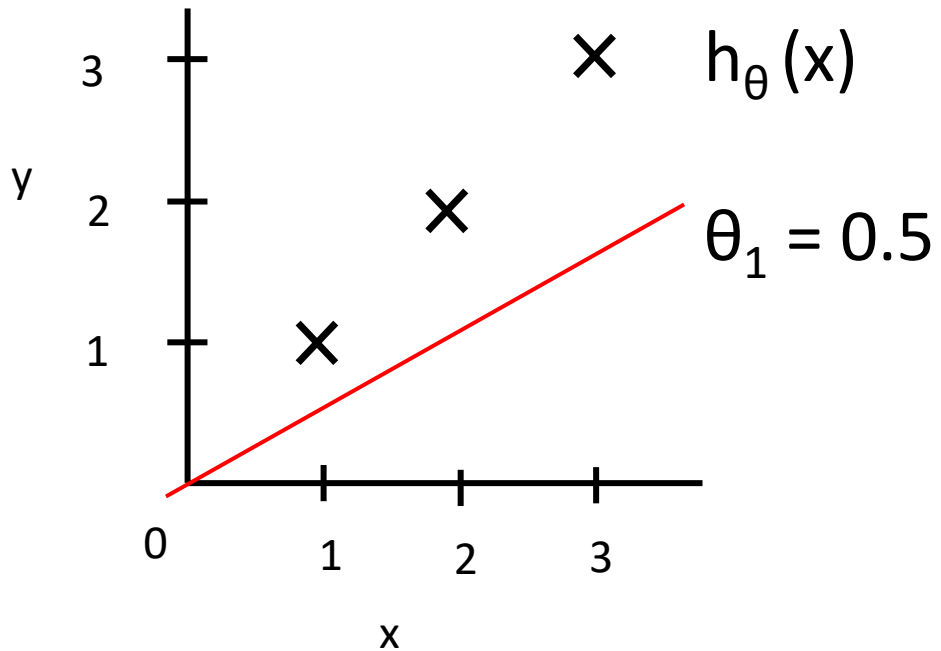
Cost Function: $J(\theta_1)$
→ Function of θ_1



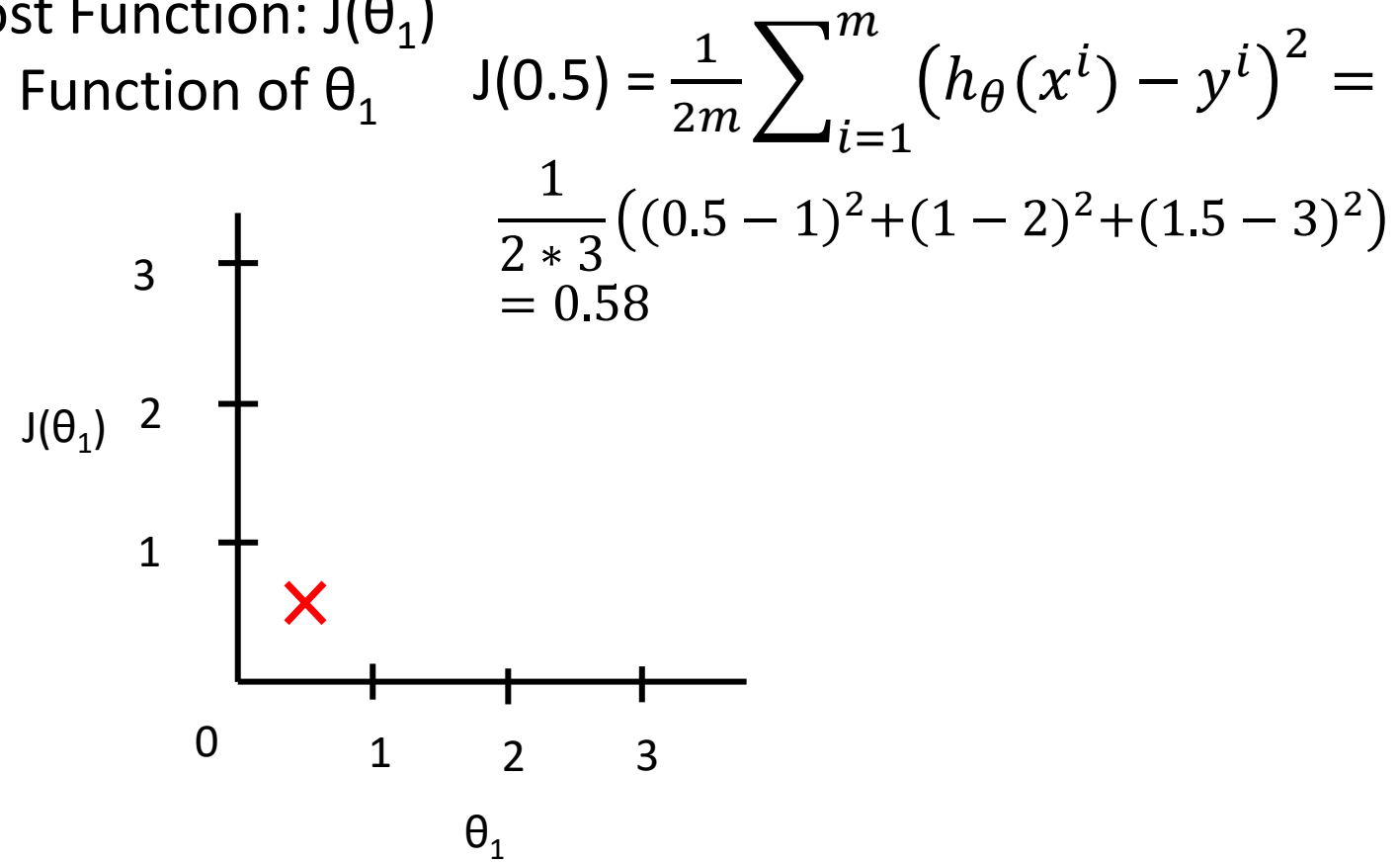
$$J(1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2 = \frac{1}{2 * 3} (0^2 + 0^2 + 0^2) = 0$$

COST FUNCTION - SIMPLIFIED

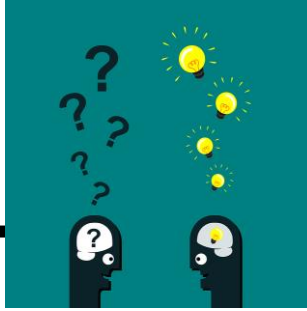
Hypothesis: $h_{\theta}(x) = \theta_1 x$
→ Function of x



Cost Function: $J(\theta_1)$
→ Function of θ_1



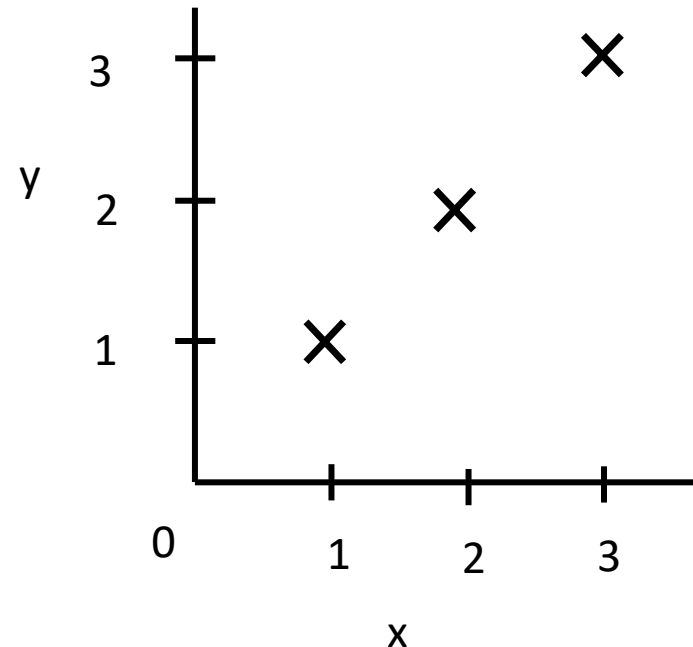
QUESTION



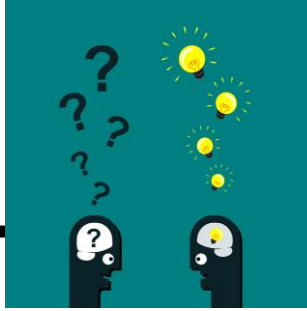
Suppose we have a training set with $m=3$ examples. Our hypothesis representation is $h_{\theta}(x) = \theta_1 x$, with parameter θ_1 .

The cost function is $J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$. What is $J(0)$?

- A: 0
- B: 1/6
- C: 1
- D: 14/6



QUESTION



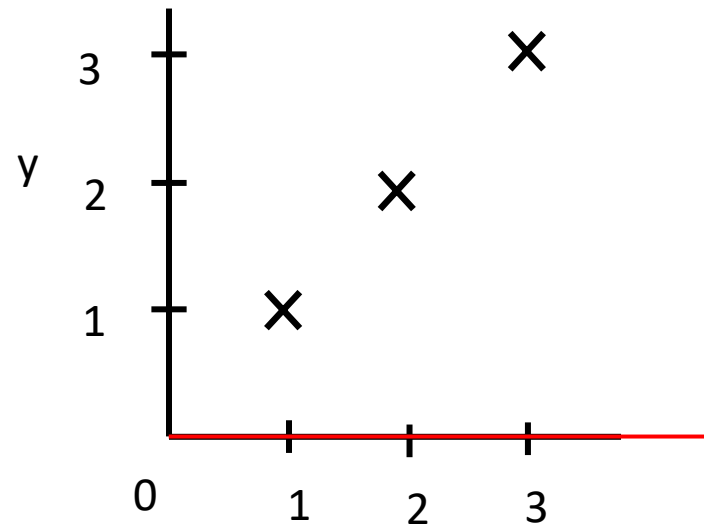
Suppose we have a training set with $m=3$ examples. Our hypothesis representation is $h_{\theta}(x) = \theta_1 x$, with parameter θ_1 . The cost function is $J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$. What is $J(0)$?

A: 0

B: $1/6$

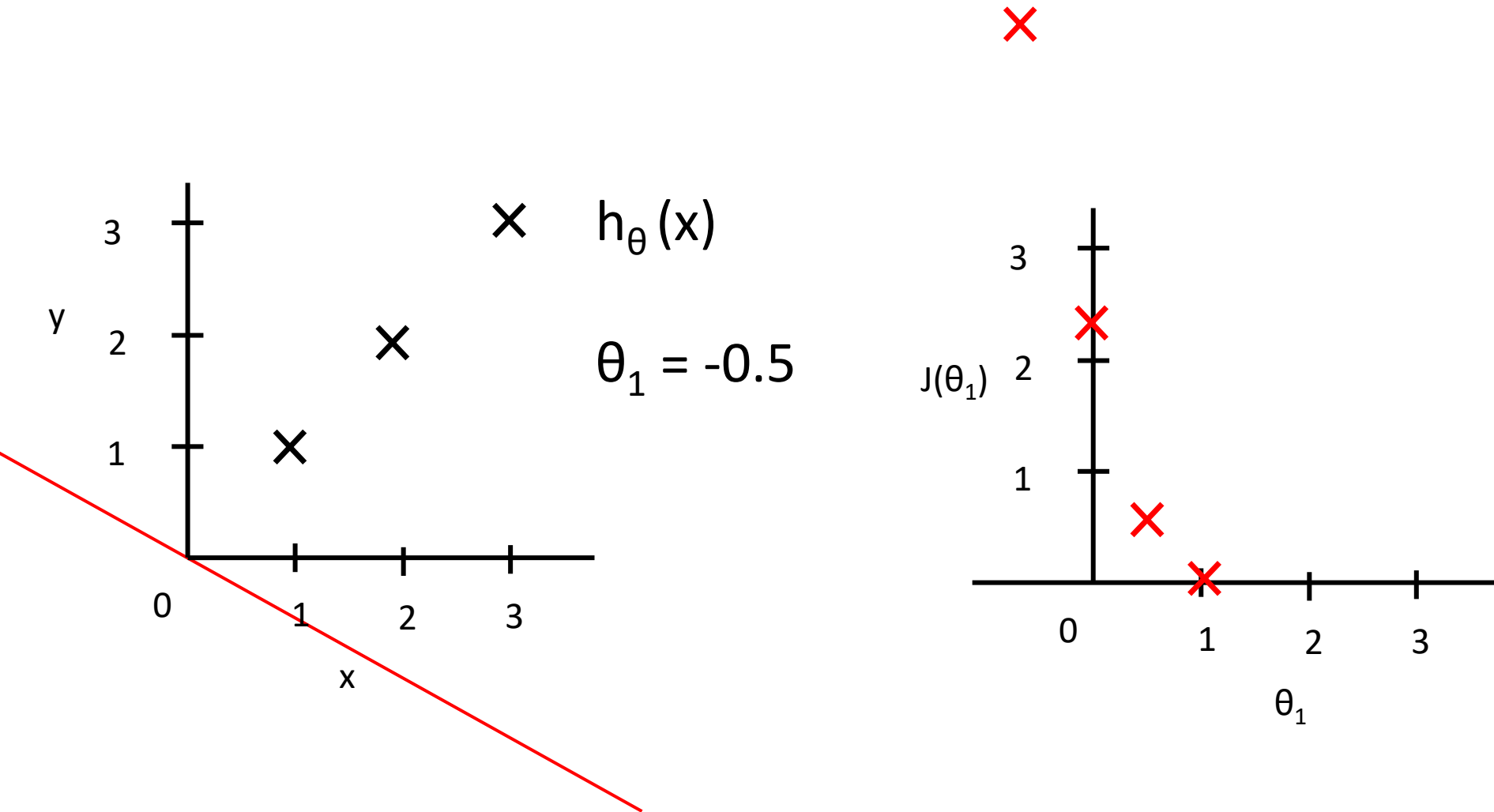
C: 1

D: $14/6$

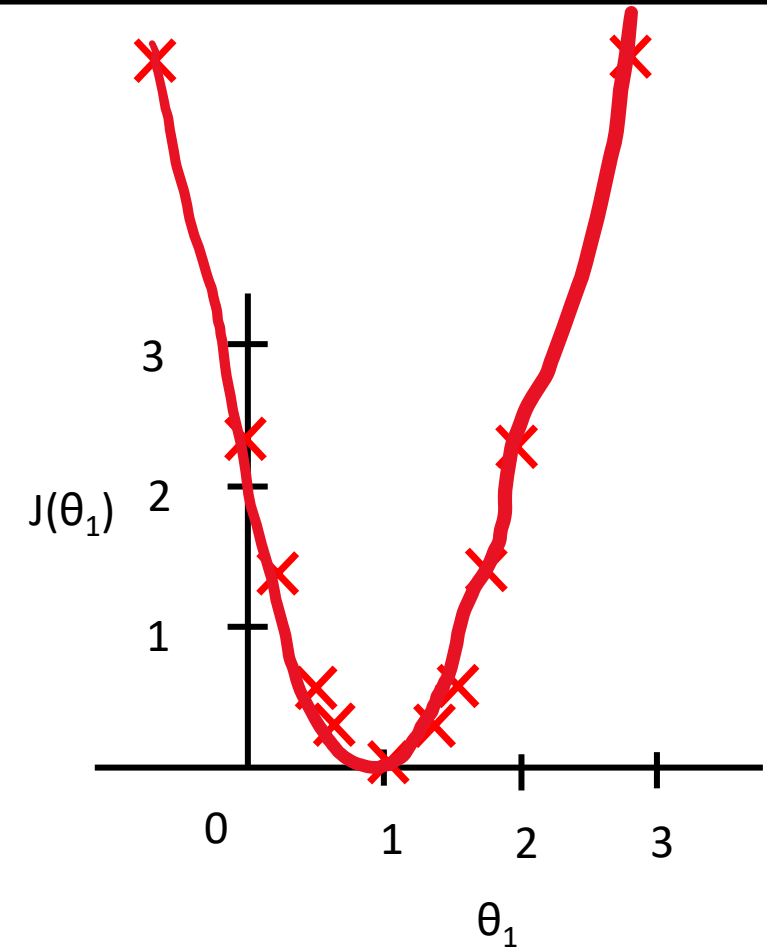
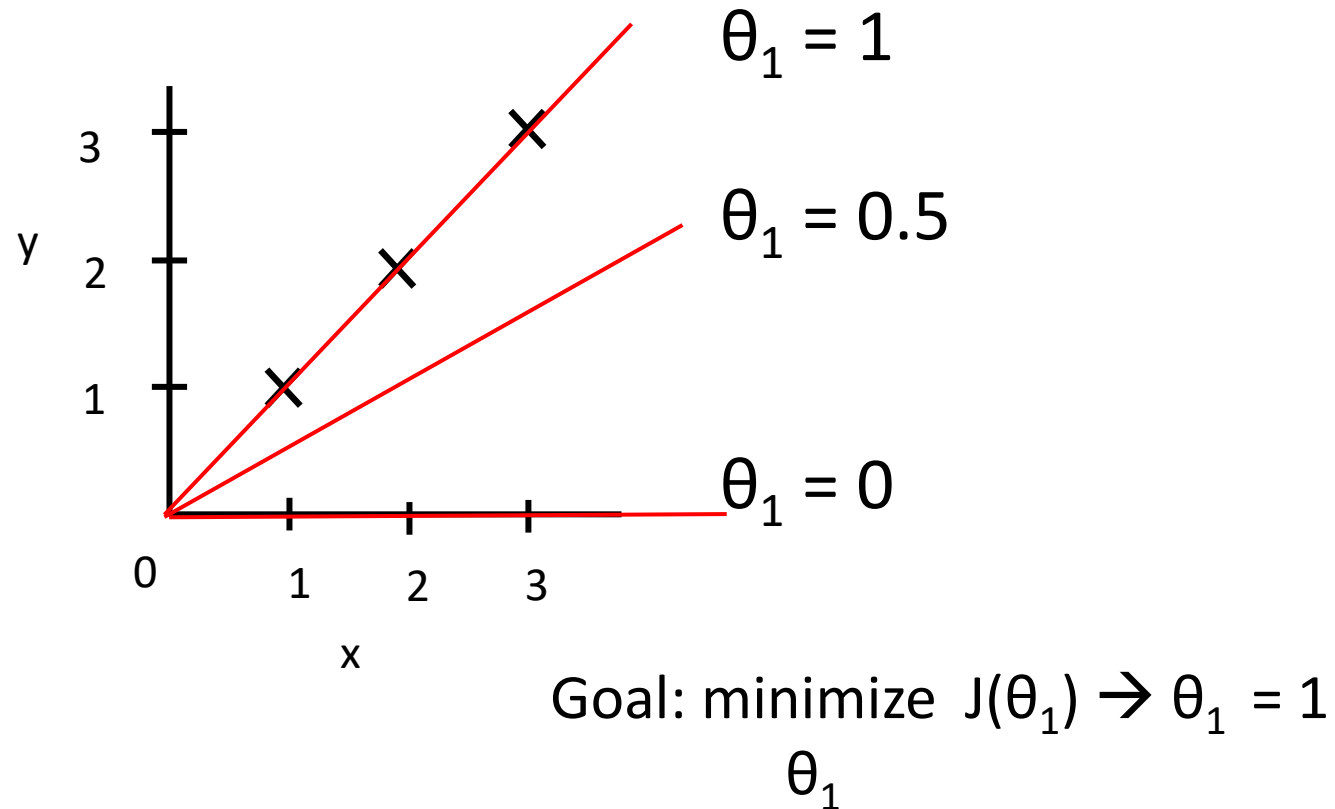


$$J(0) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta} x^i - y^i)^2 = \frac{1}{2 \cdot 3} ((0 - 1)^2 + (0 - 2)^2 + (0 - 3)^2) = 14/6 = 2.3$$

COST FUNCTION - SIMPLIFIED



COST FUNCTION - SIMPLIFIED



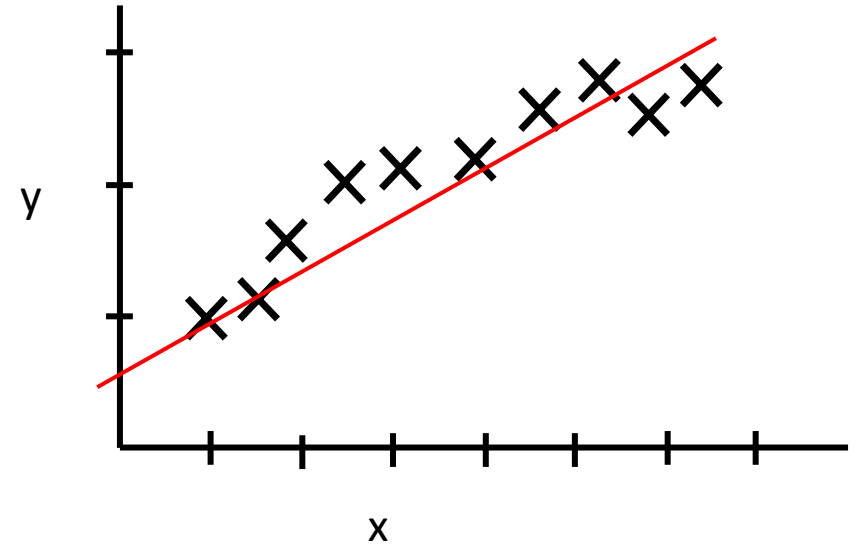
COST FUNCTION

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

Parameters: θ_0, θ_1

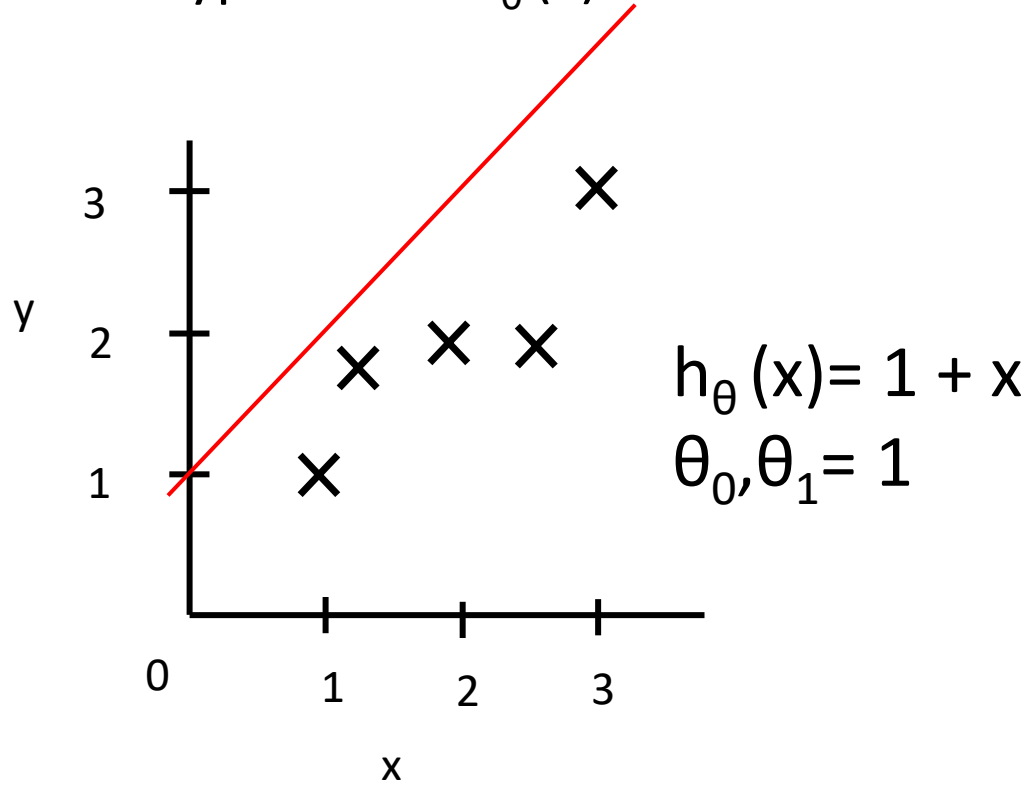
Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$

Goal: minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1



COST FUNCTION

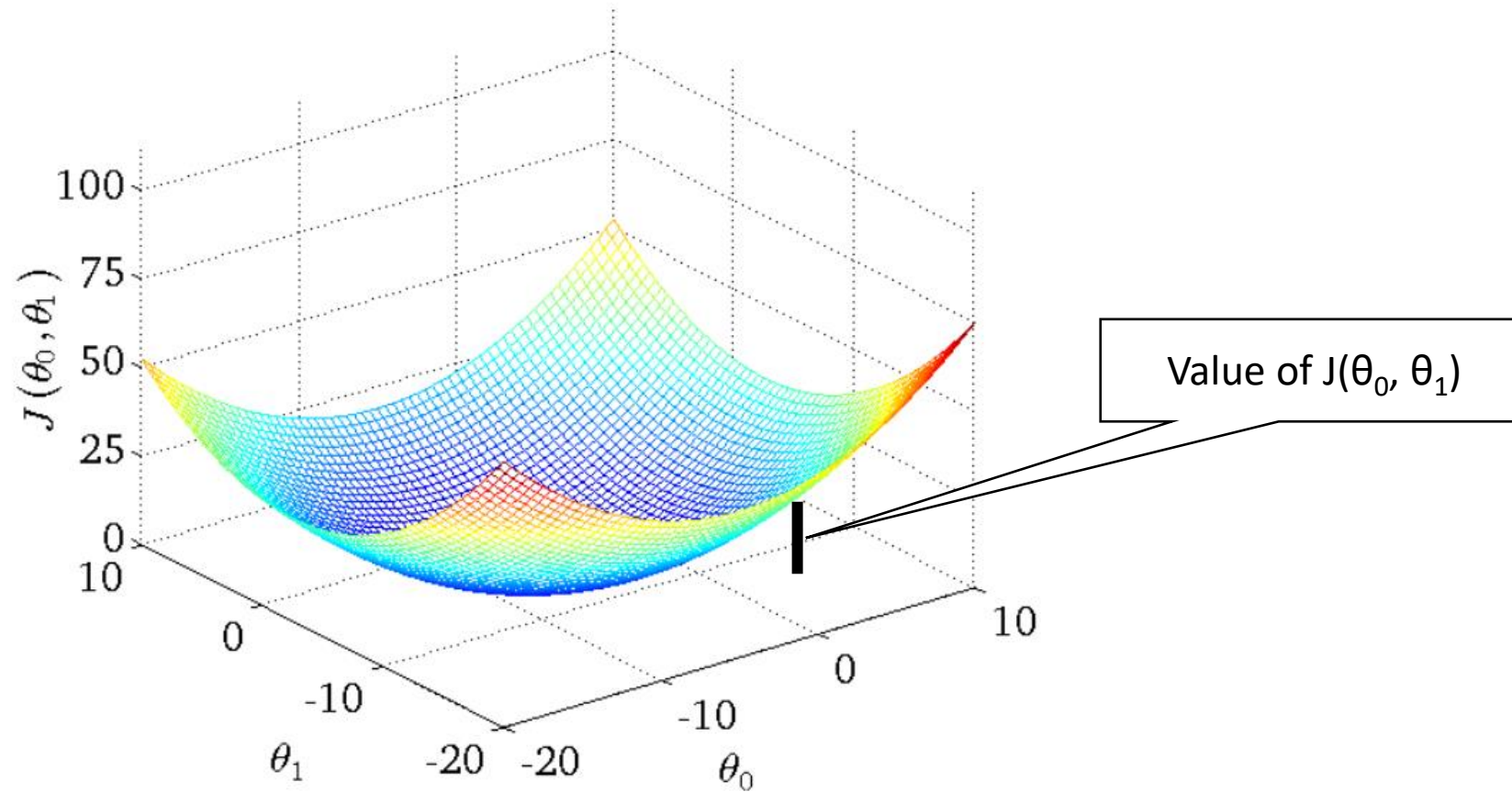
Hypothesis: $h_{\theta}(x)$



Cost Function: $J(\theta_0, \theta_1)$

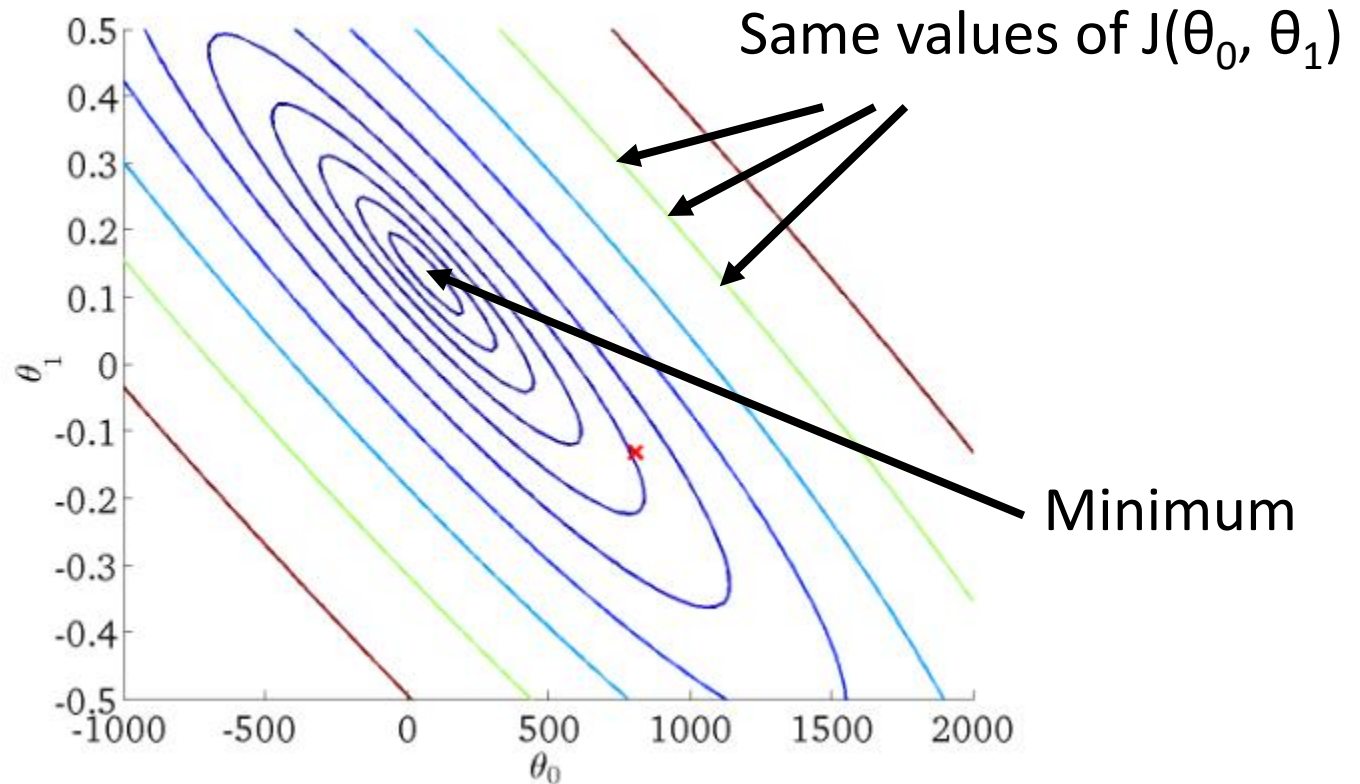
→ 3D-Plot

COST FUNCTION

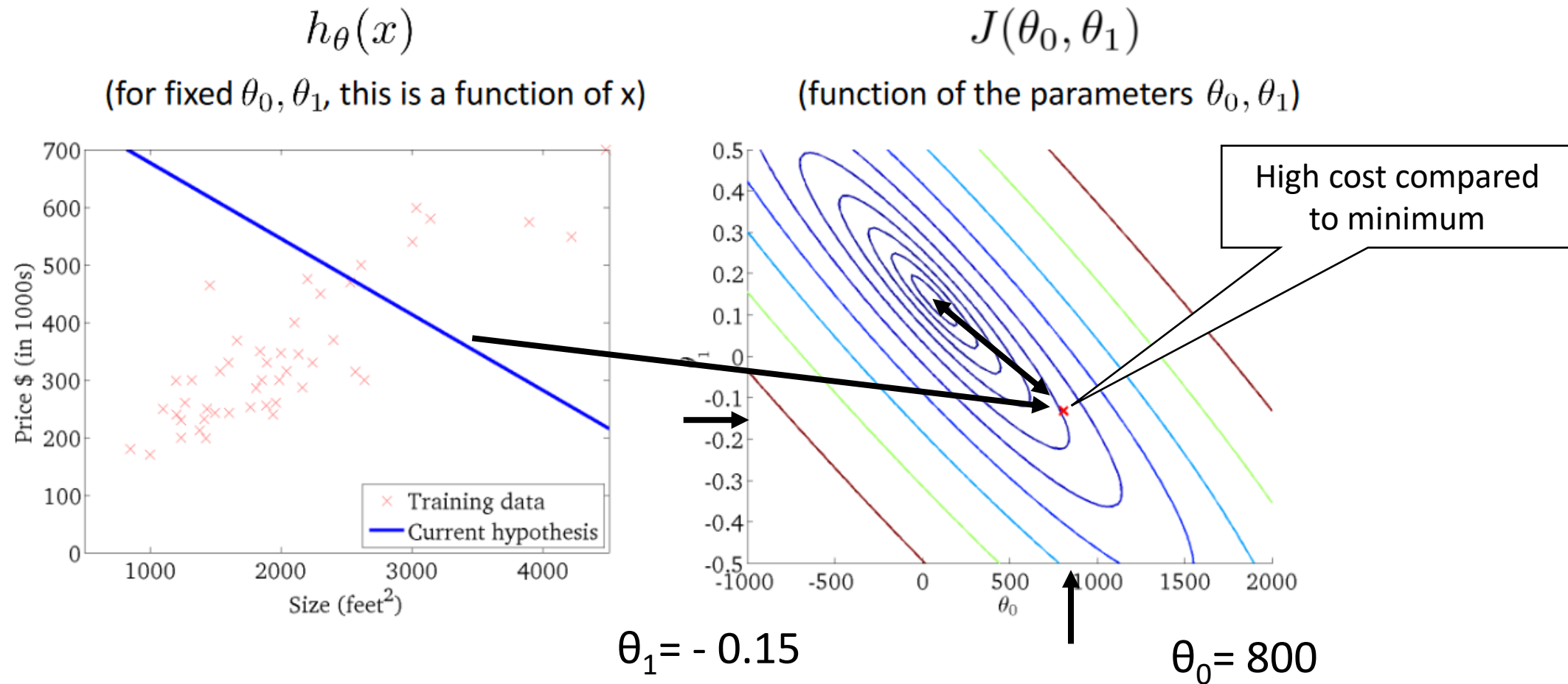


COST FUNCTION CONTOUR PLOTS

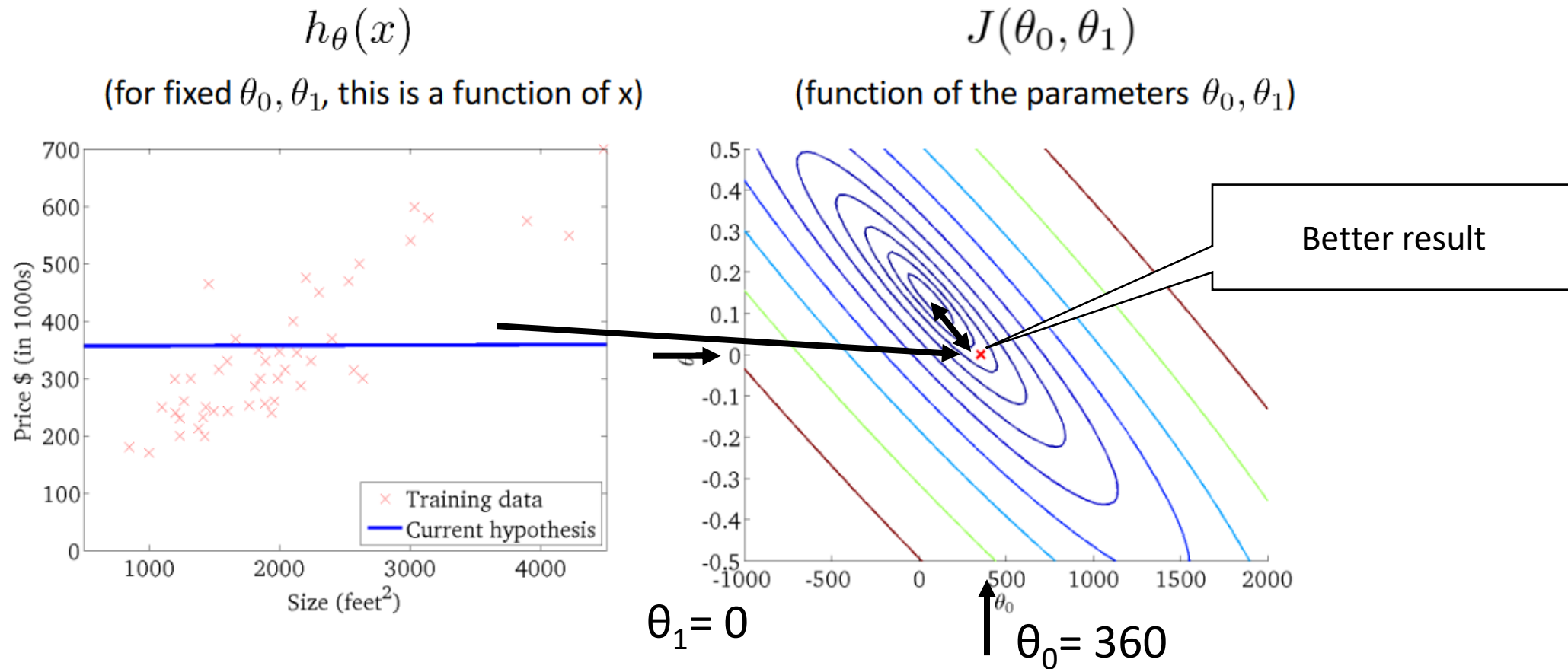
- For a function $J(\theta_0, \theta_1)$ of two variables, assigned different colours to different values of J
- Pick some values to plot
- The result will be contours—curves in the graph along which the values of $J(\theta_0, \theta_1)$ are constant



COST FUNCTION CONTOUR PLOTS



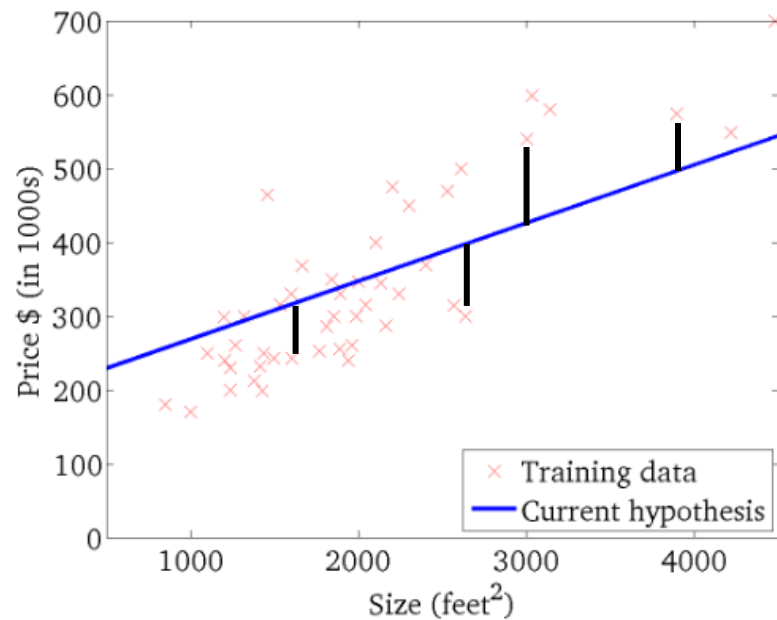
COST FUNCTION CONTOUR PLOTS



COST FUNCTION CONTOUR PLOTS

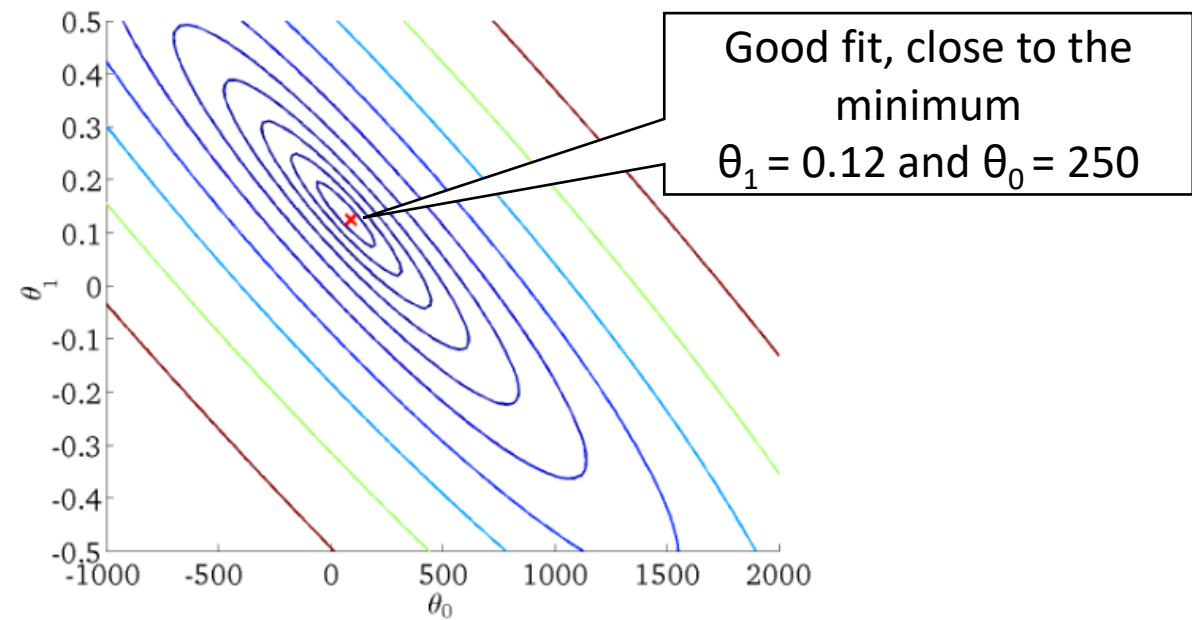
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



Our overall goal is to have an algorithm that minimizes the cost function

GRADIENT DESCENT

Cost function $J(\theta_0, \theta_1)$ – or any other function $J(\theta_0, \theta_1)$

Goal: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Idea:

- Start with any θ_0, θ_1 – e.g. $\theta_0 = 0, \theta_1 = 0$
- Keep changing to reduce $J(\theta_0, \theta_1)$ until the minimum is found

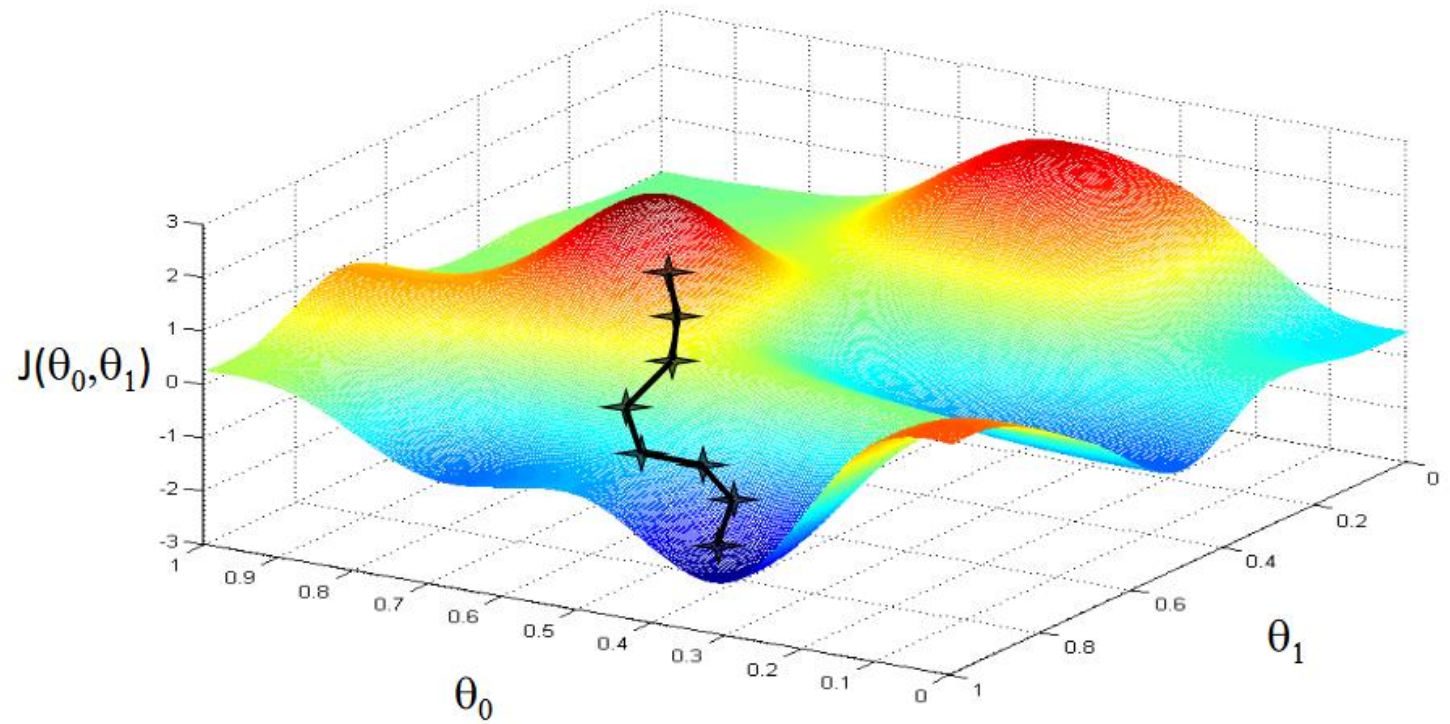
Note: Gradient Descent also works for $\min_{\theta_0, \dots, \theta_n} J(\theta_0, \dots, \theta_n)$

GRADIENT DESCENT

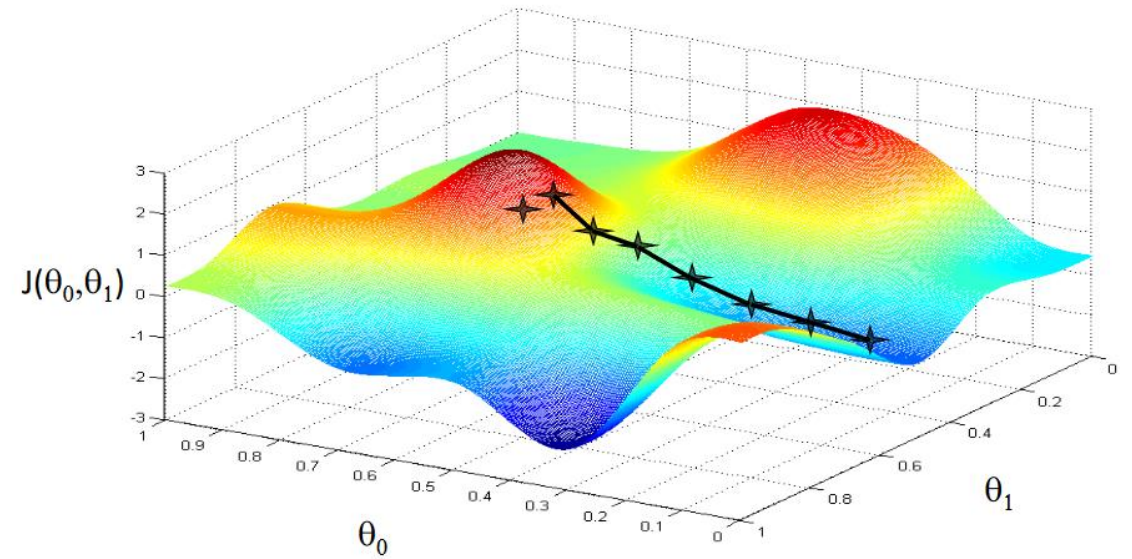
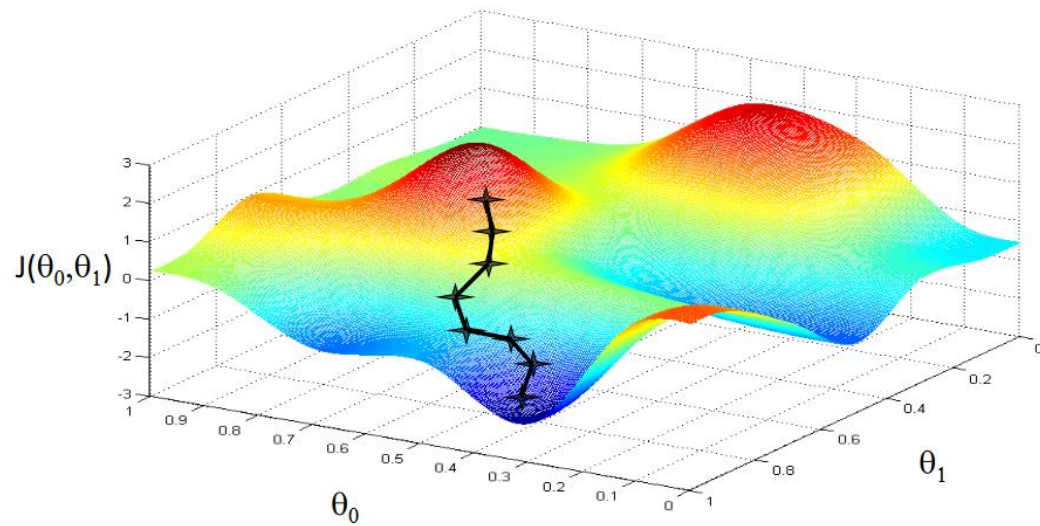
“look around 360°”

Take a step

Find a local optimum



GRADIENT DESCENT



GRADIENT DESCENT ALGORITHM

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \text{ (for } j = 0 \text{ and } j = 1)$$

}

α = learning rate \rightarrow “step size”

$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ = derivate term

GRADIENT DESCENT ALGORITHM

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \text{ (for } j = 0 \text{ and } j = 1)$$

}

Simultaneous update (of θ_0, θ_1)

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 = \text{temp0}$$

$$\theta_1 = \text{temp1}$$

Not Gradient Descent:

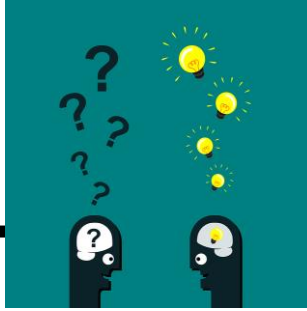
$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_0 = \text{temp0}$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_1 = \text{temp1}$$

QUESTION



Suppose $\theta_0 = 1$, $\theta_1 = 2$, and we simultaneously update θ_0 and θ_1 using the rule:

$\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$ (for $j = 0$ and $j = 1$). What are the resulting values for θ_0 and θ_1 ?

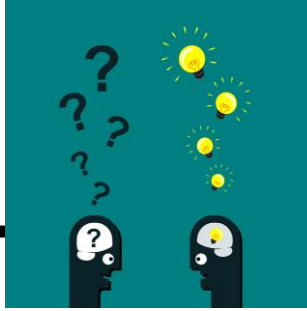
A: $\theta_0 = 1$, $\theta_1 = 2$

B: $\theta_0 = 1 + \sqrt{2}$, $\theta_1 = 2 + \sqrt{2}$

C: $\theta_0 = 2 + \sqrt{2}$, $\theta_1 = 1 + \sqrt{2}$

D: $\theta_0 = 1 + \sqrt{2}$, $\theta_1 = 2 + \sqrt{(1 + \sqrt{2}) * 2}$

QUESTION



Suppose $\theta_0 = 1$, $\theta_1 = 2$, and we simultaneously update θ_0 and θ_1 using the rule:

$\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$ (for $j = 0$ and $j=1$). What are the resulting values for θ_0 and θ_1 ?

A: $\theta_0 = 1$, $\theta_1 = 2$

B: $\theta_0 = 1 + \sqrt{2}$, $\theta_1 = 2 + \sqrt{2} \rightarrow \theta_0 := 1 + \sqrt{1 * 2}$; $\theta_1 := 2 + \sqrt{1 * 2}$

C: $\theta_0 = 2 + \sqrt{2}$, $\theta_1 = 1 + \sqrt{2}$

D: $\theta_0 = 1 + \sqrt{2}$, $\theta_1 = 2 + \sqrt{(1 + \sqrt{2}) * 2}$

GRADIENT DESCENT ALGORITHM

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \text{ (for } j=0 \text{ and } j=1)$$

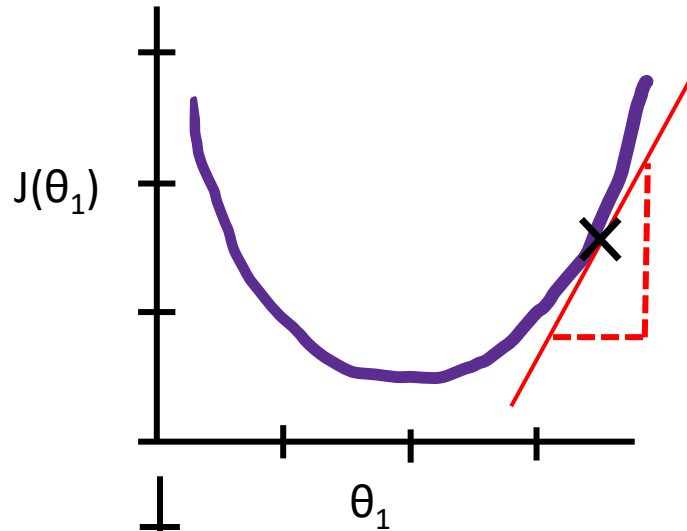
}

α = learning rate \rightarrow “step size”

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \text{derivate term}$$

Simplification: $\text{Min}_{\theta_1} J(\theta_1)$ with $\theta_1 \in \mathbb{R}$

GRADIENT DESCENT ALGORITHM – DERIVATE TERM



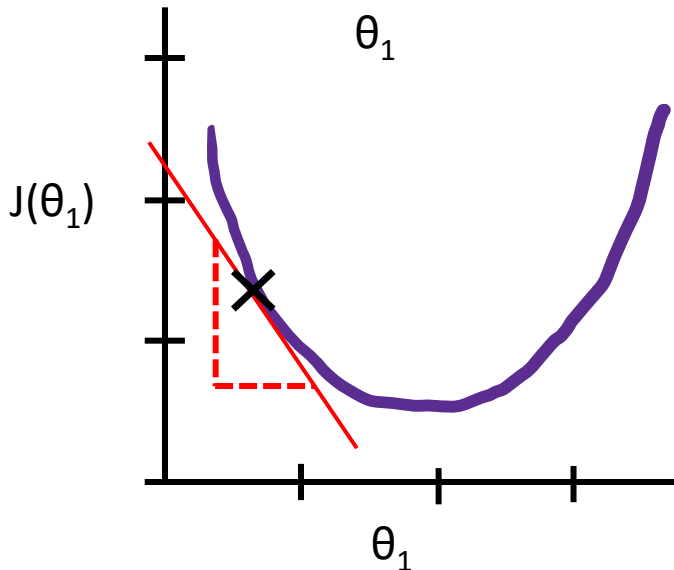
with $\theta_1 \in \mathbb{R}$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$$

Positive slope \rightarrow derivate term ≥ 0

$\rightarrow \theta_1 := \theta_1 - \alpha * (\text{positive number})$

$\rightarrow \theta_1$ decreases

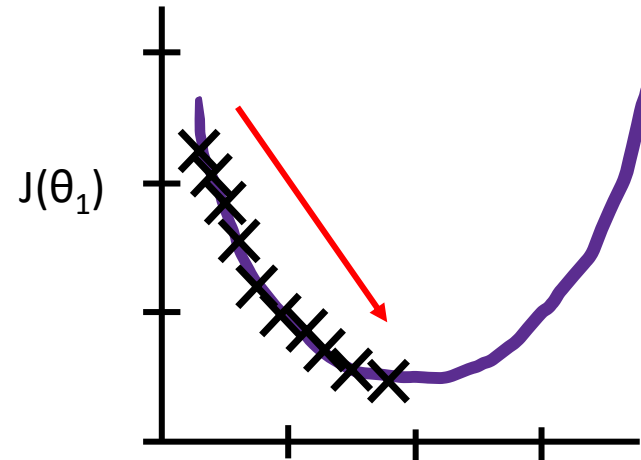


\rightarrow Negative slope \rightarrow derivate term ≤ 0

$\rightarrow \theta_1 := \theta_1 - \alpha * (\text{negative number})$

$\rightarrow \theta_1$ increases

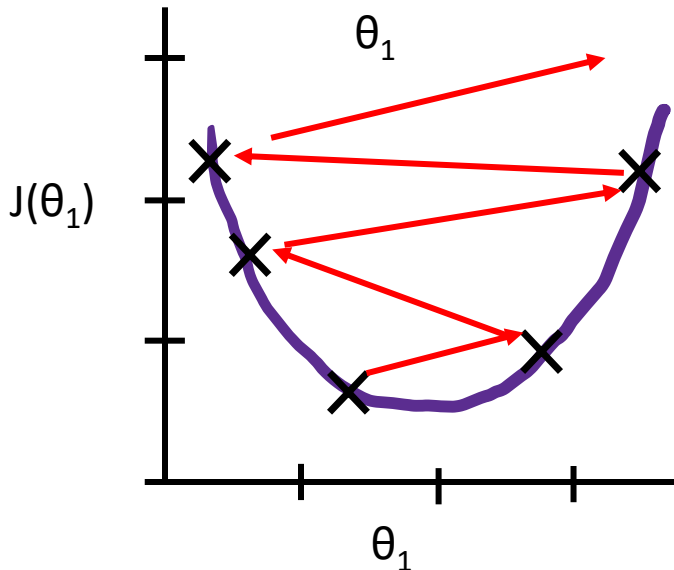
GRADIENT DESCENT ALGORITHM – ALPHA



with $\theta_1 \in \mathbb{R}$

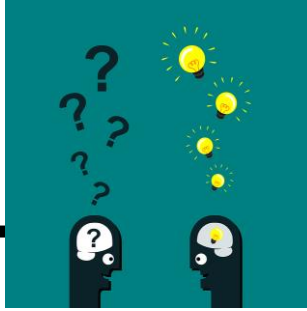
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$$

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



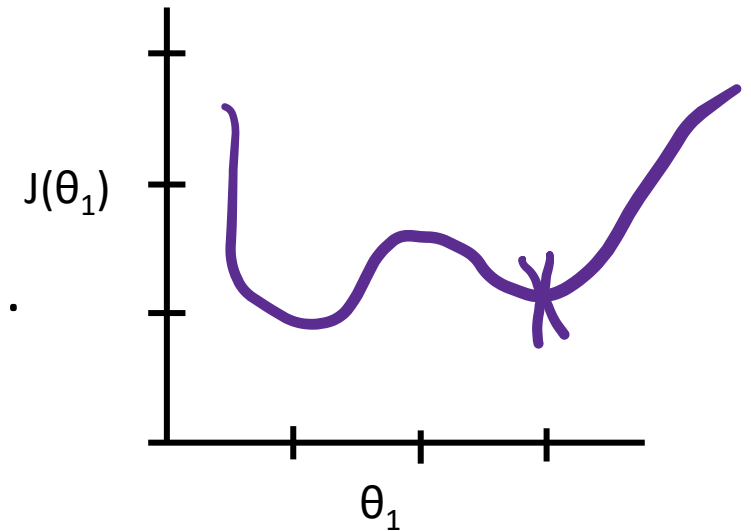
→ If α is too large, gradient descent may overshoot the minimum. It may fail to converge, or even diverge.

QUESTION

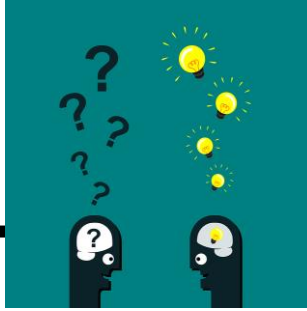


Suppose θ_1 is at a local optimum of $J(\theta_1)$, such as shown in the figure. What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$ do?

- A: Leave θ_1 unchanged.
- B: Change θ_1 in a random direction.
- C: Move θ_1 in the direction of the global minimum of $J(\theta_1)$.
- D: Decrease θ_1 .

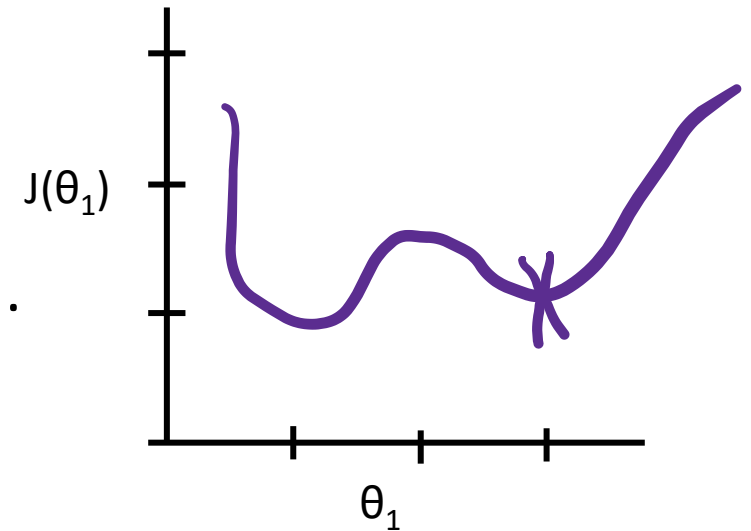


QUESTION

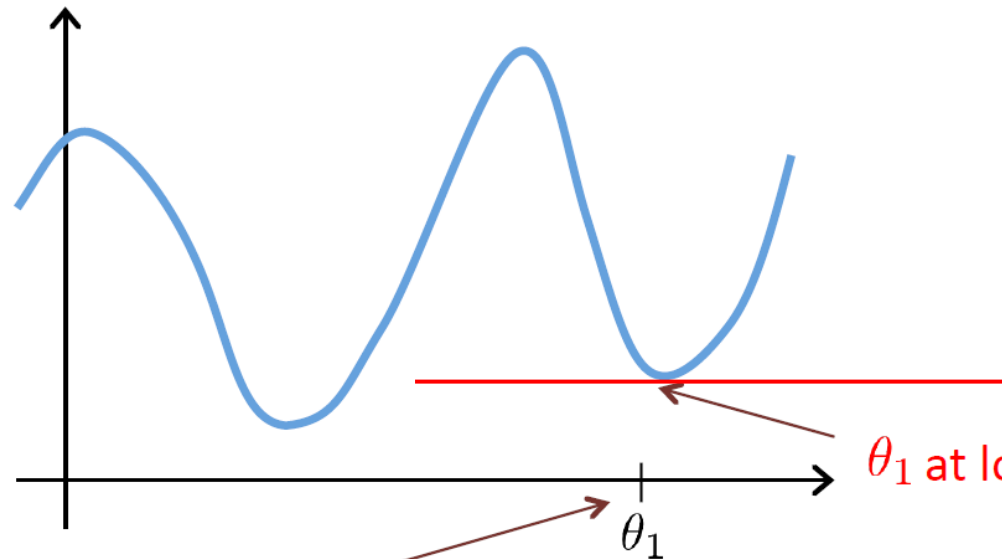
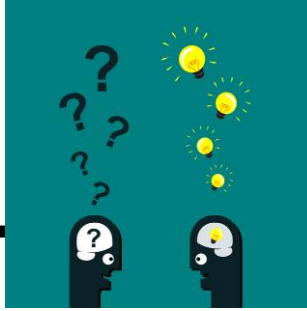


Suppose θ_1 is at a local optimum of $J(\theta_1)$, such as shown in the figure. What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$ do?

- A: Leave θ_1 unchanged.
- B: Change θ_1 in a random direction.
- C: Move θ_1 in the direction of the global minimum of $J(\theta_1)$.
- D: Decrease θ_1 .



QUESTION - DETAILS



derivate term = 0

$$\theta_1 := \theta_1 - \alpha * 0$$

$$\rightarrow \theta_1 := \theta_1$$

Current value of θ_1

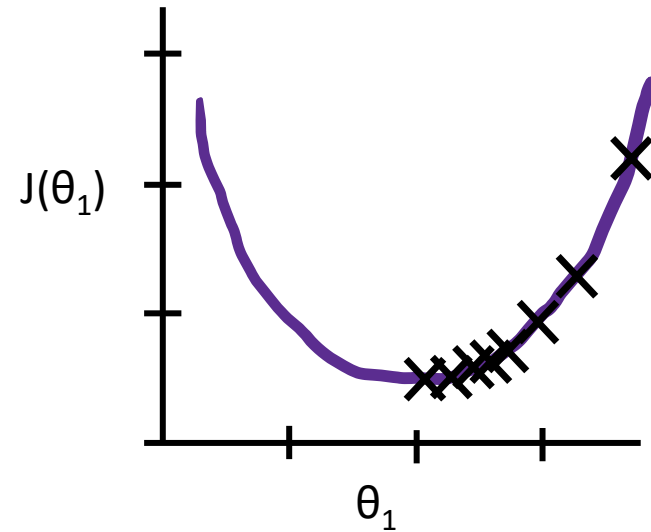
$$\theta_1 := \theta_1 - \alpha \boxed{\frac{d}{d\theta_1} J(\theta_1)}$$

GRADIENT DESCENT ALGORITHM

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps (the derivative term gets closer to 0) \rightarrow no need to decrease α over time.



GRADIENT DESCENT AND LINEAR REGRESSION

Gradient descent
algorithm

Repeat until convergence
{

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

(for $j = 0$ and $j = 1$)

}

Linear regression model

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

Parameters: θ_0, θ_1

Cost Function:

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

Goal: minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1

PARTIAL DERIVATIVE TERM

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

$$j = 0: \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})$$

$$j = 1: \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) * x^i$$

PARTIAL DERIVATIVE TERM FOR A SINGLE EXAMPLE

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_{\theta}(x) - y)^2$$

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

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} ((\theta_0 + \theta_1 x) - y)^2$$

$$j = 0 \rightarrow \frac{\partial}{\partial \theta_0} J(\theta) = \frac{\partial}{\partial \theta_0} \frac{1}{2} ((\theta_0 + \theta_1 x) - y)^2 = 2 * \frac{1}{2} * ((\theta_0 + \theta_1 x) - y) * 1$$

$$= (h_{\theta}(x^{(i)}) - y^{(i)})$$

$$j = 1 \rightarrow \frac{\partial}{\partial \theta_1} J(\theta) = \frac{\partial}{\partial \theta_1} \frac{1}{2} ((\theta_0 + \theta_1 x) - y)^2 = 2 * \frac{1}{2} * ((\theta_0 + \theta_1 x) - y) * (1 * x)$$

$$= (h_{\theta}(x^{(i)}) - y^{(i)}) * x$$


GRADIENT DESCENT ALGORITHM

Repeat until convergence {

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

(for $j=0$ and $j=1$)

}


$$j=0: \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)$$

$$j=1: \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x^i$$

Repeat until convergence {

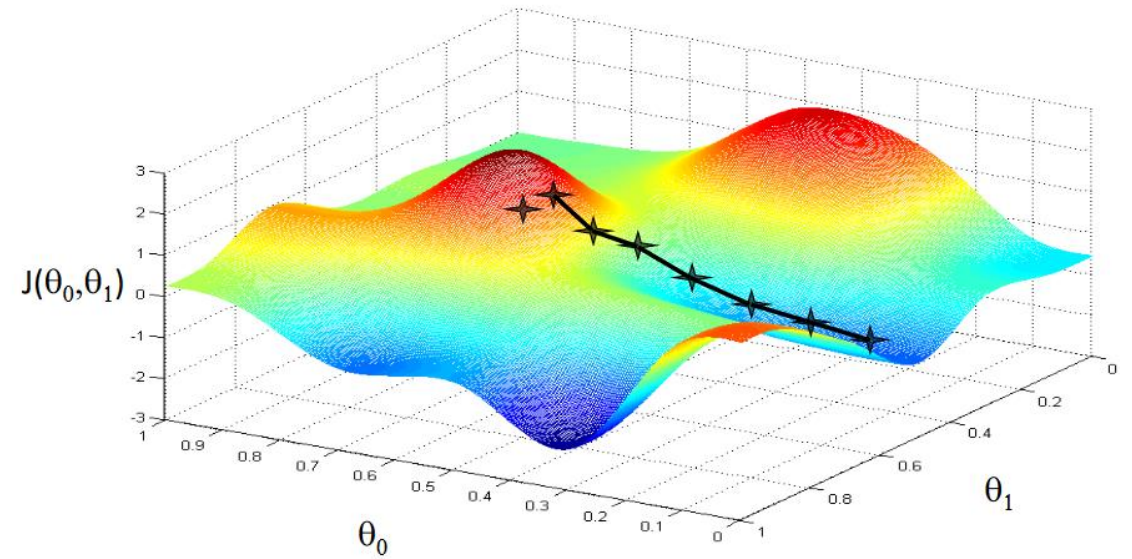
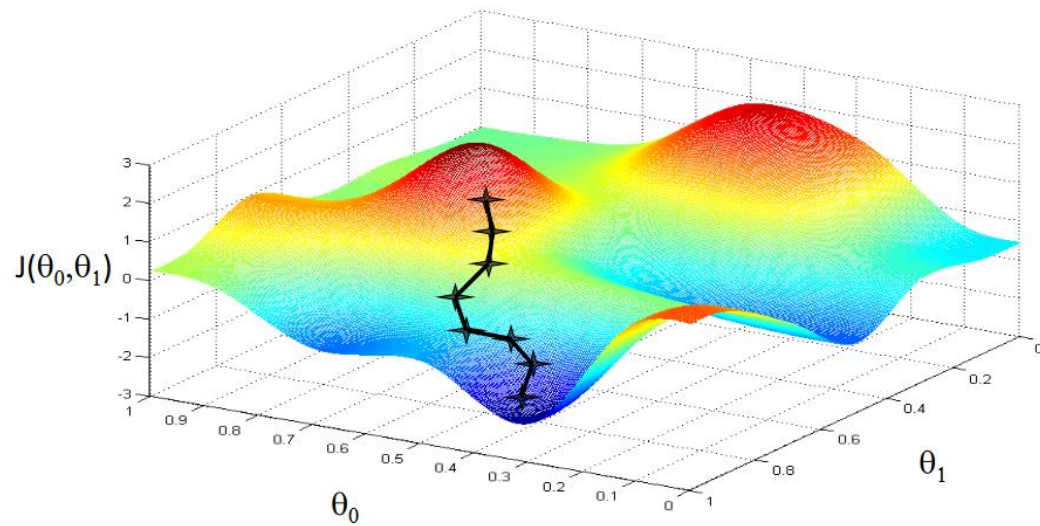
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)$$

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

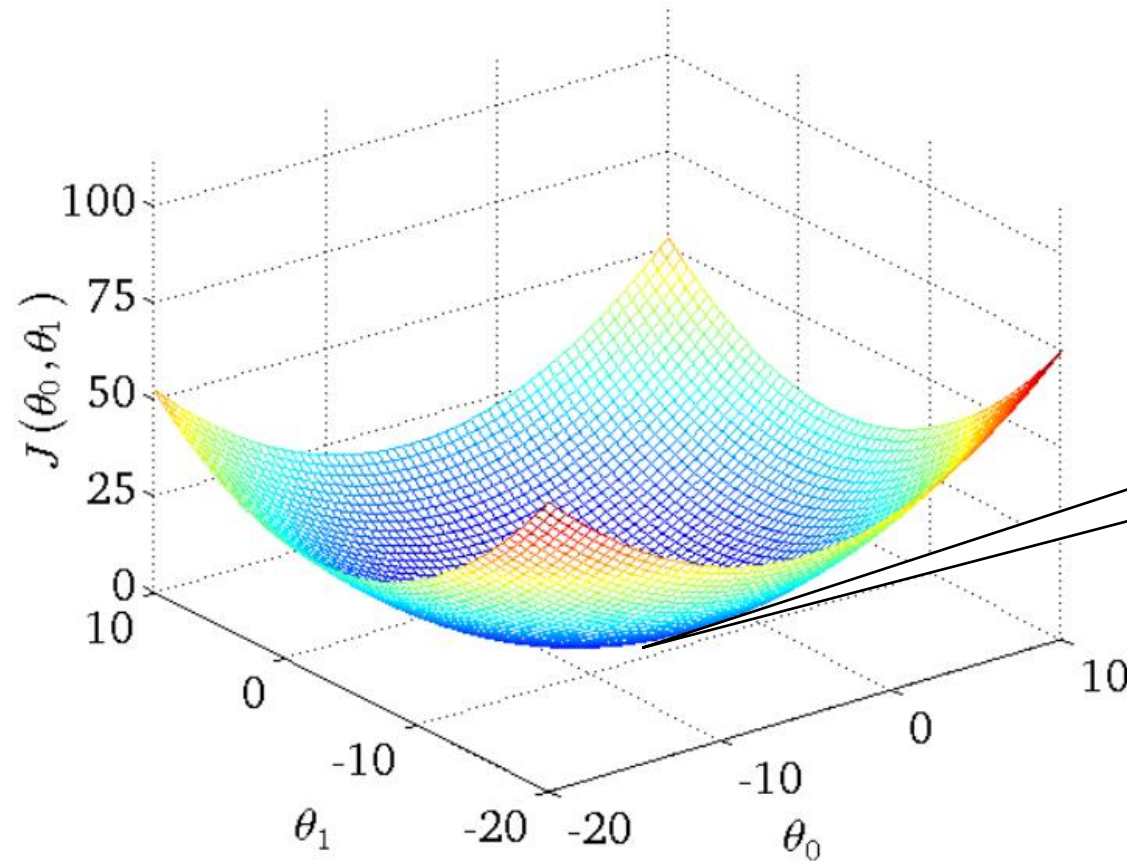
}

Update θ_0 and θ_1 simultaneously

GRADIENT DESCENT



COST FUNCTION LINEAR REGRESSION

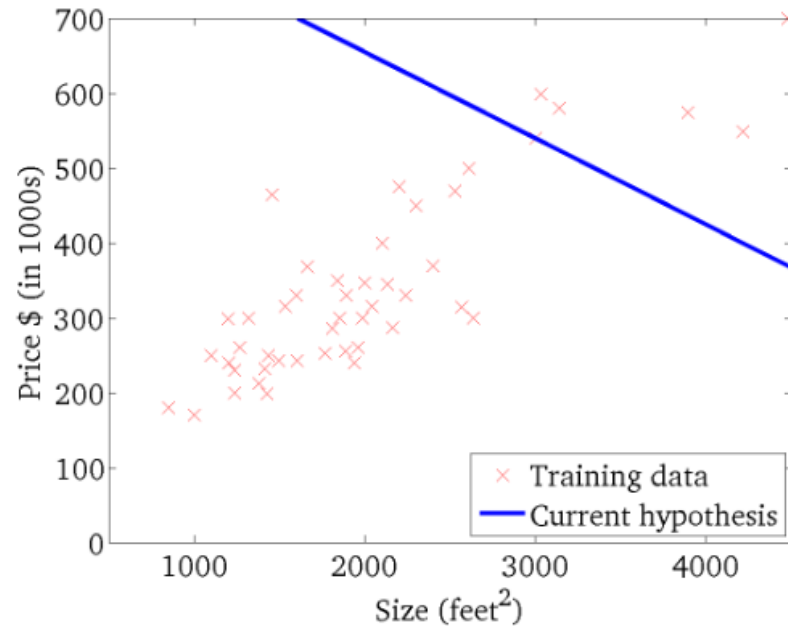


Convex function
Local optima is the global optima
Gradient Descent converges to the
global optima

GRADIENT DESCENT

$$h_{\theta}(x)$$

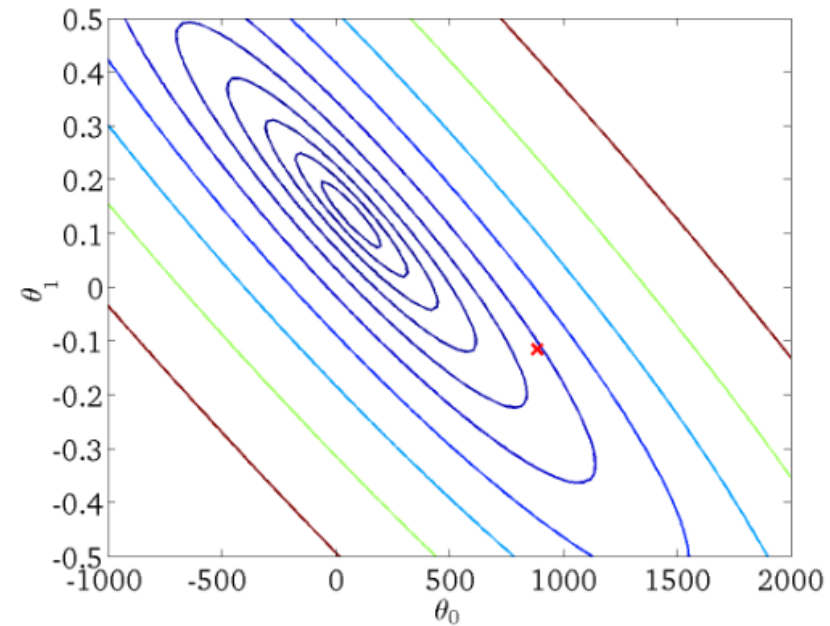
(for fixed θ_0, θ_1 , this is a function of x)



$$h_{\theta}(x) = 900 - 0.1x$$

$$J(\theta_0, \theta_1)$$

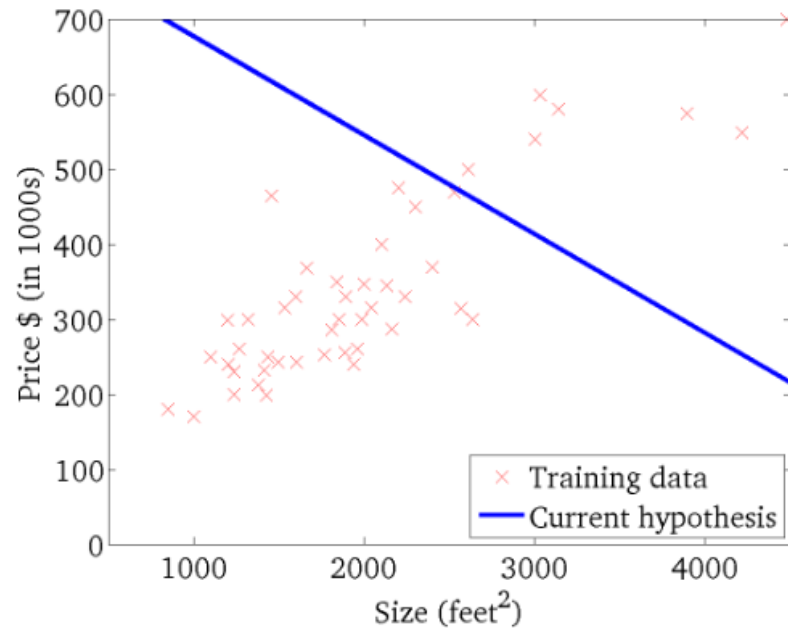
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

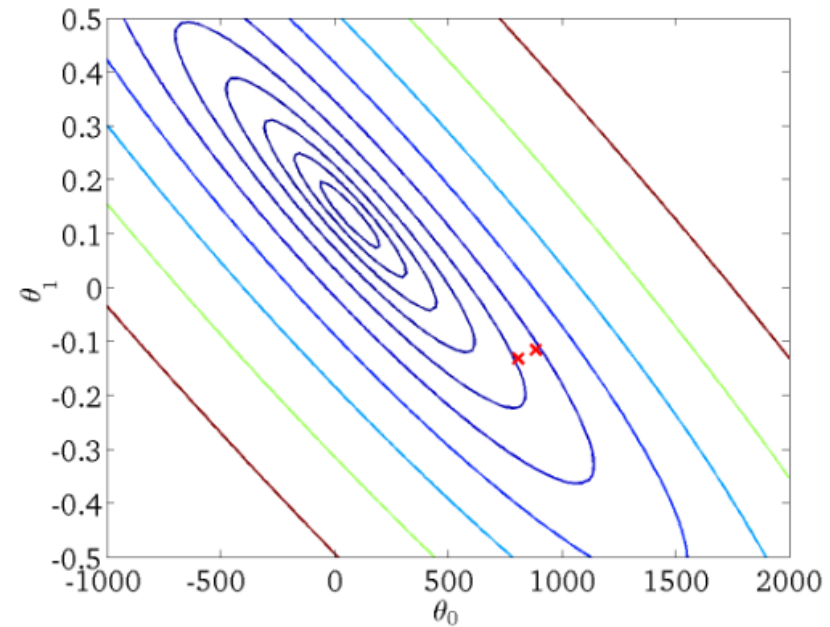
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

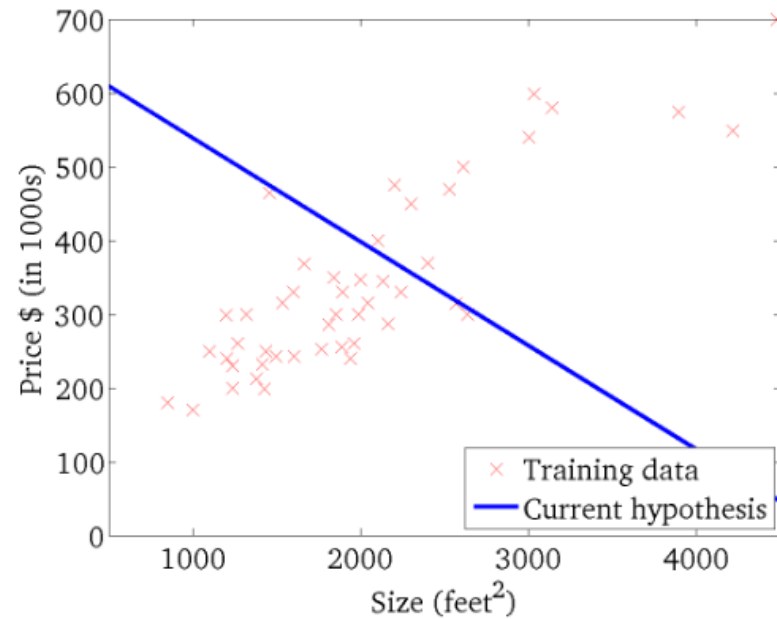
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

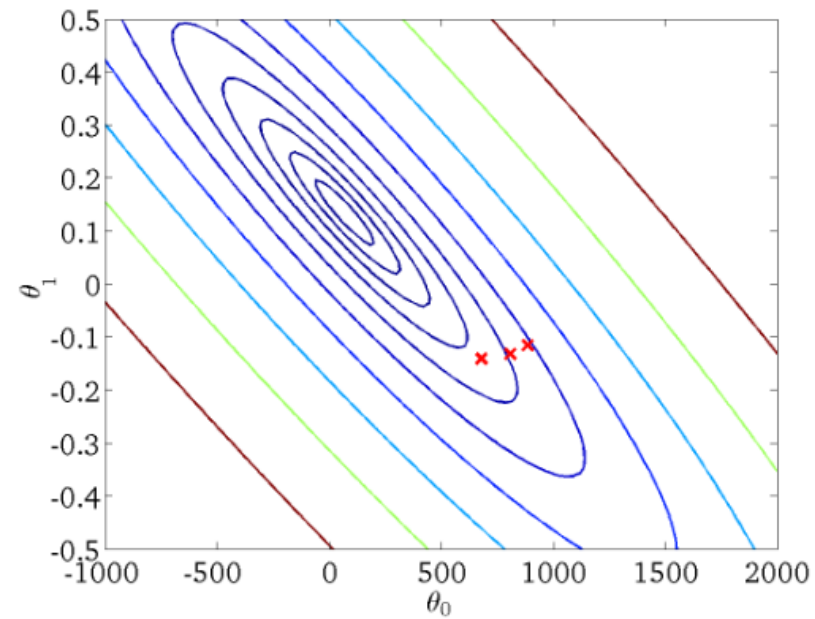
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

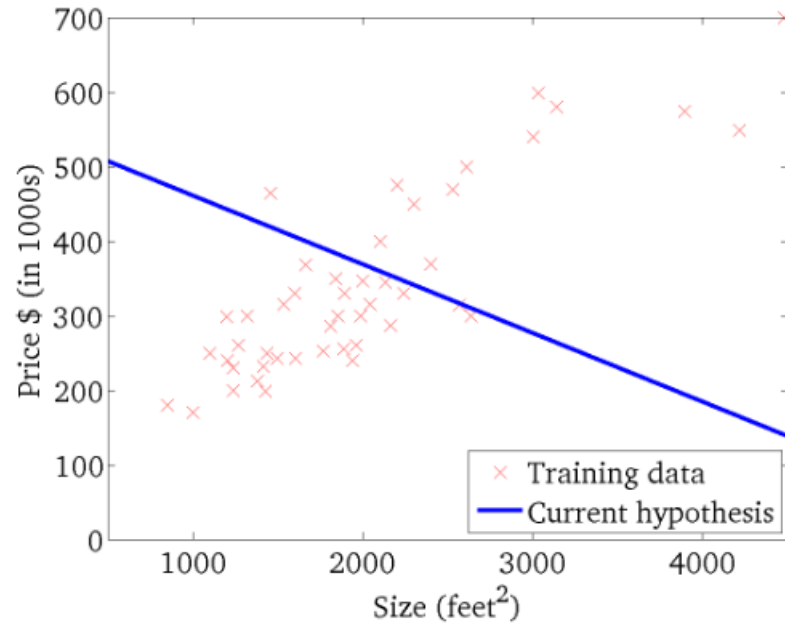
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

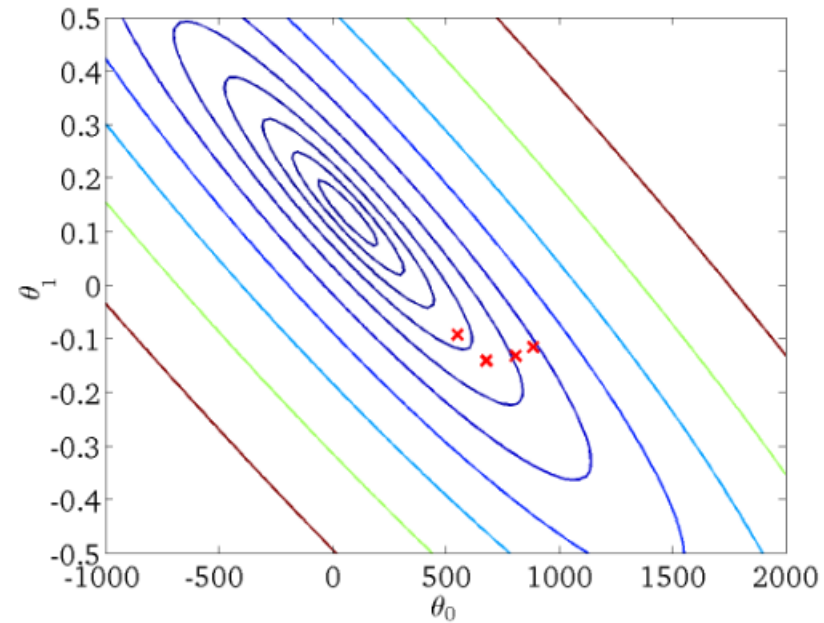
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

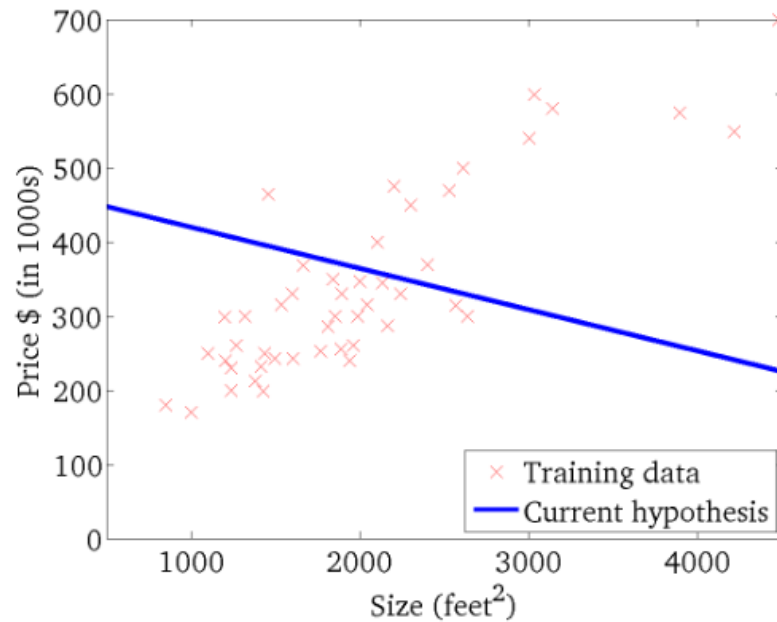
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

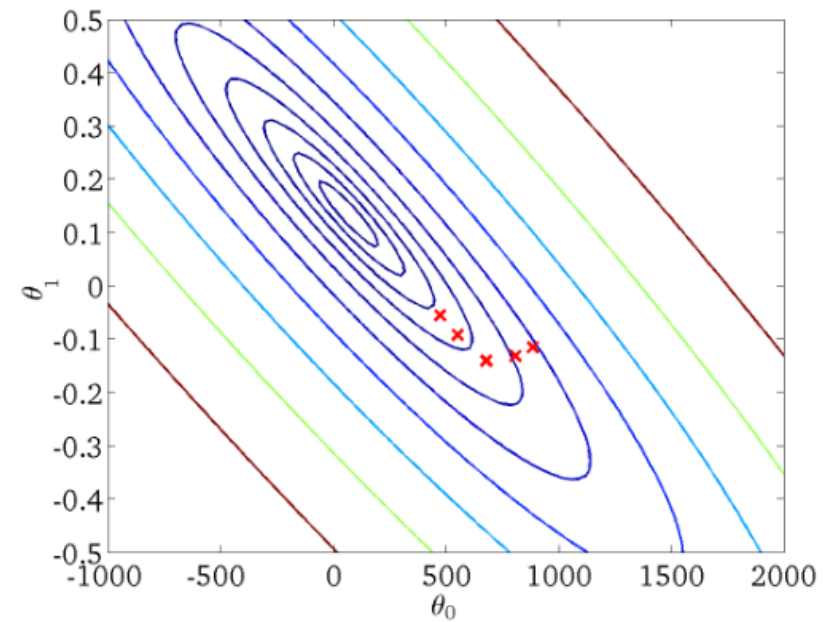
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

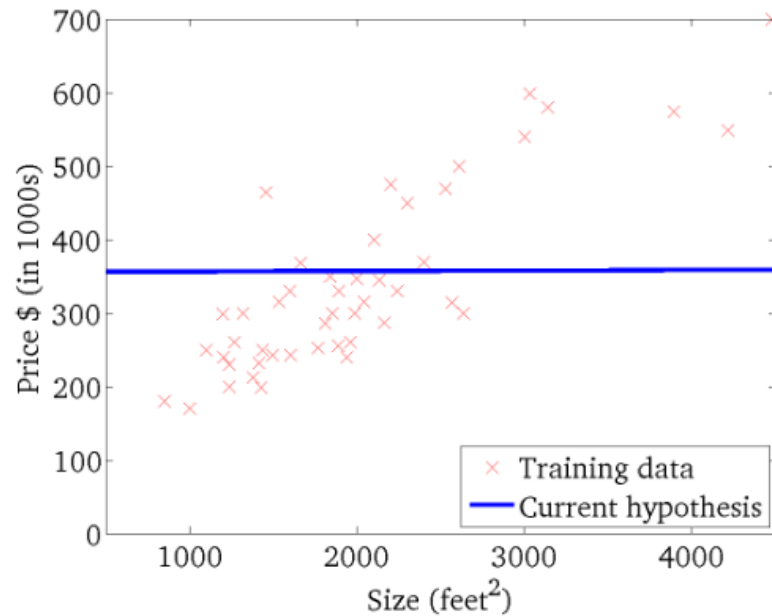
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

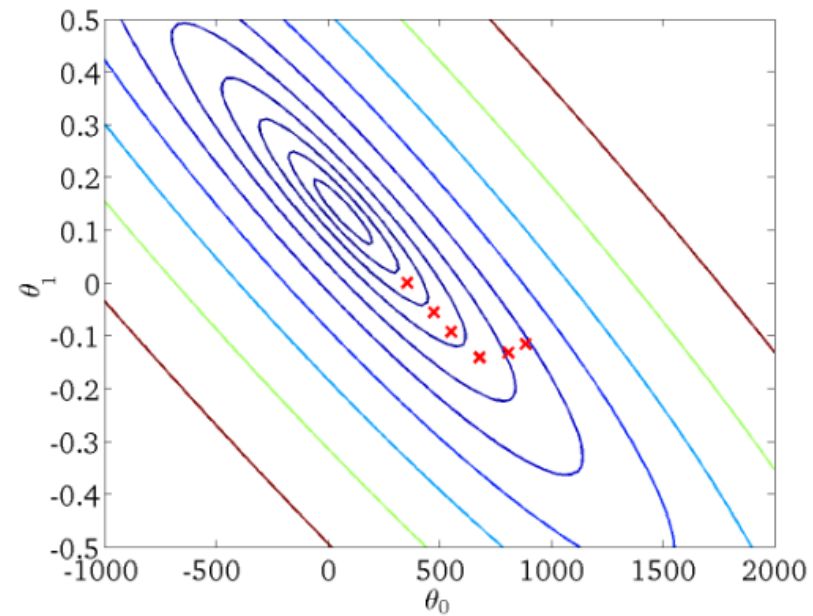
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

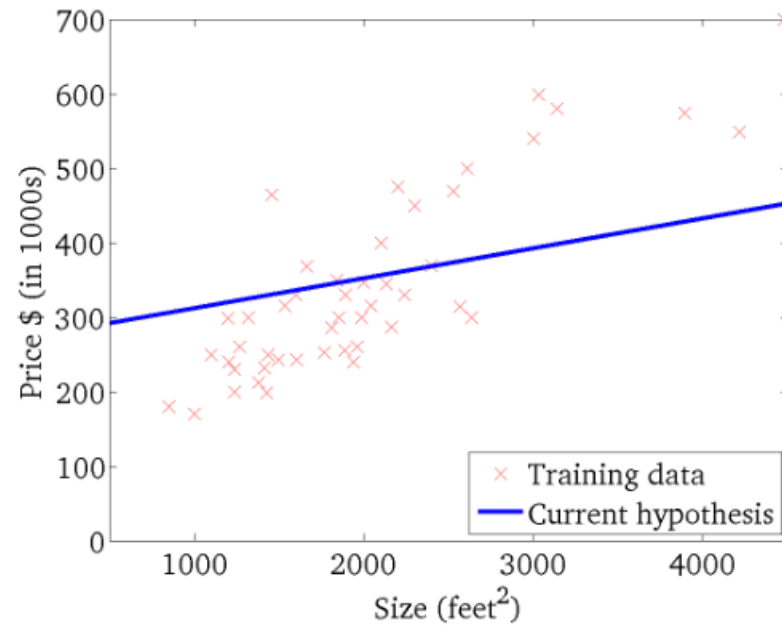
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

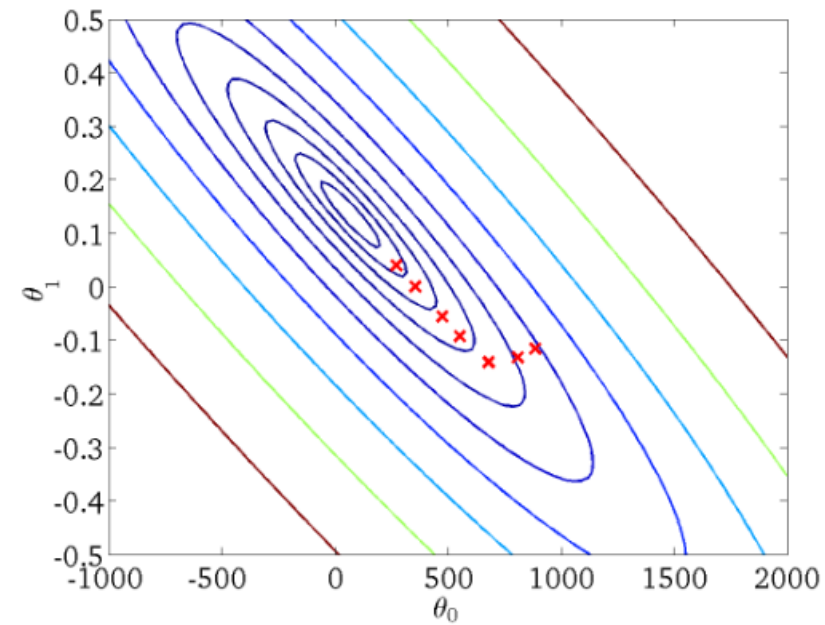
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

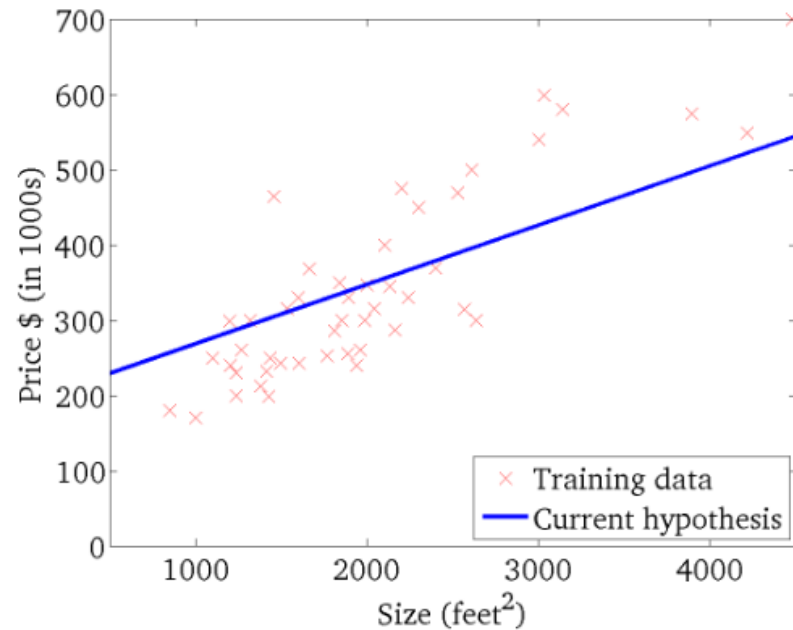
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

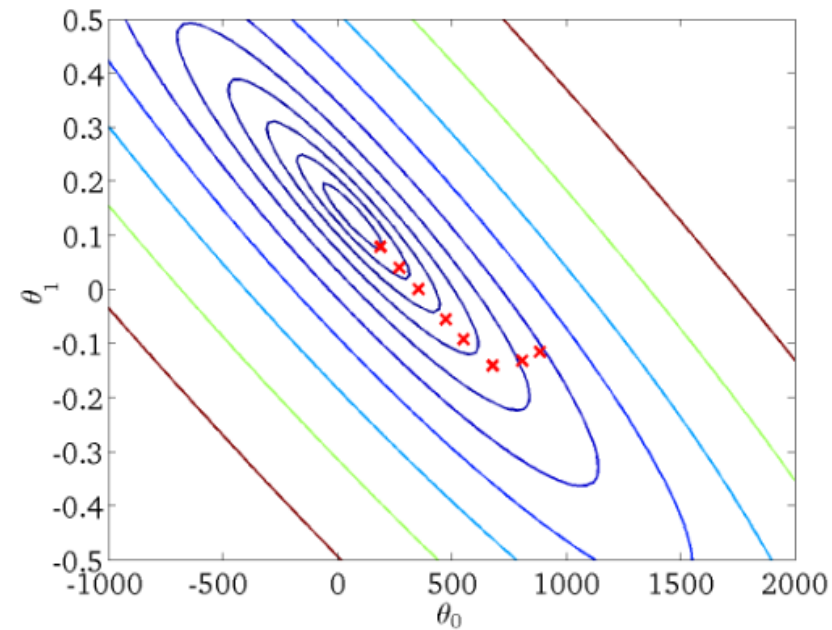
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

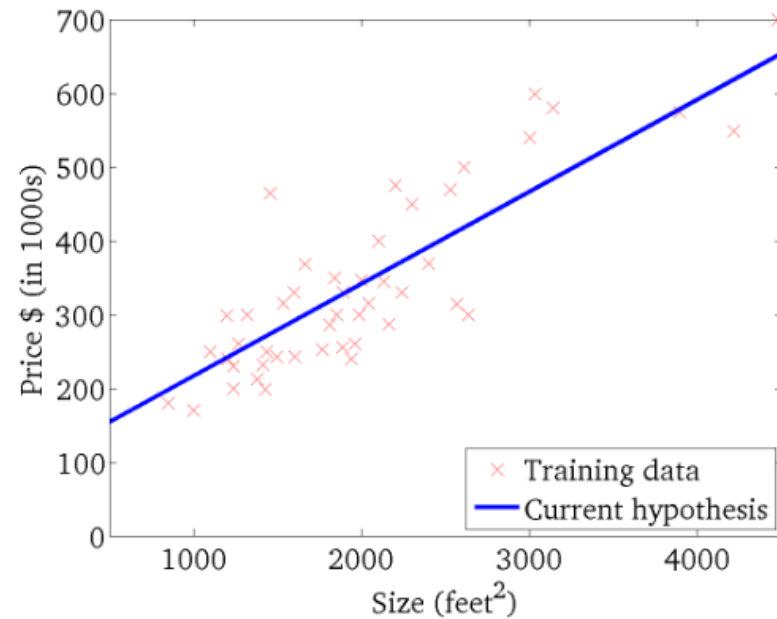
(function of the parameters θ_0, θ_1)



GRADIENT DESCENT

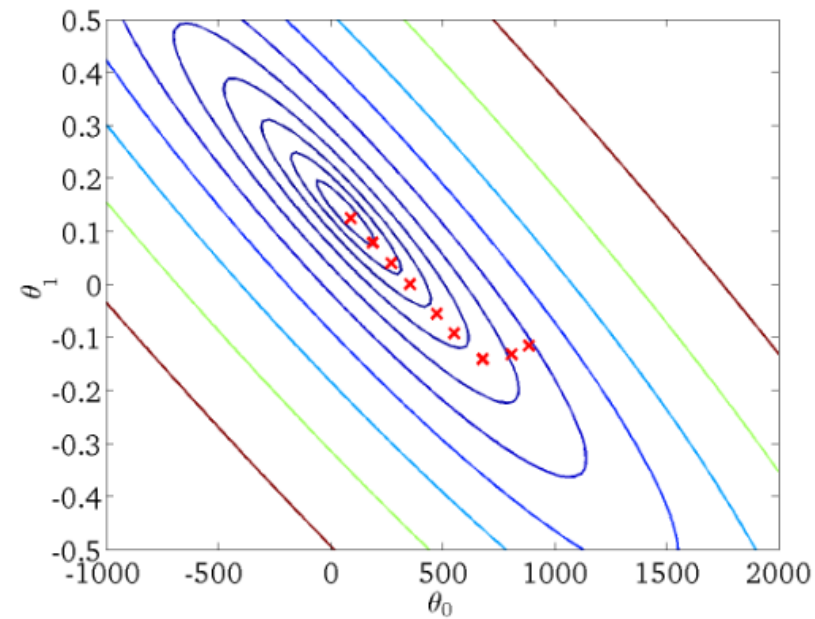
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



BATCH GRADIENT DESCENT

Batch Gradient Descent uses all of the training examples

Repeat until convergence {

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

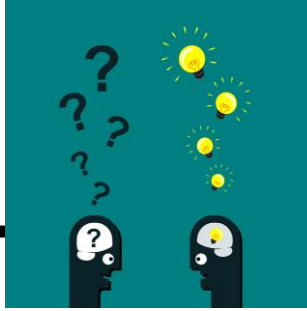
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) * x^i$$

}

m = all training examples

→ There are also versions looking just at a subset

QUESTION



Which of the following are true statements? (More than one can apply)

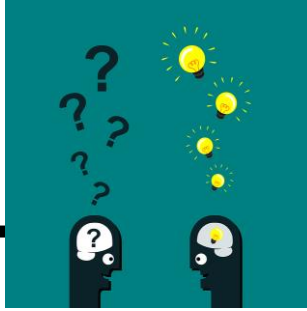
A: To make gradient descent converge, we must slowly decrease α over time.

B: Gradient descent is guaranteed to find the global minimum for any function $J(\theta_0, \theta_1)$.

C: Gradient descent can converge even if α is kept fixed. (But α cannot be too large, or else it may fail to converge.)

D: For the specific choice of cost function $J(\theta_0, \theta_1)$ used in linear regression, there are no local optima (other than the global optimum).

QUESTION



Which of the following are true statements? (More than one can apply)

A: To make gradient descent converge, we must slowly decrease α over time.

B: Gradient descent is guaranteed to find the global minimum for any function $J(\theta_0, \theta_1)$.

C: Gradient descent can converge even if α is kept fixed. (But α cannot be too large, or else it may fail to converge.)

D: For the specific choice of cost function $J(\theta_0, \theta_1)$ used in linear regression, there are no local optima (other than the global optimum).

WRAP-UP

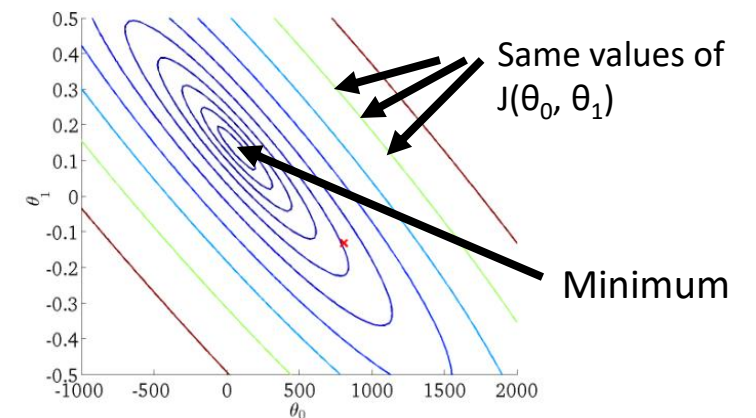
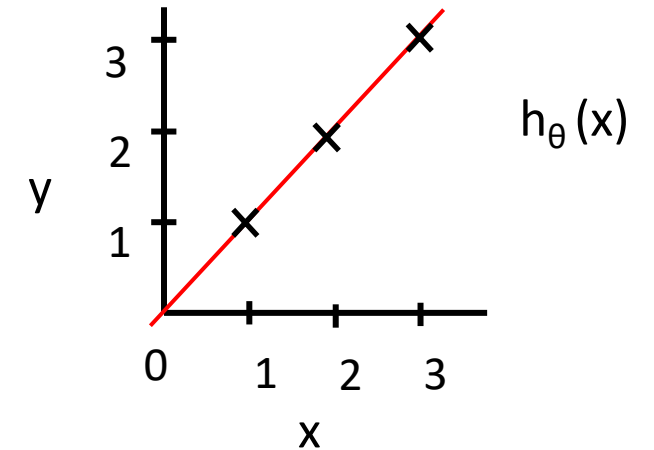
Cost Function

- We can measure the accuracy of our hypothesis function by using a cost function. This takes an average difference of all the results of the hypothesis with inputs from x 's and the actual output y 's.
- $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$
- It is $\frac{1}{2}$ of the mean of the squares of $h_{\theta}(x^i) - y^i$ or the difference between the predicted value and the actual value. The mean is halved as a convenience for computation.
- This function is otherwise called the "Squared error function", or "Mean squared error".

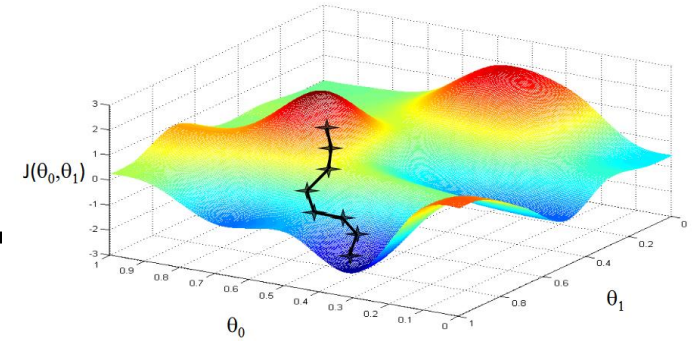
WRAP-UP

Cost Function

- If we try to think of it in visual terms, our training data set is scattered on the x-y plane. We are trying to make a straight line (defined by $h_{\theta}(x)$) which passes through these scattered data points. Our objective is to get the best possible line. The best possible line will be such so that the average squared vertical distances of the scattered points from the line will be the least. Ideally, the line should pass through all the points of our training data set. In such a case, the value of $J(\theta_0, \theta_1)$ will be 0. Thus as a goal, we should try to minimize the cost function.
- A contour plot is a graph that contains many contour lines. A contour line of a two variable function has a constant value at all points of the same line. Taking any colour and going along the 'circle', one would expect to get the same value of the cost function. If the contour plot gets closer to the centre the error of the cost function is reduced.



WRAP-UP



Gradient Descent

- We have our hypothesis function and we have a way of measuring how well it fits into the data. Now we need to estimate the parameters in the hypothesis function. That's where gradient descent comes in.
- Imagine that we graph our hypothesis function based on its fields θ_0 and θ_1 (actually we are graphing the cost function as a function of the parameter estimates). We are not graphing x and y itself, but the parameter range of our hypothesis function and the cost resulting from selecting a particular set of parameters.
- We put θ_0 on the x axis and θ_1 on the y axis, with the cost function on the vertical z axis. The points on our graph will be the result of the cost function using our hypothesis with those specific theta parameters.
- We will know that we have succeeded when our cost function is at the very bottom of the pits in our graph, i.e. when its value is the minimum.
- The way we do this is by taking the derivative (the tangential line to a function) of our cost function. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α , which is called the learning rate. Depending on where one starts on the graph, one could end up at different points.

WRAP-UP

Gradient Descent

The gradient descent algorithm is:

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \text{ (for } j=0 \text{ and } j=1)$$

}

Where $j=0,1$ represents the feature index number.

At each iteration j , one should simultaneously update the parameters $\theta_1, \theta_2, \dots, \theta_n$. Updating a specific parameter prior to calculating another one on the $j^{(\text{th})}$ iteration would yield to a wrong implementation.

WRAP-UP

Gradient Descent

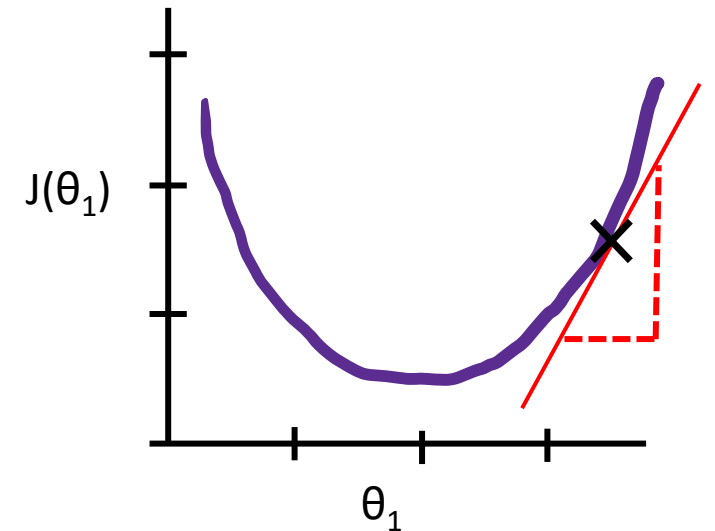
If we only use one parameter θ_1 the formula for a single parameter is:

Repeat until convergence:

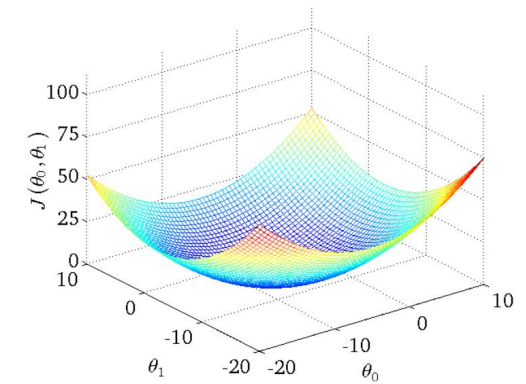
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_j} J(\theta_1)$$

θ_1 converges to its minimum value. When the slope is negative, the value of θ_1 increases and when it is positive, the value of θ_1 decreases.

We should adjust our parameter α to ensure that the gradient descent algorithm converges in a reasonable time. Failure to converge or too much time to obtain the minimum value imply that our step size is wrong. The gradient descent converges with a fixed step size α . At the minimum, the derivative will always be 0 and thus we get: $\theta_1 := \theta_1 - \alpha * 0$



WRAP-UP



Gradient Descent

When specifically applied to the case of linear regression, a new form of the gradient descent equation can be derived. We can substitute our actual cost function and our actual hypothesis function and modify the equation to:

Repeat until convergence {

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

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

}

The point of all this is that if we start with a guess for our hypothesis and then repeatedly apply these gradient descent equations, our hypothesis will become more and more accurate. This method looks at every example in the entire training set on every step, and is called batch gradient descent.

Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear regression has only one global, and no other local, optima; thus gradient descent always converges (assuming the learning rate α is not too large) to the global minimum. J is a convex quadratic function.

QUIZ – QUESTION 1

Consider the problem of predicting how well a student does in her second year of college/university, given how well she did in her first year. Specifically, let x be equal to the number of "A" grades that a student receives in their first year of college. We would like to predict the value of y , which we define as the number of "A" grades they get in their second year .

Refer to the following training set of a small sample of different students' performances:

x	y
3	4
2	1
4	3
0	1

For this training set, what is the value of m ?

QUIZ – QUESTION 1

Consider the problem of predicting how well a student does in her second year of college/university, given how well she did in her first year. Specifically, let x be equal to the number of "A" grades that a student receives in their first year of college. We would like to predict the value of y , which we define as the number of "A" grades they get in their second year .

Refer to the following training set of a small sample of different students' performances:

	x	y
1	3	4
2	2	1
3	4	3
4	0	1

For this training set, what is the value of m ? → 4

QUIZ – QUESTION 2

Consider the following training set of training examples:

Consider the linear regression model $h_{\theta}(x) = \theta_0 + \theta_1 x$

What are the values of θ_0 θ_1 that you would expect to obtain upon running gradient descent on this model? (Linear regression will be able to fit this data perfectly.)

- ☐ $\theta_0 = 0, \theta_1 = 0.5$
- ☐ $\theta_0 = 0.5, \theta_1 = 0$
- ☐ $\theta_0 = 1, \theta_1 = 0.5$
- ☐ $\theta_0 = 0.5, \theta_1 = 1$
- ☐ $\theta_0 = 0.5, \theta_1 = 0.5$

x	y
1	0.5
2	1
4	2
0	0

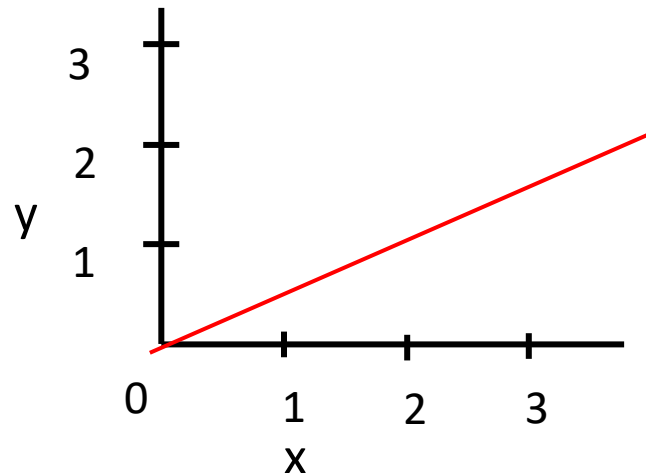
QUIZ – QUESTION 2

Consider the following training set of training examples:

Consider the linear regression model $h_{\theta}(x) = \theta_0 + \theta_1 x$

What are the values of θ_0 θ_1 that you would expect to obtain upon running gradient descent on this model? (Linear regression will be able to fit this data perfectly.)

- ☒ $\theta_0 = 0, \theta_1 = 0.5$
- ☐ $\theta_0 = 0.5, \theta_1 = 0$
- ☐ $\theta_0 = 1, \theta_1 = 0.5$
- ☐ $\theta_0 = 0.5, \theta_1 = 1$
- ☐ $\theta_0 = 0.5, \theta_1 = 0.5$



x	y
1	0.5
2	1
4	2
0	0

QUIZ – QUESTION 3

Suppose we set $\theta_0 = -1$, $\theta_1 = 0.5$ What is $h_{\theta}(4)$?

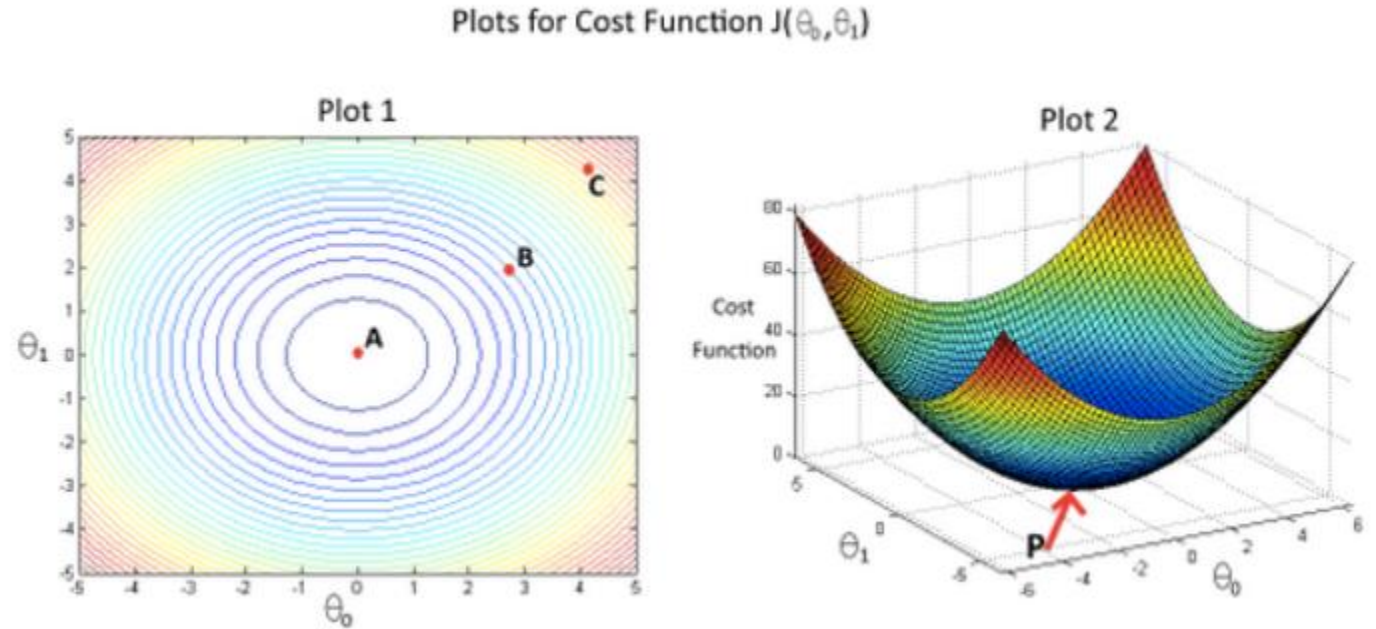
QUIZ – QUESTION 3

Suppose we set $\theta_0 = -1$, $\theta_1 = 0.5$ What is $h_\theta(4)$?

$$\rightarrow 1 \rightarrow -1 + 0.5 * 4$$

QUIZ – QUESTION 4

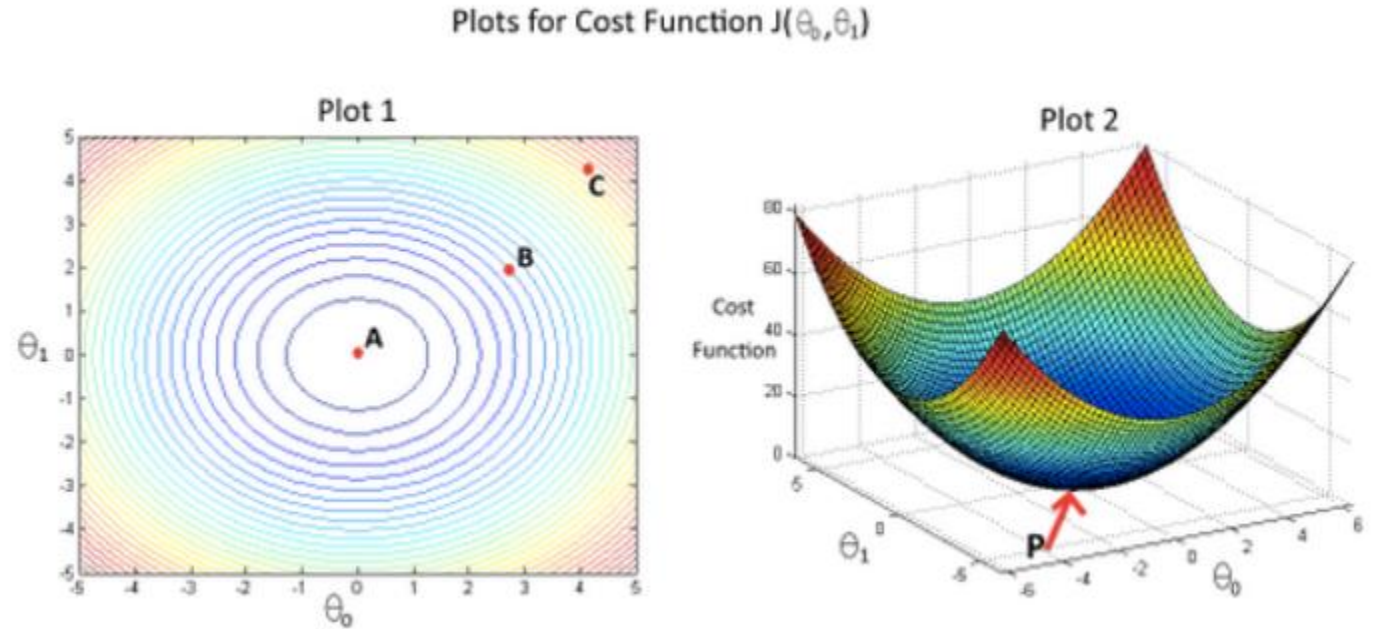
In the given figure, the cost function $J(\theta_0, \theta_1)$ has been plotted against θ_0 and θ_1 , as shown in 'Plot 2'. The contour plot for the same cost function is given in 'Plot 1'. Based on the figure, choose the correct options (check all that apply).



- If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point C, as the value of cost function $J(\theta_0, \theta_1)$ is minimum at point C.
- If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point A, as the value of cost function $J(\theta_0, \theta_1)$ is maximum at point A.
- If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point A, as the value of cost function $J(\theta_0, \theta_1)$ is minimum at A.
- Point P (the global minimum of plot 2) corresponds to point A of Plot 1.
- Point P (The global minimum of plot 2) corresponds to point C of Plot 1.

QUIZ – QUESTION 4

In the given figure, the cost function $J(\theta_0, \theta_1)$ has been plotted against θ_0 and θ_1 , as shown in 'Plot 2'. The contour plot for the same cost function is given in 'Plot 1'. Based on the figure, choose the correct options (check all that apply).



- ☐ If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point C, as the value of cost function $J(\theta_0, \theta_1)$ is minimum at point C.
- ☐ If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point A, as the value of cost function $J(\theta_0, \theta_1)$ is maximum at point A.
- ☒ If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point A, as the value of cost function $J(\theta_0, \theta_1)$ is minimum at A.
- ☒ Point P (the global minimum of plot 2) corresponds to point A of Plot 1.
- ☐ Point P (The global minimum of plot 2) corresponds to point C of Plot 1.

QUIZ – QUESTION 5

Suppose that for some linear regression problem, we have some training set, and for our training set we managed to find some θ_0, θ_1 such that $J(\theta_0, \theta_1) = 0$.

Which of the statements below must then be true? (Check all that apply.)

- ☐ For this to be true, we must have $\theta_0 = 0$ and $\theta_1 = 0$ so that $h_{\theta}(x) = 0$.
- ☐ For this to be true, we must have $y^{(i)} = 0$ for every value of $i = 1, 2, \dots, m$.
- ☐ Our training set can be fit perfectly by a straight line, i.e., all of our training examples lie perfectly on some straight line.
- ☐ Gradient descent is likely to get stuck at a local minimum and fail to find the global minimum.

QUIZ – QUESTION 5

Suppose that for some linear regression problem, we have some training set, and for our training set we managed to find some θ_0, θ_1 such that $J(\theta_0, \theta_1) = 0$.

Which of the statements below must then be true? (Check all that apply.)

- ☐ For this to be true, we must have $\theta_0 = 0$ and $\theta_1 = 0$ so that $h_{\theta}(x) = 0$.
- ☐ For this to be true, we must have $y^{(i)} = 0$ for every value of $i = 1, 2, \dots, m$.
- ☒ Our training set can be fit perfectly by a straight line, i.e., all of our training examples lie perfectly on some straight line.
- ☐ Gradient descent is likely to get stuck at a local minimum and fail to find the global minimum.