Machine Learning

Dr. Shahid Mahmood Awan
Assistant Professor
School of Systems and Technology, University of Management and Technology
shahid.awan@umt.edu.pk

Umer Saeed(MS Data Science, BSc Telecommunication Engineering)
Sr. RF Optimization & Planning Engineer
f2017313017@umt.edu.pk

Regression

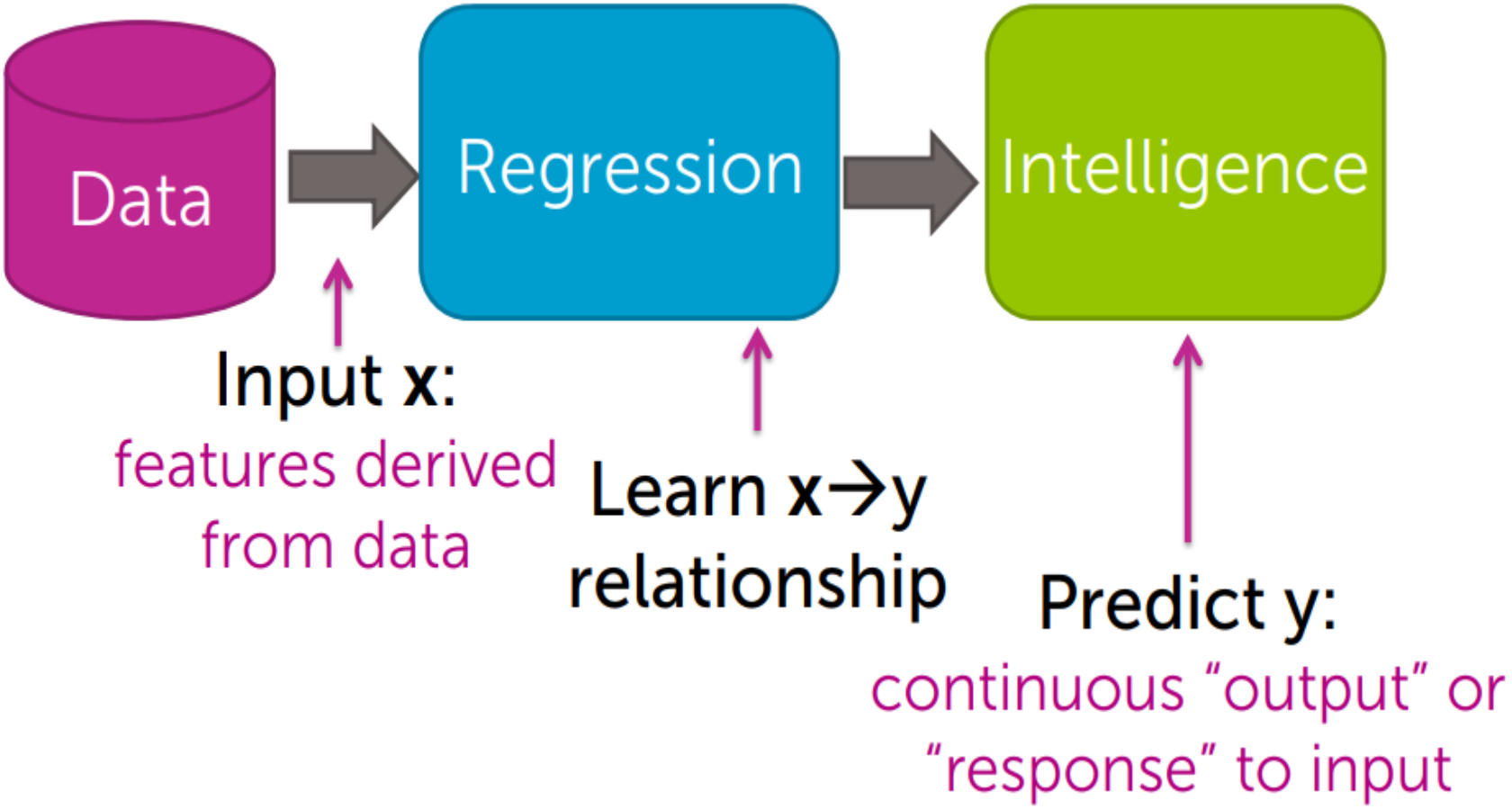
What is Regression?

Regression

- From features to predictions;

Regression

- From features to predictions;



Regression

Examples

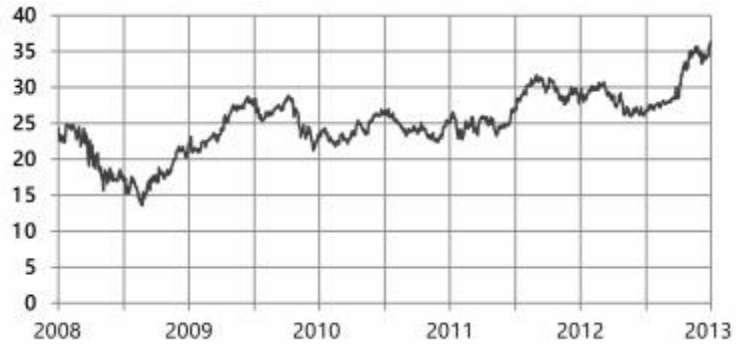
Salary after ML Subject

- How much will your salary be? (y = $$)
- Depends on x =
  - performance in courses,
  - quality of project,
  - # of forum responses, …

hard work

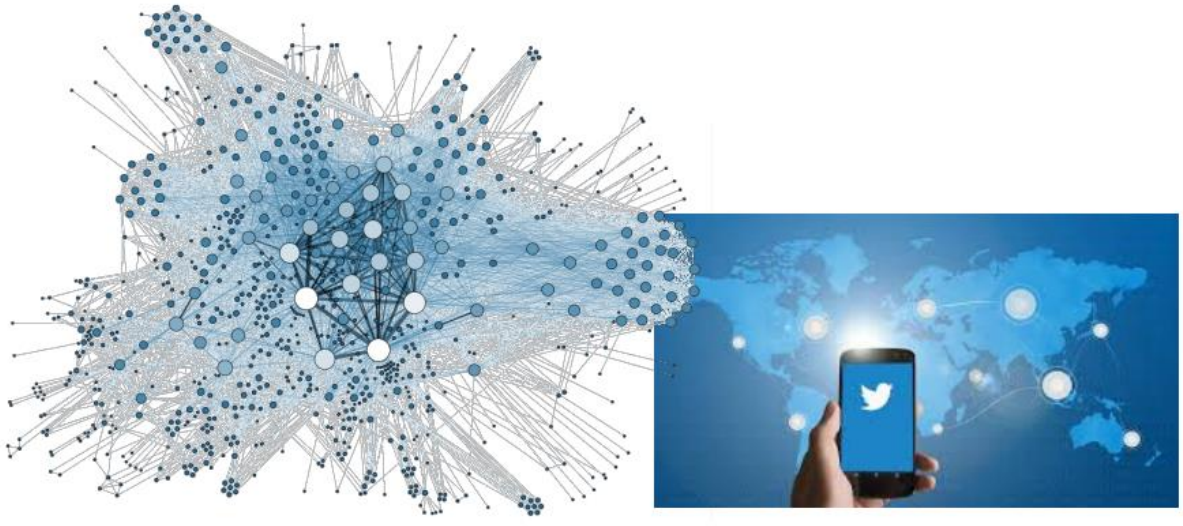
Stock Prediction

- Predict the price of a stock (y)
- Depends on x =
  - Recent history of stock price
  - News events
  - Related commodities, …

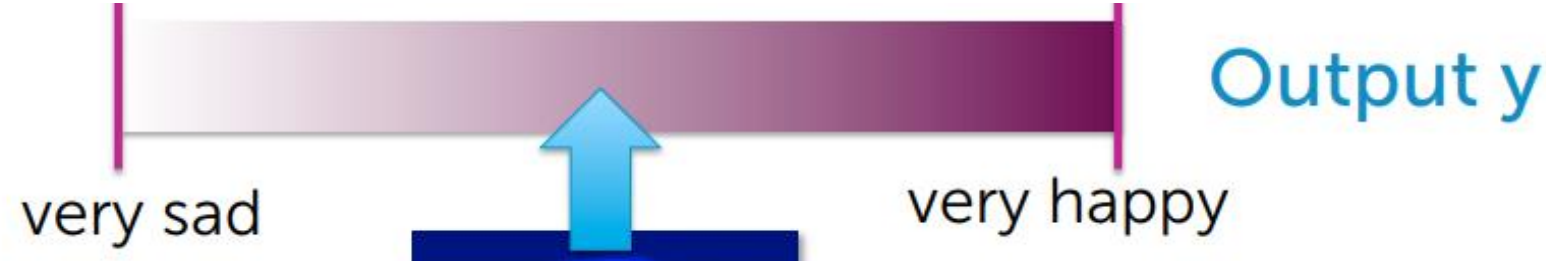


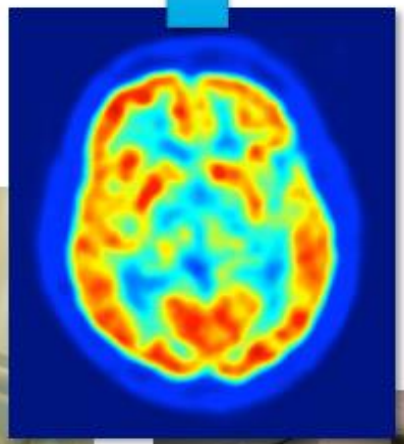
Tweet Popularity

- How many people will retweet your tweet? (y)
- Depends on x =
  - # followers
  - # of followers of followers
  - features of text tweeted
  - popularity of hashtag
  - # of past retweets, ….



Reading your mind



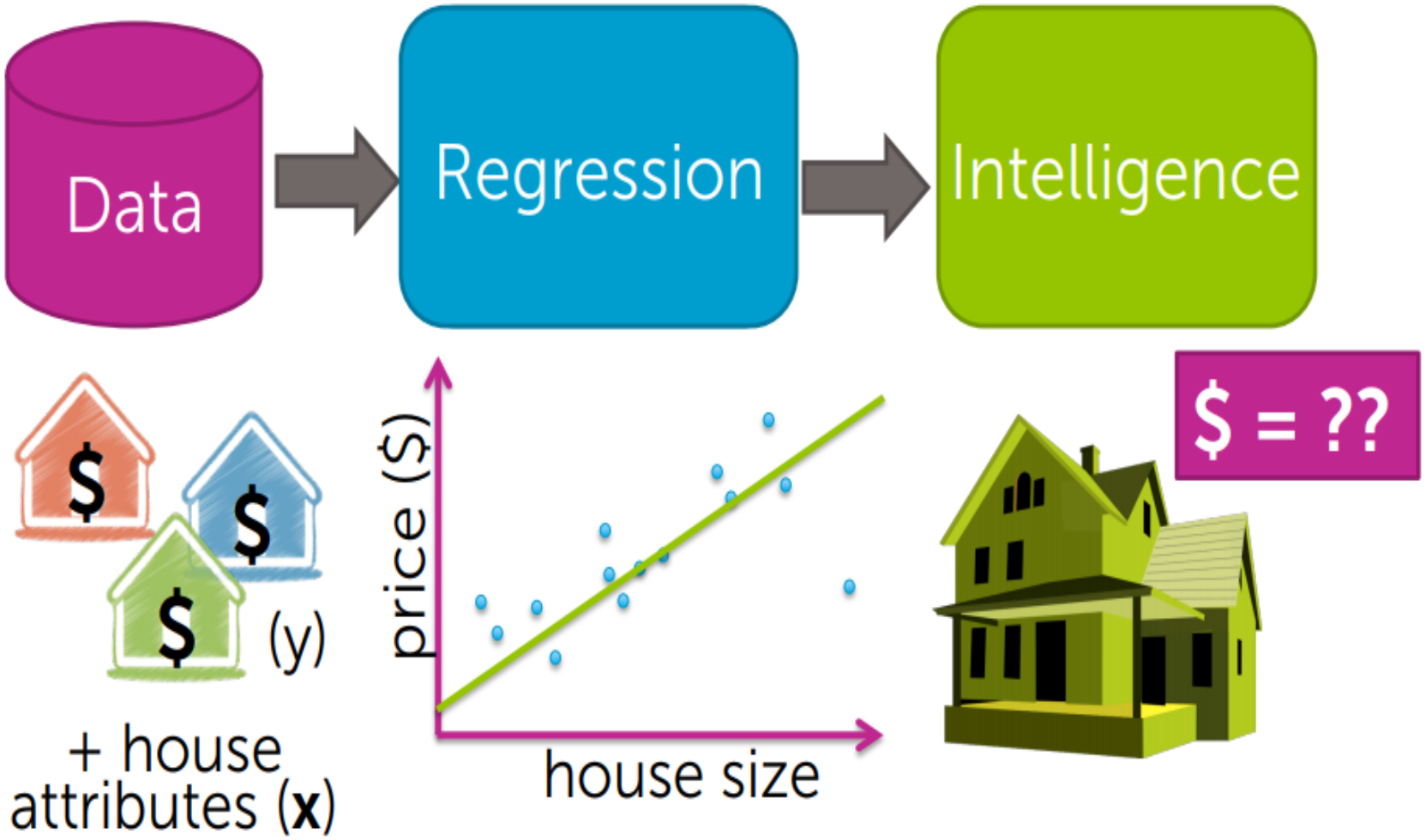


Inputs x are brain region intensities

Regression

Case Study

Predicting House Prices



Predicting House Prices

Predicting House Prices

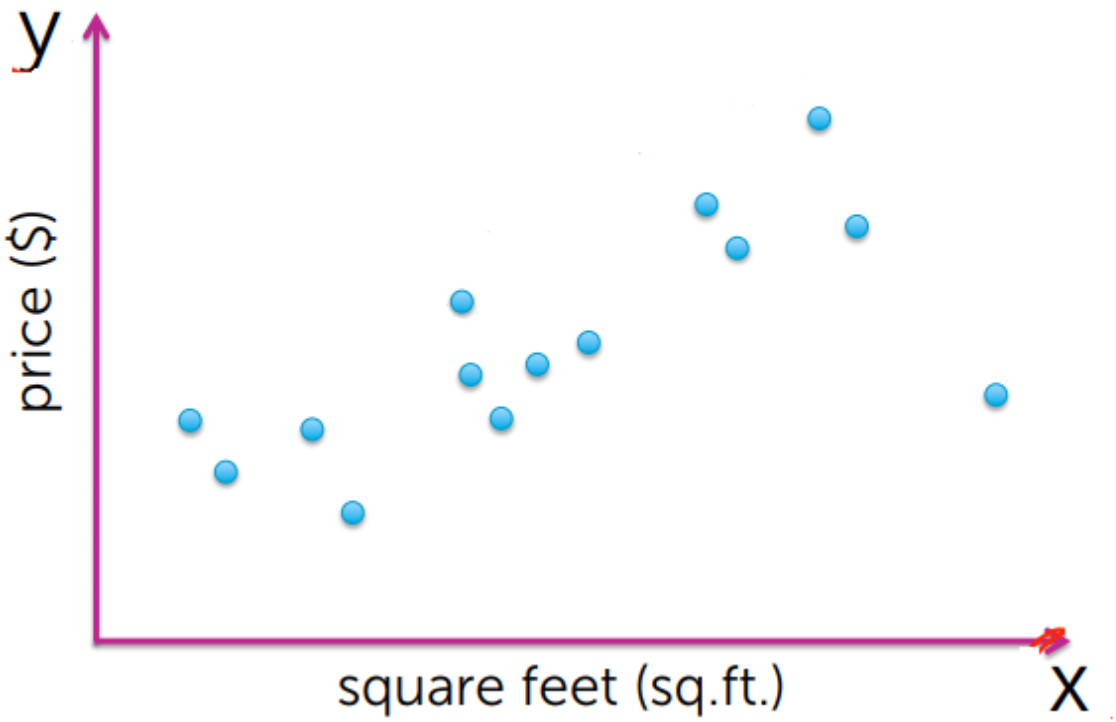
- How much is my house worth?

Predicting House Prices

- Look at recent sales in my neighborhood: How much did they sell for?

Predicting House Prices

Plot recent house sales
(Past 2 years)

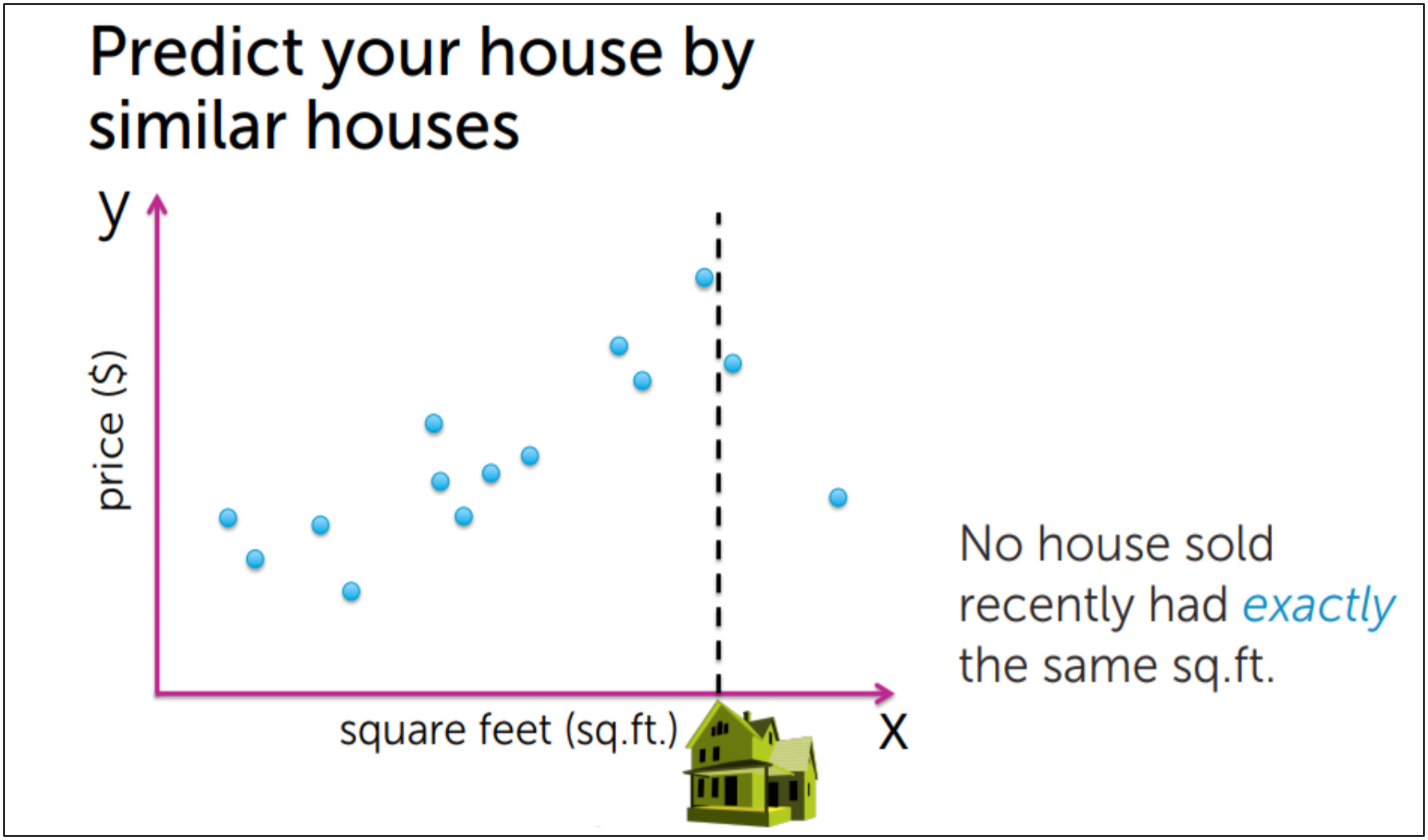


Terminology:

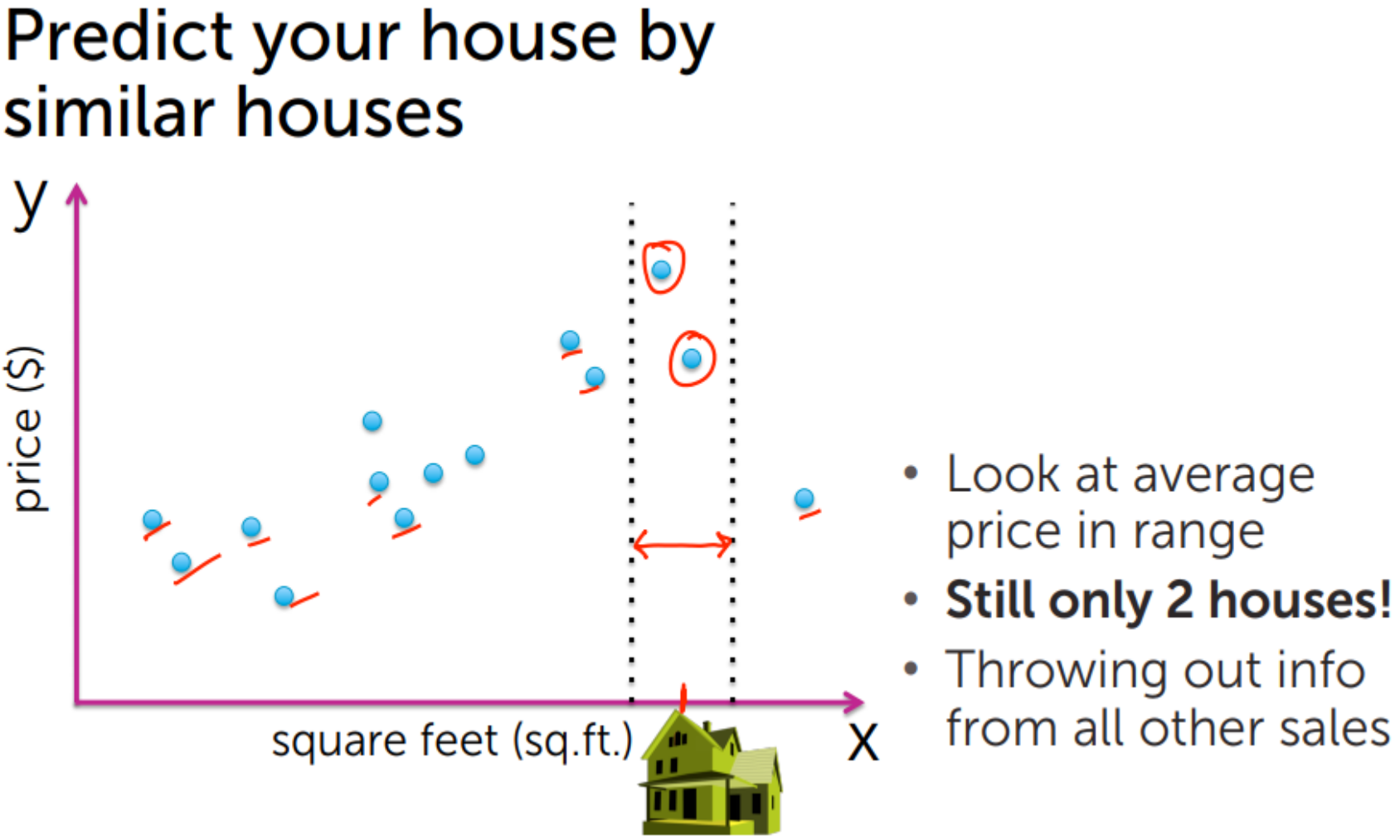
x – feature, covariate, or predictor

y – observation or response

Predicting House Prices



Predicting House Prices



Linear Regression

Uni-variate Linear Regression

Linear Regression with One Variable

- **Model Representation**

- In *regression problems*, we are taking ***input variables*** and trying **to fit the output onto a *continuous* expected result function**.

- Linear regression with one variable is also known as "**univariate linear regression**."

- Univariate linear regression is used when you want to predict a **single output** value y from a **single input** value x.

- We're doing **supervised learning** here, so that means we already have an idea about what the input/output cause and effect should be.

Predicting House Prices

Data

input *output*

$(x_1 = \text{sq.ft.}, y_1 = \$)$

$(x_2 = \text{sq.ft.}, y_2 = \$)$

$(x_3 = \text{sq.ft.}, y_3 = \$)$

$(x_4 = \text{sq.ft.}, y_4 = \$)$

$(x_5 = \text{sq.ft.}, y_5 = \$)$

⋮

Input vs. Output:

- y is the quantity of interest
- assume y can be predicted from x

Linear Regression with One Variable

- **Training Set of housing prices**

Plot Size	Price
1650	45
1700	27
1836	32.5
1905	62
1950	32
1950	49
…	…
…	…
…	

- **Supervised Learning**
- Given the “right answer” for each example in the data.

- **Regression Problem**
- Predict real-valued output

- **Notation**
- m= Number of training examples=546
- X’s= “input” variable/features/independent variable = Plot Size
- y’’s= “output” variable/”target” variable/dependent variable =Price
- (X,y)- Single training example
- $(X^{(i)},y^{(i)})$- i^{th} training example, Superscript i is not the Exponential, its the index.
- $X_j^{(i)}$= value of feature j in the i^{th} training example
- $n=|X^{(i)}|$; (the number of features)

Linear Regression with One Variable

- **Example**
- Consider the training set shown below. $(x^{(i)},y^{(i)})$ is the i^{th} training example. What is $y^{(3)}$?

Size in feet^2 (X)	Price($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	

- (a) 1416
- **(b) 1534 (Answer)**
- (c) 315
- (d) 0

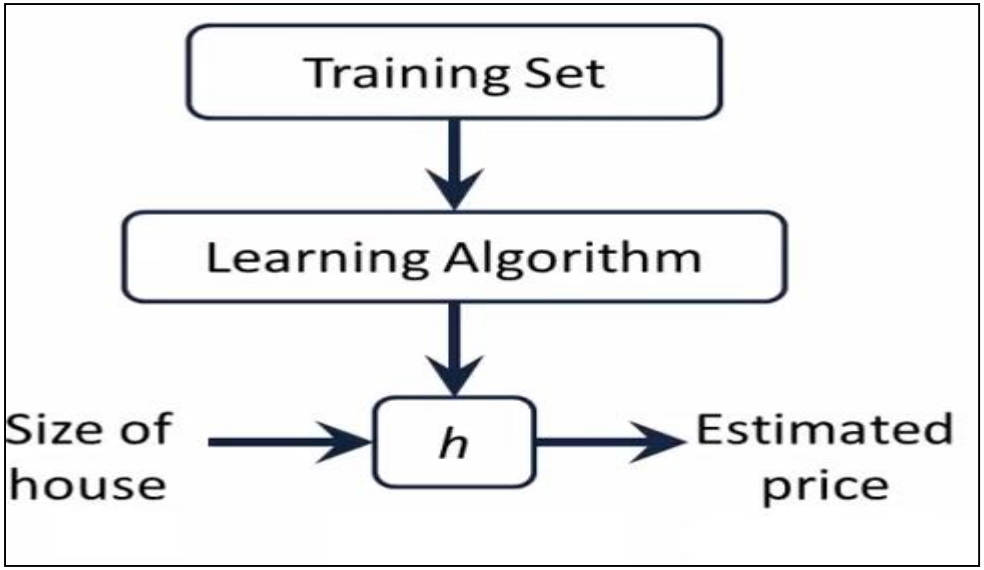
Linear Regression with one variable

The Hypothesis Function

The Hypothesis Function

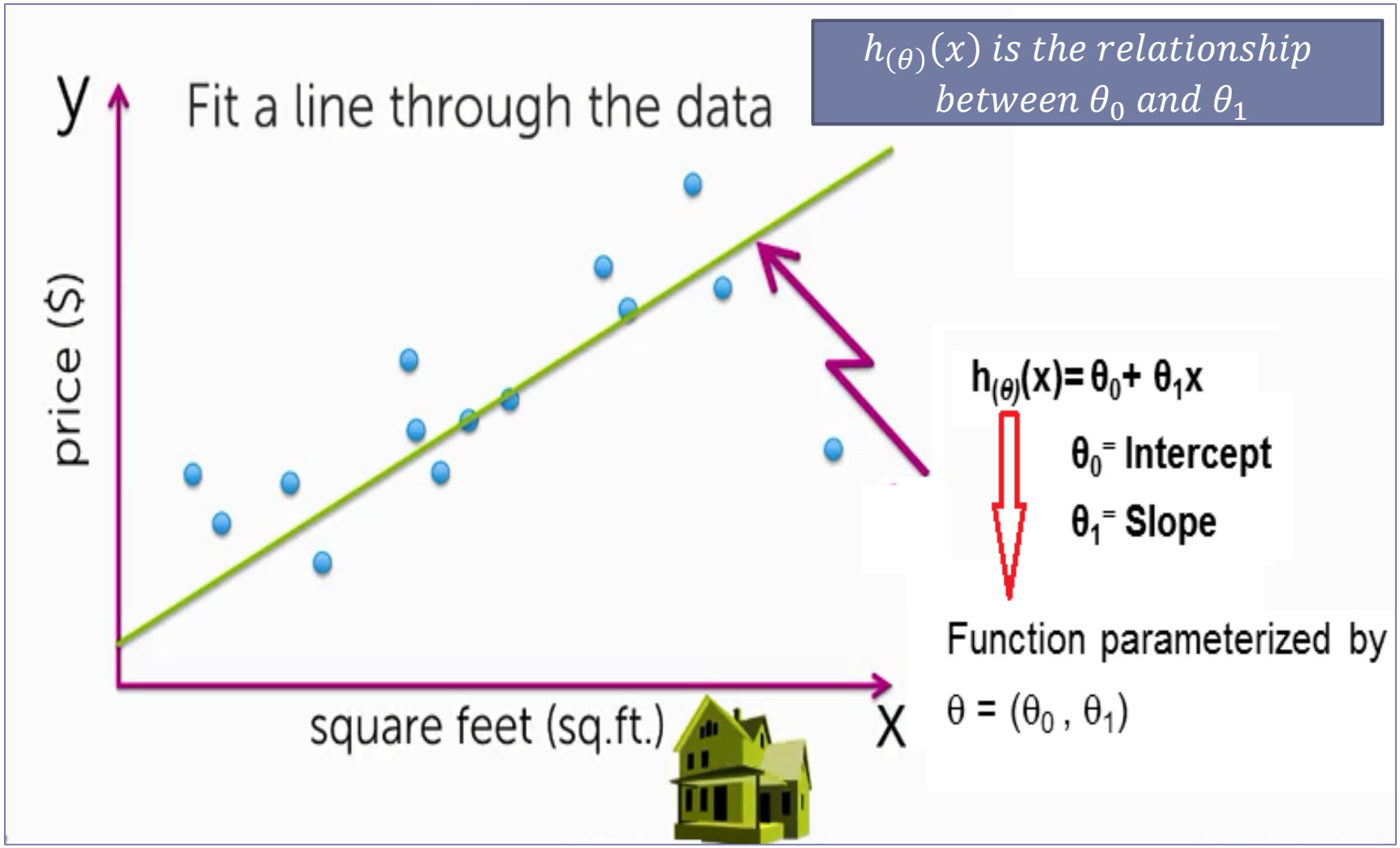
- To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function h : X → Y so that h(x) is a “good” predictor for the corresponding value of y.

- For historical reasons, this function h is called a hypothesis. Seen pictorially, the process is therefore like this:



- h maps from X’s to y’s

The Hypothesis Function



The Hypothesis Function

- Our hypothesis function has the general form: $\mathbf{h_{(\theta)}(x) = \theta_0 + \theta_1 x}$, Where $\theta_{i's}$=Parameters of model

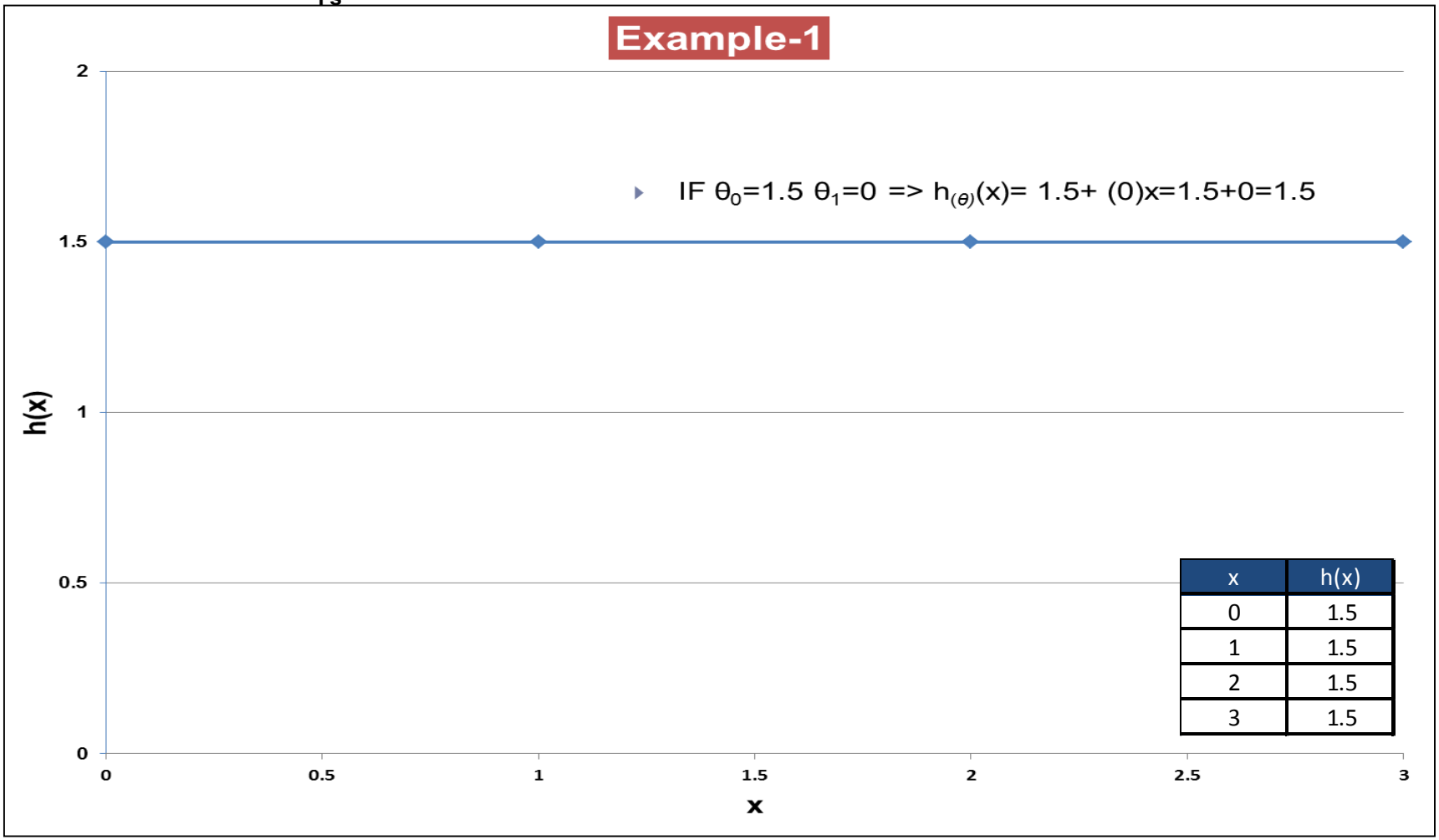
- Note that this is like the equation of a straight line.

- We give to $h_{(\theta)}(x)$ values for θ_0 and θ_1 to get our estimated output $h_{(\theta)}(x)$. In other words, we are trying to create a function called $h_{(\theta)}$that is trying to map our input data (the x's) to our output data (the y's).

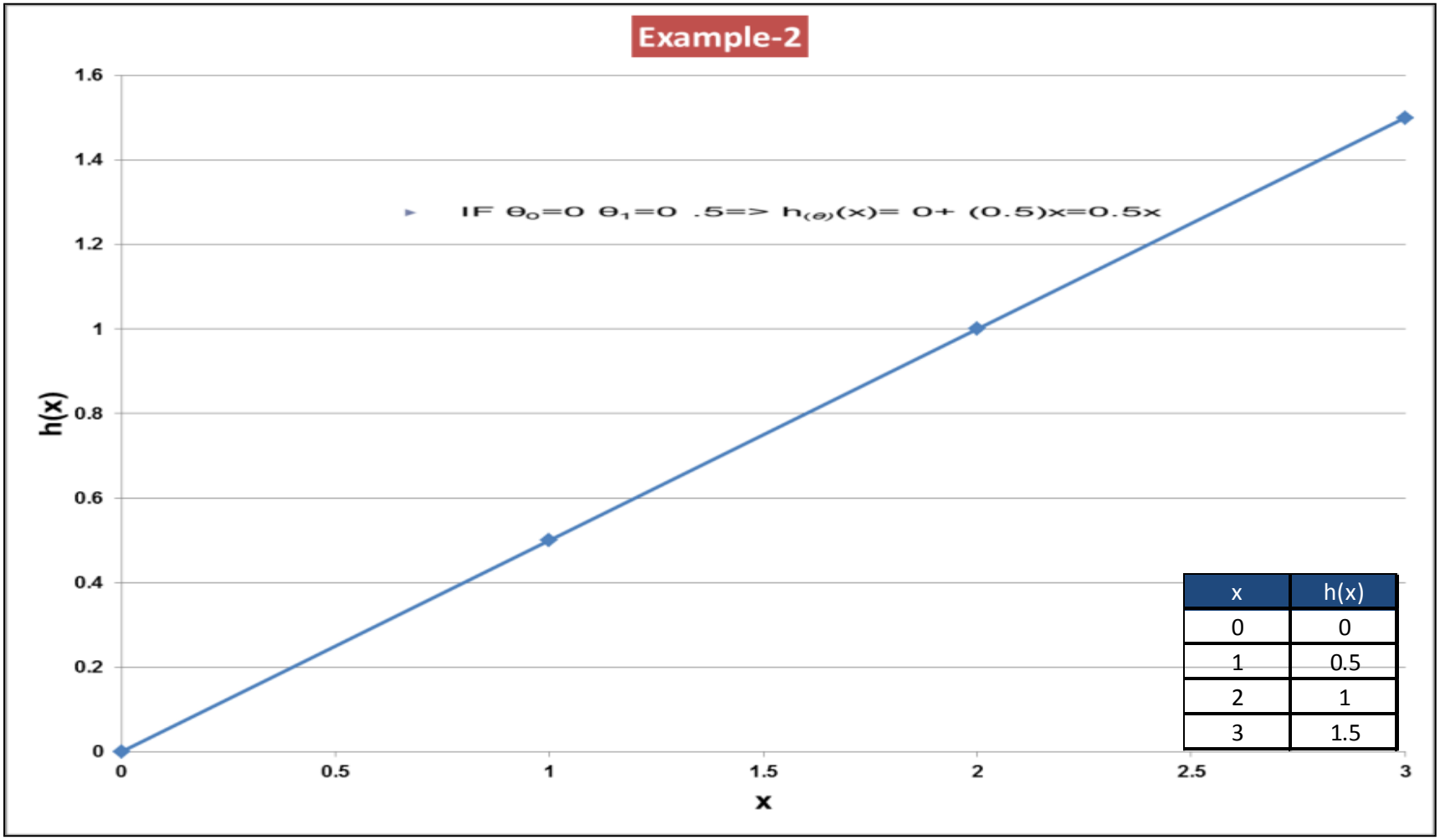
- θ_0= Intercept & θ_1= Slope or Regression Coefficient or weight on Feature X

The Hypothesis Function

- How to choose $\theta_{j's}$?

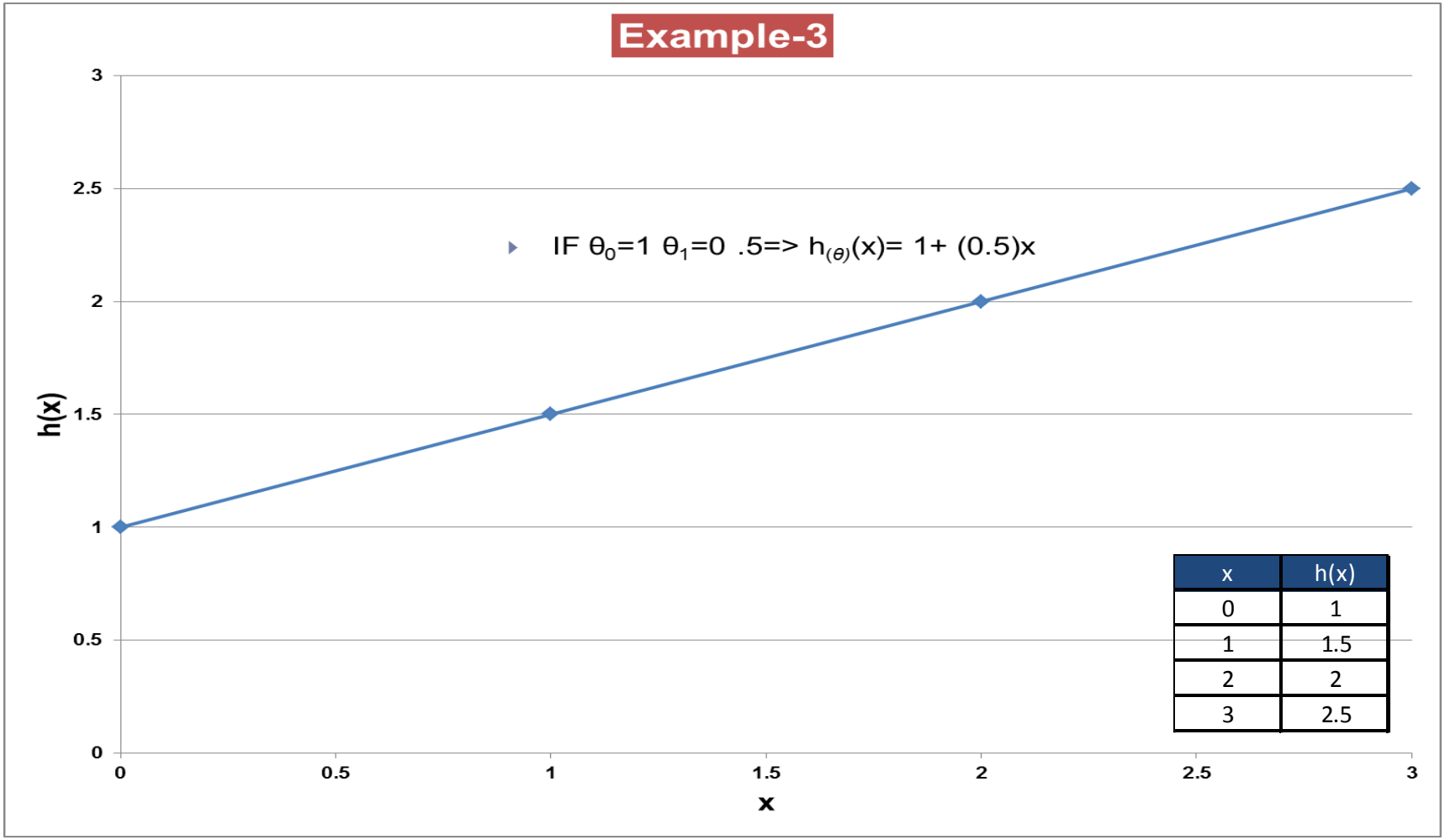


The Hypothesis Function



x	h(x)
0	0
1	0.5
2	1
3	1.5

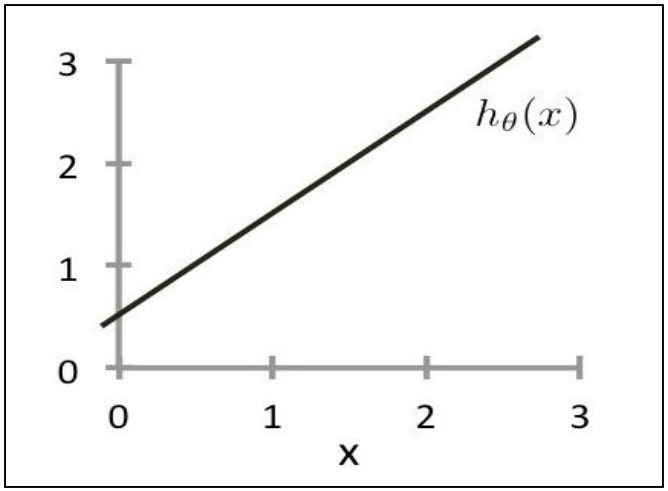
The Hypothesis Function



x	h(x)
0	1
1	1.5
2	2
3	2.5

The Hypothesis Function

- Consider the plot below of $h_{(\theta)}(x) = \theta_0 + \theta_1 x$. What are θ_0 and θ_1?



- (a) $\theta_0=0$ $\theta_1=1$
- **(b) $\theta_0=0.5$ $\theta_1=1$ (answer)**
- (c) $\theta_0=1$ $\theta_1=0.5$
- (d) $\theta_0=1$ $\theta_1=1$

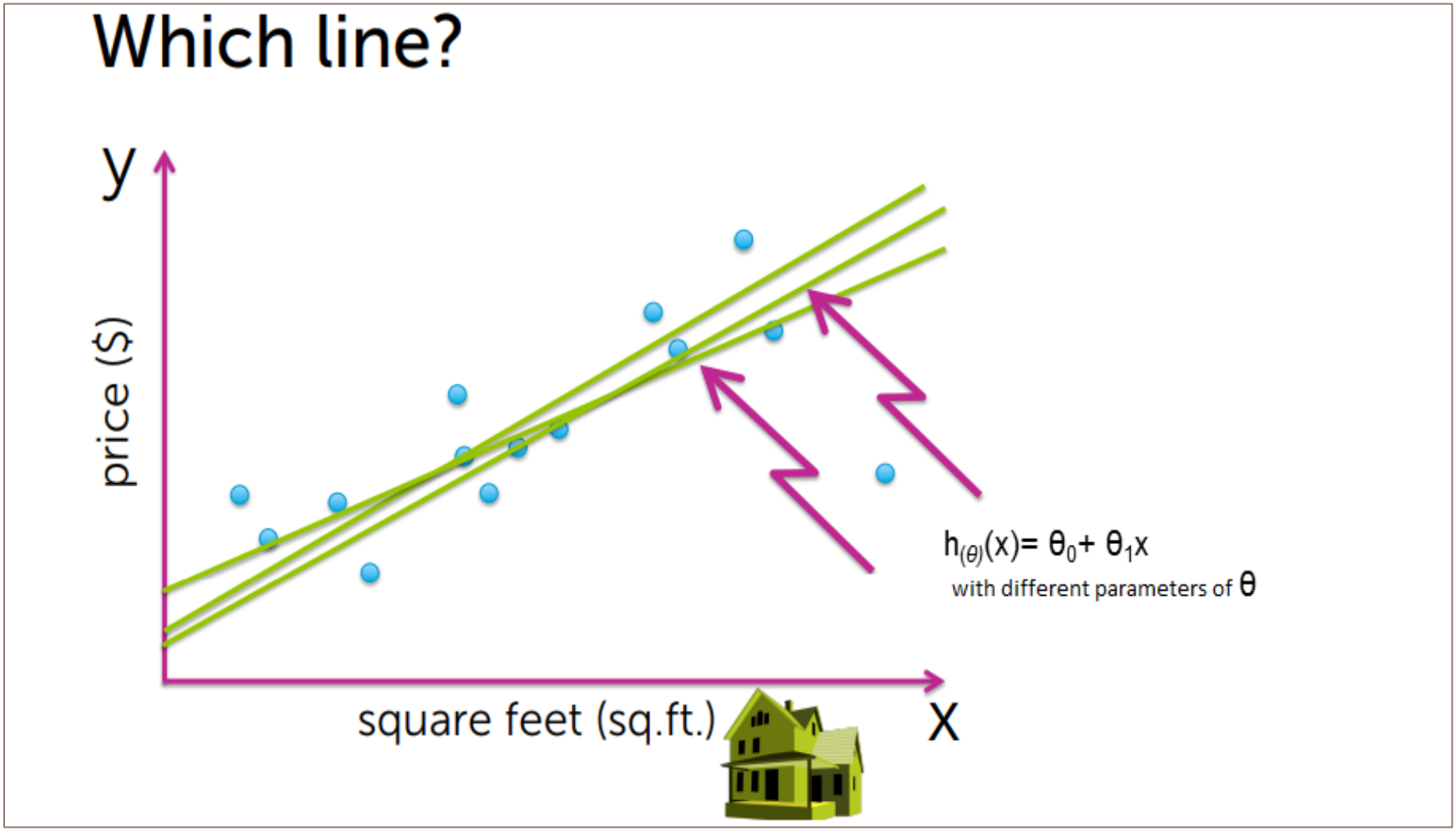
The Hypothesis Function

- **Example:**
- Suppose we have the following set of training data:

input x	output y
0	4
1	7
2	7
3	8

- Now we can make a random guess about our h_θ function: $\theta_0=2$ and $\theta_1=2$.
- Our hypothesis function has the general form: $h_{(\theta)}(x)= \theta_0+ \theta_1 x$
- The hypothesis function becomes: $h_{(\theta)}(x)=2+2X$
- So for input of 1 to our hypothesis, y will be 4. This is off by 3. Note that we will be trying out various values of θ_0 and θ_1 to try to find values which provide the best possible "fit" or the most representative "straight line" through the data points mapped on the x-y plane.
- **Idea:** Choose θ_0 and θ_1 so that $h_{(\theta)}(x)$ is close to y for our training examples (x,y).

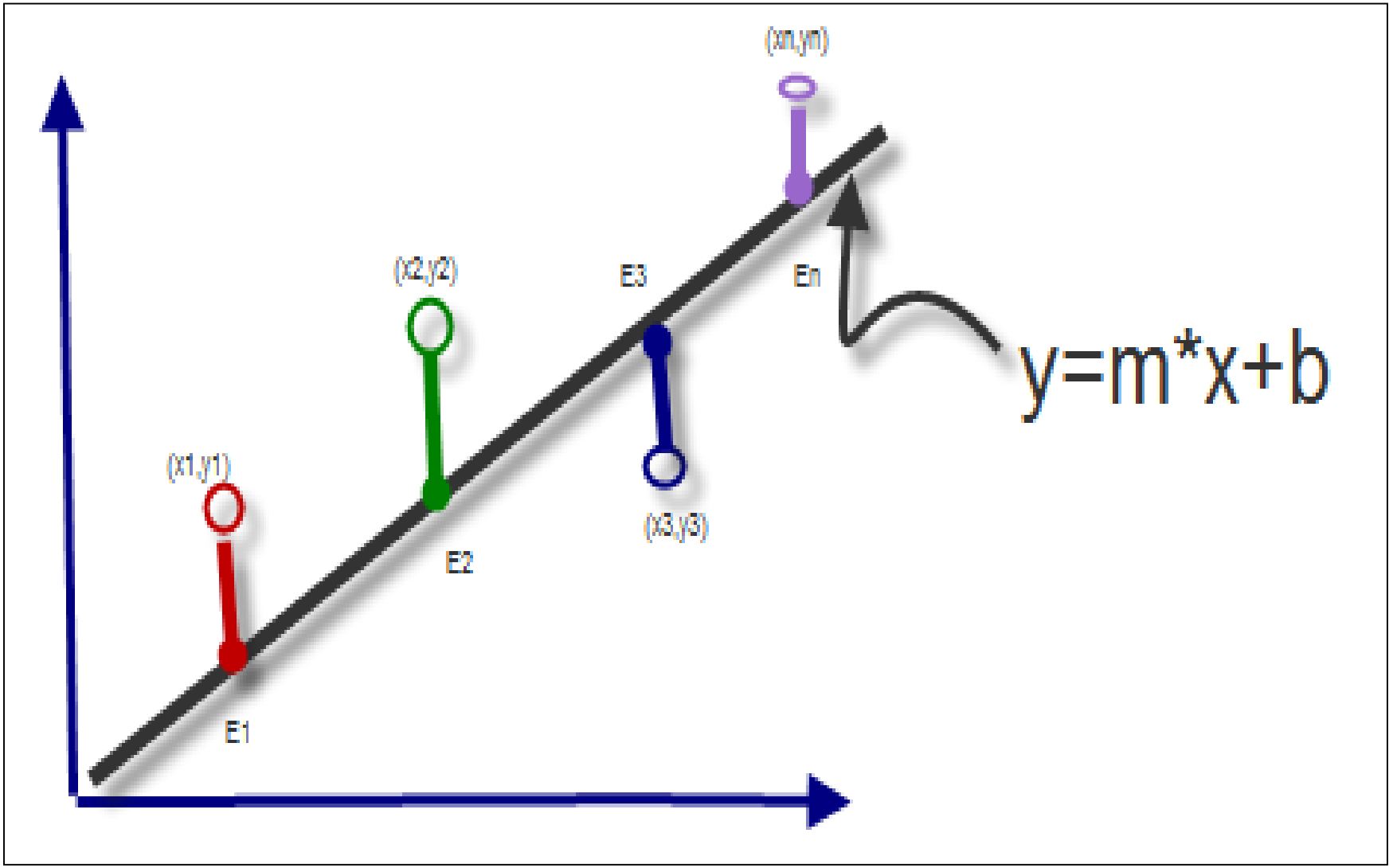
The Hypothesis Function



The Hypothesis Function

- **Residual sum of squares (RSS) (or Squared Error of Regression Line)**
- Let we have “n” points on a coordinate plane.
- We want to do is find a line that minimizes the squared distance to these different points.
- Let visualize that line for a second, I don’t know what is looks like right now.
- And what we want to do is minimize this squared error from each of these points to the line.
- Equation of the line is; y=mx+b, Where; m= slope of the line, b= y- intercept
- We want to find an “m” and “b”, so hat it minimizes the squared error.
- So for each of these points, the error between it and the line is vertical distance.

The Hypothesis Function



The Hypothesis Function

- $Error_1 = y_1-(mx_1+b)$, $Error_2 = y_2-(mx_2+b)$, -----, $Error_n = y_n-(mx_n+b)$

- $SE_{Line}=(Error_1)^2+(Error_2)^2+---+(Error_n)^2$

- $SE_{Line}=(y_1-(mx_1+b))^2+(y_2-(mx_2+b))^2+---+(y_n-(mx_n+b))^2$

- $SE_{Line}= [(y_1^2-(2)(y_1)(mx_1+b)+(mx_1+b)^2]+[(y_2^2-(2)(y_2)(mx_2+b)+(mx_2+b)^2]+ ----+ [(y_n^2-(2)(y_n)(mx_n+b)+(mx_n+b)^2]$ (Because $(a-b)^2=a^2-2ab+b^2$)

- $SE_{Line}= [y_1^2-2y_1mx_1-2y_1b+\{m^2x_1^2+2mx_1b+b^2\}]+ [y_2^2-2y_2mx_2-2y_2b+\{m^2x_2^2+2mx_2b+b^2\}]+----+ [y_n^2-2y_nmx_n-2y_nb+\{m^2x_n^2+2mx_nb+b^2\}]$ (Because $(a+b)^2=a^2+2ab+b^2$)

- $SE_{Line}= (y_1^2+y_2^2+----+y_n^2)-2m(x_1y_1+x_2y_2+---+x_ny_n)-2b(y_1+y_2+---+y_n)+m^2(x_1^2+x_2^2+----+x_n^2)+2mb(x_1+x_2+----+x_n)+nb^2$ ------ (equation-A)

The Hypothesis Function

- As we know;

$$\frac{y_1^2 + y_2^2 + \cdots + y_n^2}{n} = \overline{y^2} \qquad y_1^2 + y_2^2 + \cdots + y_n^2 = n\overline{y^2}$$

- Similarly;

$$\frac{x_1y_1 + x_2y_2 + \cdots + x_ny_n}{n} = \overline{xy} \qquad x_1y_1 + x_2y_2 + \cdots + x_ny_n = n\overline{xy}$$

$$\frac{y_1 + y_2 + \cdots + y_n}{n} = \overline{y} \qquad y_1 + y_2 + \cdots + y_n = n\overline{y}$$

$$\frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n} = \overline{x^2} \qquad x_1^2 + x_2^2 + \cdots + x_n^2 = n\overline{x^2}$$

$$\frac{x_1 + x_2 + \cdots + x_n}{n} = \overline{x} \qquad x_1 + x_2 + \cdots + x_n = n\overline{x}$$

The Hypothesis Function

- Equation (A) becomes, we have

- $SE_{Line} = (y_1^2 + y_2^2 + \text{----} + y_n^2) - 2m(x_1y_1 + x_2y_2 + \text{---} + x_ny_n) - 2b(y_1 + y_2 + \text{---} + y_n) + m^2(x_1^2 + x_2^2 + \text{----} + x_n^2) + 2mb(x_1 + x_2 + \text{----} + x_n) + nb^2$

- By putting the values, we have,

$$SE_{Line} = n\overline{y^2} - 2mn\overline{xy} - 2bn\overline{y} + m^2nx^2 + 2mbn\overline{x} + nb^2 \quad \text{-------} \quad \text{(equation-B)}$$

By taking partial derivative of equation B w.r.t ‘m’, we have

$$\frac{\partial SE_{Line}}{\partial m} = \frac{\partial}{\partial m}(n\overline{y^2} - 2mn\overline{xy} - 2bn\overline{y} + m^2nx^2 + 2mbn\overline{x} + nb^2)$$

$$\frac{\partial SE_{Line}}{\partial m} = \frac{\partial}{\partial m}(n\overline{y^2}) - \frac{\partial}{\partial m}(2mn\overline{xy}) - \frac{\partial}{\partial m}(2bn\overline{y}) + \frac{\partial}{\partial m}(m^2nx^2) + \frac{\partial}{\partial m}(2mbn\overline{x}) + \frac{\partial}{\partial m}(nb^2)$$

The Hypothesis Function

$$\frac{\partial SE_{Line}}{\partial m} = 0 - 2n\overline{xy} - 0 + 2mnx^2 + 2bn\overline{x} + 0$$

$$\frac{\partial SE_{Line}}{\partial m} = -2n\overline{xy} + 2mnx^2 + 2bn\overline{x} \quad \text{-------} \quad \text{(Equation-C)}$$

- Let;

$$\frac{\partial SE_{Line}}{\partial m} = 0$$

- So, equation C becomes,

$$-2n\overline{xy} + 2mnx^2 + 2bn\overline{x} = 0$$

$$2n(-\overline{xy} + mx^2 + b\overline{x}) = 0$$

$$-\overline{xy} + mx^2 + b\overline{x} = 0$$

$$mx^2 + b\overline{x} = \overline{xy}$$

The Hypothesis Function

$$\frac{mx^2}{\overline{x}} + \frac{b\overline{x}}{\overline{x}} = \frac{\overline{xy}}{\overline{x}}$$

$$\frac{mx^2}{\overline{x}} + b = \frac{\overline{xy}}{\overline{x}} \qquad \text{-------} \quad \text{Equation-D}$$

$$\frac{x^2}{\overline{x}}, \frac{\overline{xy}}{\overline{x}} \; lies\ on\ the\ best\ fit\ line$$

The Hypothesis Function

- Equation (B) becomes, we have

$$SE_{Line} = n\overline{y^2} - 2mn\overline{xy} - 2bn\overline{y} + m^2nx^2 + 2mbn\overline{x} + nb^2$$

By taking partial derivative of equation B w.r.t ‘b’, we have

$$\frac{\partial SE_{Line}}{\partial b} = \frac{\partial}{\partial b}(n\overline{y^2} - 2mn\overline{xy} - 2bn\overline{y} + m^2nx^2 + 2mbn\overline{x} + nb^2)$$

$$\frac{\partial SE_{Line}}{\partial b} = \frac{\partial}{\partial b}(n\overline{y^2}) - \frac{\partial}{\partial b}(2mn\overline{xy}) - \frac{\partial}{\partial b}(2bn\overline{y}) + \frac{\partial}{\partial b}(m^2nx^2) + \frac{\partial}{\partial b}(2mbn\overline{x}) + \frac{\partial}{\partial b}(nb^2)$$

$$\frac{\partial SE_{Line}}{\partial b} = 0 - 0 - 2n\overline{y} + 0 + 2mn\overline{x} + 2n\text{b}$$

$$\frac{\partial SE_{Line}}{\partial b} = -2n\overline{y} + 2mn\overline{x} + 2n\text{b} \quad \text{------ (Equation-E)}$$

The Hypothesis Function

- Let; $\frac{\partial SE_{Line}}{\partial m} = 0$
- So, equation E becomes,

$$-2n\bar{y} + 2mn\bar{x} + 2nb=0$$

$$2n(-\bar{y} + m\bar{x} + b)=0$$

$$2n(-\bar{y} + m\bar{x} + b)=0$$

$$-\bar{y} + m\bar{x} + b=0$$

$$m\bar{x} + b=\bar{y} \quad \text{-------} \quad \text{(Equation-F)}$$

$\bar{x}$ & $\bar{y}$ *lies on the line*

The Hypothesis Function

- Equation (F) – Equation (E), we get

$$(m\bar{x} + b) - \left(m\frac{\overline{x^2}}{\bar{x}} + b\right) = \bar{y} - \frac{\overline{xy}}{\bar{x}}$$

$$m\bar{x} + b - m\frac{\overline{x^2}}{\bar{x}} - b = \bar{y} - \frac{\overline{xy}}{\bar{x}}$$

$$m\bar{x} - m\frac{\overline{x^2}}{\bar{x}} = \bar{y} - \frac{\overline{xy}}{\bar{x}}$$

$$m\left(\bar{x} - \frac{\overline{x^2}}{\bar{x}}\right) = \bar{y} - \frac{\overline{xy}}{\bar{x}}$$

The Hypothesis Function

$$m = \frac{\bar{y} - \frac{\overline{xy}}{\bar{x}}}{\bar{x} - \frac{\overline{x^2}}{\bar{x}}}$$

$$m = \frac{\bar{x}\,\bar{y} - \overline{xy}}{(\bar{x})^2 - \overline{x^2}}$$

- Equation F becomes, we have

$$m\bar{x} + b = \bar{y}$$

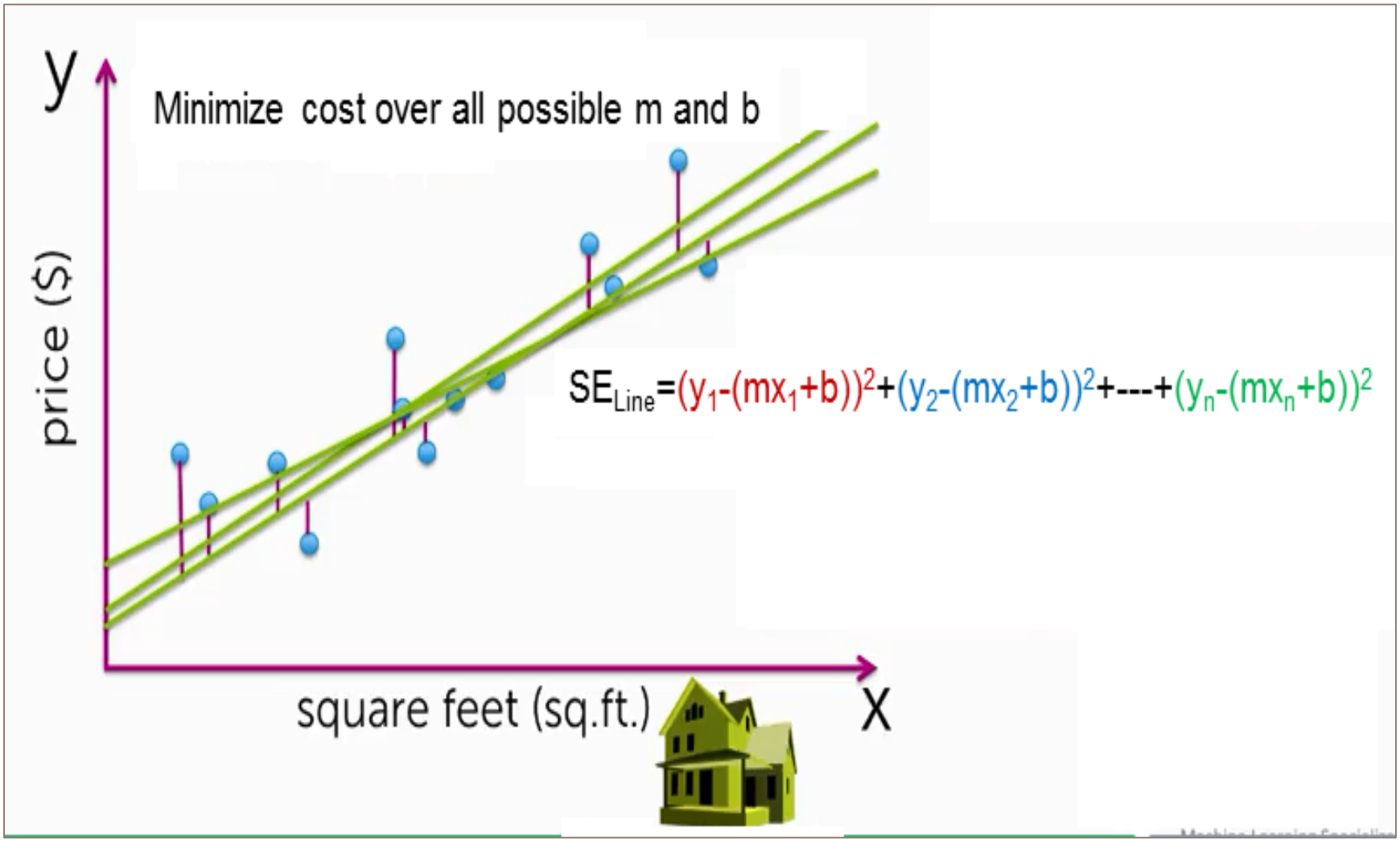
$$b = \bar{y} - m\bar{x}$$

The Hypothesis Function

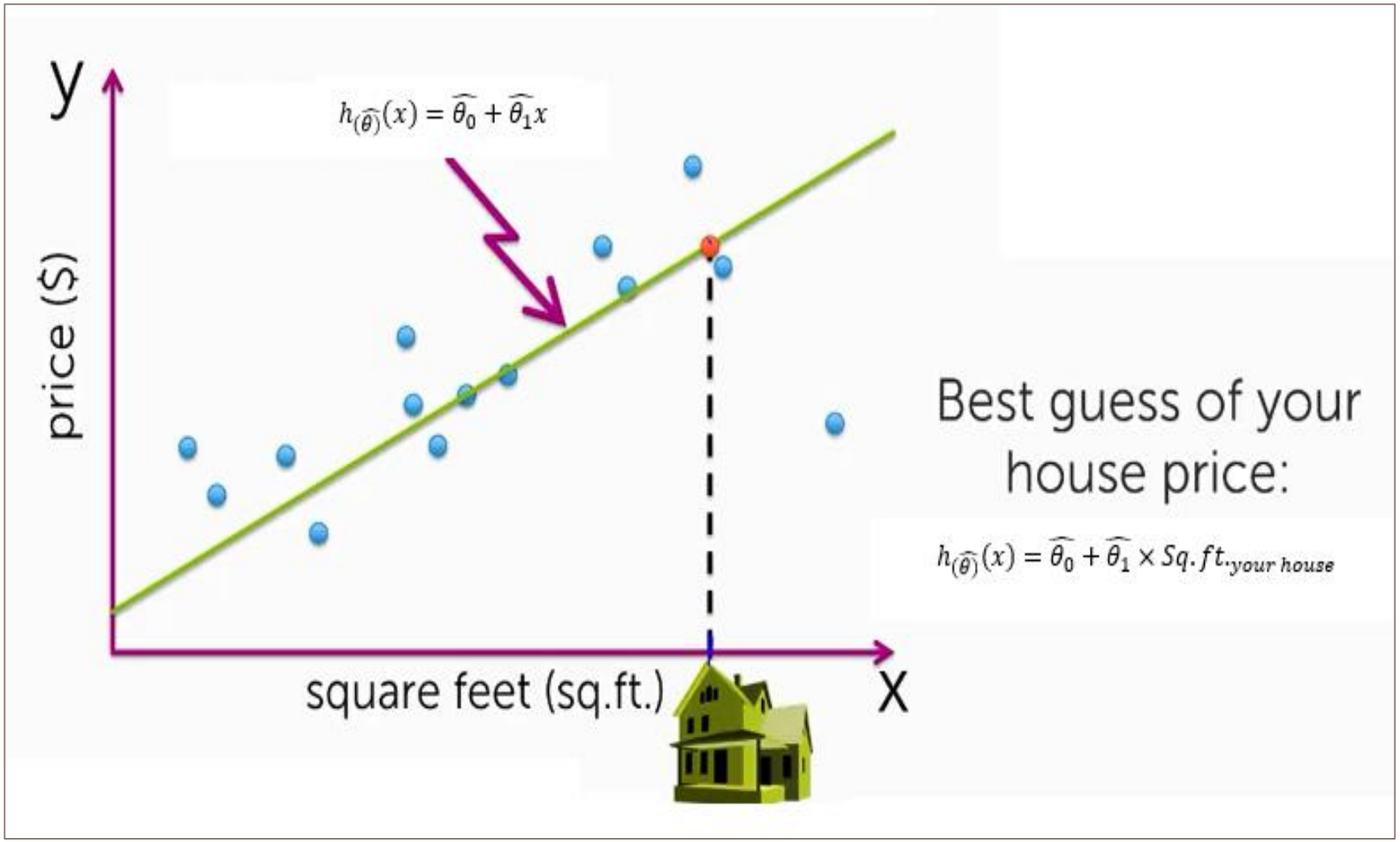
- By putting the value of ‘m’, we have

$$b=\overline{y} - \left(\frac{\overline{x}\,\overline{y}-\overline{xy}}{(\overline{x})^2-\overline{x^2}}\right)\overline{x}$$

The Hypothesis Function



The Hypothesis Function



Linear Regression with one variable

Cost Function

Cost Function

- We can measure the accuracy of our hypothesis function by using a **cost function**.

- This takes an average (actually a fancier version of an average) of all the results of the hypothesis with inputs from x's compared to the actual output y's.

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

- To break it apart, it is (1/2) $\bar{x}$ where $\bar{x}$ is the mean of the squares of $h(x_i)-y_i$, or the difference between the predicted value and the actual value.

- This function is otherwise called the "Squared error function", or "Mean squared error".

- The mean is halved (1/2m) as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the (1/2) term.

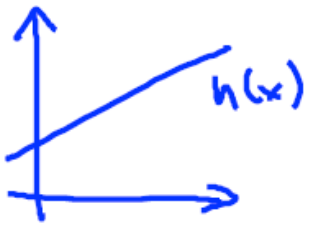
Cost Function

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$



Cost Function:

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

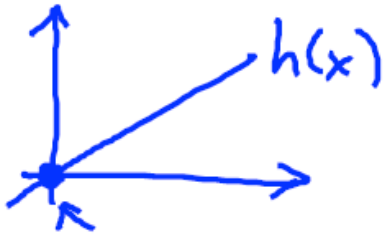
Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

Simplified

$$h_\theta(x) = \theta_1 x$$

- If θ_0=0 then

θ_1



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

$\underset{\theta_1}{\text{minimize}}\ J(\theta_1)$

- h(x) for fixed θ_1, this is the function of x (size of the house)
- $J(\theta_1)$ function of the parameter θ_1 (which control the slop of the straight line

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 $h(x)$) 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.

Cost Function

The following example shows the ideal situation where we have a cost function of 0.

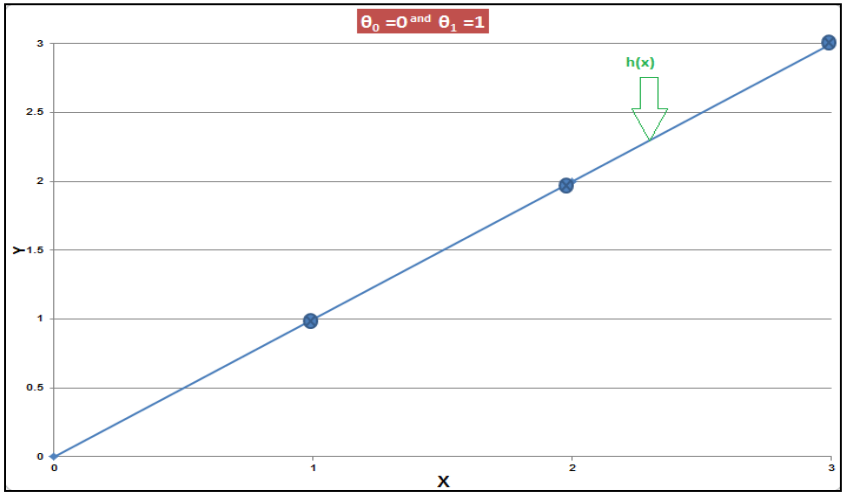
IF $\theta_0 = 0$ and $\theta_1 = 1$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + 1(x)$

$h(x) = 0 + x$

$h(x) = x$



	x	y	h(x)=x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	1	0	0
	2	2	2	0	0
	3	3	3	0	0
Sum	6	6	6	0	0
m=3					

$$J(0,1) = \frac{1}{2 \times 3}[0] = 0$$

When *θ1=1, we get a slope of 1 which goes through every single data point in our model.*

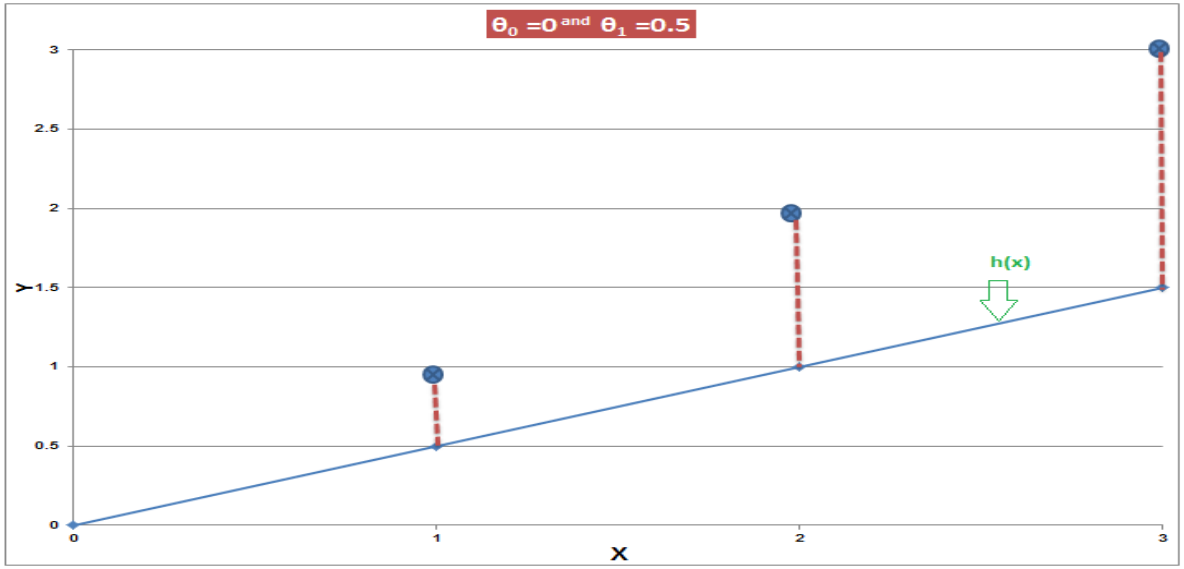
Cost Function

IF $\theta_0 = 0$ and $\theta_1 = 0.5$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + 0.5(x)$

$h(x) = 0.5x$



	x	y	h(x)=0.5x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	0.5	-0.5	0.25
	2	2	1	-1	1
	3	3	1.5	-1.5	2.25
Sum	6	6	3	-3	3.5
m=3					

$$J(0,0.5) = \frac{1}{2\times 3}[3.5] = 0.58$$

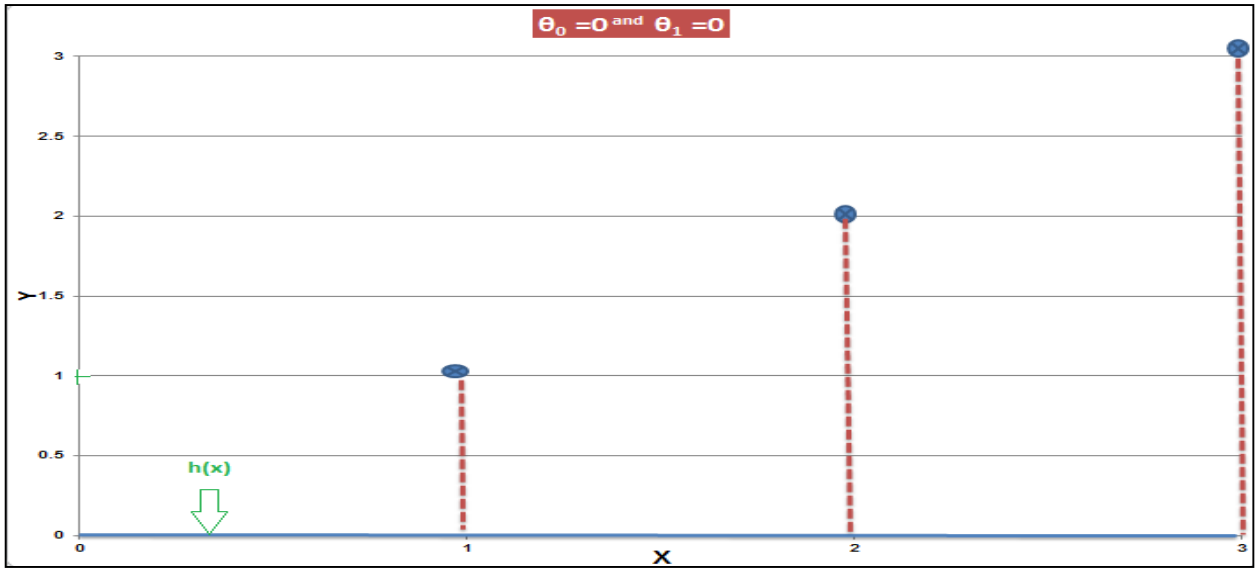
When $\theta 1=0.5$, we see the vertical distance from our fit to the data points increase.

Cost Function

IF $\theta_0 = 0$ and $\theta_1 = 0$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0$



	x	y	h(x)=0	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	0	-1	1
	2	2	0	-2	4
	3	3	0	-3	9
Sum	6	6	0	-6	14
m=3					

$$J(0,0) = \frac{1}{2 \times 3}[14] = 2.3$$

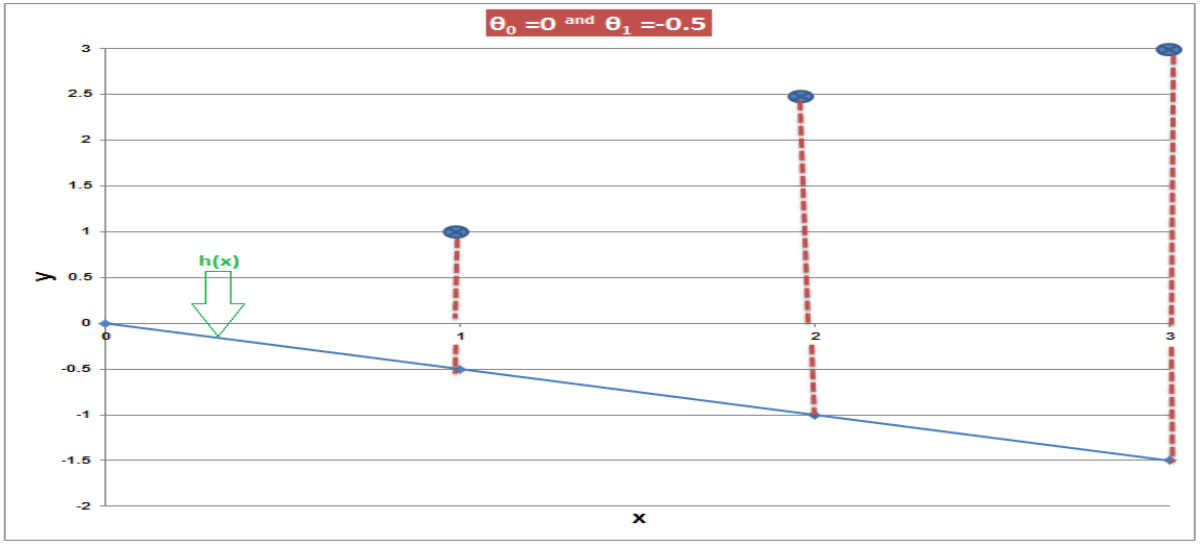
Cost Function

IF $\theta_0 = 0$ and $\theta_1 = -0.5$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + (-0.5)x$

$h(x) = -0.5x$



	x	y	h(x)=-0.5x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	-0.5	-1.5	2.25
	2	2	-1	-3	9
	3	3	-1.5	-4.5	20.25
Sum	6	6	-3	-9	31.5
m=3					

$$J(0, -0.5) = \frac{1}{2 \times 3}[31.5] = 5.25$$

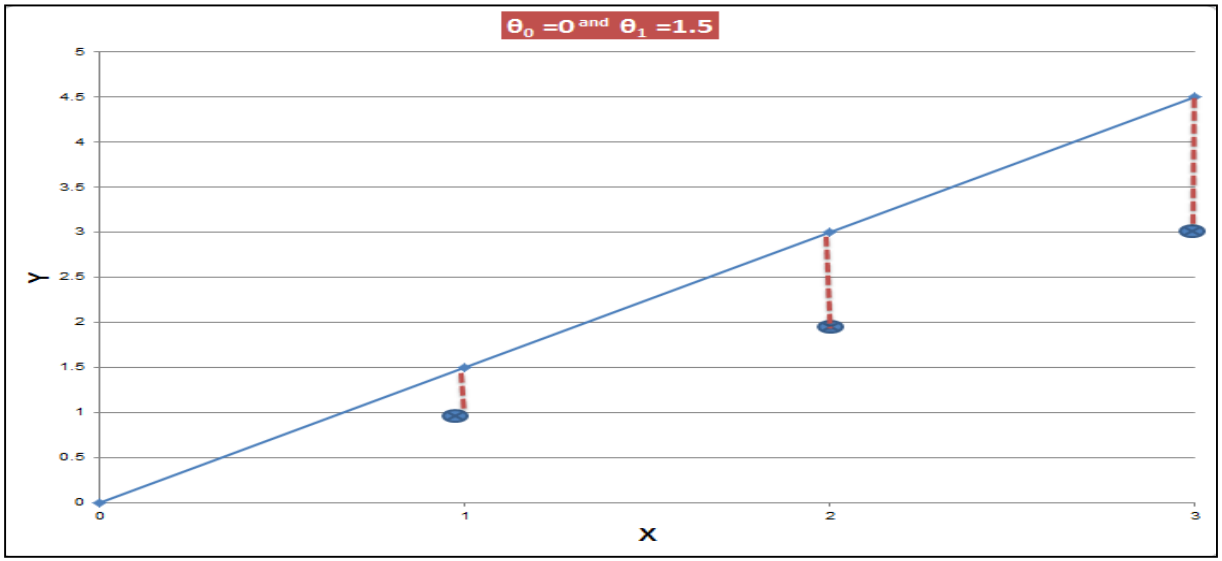
Cost Function

IF $\theta_0 = 0$ and $\theta_1 = 1.5$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + (1.5)x$

$h(x) = 1.5x$



	x	y	h(x)=1.5x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	1.5	0.5	0.25
	2	2	3	1	1
	3	3	4.5	1.5	2.25
Sum	6	6	9	3	3.5
m=3					

$$J(0,1.5) = \frac{1}{2 \times 3}[3.5] = 0.58$$

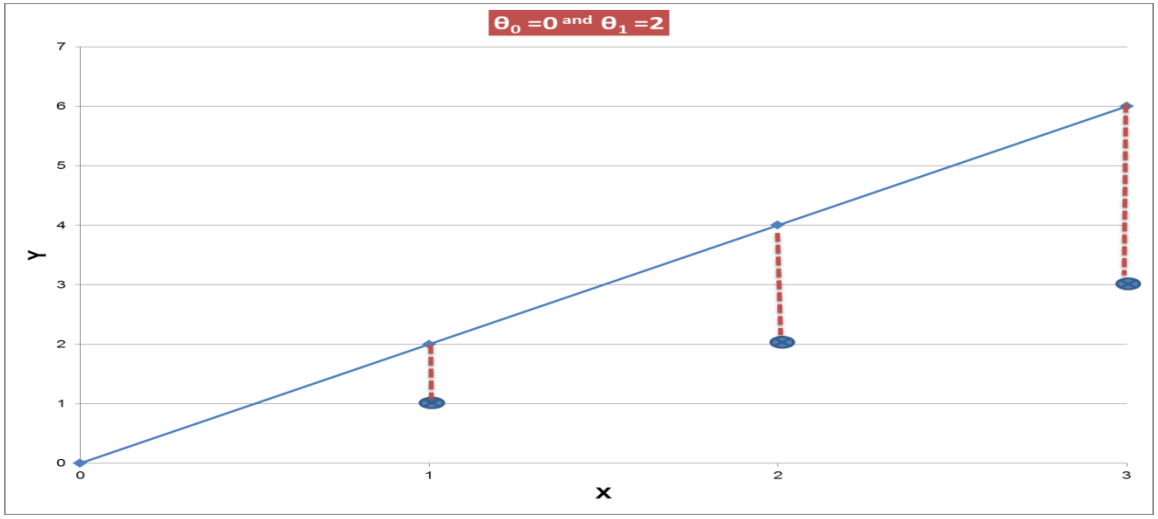
Cost Function

IF $\theta_0 = 0$ and $\theta_1 = 2$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + (2)x$

$h(x) = 2x$



	x	y	h(x)=2x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	2	1	1
	2	2	4	2	4
	3	3	6	3	9
Sum	6	6	11	5	14
m=3					

$$J(0,2) = \frac{1}{2 \times 3}[14] = 2.33$$

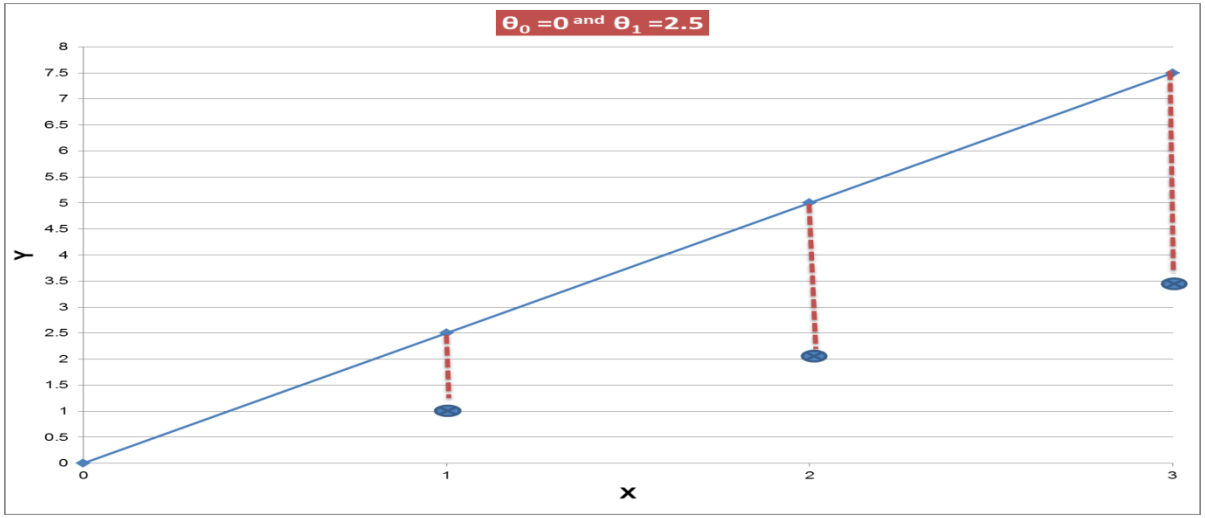
Cost Function

IF $\theta_0 = 0$ and $\theta_1 = 2.5$

$h(x) = \theta_0 + \theta_1 x$

$h(x) = 0 + (2.5)x$

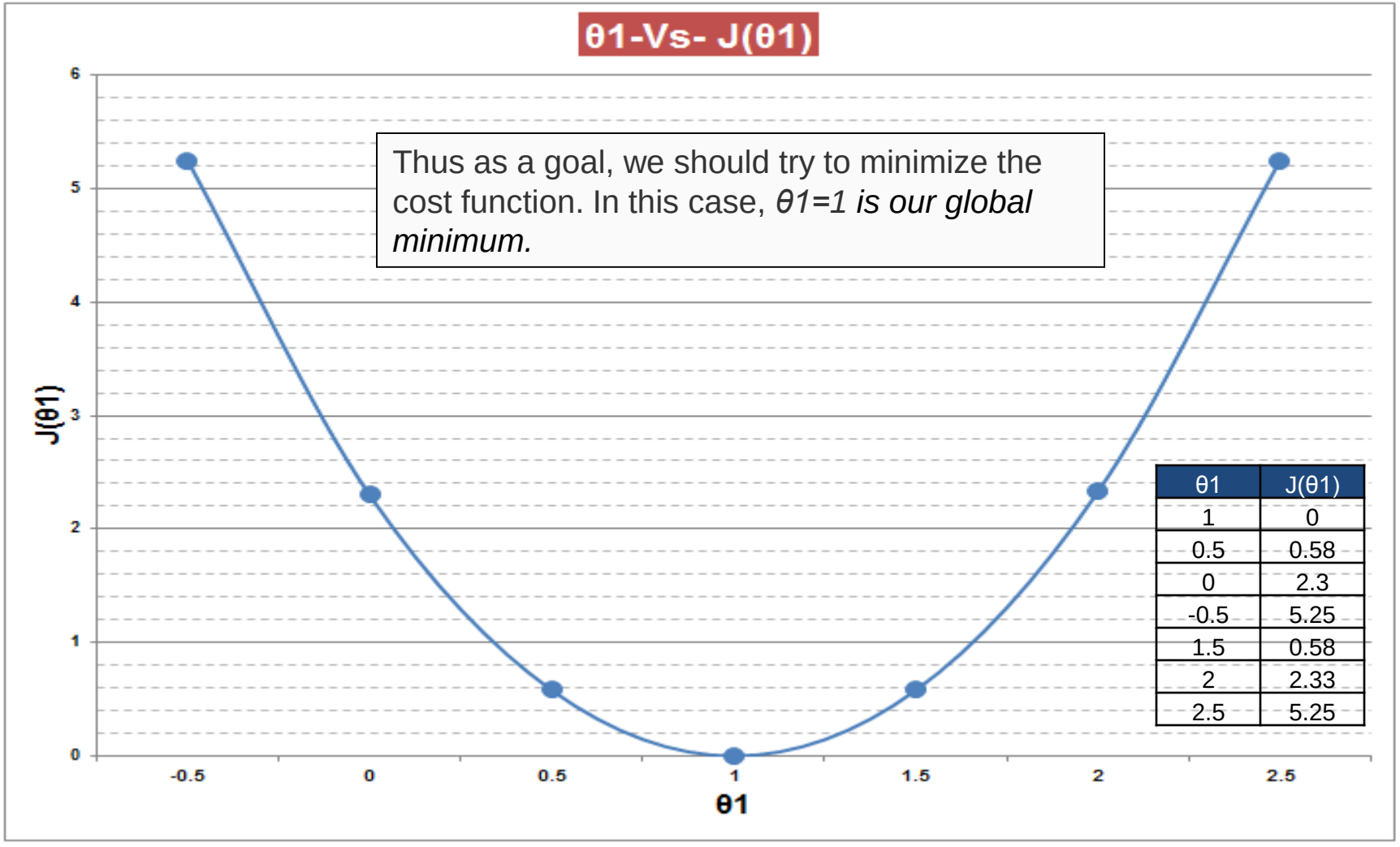
$h(x) = 2.5x$



	x	y	h(x)=2.5x	h(x)-y	(h(x)-y)^2
	0	0	0	0	0
	1	1	2.5	1.5	2.25
	2	2	5	3	9
	3	3	7.5	4.5	20.25
Sum	6	6	15	9	31.5
m=3					

$$J(0,2.5) = \frac{1}{2 \times 3}[31.5] = 5.25$$

Cost Function

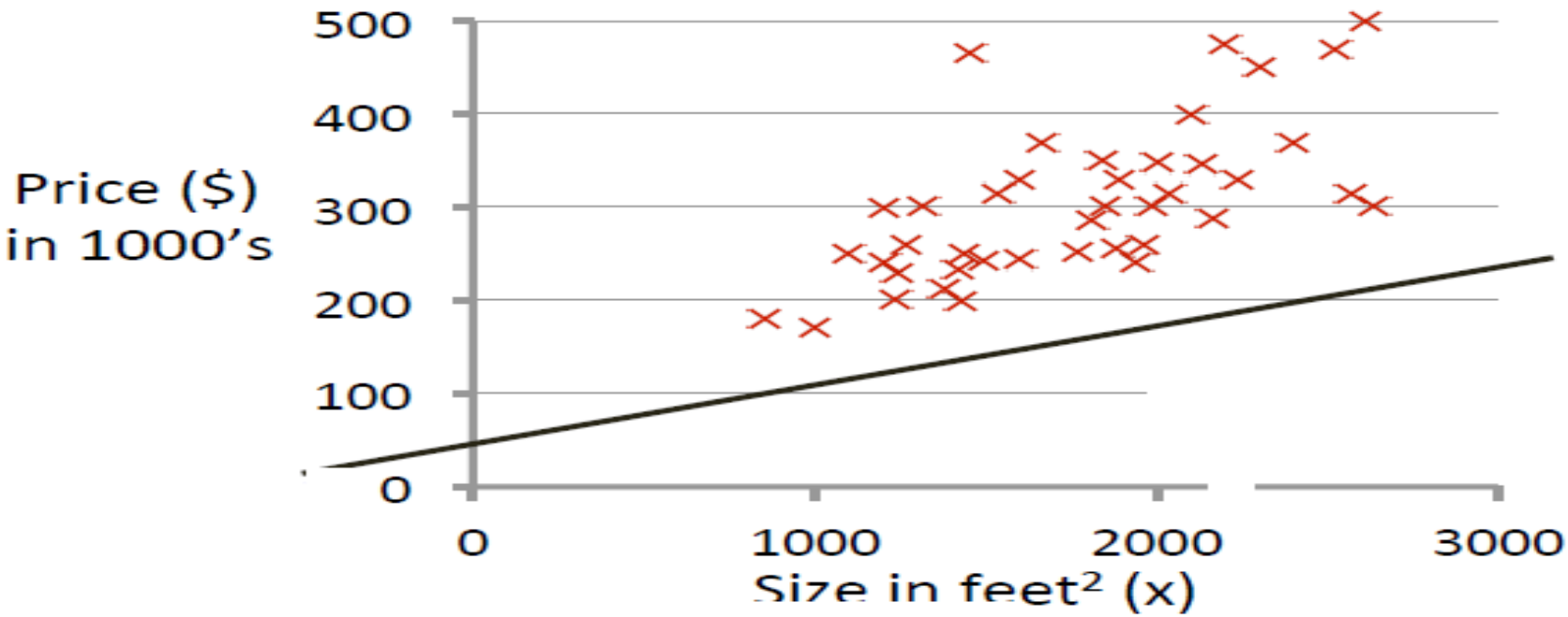


θ1	J(θ1)
1	0
0.5	0.58
0	2.3
-0.5	5.25
1.5	0.58
2	2.33
2.5	5.25

Cost Function

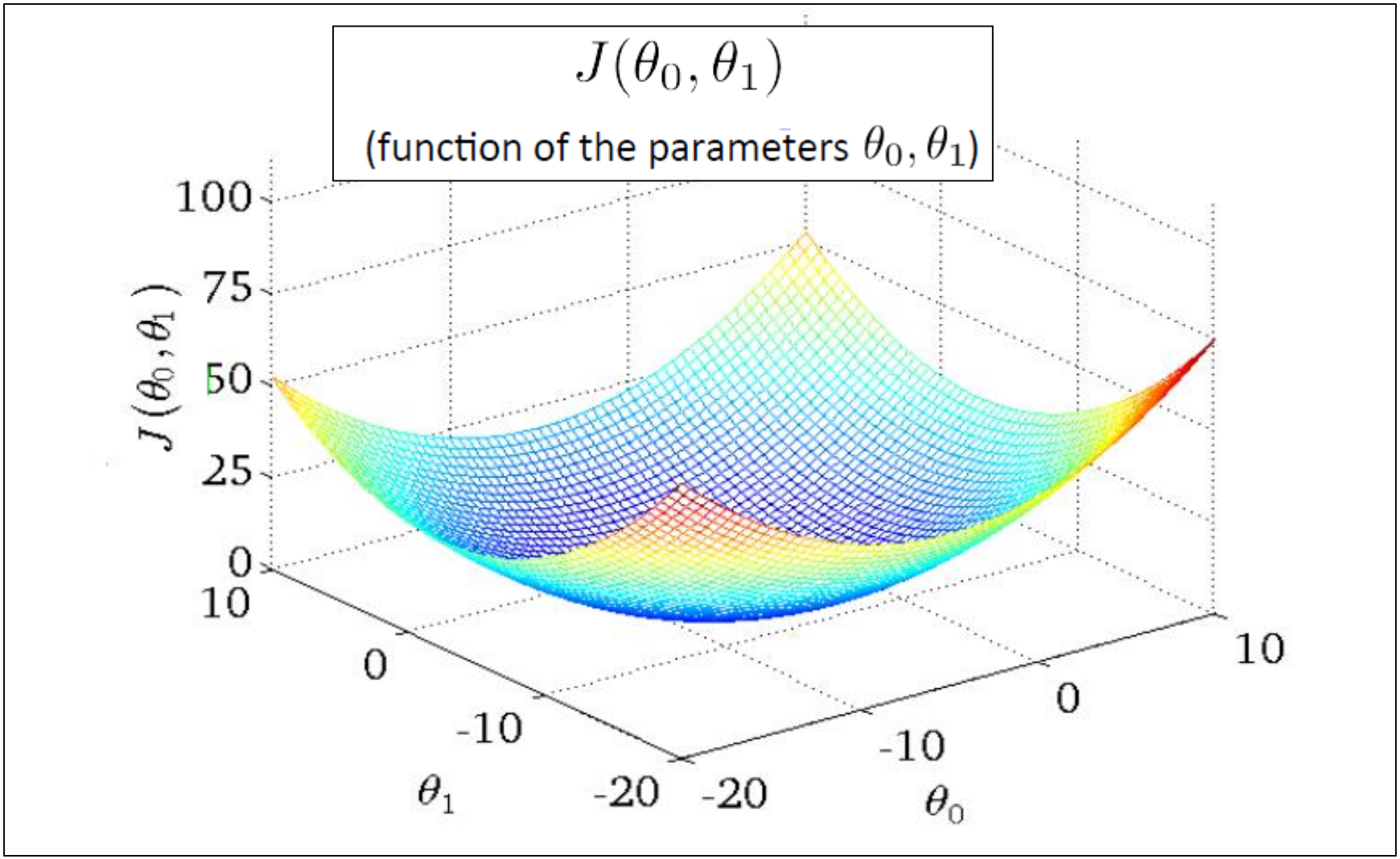
$h_\theta(x)$

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



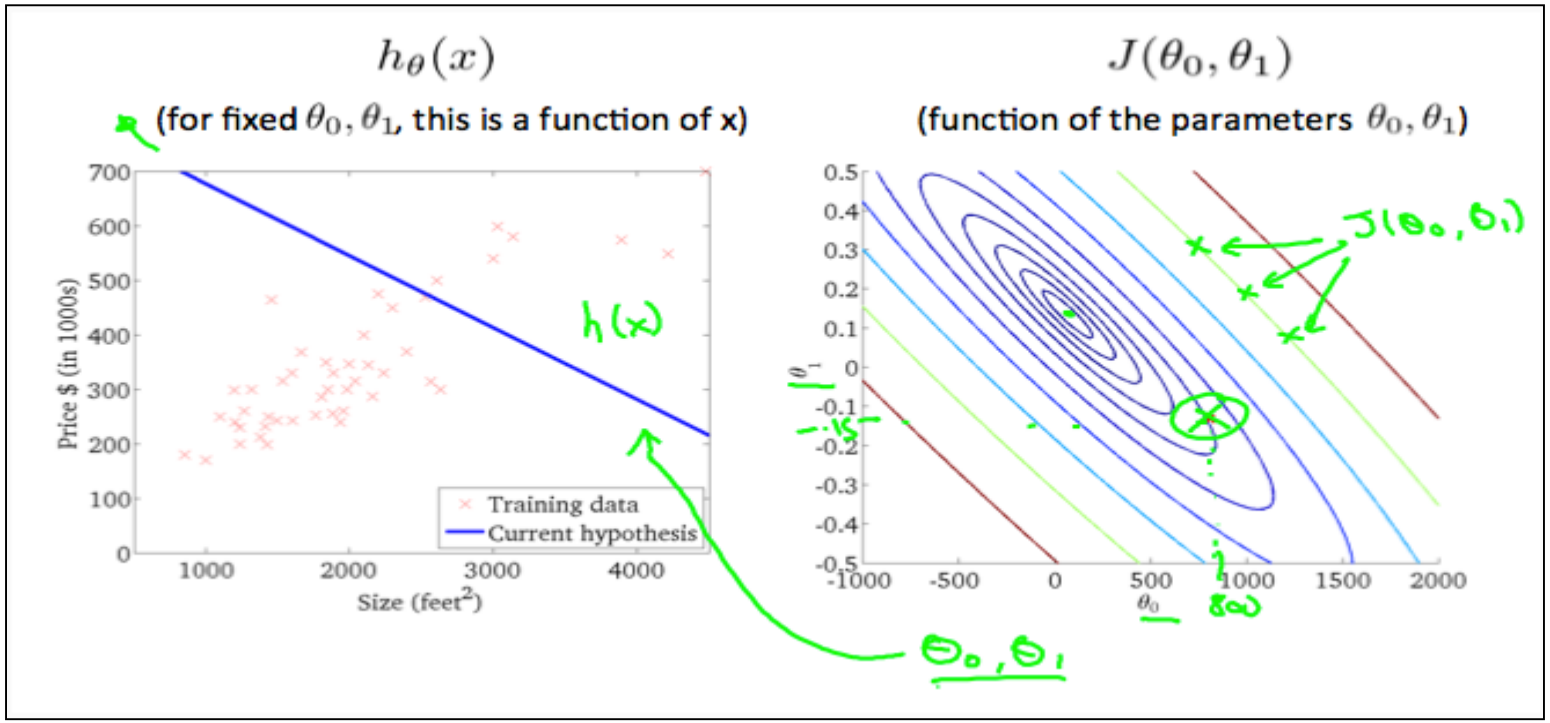
$$h_\theta(x) = 50 + 0.06x$$

Cost Function



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. An example of such a graph is the one to the right below.



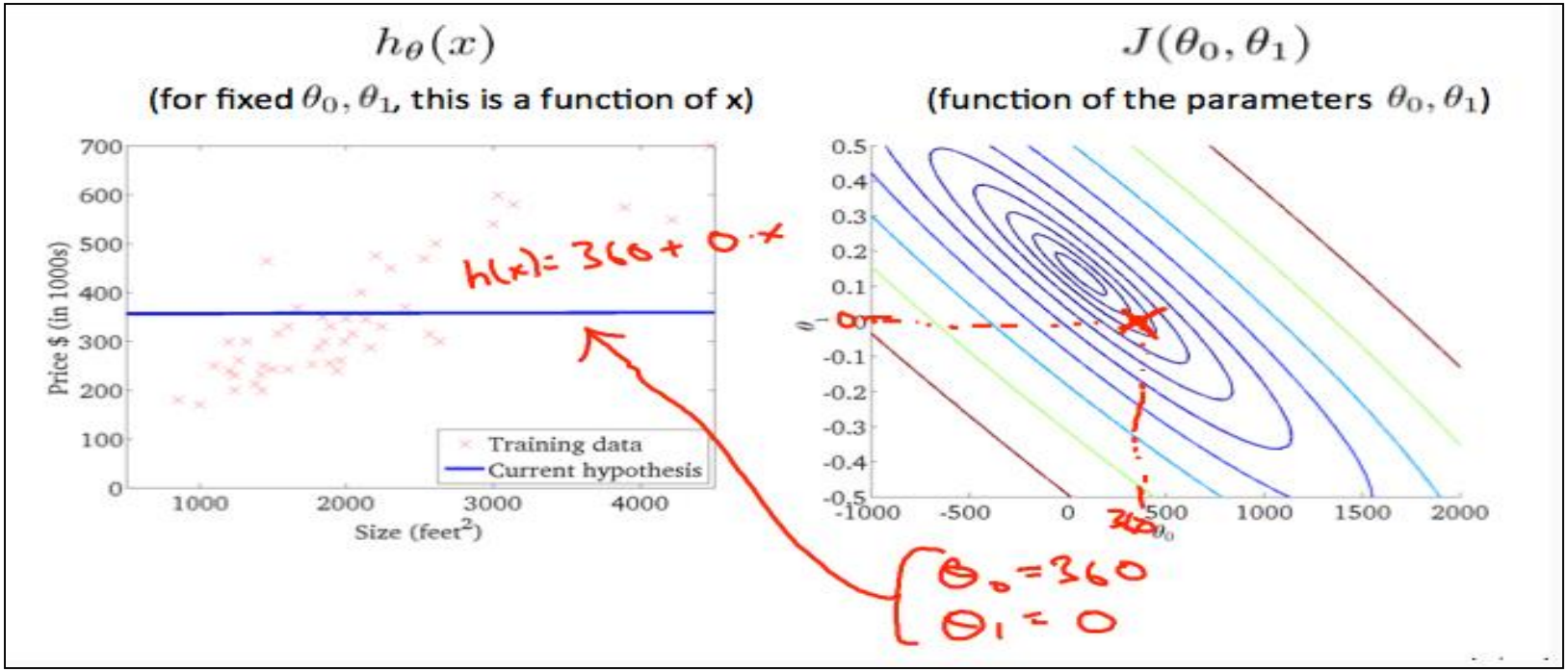
Cost Function

- Taking any color and going along the 'circle', one would expect to get the same value of the cost function.
- For example, the three green points found on the green line above have the same value for $J(\theta_0, \theta_1)$ and as a result, they are found along the same line.

- The circled x displays the value of the cost function for the graph on the left when $\theta_0 = 800$ and $\theta_1 = -0.15$.

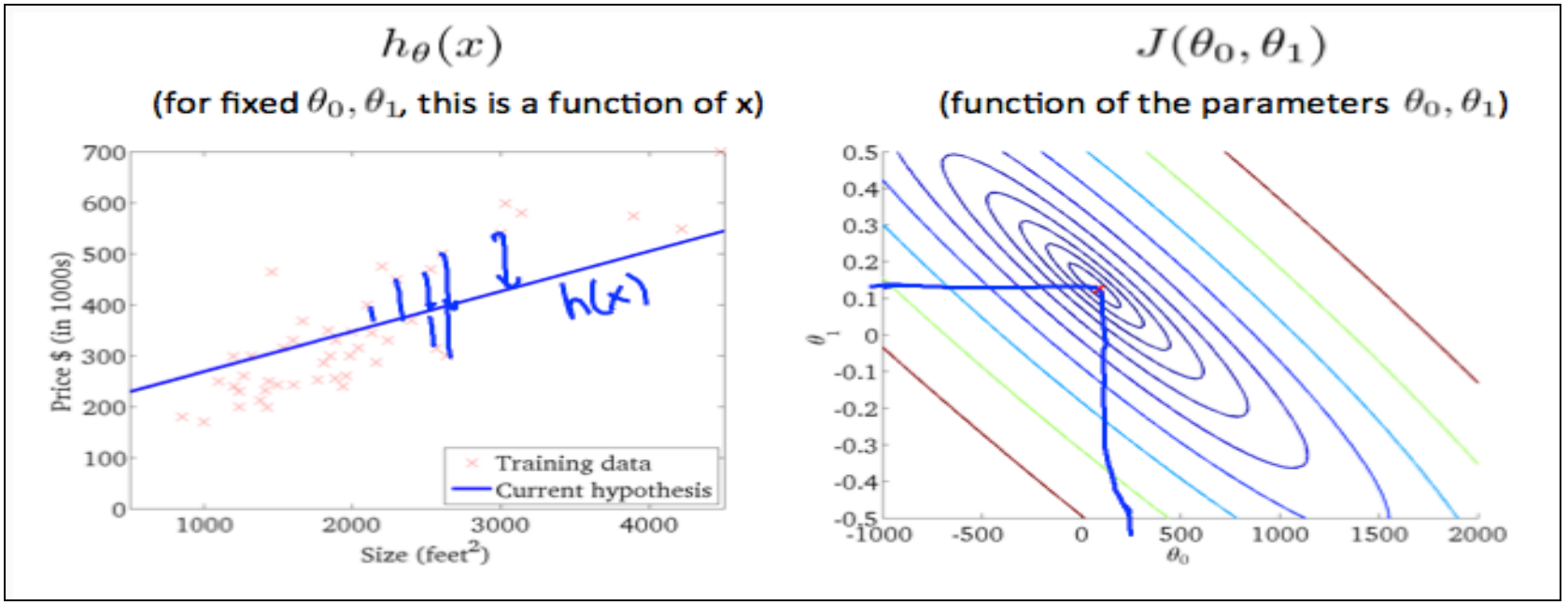
Cost Function

Taking another h(x) and plotting its contour plot, one gets the following graphs:



- When θ_0 = 360 and θ_1 = 0, the value of $J(\theta_0, \theta_1)$ in the contour plot gets closer to the center thus reducing the cost function error.
- Now giving our hypothesis function a slightly positive slope results in a better fit of the data.

Cost Function



- The graph above minimizes the cost function as much as possible and consequently, the result of θ_1 and θ_1 tend to be around 0.12 and 250 respectively.

- Plotting those values on our graph to the right seems to put our point in the center of the inner most 'circle'.

Linear Regression with one variable

Gradient Descent For Linear Regression

Gradient Descent

- Have some function $J(\theta_0, , \theta_1)$
- Want minimum $J(\theta_0, , \theta_1)$

- **Outline:**

- Start with some θ_0, θ_1 (say 0,0)
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up a minimum.

repeat until convergence: {

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

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i)$$

}

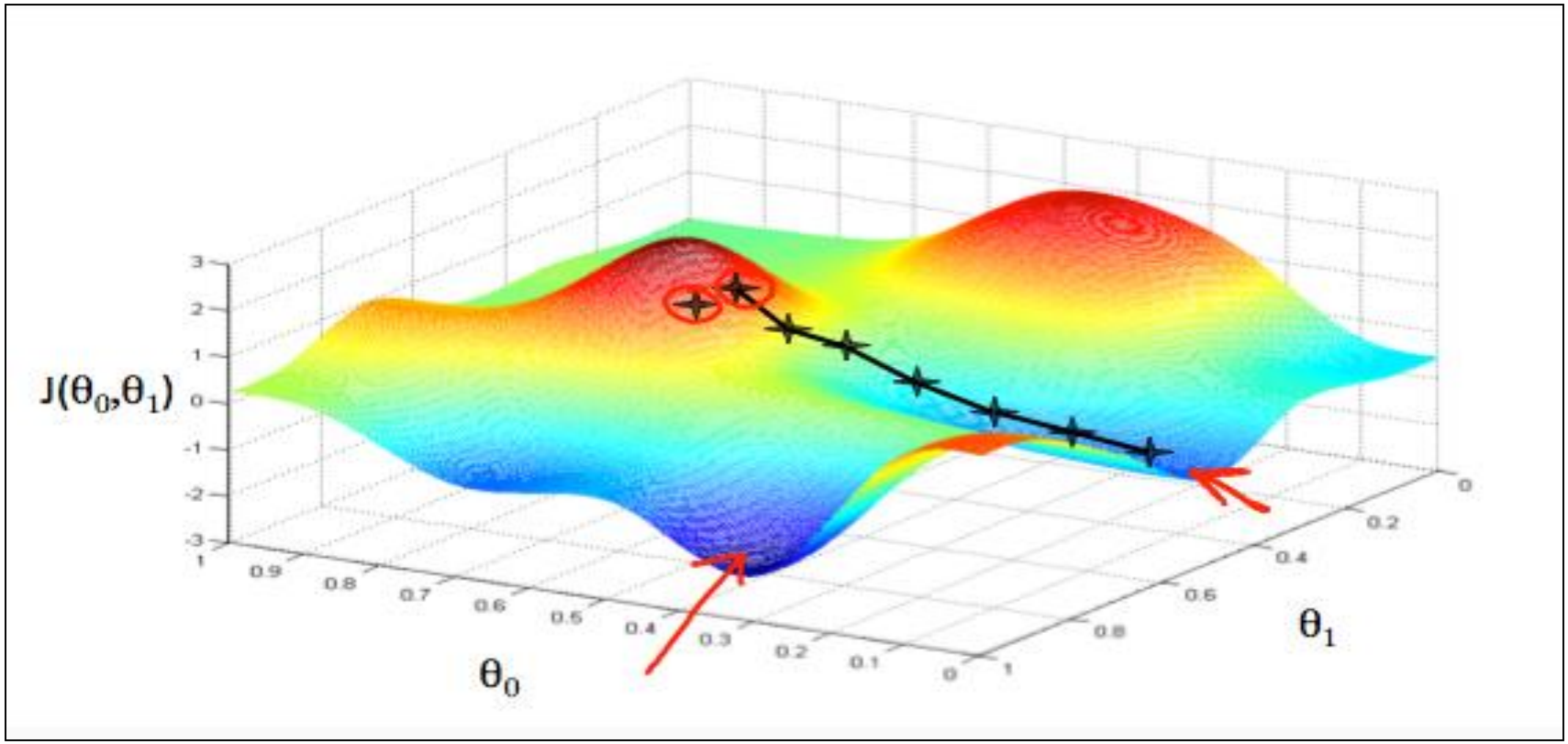
Gradient Descent

- 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. The graph below depicts such a setup.

Gradient Descent



- 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 red arrows show the minimum points in the graph.

Gradient Descent

- 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.
- For example, the distance between each 'star' in the graph above represents a step determined by our parameter α.
- A smaller α would result in a smaller step and a larger α results in a larger step.
- The direction in which the step is taken is determined by the partial derivative of $J(\theta_0, \theta_1)$. Depending on where one starts on the graph, one could end up at different points.
- The image above shows us two different starting points that end up in two different places.

Gradient Descent

- We explored the scenario where we used one parameter θ_1 and plotted its cost function to implement a gradient descent. Our formula for a single parameter was :

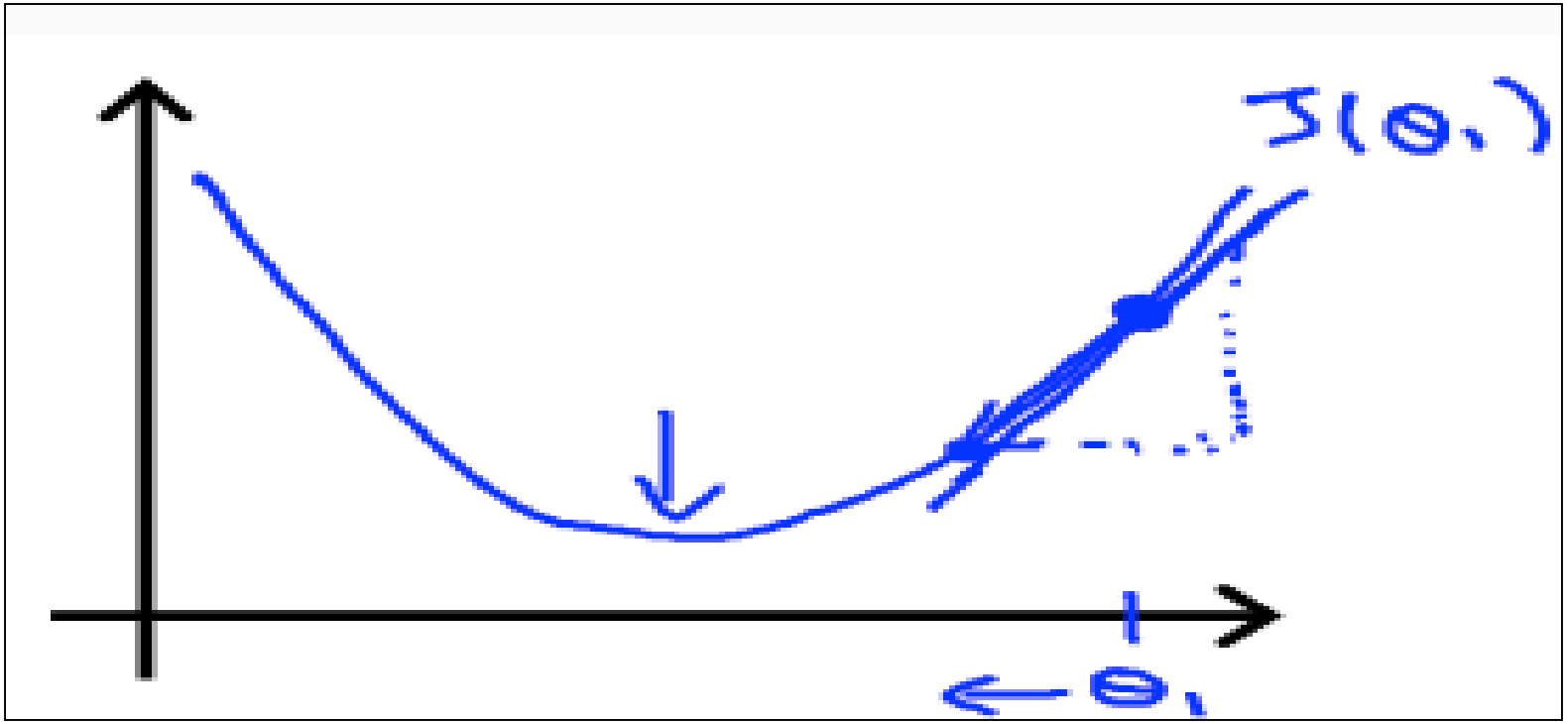
- Repeat until convergence:

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

- Regardless of the slope's sign for $d/d\ \theta_1\ J(\theta_1)$, θ_1 eventually converges to its minimum value.

- The following graph shows that when the slope is negative, the value of $\theta 1$ increases and when it is positive, the value of θ_1 decreases.

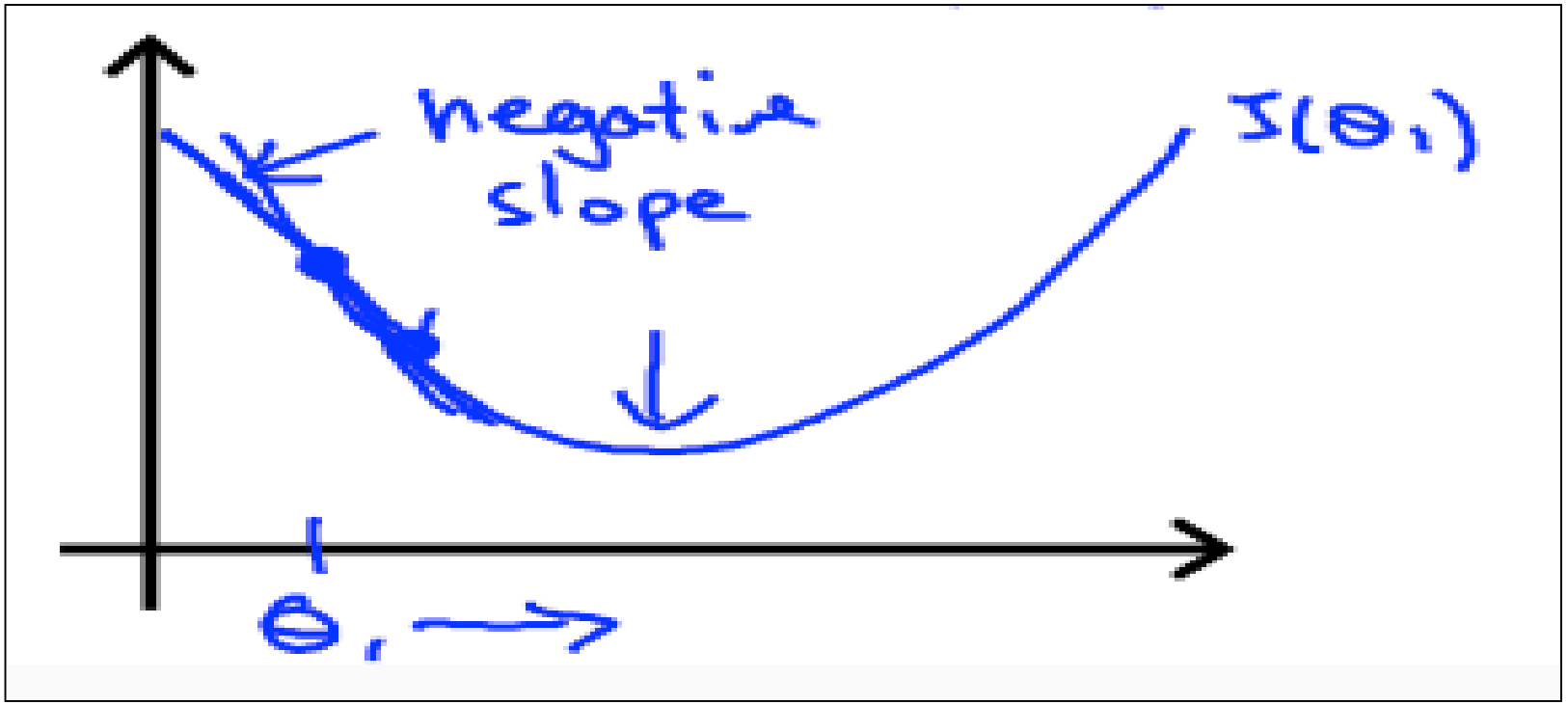
Gradient Descent



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

$\theta_1 := \theta_1 - \alpha$ (positive number)

Gradient Descent



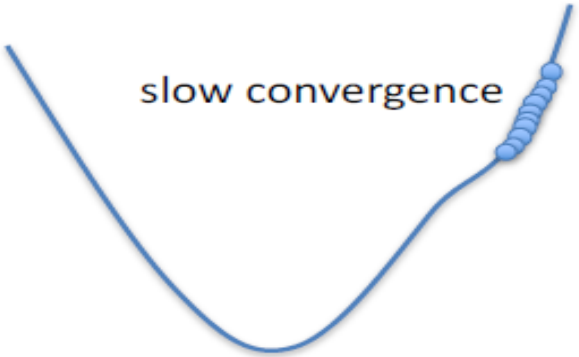
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

θ_1:= θ_1-α (negative number)

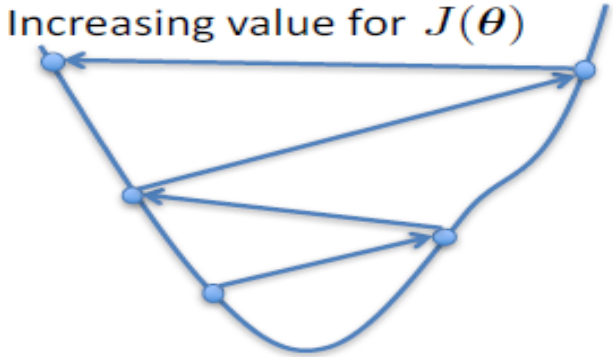
Gradient Descent

Choosing α

α too small



α too large



- May overshoot the minimum
- May fail to converge
- May even diverge

To see if gradient descent is working, print out $J(\theta)$ each iteration

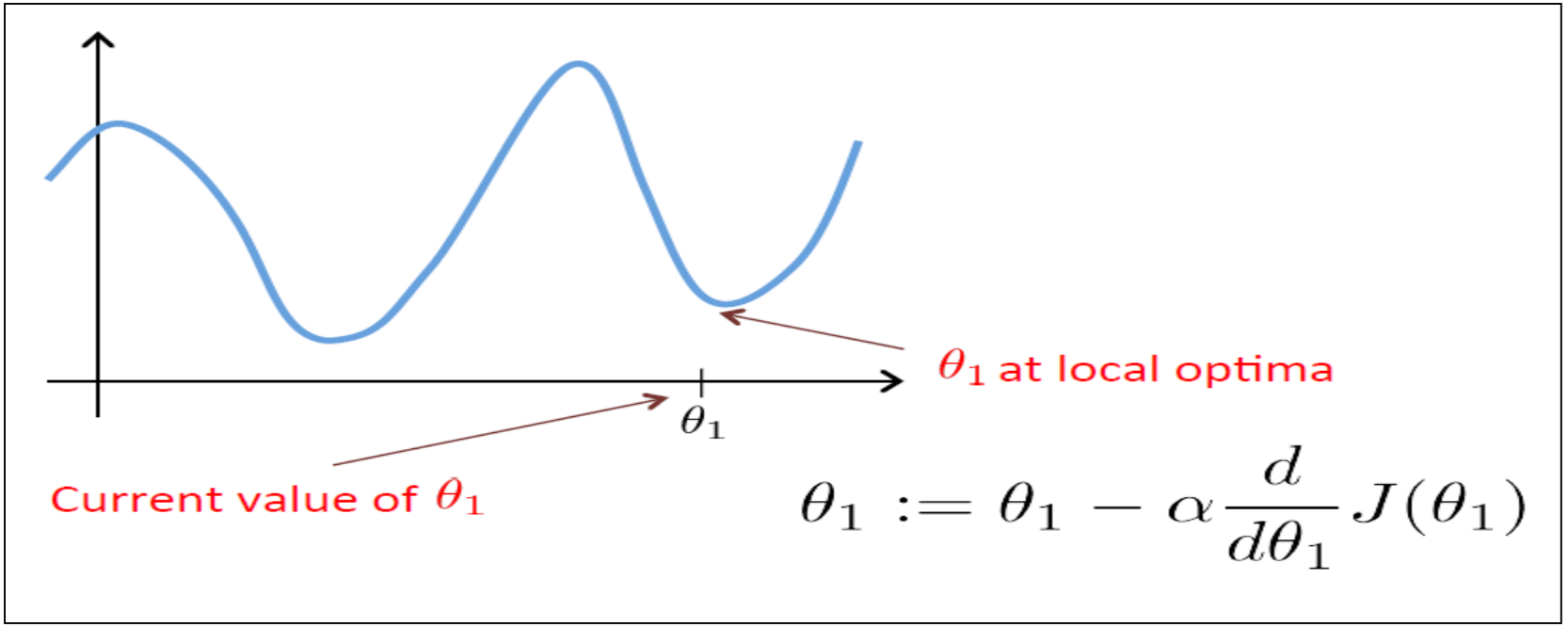
- The value should decrease at each iteration
- If it doesn't, adjust α

Gradient Descent

- The intuition behind the convergence is that $d/d\theta_1 J(\theta_1)$ approaches 0 as we approach the bottom of our convex function.

- At the minimum, the derivative will always be 0 and thus we get:

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

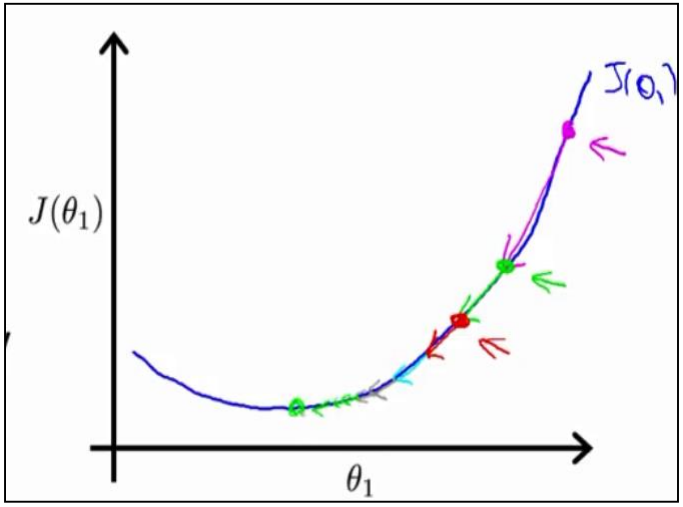


Gradient Descent

- Gradient Descent can converge to a local minimum, even with the leaning rate α fixed.

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

- As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.



Gradient Descent Algorithm

$$Repeat\ until\ convergence;$$

$$\{$$

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

$$\} \qquad ----------(Equation - A)$$

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

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

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \left[\frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2 \right] ------ (Equation - B)$$

Gradient Descent Algorithm

- **Case-1, j=0**
- By putting the value of j in equation (B), we have

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{\partial}{\partial\theta_0}\left[\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})^2\right]$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{2}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})\frac{\partial}{\partial\theta_0}(\theta_0+\theta_1x^{(i)}-y^{(i)})$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}\left(\theta_0+\theta_1x^{(i)}-y^{(i)}\right)[\frac{\partial}{\partial\theta_0}(\theta_0)+\frac{\partial}{\partial\theta_0}(\theta_1x^{(i)})-\frac{\partial}{\partial\theta_0}y^{(i)})]$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}\left(\theta_0+\theta_1x^{(i)}-y^{(i)}\right)[1+0-0]$$

Gradient Descent Algorithm

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)[1]$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)[x_0^{(i)}]$$

Gradient Descent Algorithm

- **Case-2, j=1,** By putting the value of j in equation (B), we have

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{\partial}{\partial\theta_1}\left[\frac{1}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})^2\right]$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{2}{2m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})\frac{\partial}{\partial\theta_1}(\theta_0+\theta_1x^{(i)}-y^{(i)})$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})[\frac{\partial}{\partial\theta_1}(\theta_0)+\frac{\partial}{\partial\theta_1}(\theta_1x^{(i)})-\frac{\partial}{\partial\theta_1}(y^{(i)})]$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})[0+x^{(i)}-0]$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)=\frac{1}{m}\sum_{i=1}^{m}(\theta_0+\theta_1x^{(i)}-y^{(i)})[x^{(i)}]$$

Gradient Descent Algorithm

By putting the values in equation (A), we have

$$Repeat\ until\ convergence;$$

$$\{$$

$$\theta_0 := \theta_0 - \frac{\propto}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1\, x^{(i)} - y^{(i)})[x_0^{(i)}]$$

$$\theta_1 := \theta_1 - \frac{\propto}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1\, x^{(i)} - y^{(i)})[x^{(i)}]$$

$$\}$$

$$Update\ \theta_0 and\ \theta_1 simultaneously$$

Gradient Descent Algorithm

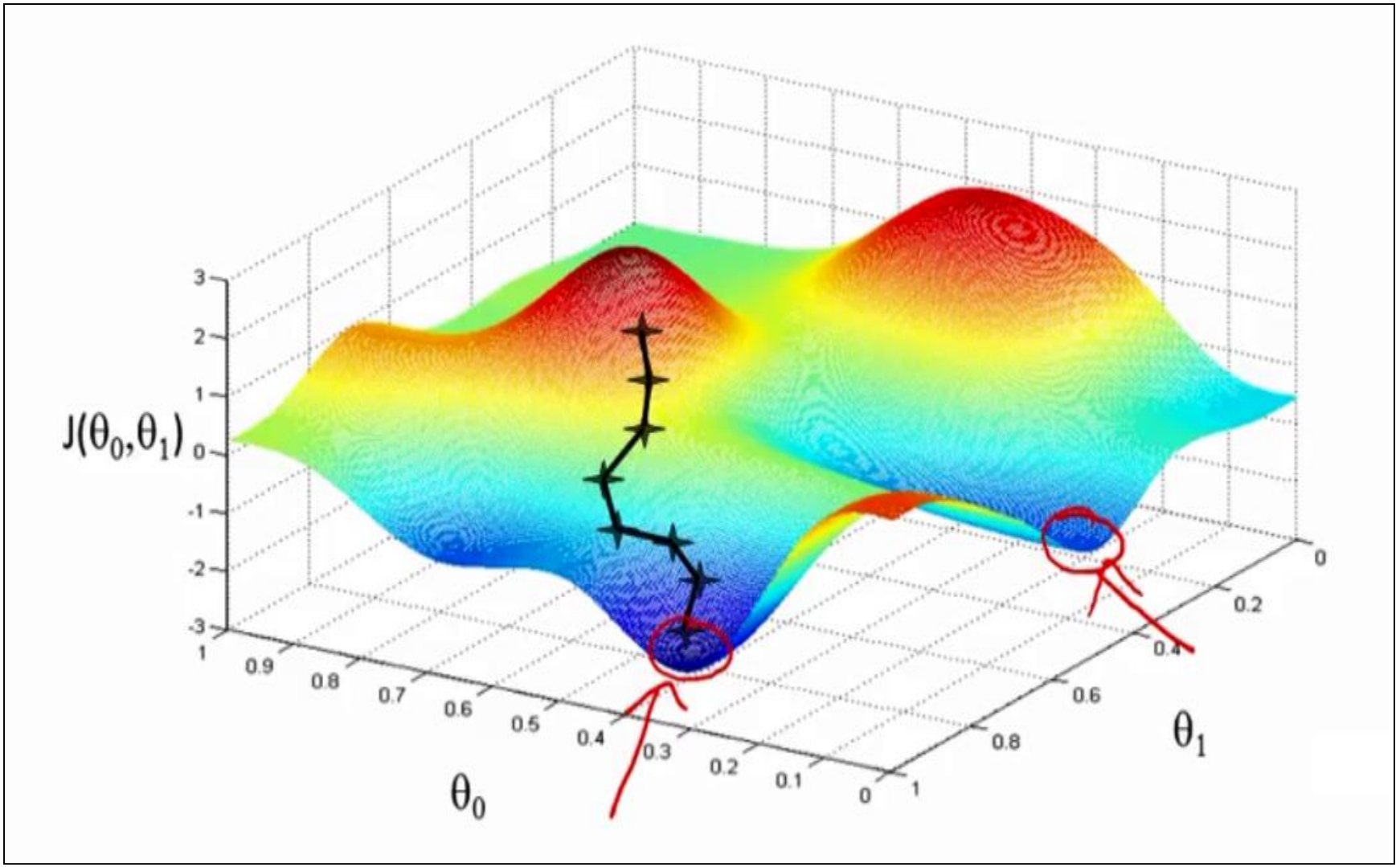
- The point of all this is that if we start with a guess for our hypothesis and then repeatedly apply these gradient descent equations, our hypothesis will become more and more accurate.

- **Batch Gradient Descent:**

- So, this is simply gradient descent on the original cost function J. This method looks at every example in the entire training set on every step, and is called **batch gradient descent**.

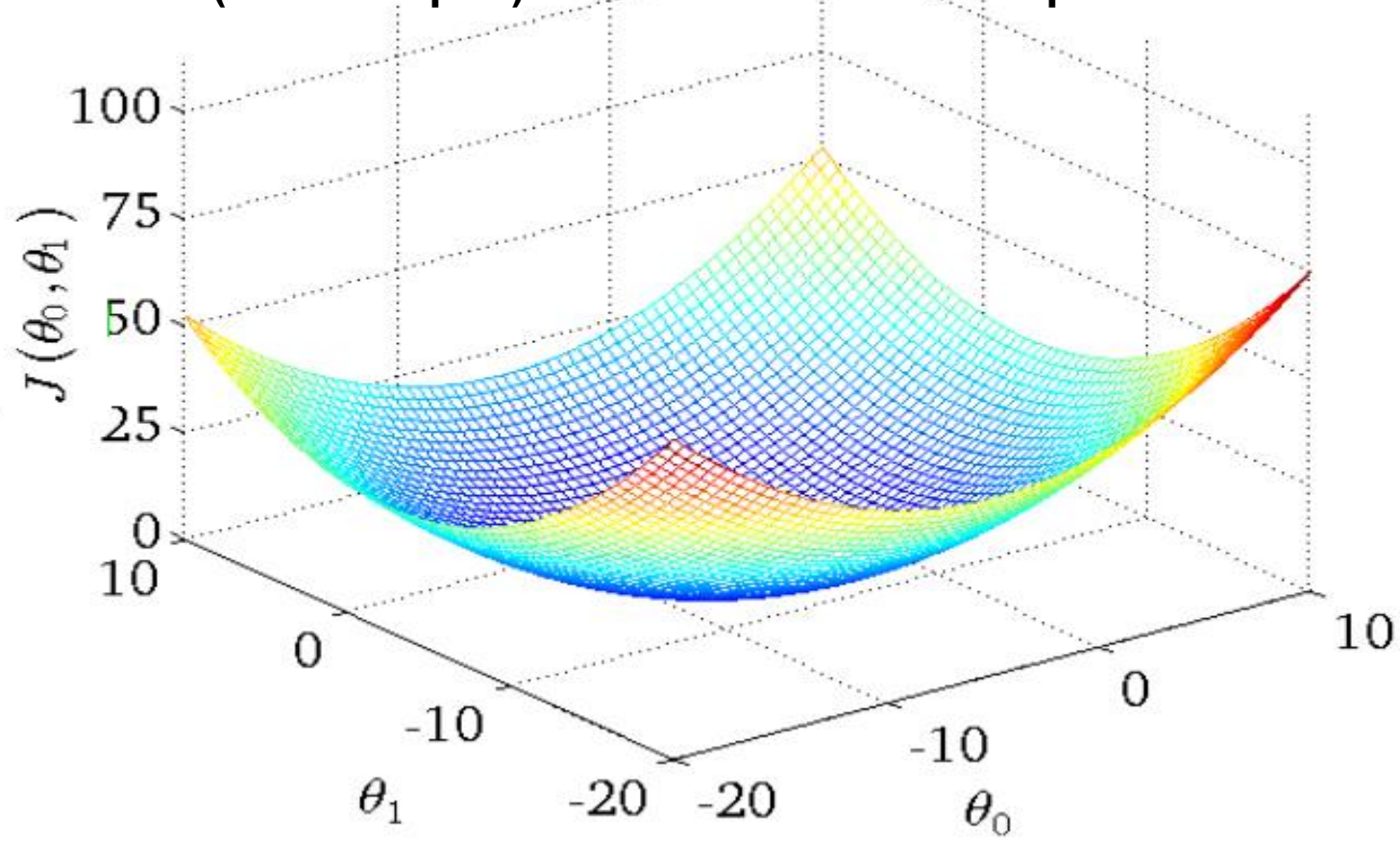
- Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear regression has only one global, and no other local, optima; thus gradient descent always converges (assuming the learning rate α is not too large) to the global minimum.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

Gradient Descent Algorithm



Gradient Descent Algorithm

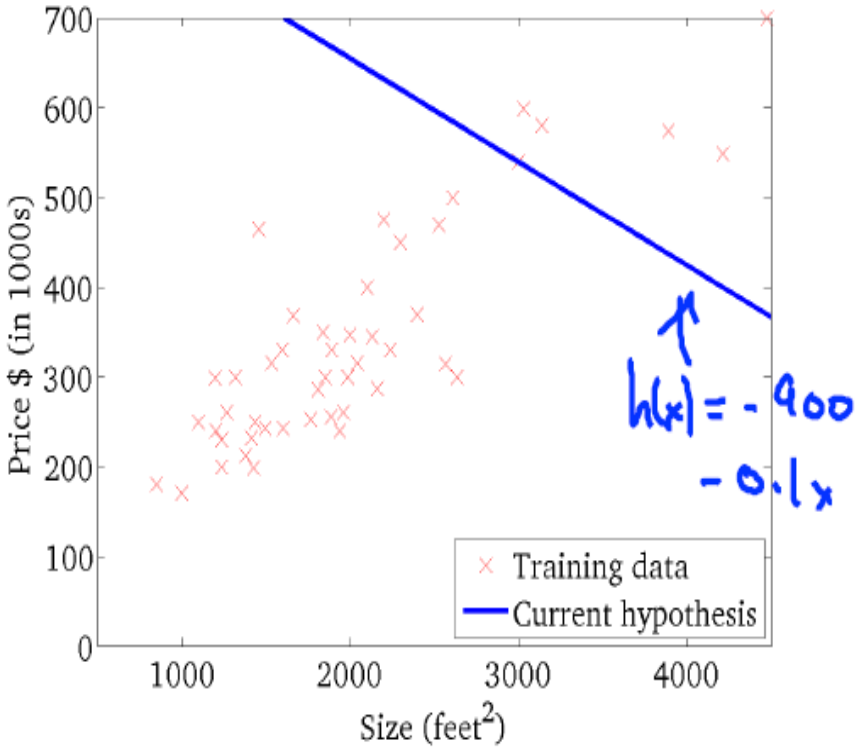
Cost Function for Linear Regression is always like "Convex Function (Bowl shaped)" No Local minima except Global minima.



Gradient Descent Algorithm

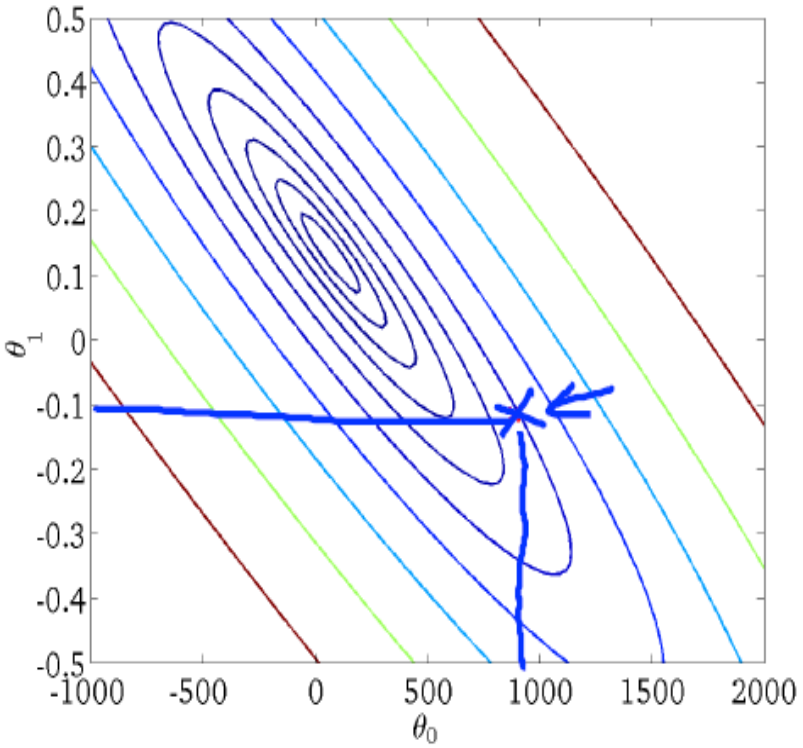
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

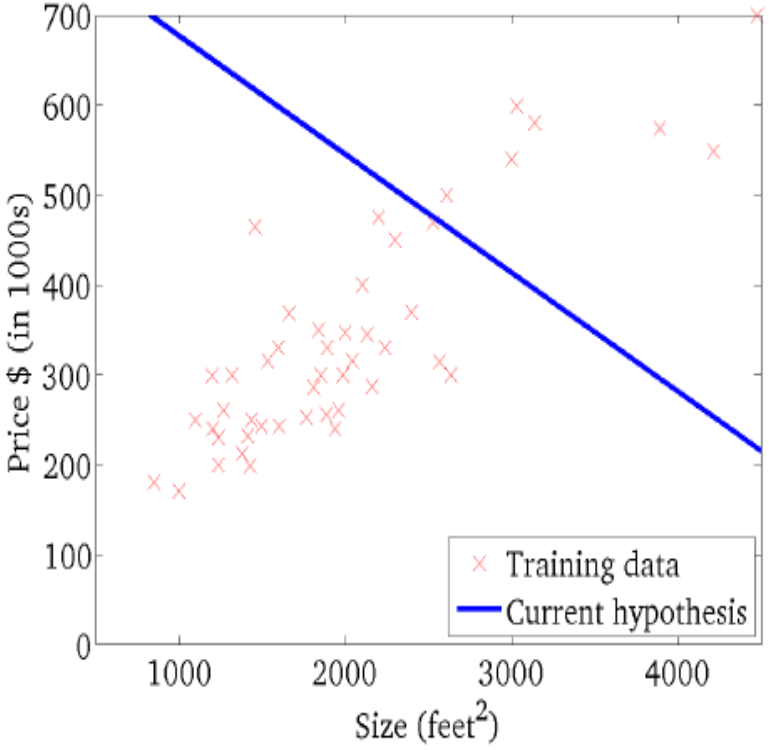
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

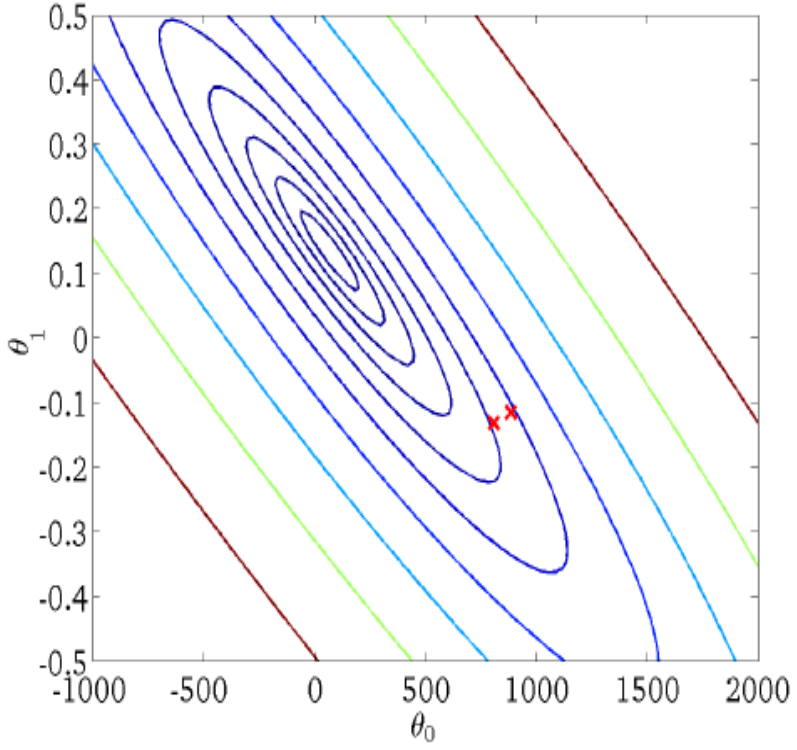
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

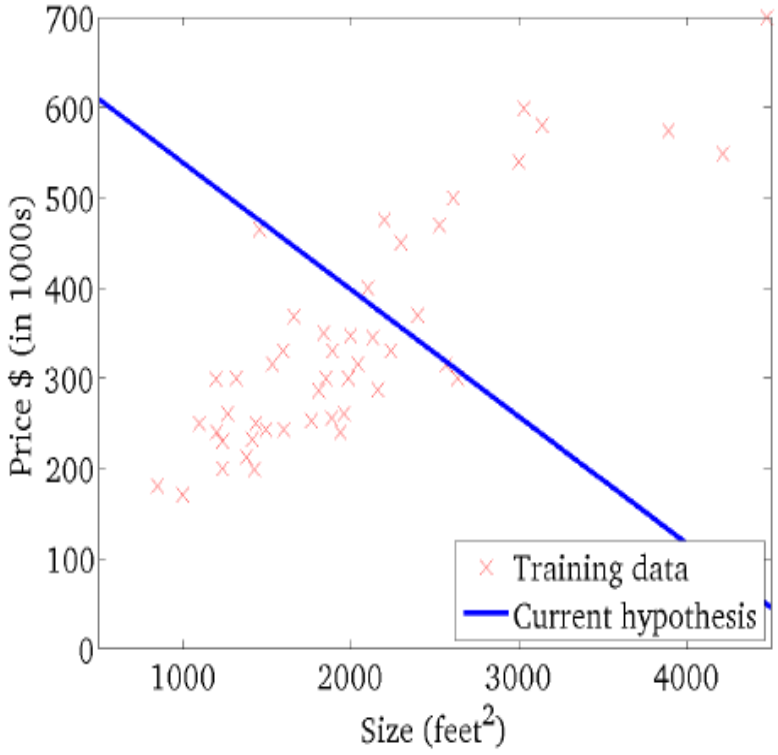
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

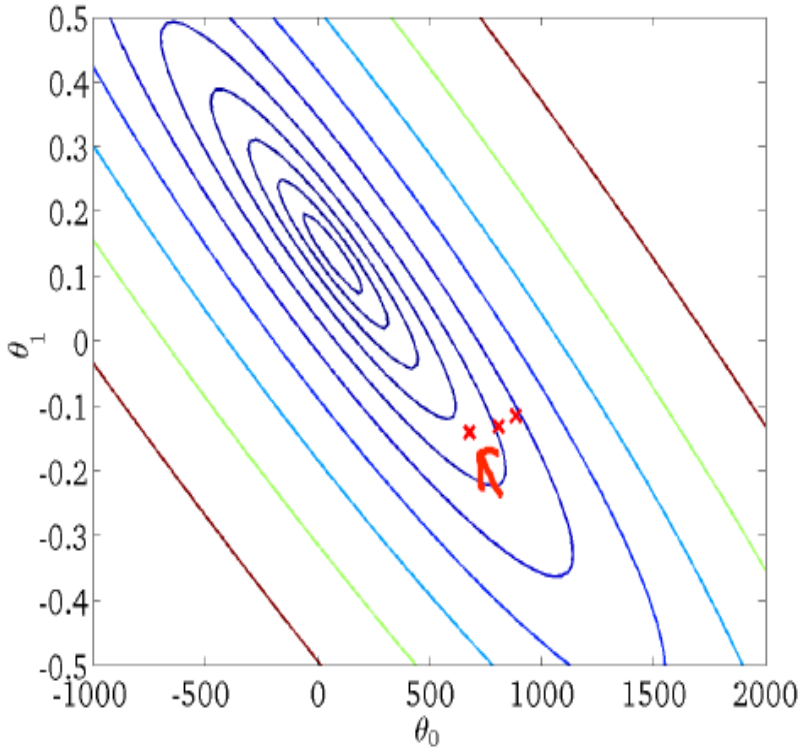
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

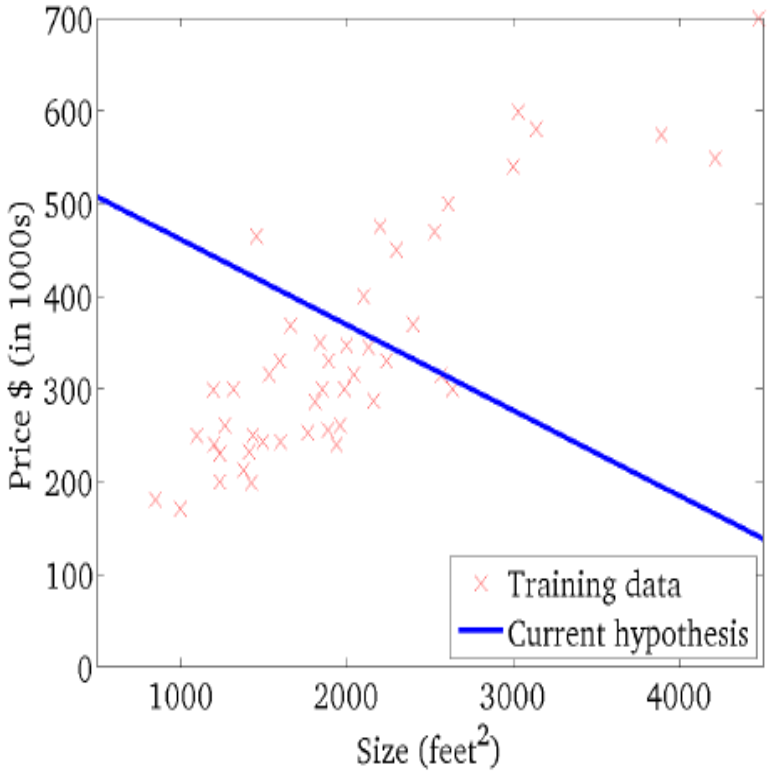
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

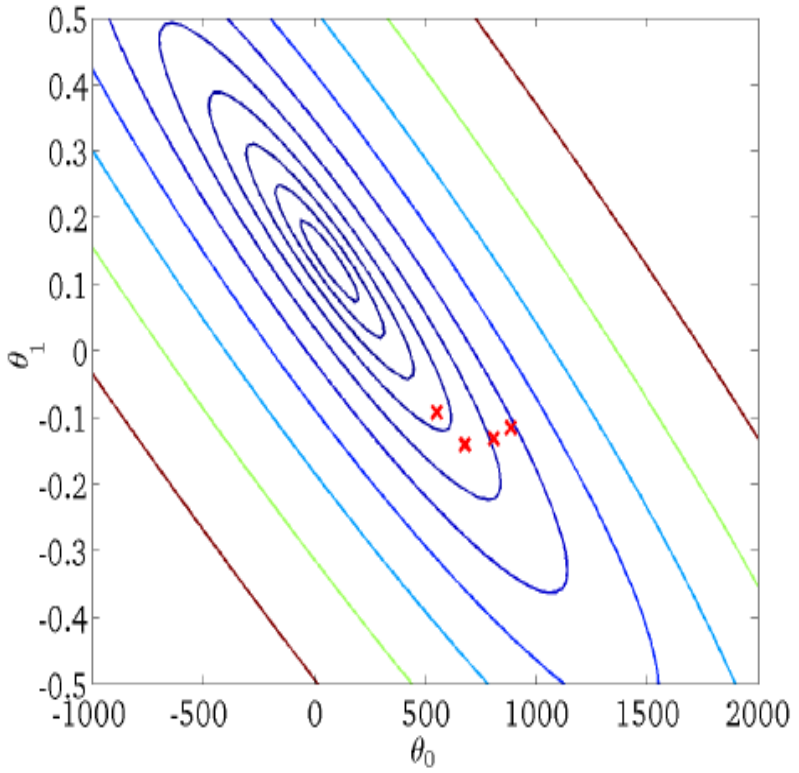
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

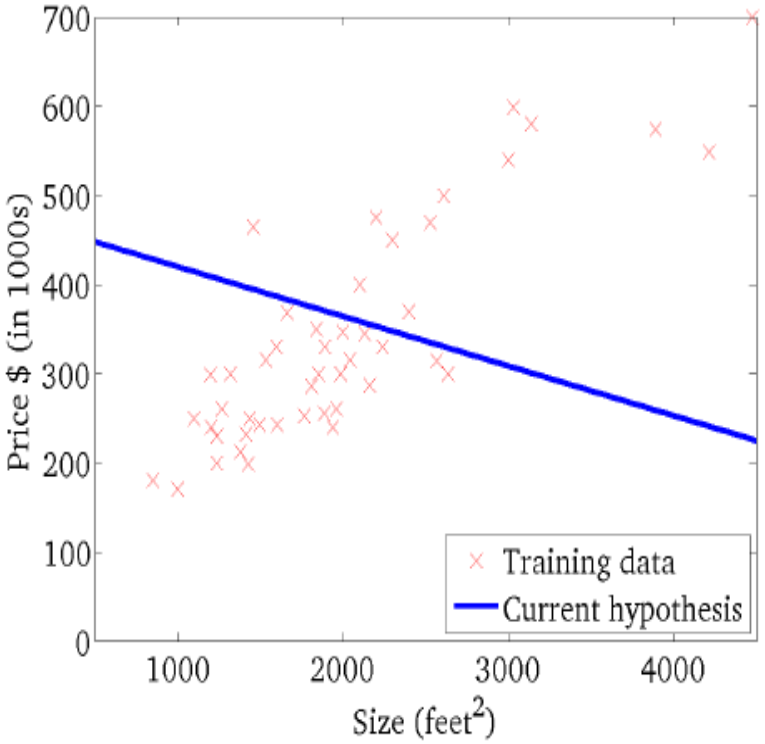
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

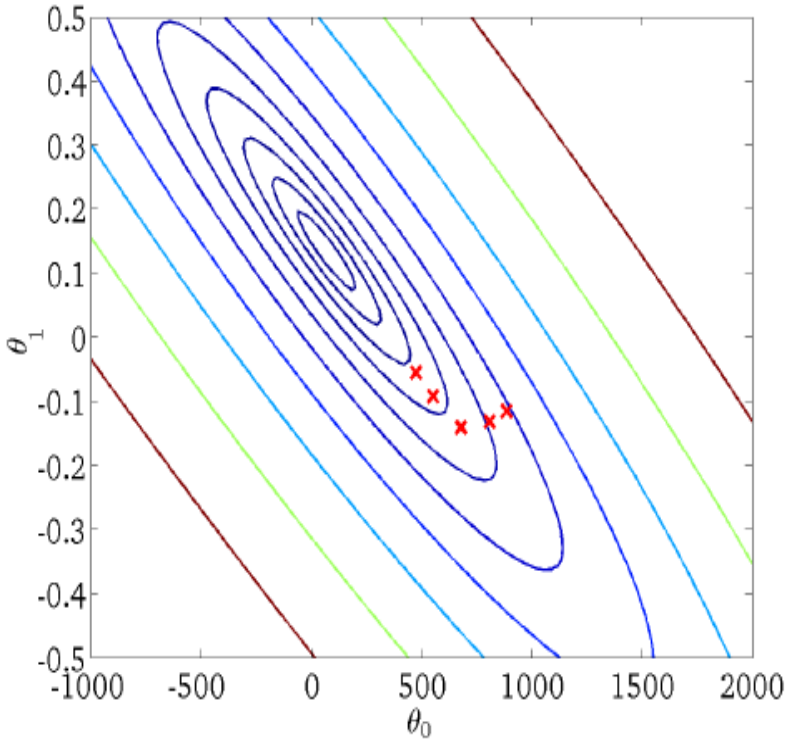
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

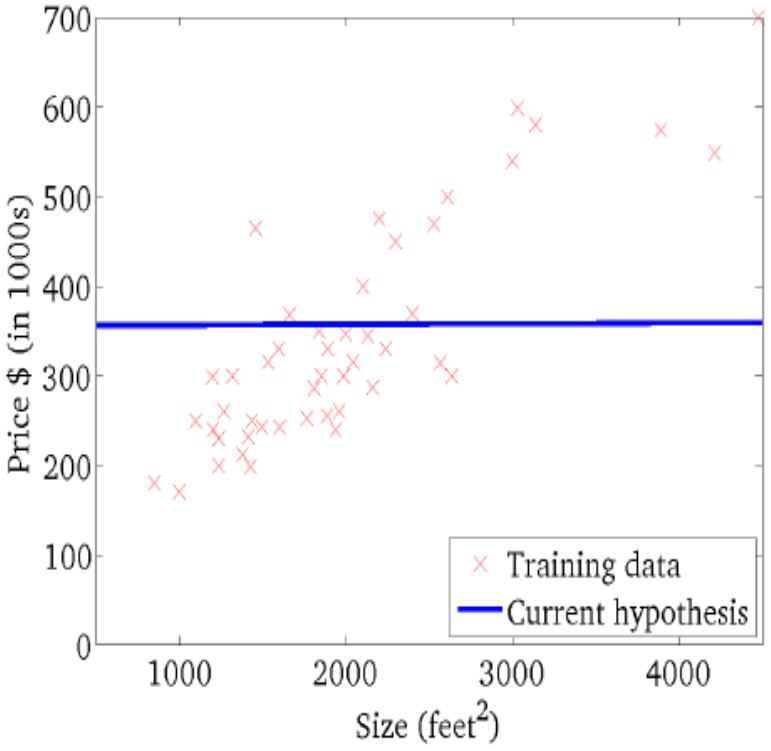
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

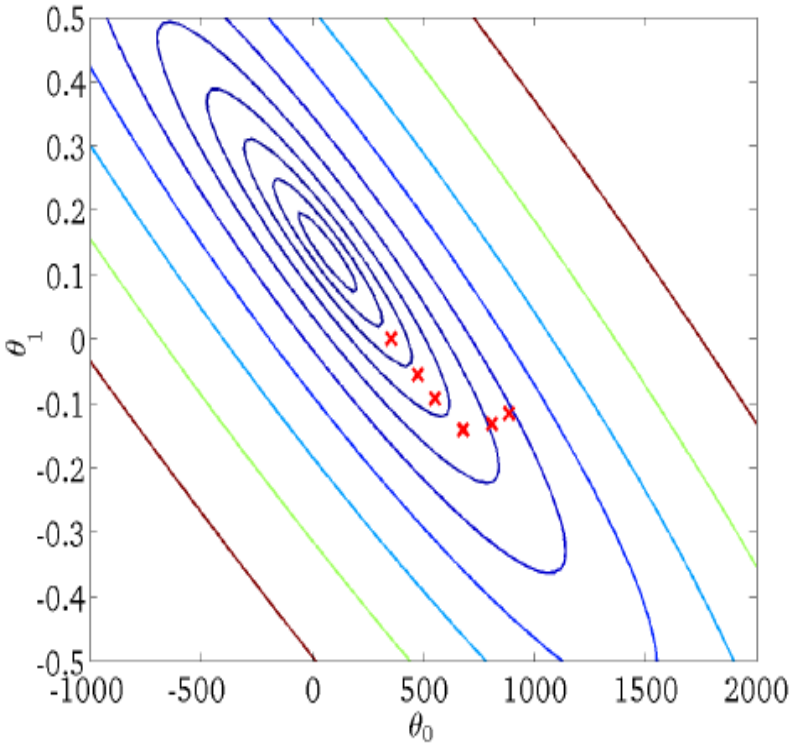
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

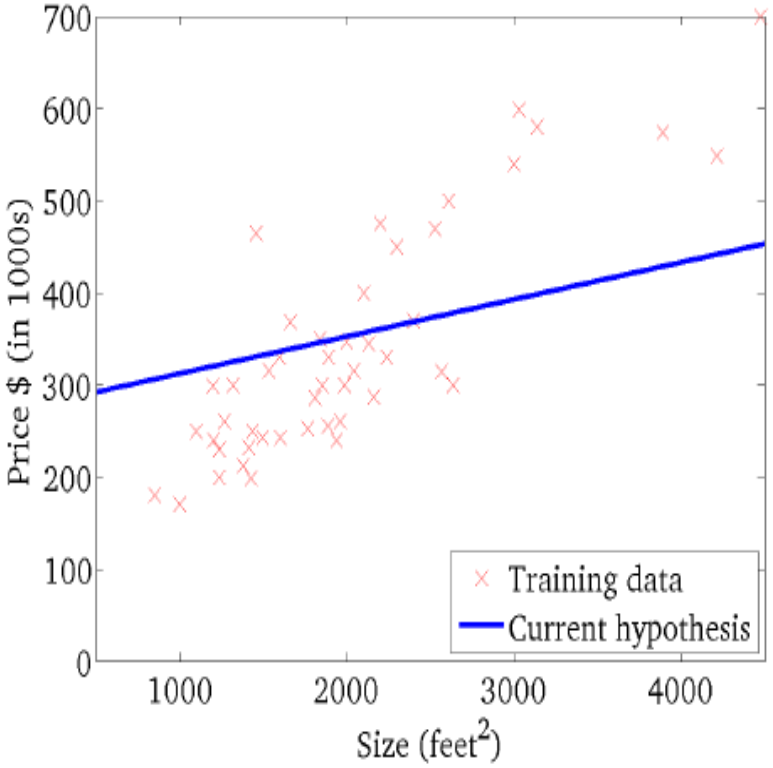
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

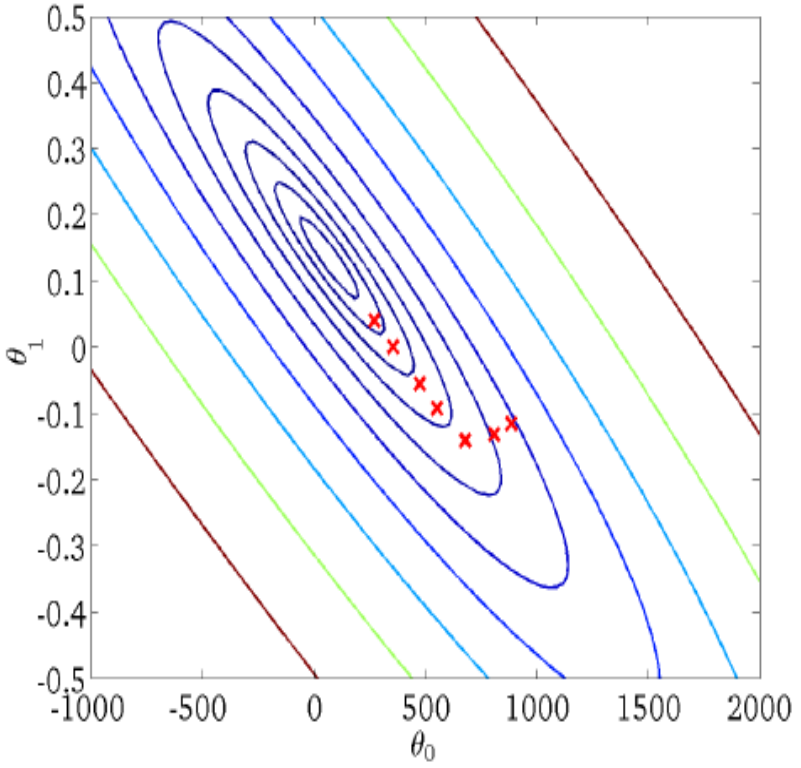
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

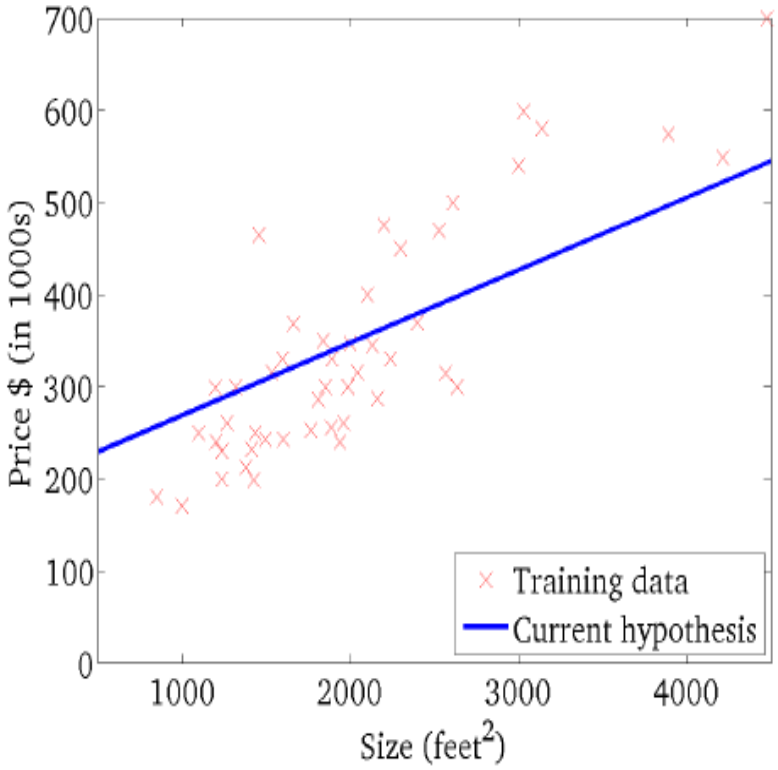
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

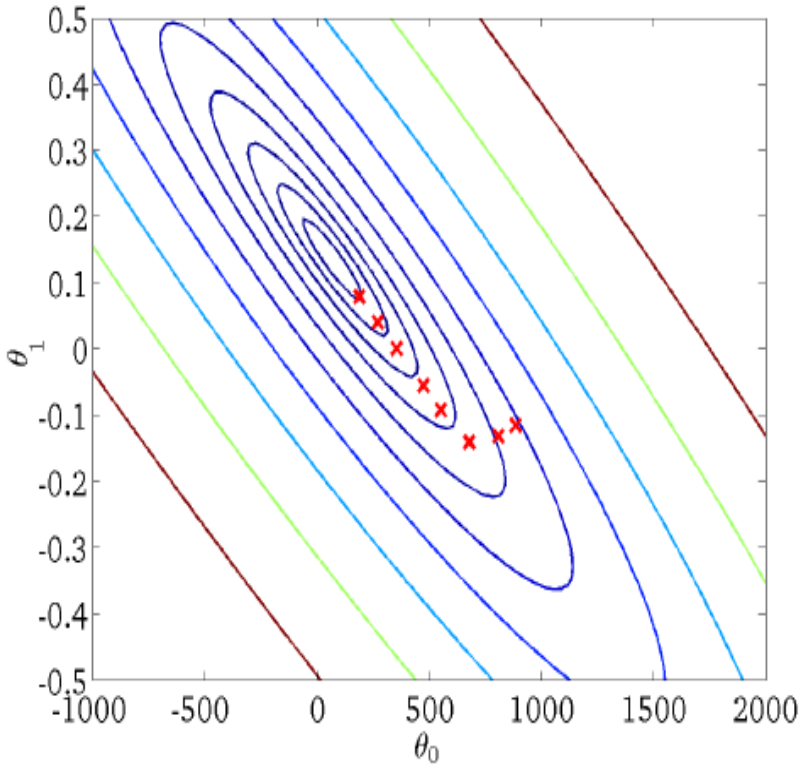
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

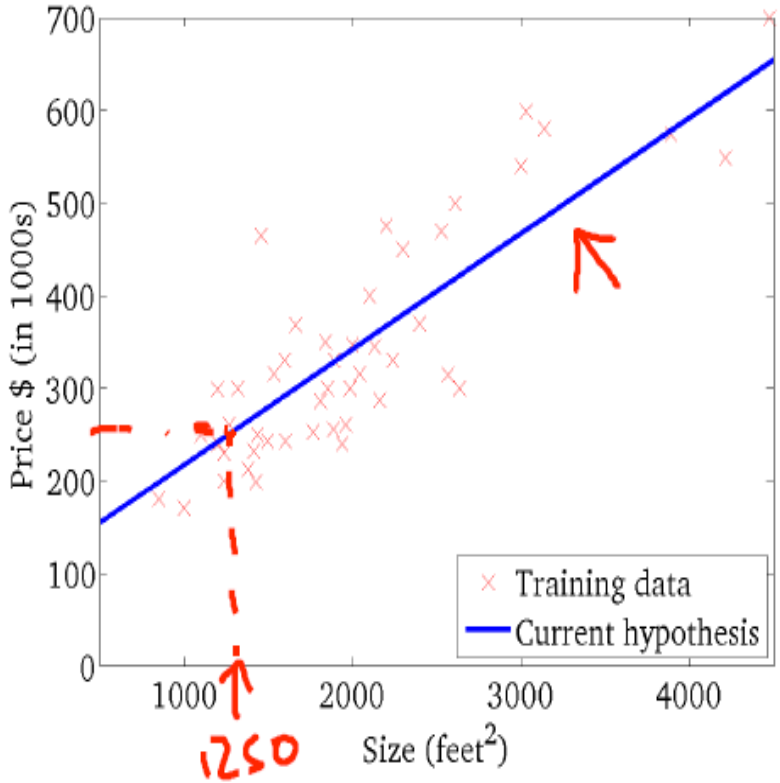
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

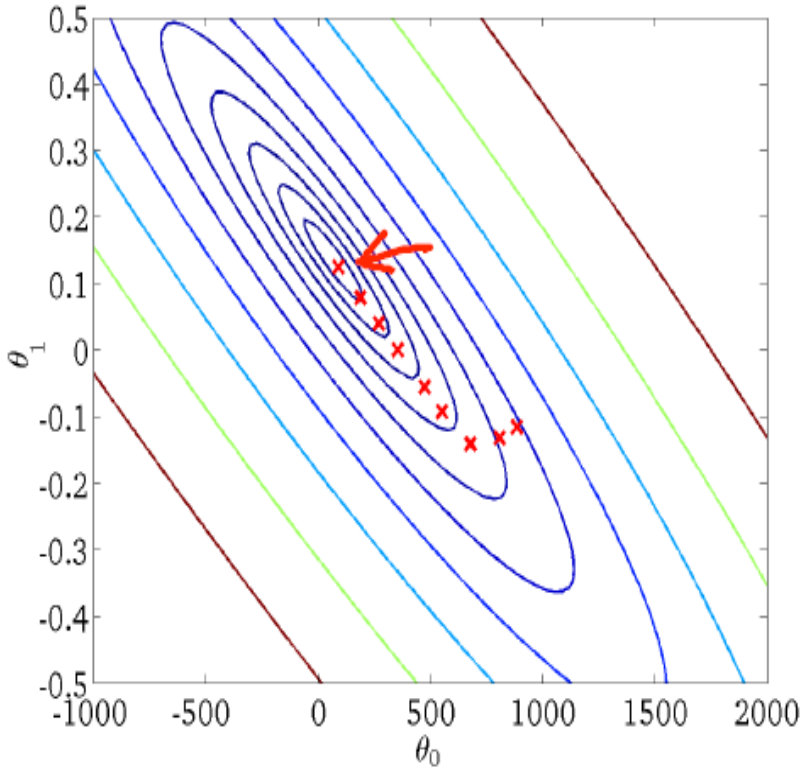
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

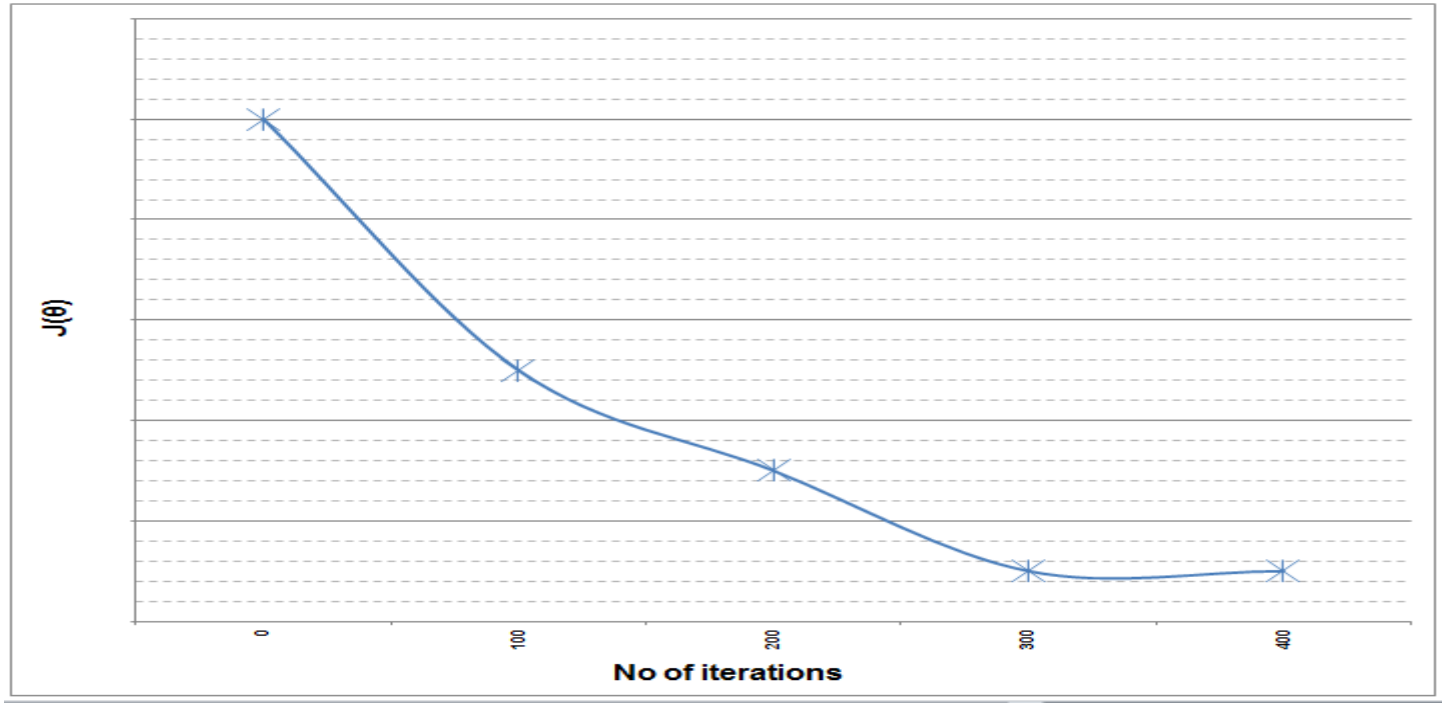
(function of the parameters θ_0, θ_1)



Gradient Descent Algorithm

- **Making sure gradient descent is working correctly**

- **Debugging gradient descent.** Make a plot with *number of iterations* on the x-axis. Now plot the cost function, J(θ) over the number of iterations of gradient descent. J(θ)should decreases after every iterations. If J(θ) ever increases, then you probably need to decrease α.

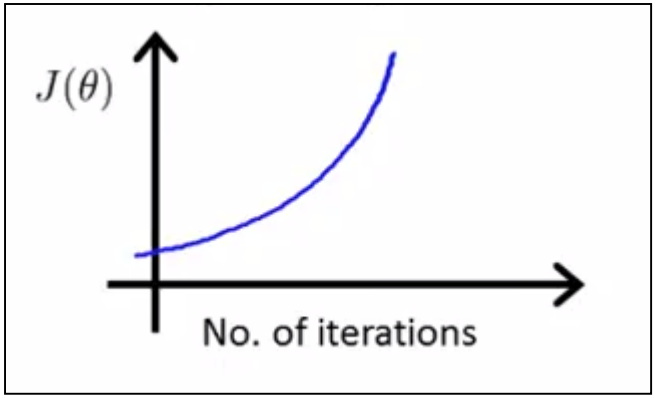


Gradient Descent Algorithm

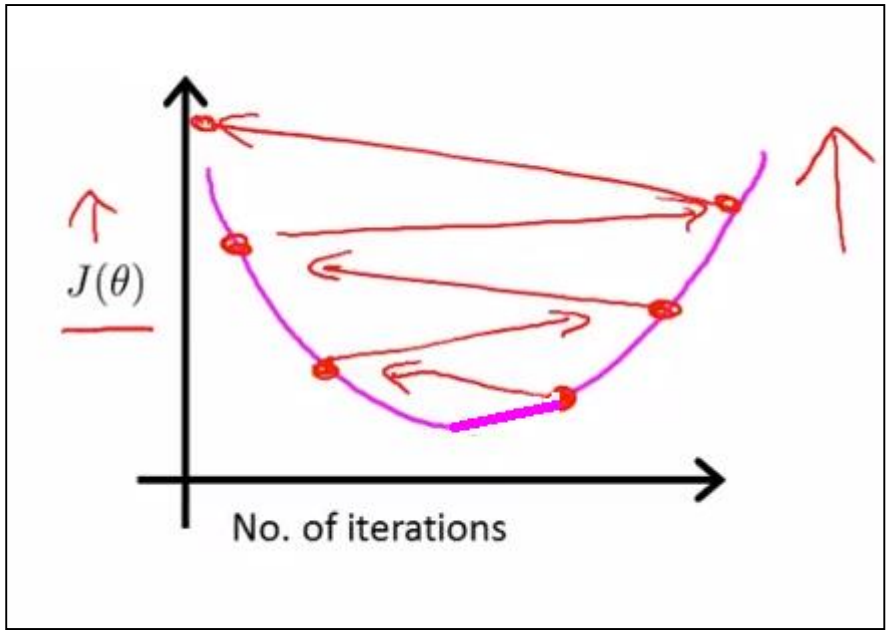
- **Automatic convergence test.** Declare convergence if J(θ) decreases by less than E in one iteration, where E is some small value such as 10^{-3}. However in practice it's difficult to choose this threshold value.

Gradient Descent Algorithm

- Gradient descent not working

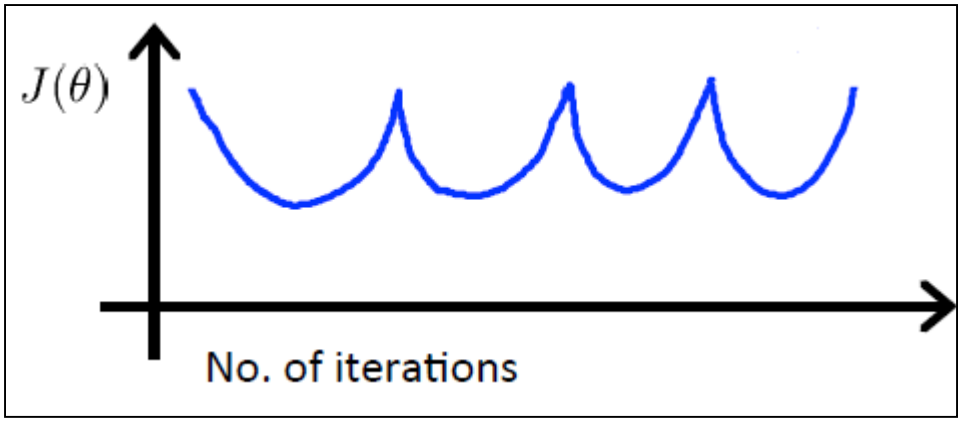


- Value of α is large.



Gradient Descent Algorithm

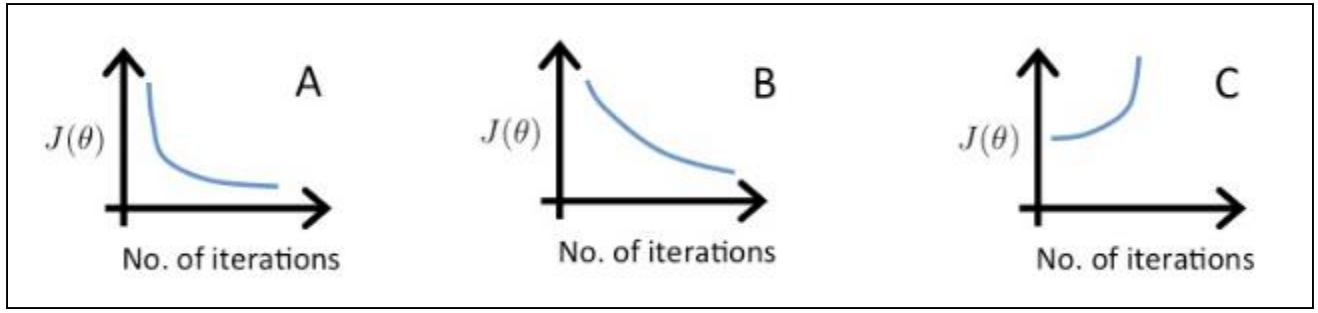
- Use smaller α



- Note:
- For sufficiently small α, J(θ) should decrease on every iteration.
- But if α is too small, gradient descent can be slow to converge.

Gradient Descent Algorithm

- **Question:**
- Suppose a friend ran gradient descent three times, with α=0.01, α=0.1, and α=1, and got the following three plots (labeled A, B, and C):



- Which plots corresponds to which values of α?
- (a) A is α=0.01, B is α=0.1, C is α=1.
- **(b) A is α=0.1, B is α=0.01, C is α=1. (Answer)**
- (c) A is α=1, B is α=0.01, C is α=0.1.
- (d) A is α=1, B is α=0.1, C is α=0.01.

Linear Algebra Review

Matrices and Vectors

Matrices and Vectors

- Matrices are 2-dimensional arrays:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{bmatrix}$$

- The above matrix has four rows and three columns, so it is a 4 x 3 matrix.

- A vector is a matrix with one column and many rows:

$$\begin{bmatrix} w \\ x \\ y \\ z \end{bmatrix}$$

- So vectors are a subset of matrices. The above vector is a 4 x 1 matrix.

Matrices and Vectors

- **Notation and terms**:
- *Aij* refers to the element in the ith row and jth column of matrix A.
- A vector with 'n' rows is referred to as an 'n'-dimensional vector.
- *vi* refers to the element in the ith row of the vector.
- In general, all our vectors and matrices will be 1-indexed. Note that for some programming languages, the arrays are 0-indexed.
- Matrices are usually denoted by uppercase names while vectors are lowercase.
- "Scalar" means that an object is a single value, not a vector or matrix.
- $\Re$ refers to the set of scalar real numbers.
- $\Re^n$ refers to the set of n-dimensional vectors of real numbers.

Addition

- Addition are **element-wise**, so you simply add each corresponding element:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} a+w & b+x \\ c+y & d+z \end{bmatrix}$$

- **Example:**

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

- To add two matrices, their dimensions must be **the same**.

Subtraction

- Subtraction are **element-wise**, so you simply subtract each corresponding element:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} - \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} a-w & b-x \\ c-y & d-z \end{bmatrix}$$

- **Example**

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A - B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} - \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 1-5 & 2-6 \\ 3-7 & 4-8 \end{bmatrix}$$

$$A + B = \begin{bmatrix} -4 & -4 \\ -4 & -4 \end{bmatrix}$$

- To subtract two matrices, their dimensions must be **the same**.

Scalar Multiplication

- In scalar multiplication, we simply multiply every element by the scalar value:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * x = \begin{bmatrix} a*x & b*x \\ c*x & d*x \end{bmatrix}$$

- **Example:**

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$2A = 2\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$2A = 2\begin{bmatrix} 1 \times 2 & 2 \times 2 \\ 3 \times 2 & 4 \times 2 \end{bmatrix}$$

$$2A = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}$$

Scalar Division

- In scalar division, we simply divide every element by the scalar value:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} /x = \begin{bmatrix} a/x & b/x \\ c/x & d/x \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

- **Example:**

$$\frac{1}{2}A = \frac{1}{2}\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$\frac{1}{2}A = \begin{bmatrix} 1 \times \frac{1}{2} & 2 \times \frac{1}{2} \\ 3 \times \frac{1}{2} & 4 \times \frac{1}{2} \end{bmatrix}$$

$$\frac{1}{2}A = \begin{bmatrix} \frac{1}{2} & 1 \\ \frac{3}{2} & 2 \end{bmatrix}$$

Matrix-Vector Multiplication

- We map the column of the vector onto each row of the matrix, multiplying each element and summing the result.

$$\begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a*x+b*y \\ c*x+d*y \\ e*x+f*y \end{bmatrix}$$

- The result is a **vector**.
- The number of **columns** of the matrix must equal the number of **rows** of the vector.
- An **m x n matrix** multiplied by an **n x 1 vector** results in an **m x 1 vector**.

Matrix-Vector Multiplication

- Example

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}, B = \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

$$AB = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

$$AB = \begin{bmatrix} 1 \times 7 + 2 \times 8 \\ 3 \times 7 + 4 \times 8 \\ 5 \times 7 + 6 \times 8 \end{bmatrix}$$

$$AB = \begin{bmatrix} 7 + 16 \\ 21 + 32 \\ 35 + 48 \end{bmatrix}$$

$$AB = \begin{bmatrix} 23 \\ 53 \\ 83 \end{bmatrix}$$

Matrix Multiplication

- We multiply two matrices by breaking it into several vector multiplications and concatenating the result.

$$\begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} * \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} a*w+b*y & a*x+b*z \\ c*w+d*y & c*x+d*z \\ e*w+f*y & e*x+f*z \end{bmatrix}$$

- An **m x n matrix** multiplied by an **n x o matrix** results in an **m x o** matrix.
- In the above example, a 3 x 2 matrix times a 2 x 2 matrix resulted in a 3 x 2 matrix.
- To multiply two matrices, the number of **columns** of the first matrix must equal the number of **rows** of the second matrix.

Matrix Multiplication

- **Example**

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$AB = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \times 5 + 2 \times 7 & 1 \times 6 + 2 \times 8 \\ 3 \times 5 + 4 \times 7 & 3 \times 6 + 4 \times 8 \end{bmatrix} = \begin{bmatrix} 5 + 14 & 6 + 16 \\ 15 + 28 & 18 + 32 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

Matrix Multiplication Properties

- Matrices are not commutative: $A*B \neq B*A$

- **Example**

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$AB = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \times 5 + 2 \times 7 & 1 \times 6 + 2 \times 8 \\ 3 \times 5 + 4 \times 7 & 3 \times 6 + 4 \times 8 \end{bmatrix} = \begin{bmatrix} 5 + 14 & 6 + 16 \\ 15 + 28 & 18 + 32 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

$$BA = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 5 \times 1 + 6 \times 3 & 5 \times 2 + 6 \times 4 \\ 7 \times 1 + 8 \times 3 & 7 \times 2 + 8 \times 4 \end{bmatrix} = \begin{bmatrix} 5 + 18 & 10 + 24 \\ 7 + 24 & 14 + 32 \end{bmatrix} = \begin{bmatrix} 23 & 34 \\ 31 & 46 \end{bmatrix}$$

- Hence proof, $A*B \neq B*A$

Matrix Multiplication Properties

- Matrices are associative: $(A*B)*C=A*(B*C)$
- $$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}, C = \begin{bmatrix} 9 & 10 \\ 11 & 12 \end{bmatrix}$$

$$A * B = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix} \qquad (A * B) * C = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix} \begin{bmatrix} 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 413 & 454 \\ 937 & 1030 \end{bmatrix}$$

$$B * C = \begin{bmatrix} 111 & 122 \\ 151 & 166 \end{bmatrix} \qquad A * (B * C) = \begin{bmatrix} 413 & 454 \\ 937 & 1030 \end{bmatrix}$$

- Hence proof, Matrices are associative: $(A*B)*C=A*(B*C)$

Matrix Multiplication Properties

- Matrices are Distributive: A*(B+C)=A*B+A*C

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}, C = \begin{bmatrix} 9 & 10 \\ 11 & 12 \end{bmatrix}$$

$$B + C = \begin{bmatrix} 14 & 16 \\ 18 & 20 \end{bmatrix}, A * (B + C) = \begin{bmatrix} 50 & 56 \\ 114 & 128 \end{bmatrix}$$

$$A * B = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}, A * C = \begin{bmatrix} 31 & 34 \\ 71 & 78 \end{bmatrix}, A * B + A * C = \begin{bmatrix} 50 & 56 \\ 114 & 128 \end{bmatrix}$$

Matrix Multiplication Properties

Identity Matrix

- The identity matrix simply has 1's on the diagonal (upper left to lower right diagonal) and 0's elsewhere.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Matrix Multiplication Properties

- The **identity matrix**, when multiplied by any matrix of the same dimensions, results in the original matrix. It's just like multiplying numbers by 1.

- When multiplying the identity matrix after some matrix (A∗I), the square identity matrix's dimension should match the other matrix's **columns**. When multiplying the identity matrix before some other matrix (I∗A), the square identity matrix's dimension should match the other matrix's **rows**.

$$A * I = I * A = A$$

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$AI = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$IA = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

- Hence Proof, $A * I = I * A = A$

Inverse and Transpose

- The **inverse** of a matrix A is denoted A^{-1}
- Multiplying by the inverse results in the identity matrix.
- A non square matrix does not have an inverse matrix.
- Matrices that don't have an inverse are *singular* or *degenerate*.
- Formula for **inverse** of a matrix A is;

$$A^{-1} = \frac{adj(A)}{|A|}$$

Inverse and Transpose

- **Calculate Determinants of a 2 X 2 Matrix:**
- **Formula;**

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\det(A) = det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\det(A) = ad - bc$$

- **Example;**

$$A = \begin{bmatrix} 4 & -2 \\ 1 & -3 \end{bmatrix}$$

$$\det(A) = det \begin{bmatrix} 4 & -2 \\ 1 & -3 \end{bmatrix}$$

$$\det(A) = (4 \times -3) - (-2 \times 1) = -12 - (-2) = -12 + 2 = -10$$

- $|A| \neq 0$, so A^{-1} solution is possible.

Inverse and Transpose

- **Calculate Adjoint of a 2 X 2 Matrix:**
- **Formula;**

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad adj(A) = adj \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad adj(A) = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

- **Example**

$$A = \begin{bmatrix} 4 & -2 \\ 1 & -3 \end{bmatrix}$$

$$\text{adj}(A) = ajd \begin{bmatrix} 4 & -2 \\ 1 & -3 \end{bmatrix}$$

$$adj(A) = \begin{bmatrix} -3 & 2 \\ -1 & 4 \end{bmatrix}$$

Inverse and Transpose

- **Calculate Inverse of a 2 X 2 Matrix:**

$$A^{-1} = \frac{adj(A)}{|A|}$$

- **By putting the values, we have**

$$A^{-1} = \frac{\begin{bmatrix} -3 & 2 \\ -1 & 4 \end{bmatrix}}{-10}$$

$$A^{-1} = \begin{bmatrix} \frac{-3}{-10} & \frac{2}{-10} \\ \frac{-1}{-10} & \frac{4}{-10} \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} \frac{3}{10} & \frac{-1}{5} \\ \frac{1}{10} & \frac{-2}{5} \end{bmatrix}$$

Inverse and Transpose

- **Calculate Determinants of a 3 X 3 Matrix:**
- **Formula;**

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

$$|A| = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix}$$

$$|A| = a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

$$|A| = a(ei - fh) - b(di - fg) + c(dh - eg)$$

Inverse and Transpose

- **Calculate Determinants of a 3 X 3 Matrix:**

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 2 & 3 \\ 1 & 2 & 1 \end{bmatrix}$$

$$|A| = \begin{vmatrix} 1 & 0 & 1 \\ 0 & 2 & 3 \\ 1 & 2 & 1 \end{vmatrix}$$

$$|A| = 1\begin{vmatrix} 2 & 3 \\ 2 & 1 \end{vmatrix} - 0\begin{vmatrix} 0 & 3 \\ 1 & 1 \end{vmatrix} + 1\begin{vmatrix} 0 & 2 \\ 1 & 2 \end{vmatrix}$$

$$|A| = 1(2 \times 1 - 2 \times 3) - 0(0 \times 1 - 3 \times 1) + 1(0 \times 2 - 2 \times 1)$$

$$|A| = 1(2 - 6) - 0(0 - 3) + 1(0 - 2)$$

$$|A| = 1(-4) - 0(-3) + 1(-2) = -4 + 0 - 2 = -6$$

- $|A| \neq 0$, so A^{-1} solution is possible.

Inverse and Transpose

- The **transposition** of a matrix is like rotating the matrix 90° in clockwise direction and then reversing it.
- In other words: $A_{ij} = A^T_{ji}$

$$A = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix}$$

$$A^T = \begin{bmatrix} a & c & e \\ b & d & f \end{bmatrix}$$

- **Example:**

$$A = \begin{bmatrix} 1 & -2 & -5 \\ 4 & 3 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 1 & -2 & -5 \\ 4 & 3 & 0 \\ -1 & -1 & -1 \end{bmatrix}^T$$

$$A^T = \begin{bmatrix} 1 & 4 & -1 \\ -2 & 3 & -1 \\ -5 & 0 & -1 \end{bmatrix}$$

Inverse and Transpose

- **Calculate Adjoint of a 3 X 3 Matrix:**
- **Formula;**

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- **Step-1(Calculate cof A)**

$$Cof(A) = \begin{bmatrix} +\begin{vmatrix} e & f \\ h & i \end{vmatrix} & -\begin{vmatrix} d & f \\ g & i \end{vmatrix} & +\begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ -\begin{vmatrix} b & c \\ h & i \end{vmatrix} & +\begin{vmatrix} a & c \\ g & i \end{vmatrix} & -\begin{vmatrix} a & b \\ g & h \end{vmatrix} \\ +\begin{vmatrix} b & c \\ e & f \end{vmatrix} & -\begin{vmatrix} a & c \\ d & f \end{vmatrix} & +\begin{vmatrix} a & b \\ d & e \end{vmatrix} \end{bmatrix}$$

$$Cof(A) = \begin{bmatrix} +(ei - fh) & -(di - fg) & +(dh - eg) \\ -(bi - ch) & +(ai - cg) & -(ah - bg) \\ +(bf - ce) & -(af - cd) & +(ae - bd) \end{bmatrix}$$

Inverse and Transpose

- **Step-2 (adj(A))**
- **Formula;**
- $adj(A) = [Cof(A)]^T$
- By putting the values, we have

$$adj(A) = \begin{bmatrix} +(ei - fh) & -(di - fg) & +(dh - eg) \\ -(bi - ch) & +(ai - cg) & -(ah - bg) \\ +(bf - ce) & -(af - cd) & +(ae - bd) \end{bmatrix}^T$$

$$adj(A) = \begin{bmatrix} +(ei - fh) & -(bi - ch) & +(bf - ce) \\ -(di - fg) & +(ai - cg) & -(af - cd) \\ +(dh - eg) & -(ah - bg) & +(ae - bd) \end{bmatrix}$$

Inverse and Transpose

- **Calculate Adjoint of a 3 X 3 Matrix:**
- **Example:**

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 2 & 3 \\ 1 & 2 & 1 \end{bmatrix}$$

- **Step-1(Calculate cof A)**

$$Cof(A) = \begin{bmatrix} +\begin{vmatrix} 2 & 3 \\ 2 & 1 \end{vmatrix} & -\begin{vmatrix} 0 & 3 \\ 1 & 1 \end{vmatrix} & +\begin{vmatrix} 0 & 2 \\ 1 & 2 \end{vmatrix} \\ -\begin{vmatrix} 0 & 1 \\ 2 & 1 \end{vmatrix} & +\begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} & -\begin{vmatrix} 1 & 0 \\ 1 & 2 \end{vmatrix} \\ +\begin{vmatrix} 0 & 1 \\ 2 & 3 \end{vmatrix} & -\begin{vmatrix} 1 & 1 \\ 0 & 3 \end{vmatrix} & +\begin{vmatrix} 1 & 0 \\ 0 & 2 \end{vmatrix} \end{bmatrix}$$

$$Cof(A) = \begin{bmatrix} +(2-6) & -(0-3) & +(0-2) \\ -(0-2) & +(1-1) & -(2-0) \\ +(0-2) & -(3-0) & +(2-0) \end{bmatrix}$$

$$Cof(A) = \begin{bmatrix} -4 & 3 & -2 \\ 2 & 0 & -2 \\ -2 & -3 & 2 \end{bmatrix}$$

Inverse and Transpose

- **Step-2 (adj(A))**
- **Formula;**
- $adj(A) = [Cof(A)]^T$
- By putting the values, we have

$$adj(A) = \begin{bmatrix} -4 & 3 & -2 \\ 2 & 0 & -2 \\ -2 & -3 & 2 \end{bmatrix}^T$$

$$adj(A) = \begin{bmatrix} -4 & 2 & -2 \\ 3 & 0 & -3 \\ -2 & -2 & 2 \end{bmatrix}$$

Inverse and Transpose

- **Calculate Inverse of a 3 X 3 Matrix:**

$$A^{-1} = \frac{adj(A)}{|A|}$$

- By putting the values, we have

$$A^{-1} = \begin{bmatrix} \frac{-4}{-6} & \frac{2}{-6} & \frac{-2}{-6} \\ \frac{3}{-6} & \frac{0}{-6} & \frac{-3}{-6} \\ \frac{-2}{-6} & \frac{-2}{-6} & \frac{2}{-6} \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} \frac{2}{3} & \frac{-1}{3} & \frac{1}{3} \\ \frac{-1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & \frac{-1}{3} \end{bmatrix}$$

Linear Regression

Linear Regression with Multiple Variables

Linear Regression with Multiple Variables

Plot Size (x)	Price (y)
1650	45
1700	27
1836	32.5
1905	62
1950	32
1950	49
…	…
…	…
…	

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

Linear Regression with Multiple Variables

Sr no	Price	Plotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw	airco	garagepl	prefarea
1	42000	5850	3	1	2	yes	no	yes	no	no	1	no
2	38500	4000	2	1	1	yes	no	no	no	no	0	no
3	49500	3060	3	1	1	yes	no	no	no	no	0	no
4	60500	6650	3	1	2	yes	yes	no	no	no	0	no
5	61000	6360	2	1	1	yes	no	no	no	no	0	no
6	66000	4160	3	1	1	yes	yes	yes	no	yes	0	no
7	66000	3880	3	2	2	yes	no	yes	no	no	2	no
8	69000	4160	3	1	3	yes	no	no	no	no	0	no
9	83800	4800	3	1	1	yes	yes	yes	no	no	0	no
10	88500	5500	3	2	4	yes	yes	no	no	yes	1	no

Notation

m= Number of training examples=546

X's= "input" variable/features = Plot Size

y''s= "output" variable/"target" variable=Price

(X,y)- Single training example

$(X^{(i)},y^{(i)})$- i^{th} training example, Superscript i is not the Exponential, its the index.

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

$n=|X^{(i)}|$; (the number of features)

Linear Regression with Multiple Variables

- **Example:**

Size (feet)^2 (X1)	Number of bedrooms(X2)	Number of floors (X3)	Age of home (years) (X4)	Price ($1000) (y)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

- In the training set above, what is $x_1^{(4)}$?
- (a) The size (in feet^2) of the 1st home in the training set.
- (b) The age (in years) of the 1st home in the training set.
- **(c) The size (in feet^2) of the 4th home in the training set. (answer)**
- (d) The age (in years) of the 4th home in the training set

Linear Regression with Multiple Variables

- **Example 2:**
- In the training set above, what is $x^{(2)}$?
- **Answer**

$$\begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$$

Linear Regression with Multiple Variables

- The Hypothesis Function (for one variable): $h_{(\theta)}(x) = \theta_0 + \theta_1 x$
- The Hypothesis Function (for multiple variable): $h_{(\theta)}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \text{----} + \theta_n x_n$
- For convenience of notation, define $x_0 = 1$

$$X = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x^n \end{bmatrix}, \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta^n \end{bmatrix}, X \; and \theta \; are \; n+1 \text{ dimension } vectors$$

- The Hypothesis Function (for multiple variable): $h_{(\theta)}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \text{----} + \theta_n x_n$

$$h_{(\theta)}(x) = \theta^T{}_X$$

- This is also called Multivariate linear regression.

Linear Regression with Multiple Variables

Gradient Descent for multiple variables

Gradient Descent for Multiple Variables

- **Hypothesis:** $h(x) = \theta^T X = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 - - - - + \theta_n x_n$
- **Parameters:** $\theta, where \theta$ is the n+1 dimension $vector$
- **Cost Function:**

$$J(\theta\ ) = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2 = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x_i) - y_i)^2$$

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: {

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

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

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

$\dots$

}

- In other words;

repeat until convergence: {

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

}

Gradient Descent for Multiple Variables

- The following image compares gradient descent with one variable to gradient descent with multiple variables:

Previously (n=1):

Repeat $\{$

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

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

(simultaneously update θ_0, θ_1)

$\}$

Gradient Descent for Multiple Variables

New algorithm $(n \geq 1)$:

Repeat $\{$

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

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

$\}$

repeat until convergence: {

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

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

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

...

}

Gradient Descent for Multiple Variables

- **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(θ)?

- [ ] $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})^2$
- [ ] $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\sum_{j=0}^{n}\theta_j x_j^{(i)}\right) - y^{(i)}\right)^2$ (Inner sum starts at 0)
- [ ] $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\sum_{j=1}^{n}\theta_j x_j^{(i)}\right) - y^{(i)}\right)^2$ (Inner sum starts at 1)
- [ ] $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\sum_{j=0}^{n}\theta_j x_j^{(i)}\right) - \left(\sum_{j=0}^{n} y_j^{(i)}\right)\right)^2$

Answers

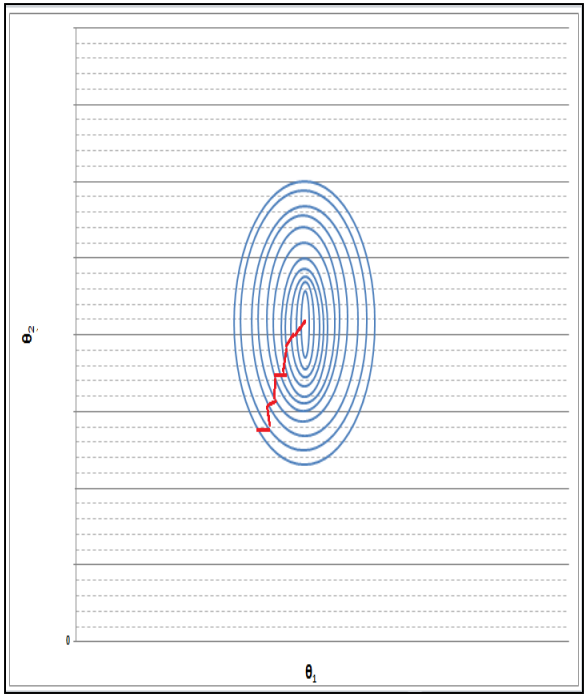
Linear Regression with Multiple Variables

Gradient Descent in Practice

Feature Scaling

- **Idea:** Make sure features are on a similar scale. If features are in the similar scale then Gradient Descent converge quickly.

- E.g. $x_1 = size(0 - 2000 feet^2), x_2 = number\ of\ bedrooms(1 - 5)$

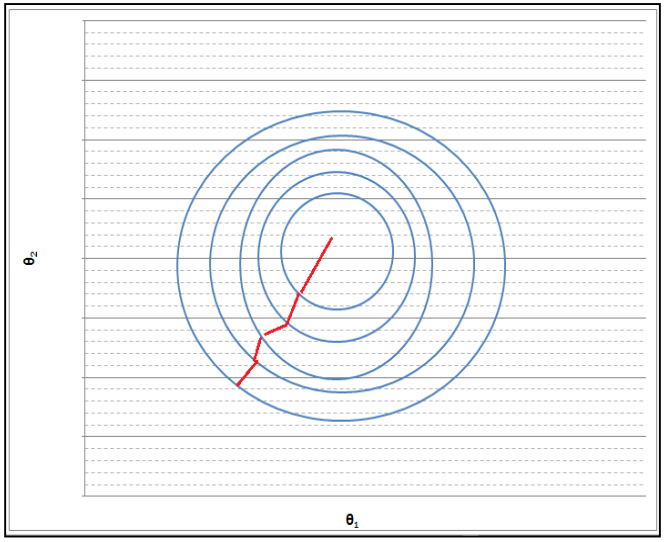


Feature Scaling

$$x_1 = size(0 - 2000 feet^2), x_2 = number\ of\ bedrooms(1 - 5)$$

$$x_1 = \frac{size(feet^2)}{2000}, x_2 = \frac{number\ of\ bedrooms}{5}$$

$$0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1$$



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.

Feature Scaling

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

$$-1 \leq x_i \leq 1 \text{ or } -0.5 \leq x_i \leq 0.5$$

- 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, give or take a few

Range	Comments
$0 \leq x_1 \leq 3$	OK
$-2 \leq x_2 \leq 0.5$	OK
$-100 \leq x_3 \leq 100$	Not OK
$-0.0001 \leq x_4 \leq 0.0001$	Not OK
$-3 \leq x_5 \leq 3$	OK
$-\frac{1}{3} \leq x_6 \leq \frac{1}{3}$	Ok

- 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

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

- 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:

$$x_i = \frac{x_i - \mu_i}{range}$$

$$range = max - min$$

Or

$$x_i = \frac{x_i - \mu_i}{\sigma}$$

$$\mu_1 = average\ of\ x_i\ in\ the\ training\ test$$

$$\sigma = \text{standard deviation}$$

Mean Normalization

- The formula for standard deviation (SD) is;

$$\sigma = \sqrt{\sum \frac{|x - \mu|^2}{N}}$$

- **Example (using Range)**

$$x_1 = size(0 - 2000 feet^2), x_2 = number\ of\ bedrooms(1 - 5)$$

$$range_1 = 2000, range_2 = 4$$

$$\mu_0 = 1000, \mu_1 = 3$$

$$-0.5 \leq x_1 \leq 0.5\ , -0.5 \leq x_2 \leq 0.5$$

Mean Normalization

- **Example (using standard deviation)**

$$x_1 = size(0 - 2000 feet^2), x_2 = number\ of\ bedrooms(1 - 5)$$

$$\mu_0 = 1000, \mu_1 = 3$$

$$\sigma_1 = 577.63, \sigma_2 = 1.41$$

$$-1.73 \leq x_1 \leq 1.73\ , -1.41 \leq x_2 \leq 1.41$$

Mean Normalization

- Suppose you are using a learning algorithm to estimate the price of houses in a city. You want one of your features *xi* 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?

$$(a) x_i = age\ of\ the\ house$$

$$(b) x_i = \frac{age\ of\ the\ house}{50}$$

$$(c) x_i = \frac{age\ of\ the\ house - 38}{50}$$

$$(d) x_i = \frac{age\ of\ the\ house - 38}{20}$$

- Answer (d)

Linear Regression with Multiple Variables

Features Polynomial Regression

Features

- 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 $x_1 * x_2$

- **Example (Housing Prices Prediction):**
- Let we have two features; $x_1 = frontage$, $x_2 = depth$

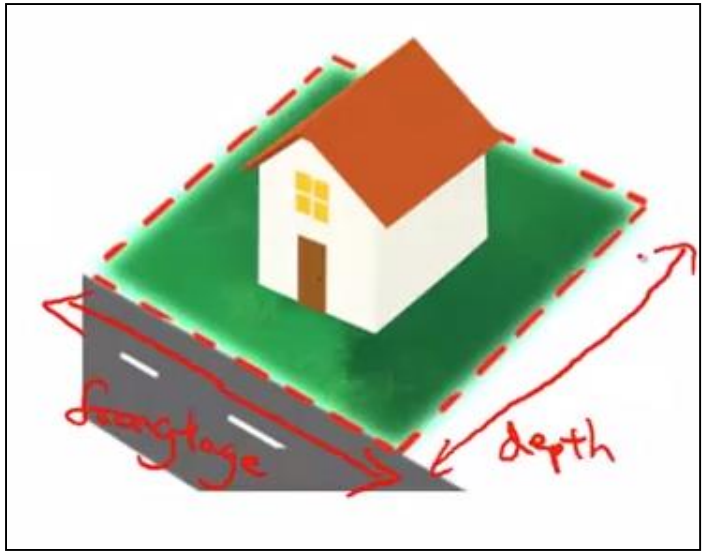
$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$h_\theta(x) = \theta_0 + \theta_1 \times frontage + \theta_2 \times depth$$

$$Area = frontage \times depth = x_3$$

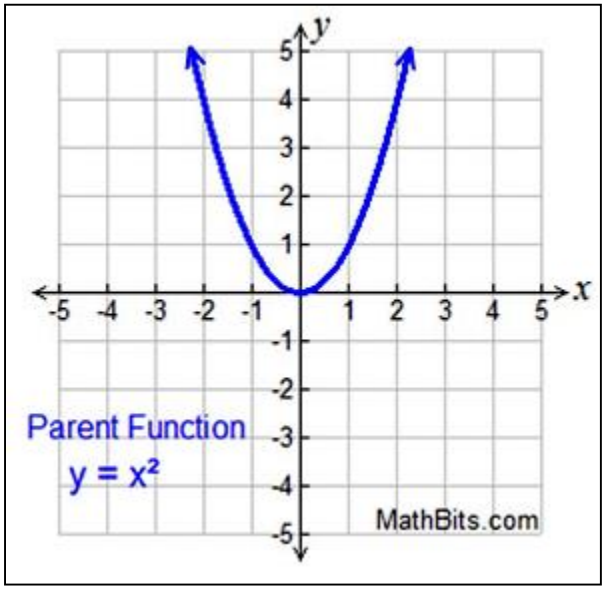
$$h_\theta(x) = \theta_0 + \theta_1 \times Area$$

$$h_\theta(x) = \theta_0 + \theta_1 \times x_3$$



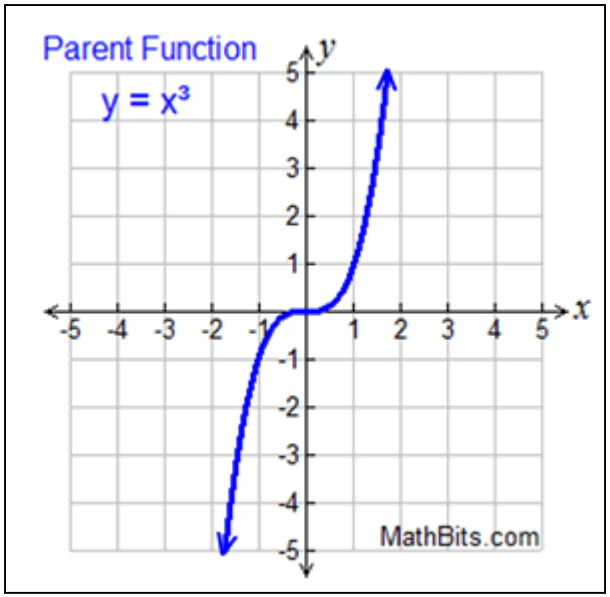
Polynomial Regression

- Our hypothesis function need not be linear (a straight line) if that does not fit the data well.
- We can **change the behavior or curve** of our hypothesis function by making it a quadratic, cubic or square root function (or any other form).
- **Quadratic Function Shape:**



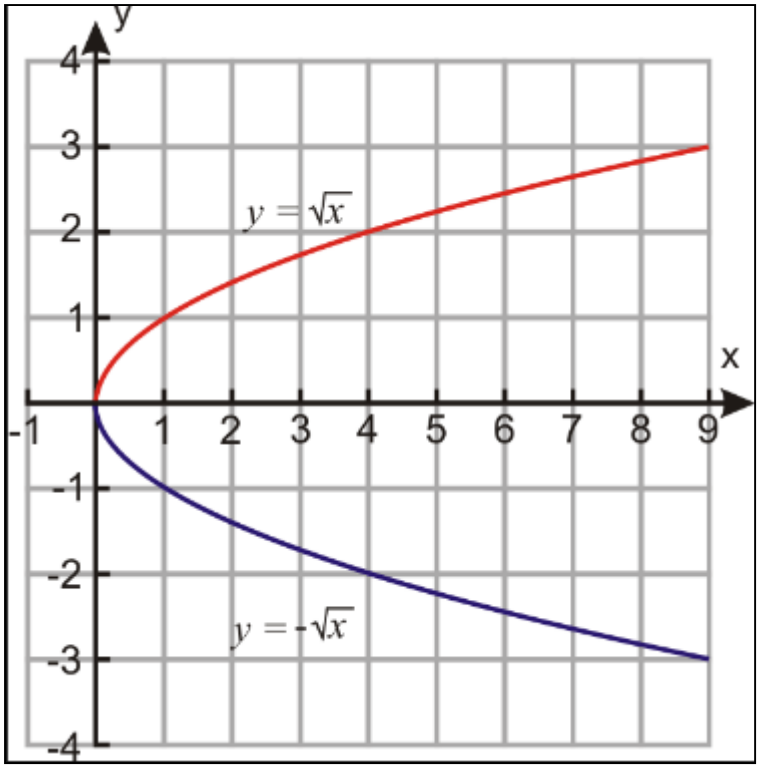
Polynomial Regression

- **Cubic Function Shape:**



Polynomial Regression

- **square root function Shape:**



Polynomial Regression

- 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;

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^3$$

- To make it a square root function, we could do:

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

Polynomial Regression

- One important thing to keep in mind is, if you choose your features this way then feature scaling becomes very important.

- e.g. if x_1 has rage 1-1000 then rage of $x_1^2 = 1 - 10^6$ $and\ that\ of\ x_1^3 = 1 - 10^9$

Polynomial Regression

- **Question**
- Suppose you want to predict a house's price as a function of its size. Your model is;

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

- Suppose size ranges from 1 to 1000 (feet^2). 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: $\sqrt{1000} \approx 32$)

- $x_1 = \text{size},\ x_2 = 32\sqrt{(\text{size})}$
- $x_1 = 32(\text{size}),\ x_2 = \sqrt{(\text{size})}$
- $x_1 = \frac{\text{size}}{1000},\ x_2 = \frac{\sqrt{(\text{size})}}{32}$ — Answer
- $x_1 = \frac{\text{size}}{32},\ x_2 = \sqrt{(\text{size})}.$

Linear Regression with Multiple Variables

Computing Parameters Analytically

Normal Equation

- Gradient descent gives one way of minimizing J.
- Let's discuss a second way of doing so, this time performing the minimization explicitly and without resorting to an iterative algorithm.
- "Normal equation Method" to solve for θ analytically.
- In the "Normal Equation" method, we will minimize J by explicitly taking its derivatives with respect to the θj 's, and setting them to zero.
- This allows us to find the optimum theta without iteration.
- Formula for Normal equation is;

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

Normal Equation

Size (feet)^2 (X1)	Number of bedrooms(X2)	Number of floors (X3)	Age of home (years) (X4)	Price ($1000) (y)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

(X0)	Size (feet)^2 (X1)	Number of bedrooms(X2)	Number of floors (X3)	Age of home (years) (X4)	Price ($1000) (y)
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

Normal Equation

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}, y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

- Now we can solve the normal equation using above matrices.

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

- There is **no need** to do feature scaling with the normal equation.

Gradient Descent-Vs- Normal Equation

- The following is a comparison of gradient descent and the normal equation:

Gradient Descent	Normal Equation
Need to choose alpha	No need to choose alpha
Needs many iterations	No need to iterate
O (kn^2)	O (n^3), need to calculate inverse of $X^T X$
Works well when n is large	Slow if n is very large

- With the normal equation, computing the inversion has complexity O(n^3).
- So if we have a very large number of features, the normal equation will be slow.
- In practice, when n exceeds 10,000 it might be a good time to go from a normal solution to an iterative process.

Normal Equation

- **Question**
- Suppose you have the training in the table below:

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

- You would like to predict a child's weight as a function of his age and height with the model;

$$h_\theta(x) = \theta_0 + \theta_1 \times age + \theta_2 \times height$$

- What are *X* and *y*?

Normal Equation

- $X = \begin{bmatrix} 4 & 89 \\ 9 & 124 \\ 5 & 103 \end{bmatrix}, y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$
- $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}, y = \begin{bmatrix} 1 & 16 \\ 1 & 28 \\ 1 & 20 \end{bmatrix}$
- $X = \begin{bmatrix} 4 & 89 & 1 \\ 9 & 124 & 1 \\ 5 & 103 & 1 \end{bmatrix}, y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$
- **(selected)** $X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}, y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$

Answer

Linear Regression with Multiple Variables

Normal Equation and non-invertibility

Normal Equation Non-invertibility:

- **Normal Equation Non-invertibility:**
- Formula for Normal equation is; $$\theta = (X^T X)^{-1} X^T y$$
- What if $X^T X$ is non-invertible? (Singular or degenerate matrix)
- A square **matrix** that is not invertible is called singular or **degenerate**.
- A square **matrix** is singular if and only if its determinant is 0.

Normal Equation Non-invertibility:

- What if $X^T X$ is non-invertible, the common causes might be having :
- Redundant features, where two features are very closely related (i.e. they are linearly dependent)

$$e.g. x_1 = size\ in\ feet^2\ , x_2 = size\ in\ m^2$$

- Too many features (e.g. m ≤ n). In this case, delete some features or use "regularization“
- 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.

Reference

- https://www.coursera.org/