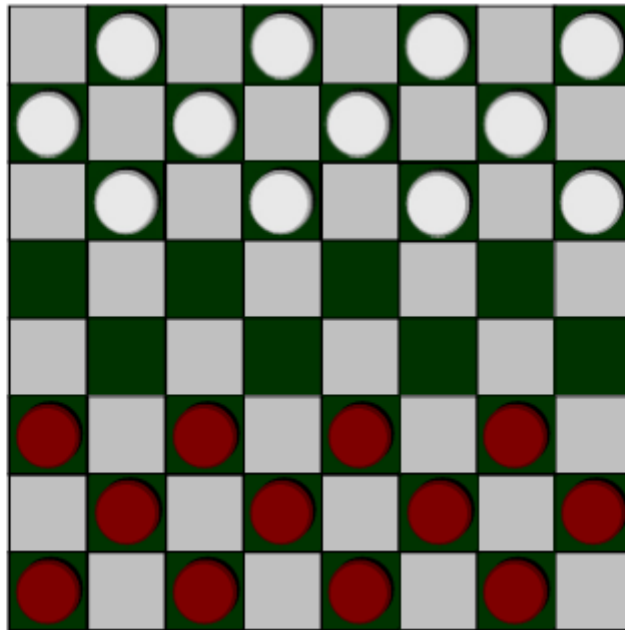


An abstract graphic on the left side of the slide, consisting of a network of white lines and small circles on a blue gradient background. The lines are vertical and horizontal, with some diagonal segments, and the circles are of varying sizes, creating a circuit-like or neural network aesthetic.

MACHINE LEARNING

Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.



Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.
- Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to *learn* from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E .

"A computer program is said to *learn* from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task T in this setting?

- ☐ Classifying emails as spam or not spam.
- ☐ Watching you label emails as spam or not spam.
- ☐ The number (or fraction) of emails correctly classified as spam/not spam.
- ☐ None of the above—this is not a machine learning problem.

"A computer program is said to *learn* from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task T in this setting?



- ☒ Classifying emails as spam or not spam. $T \leftarrow$
- ☐ Watching you label emails as spam or not spam. $E \leftarrow$
- ☐ The number (or fraction) of emails correctly classified as spam/not spam.
- ☐ None of the above—this is not a machine learning problem. $P \leftarrow$



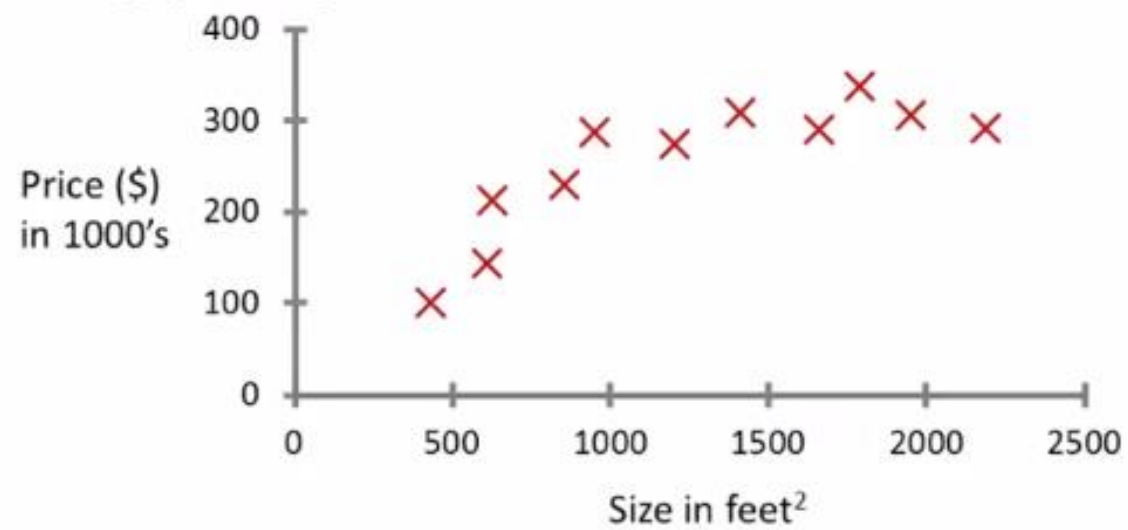
Machine learning algorithms:

- Supervised learning
- Unsupervised learning

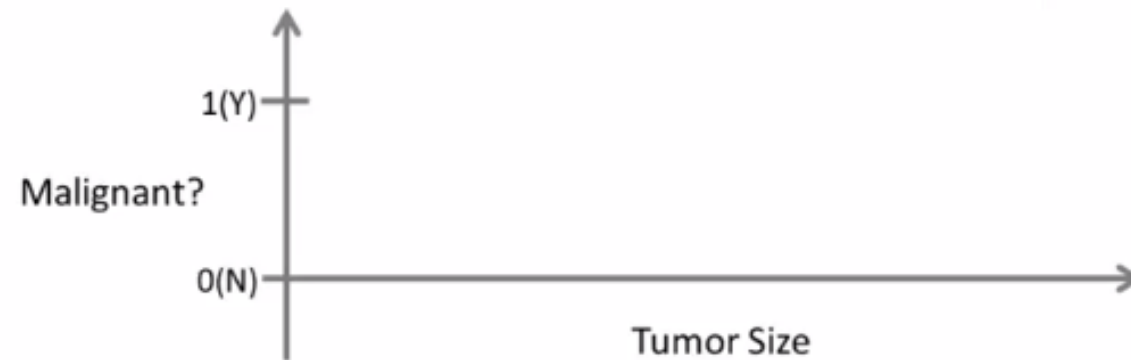
Others: Reinforcement learning, recommender systems.



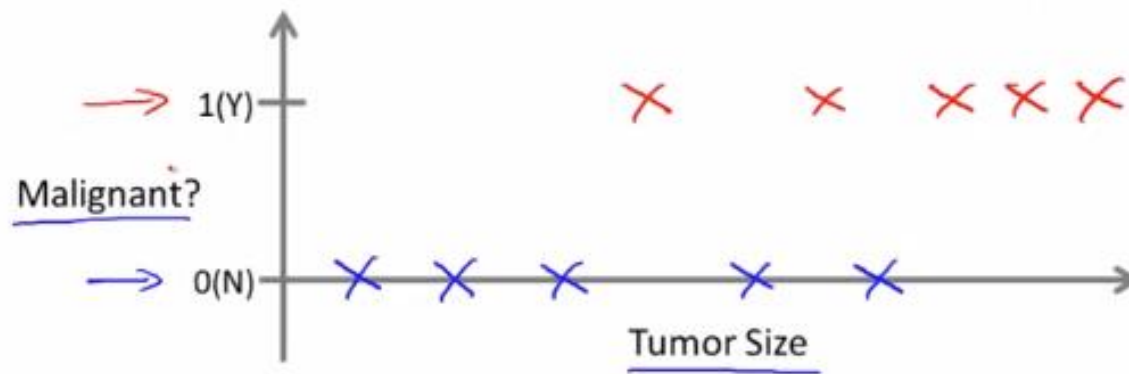
Housing price prediction.

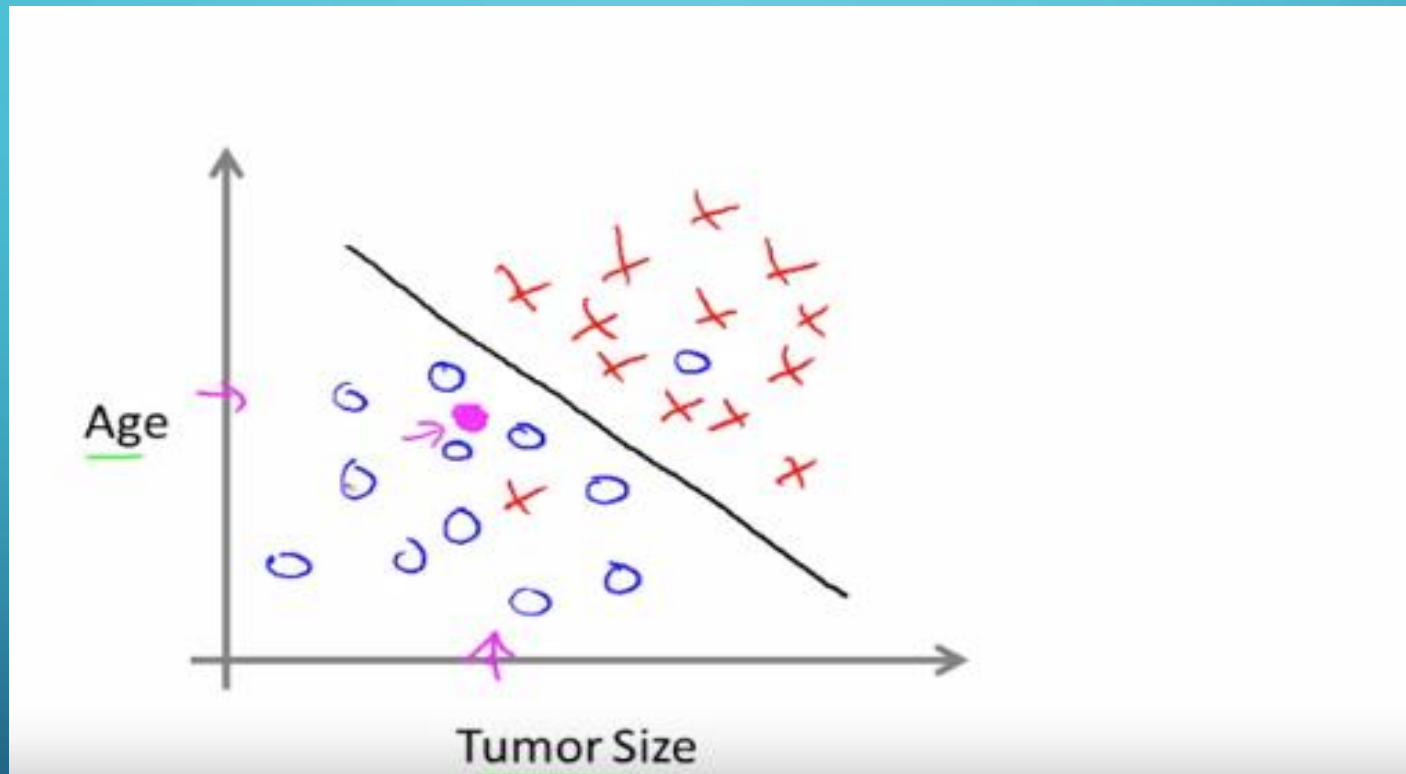


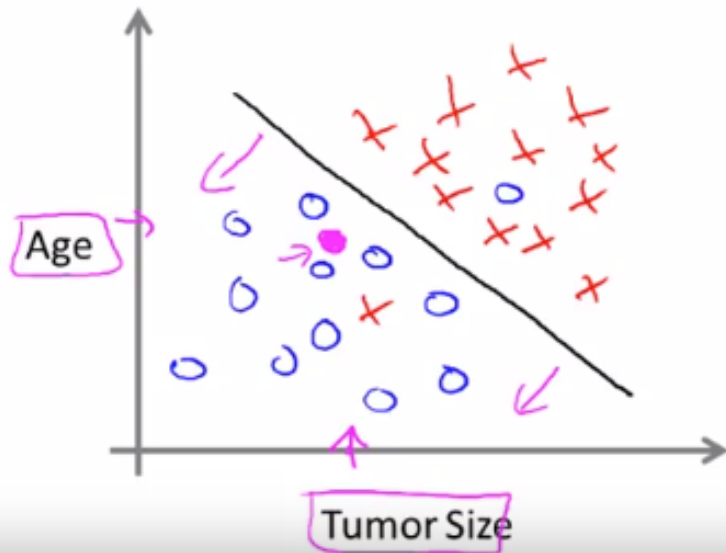
Breast cancer (malignant, benign)



Breast cancer (malignant, benign)







- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape
- ...

You're running a company, and you want to develop learning algorithms to address each of two problems.

Problem 1: You have a large inventory of identical items. You want to predict how many of these items will sell over the next 3 months.

Problem 2: You'd like software to examine individual customer accounts, and for each account decide if it has been hacked/compromised.

Should you treat these as classification or as regression problems?

- ☐ Treat both as classification problems.
- ☐ Treat problem 1 as a classification problem, problem 2 as a regression problem.
- ☐ Treat problem 1 as a regression problem, problem 2 as a classification problem.
- ☐ Treat both as regression problems.

Supervised Learning

In supervised learning, we are given a data set and already know what our correct output should look like, having the idea that there is a relationship between the input and the output.

Supervised learning problems are categorized into "regression" and "classification" problems. In a regression problem, we are trying to predict results within a continuous output, meaning that we are trying to map input variables to some continuous function. In a classification problem, we are instead trying to predict results in a discrete output. In other words, we are trying to map input variables into discrete categories.

Example 1:

Given data about the size of houses on the real estate market, try to predict their price. Price as a function of size is a continuous output, so this is a regression problem.

We could turn this example into a classification problem by instead making our output about whether the house "sells for more or less than the asking price." Here we are classifying the houses based on price into two discrete categories.

Example 2:

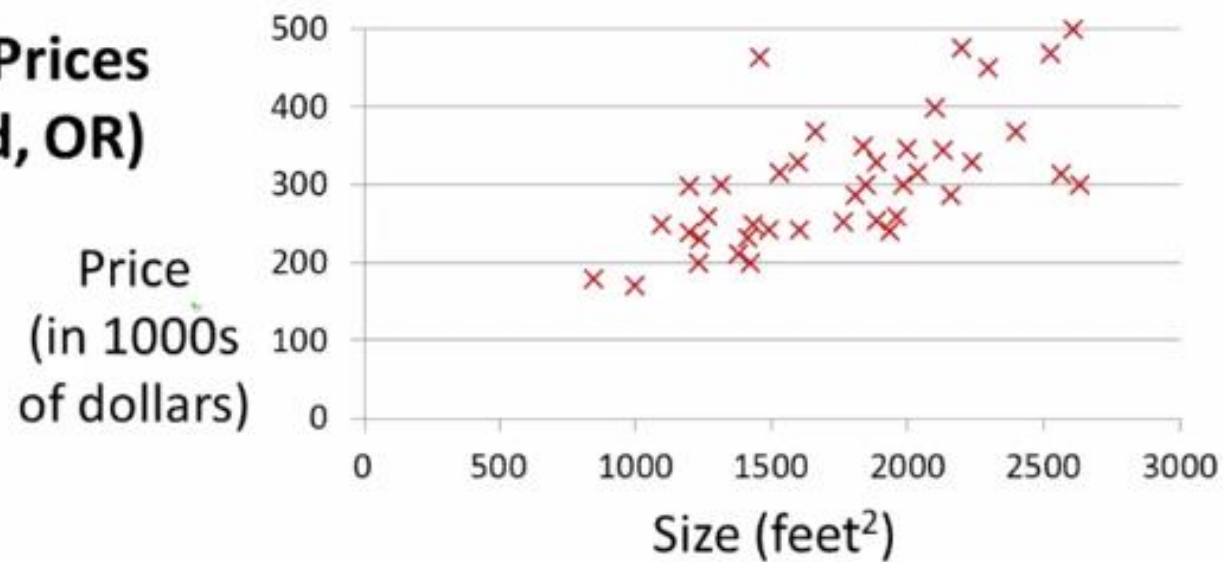
(a) Regression - Given a picture of a person, we have to predict their age on the basis of the given picture

(b) Classification - Given a patient with a tumor, we have to predict whether the tumor is malignant or benign.

LINEAR REGRESSION

- Simple Linear Regression
- Multivariate Linear Regression
- Polynomial Regression

Housing Prices (Portland, OR)



**Training set of
housing prices
(Portland, OR)**

Size in feet² (x)

Price (\$) in 1000's (y)

2104

460

1416

232

1534

315

852

178

...

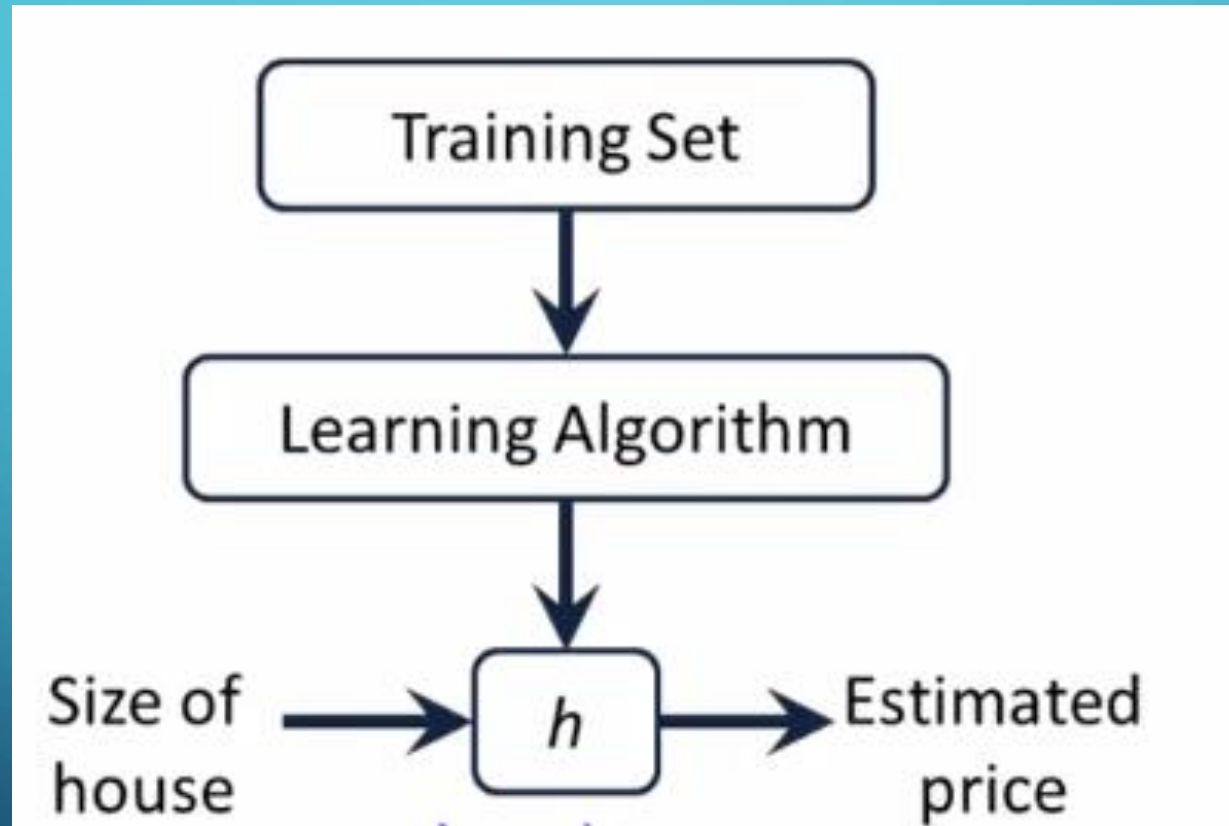
...

Notation:

m = Number of training examples

x's = "input" variable / features

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



Regressions

Simple
Linear
Regression

$$y = b_0 + b_1 * x_1$$

Simple Linear Regression

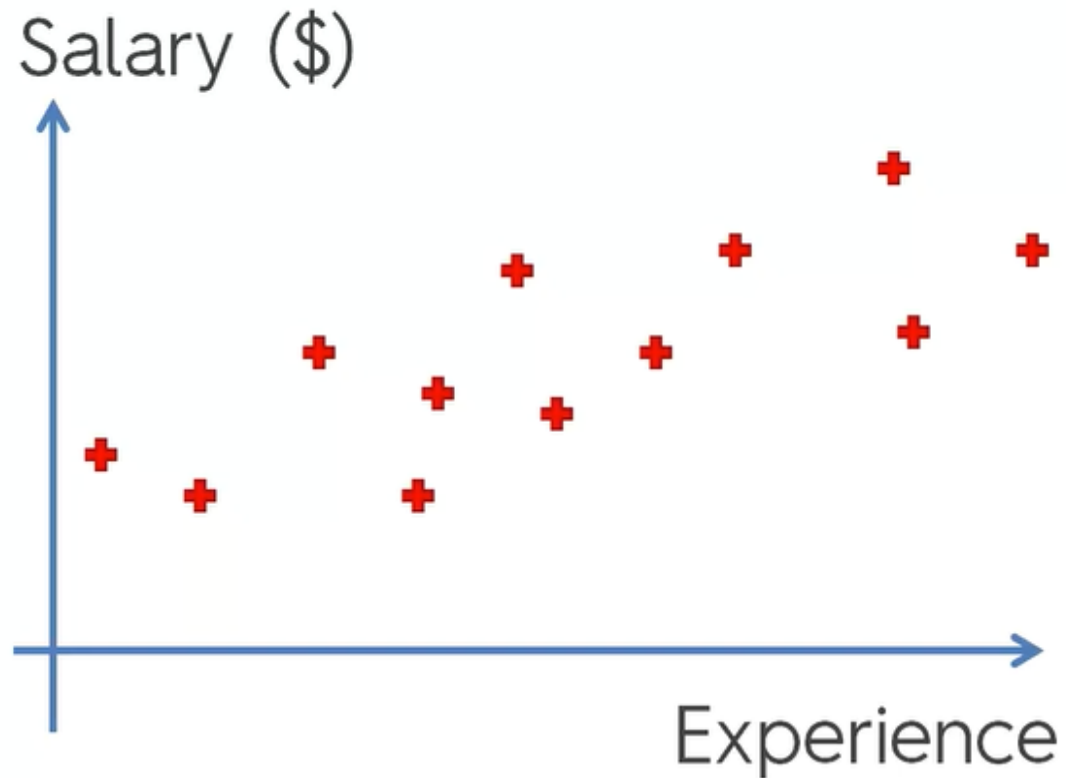
$$y = b_0 + b_1 * x_1$$

Coefficient

Dependent variable (DV)

Independent variable (IV)

Simple Linear Regression:

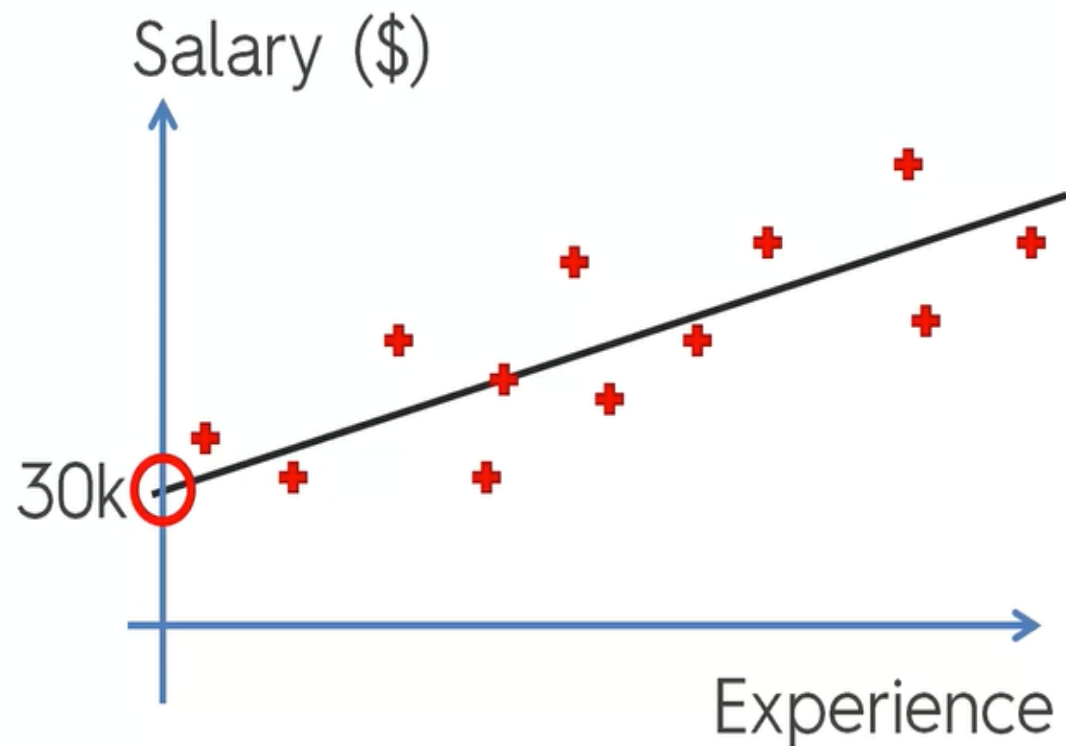


$$y = b_0 + b_1 * x$$



$$\text{Salary} = b_0 + b_1 * \text{Experience}$$

Simple Linear Regression:



$$y = b_0 + b_1 * x$$

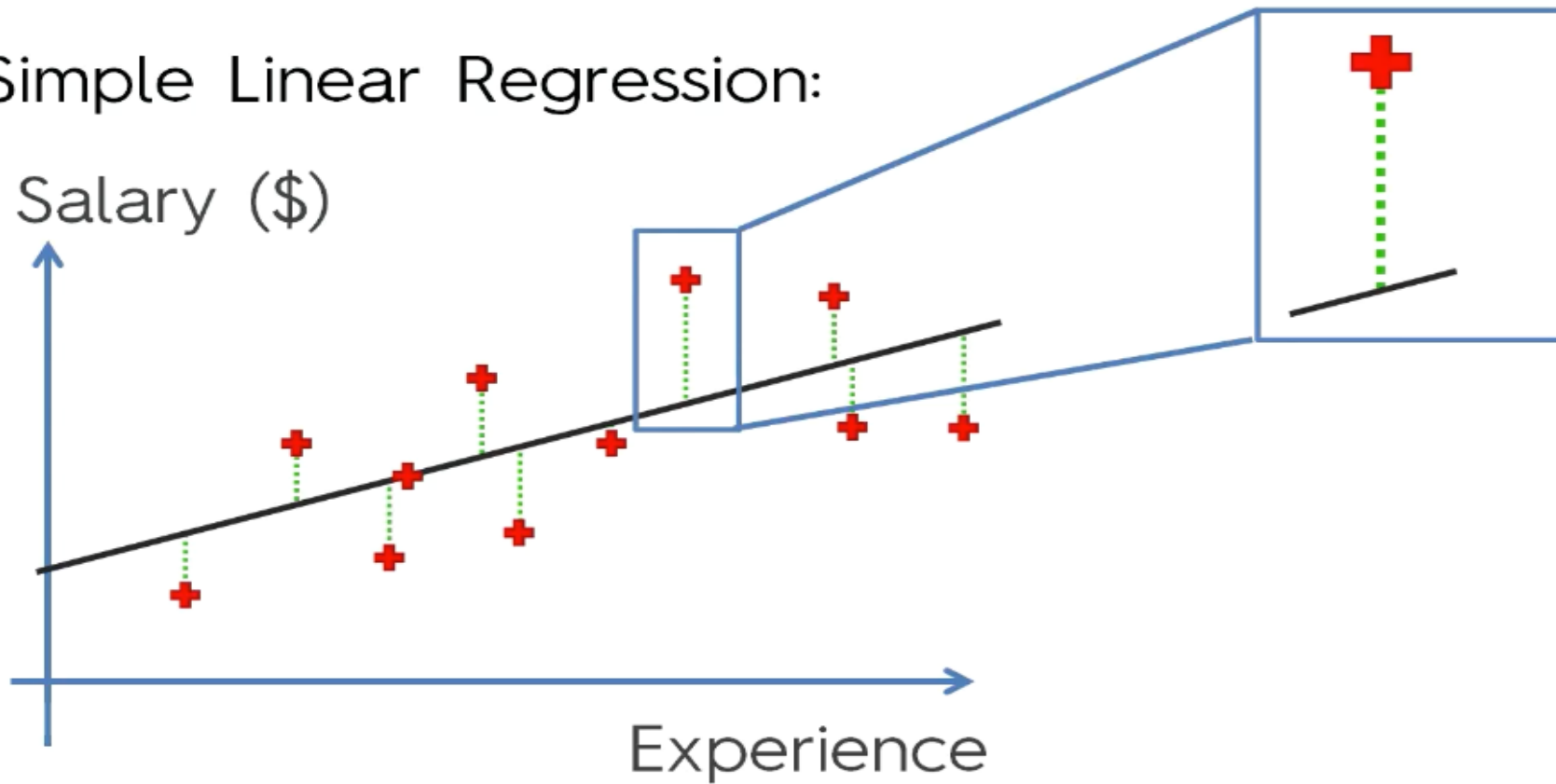


$$\text{Salary} = \text{b}_0 + \text{b}_1 * \text{Experience}$$

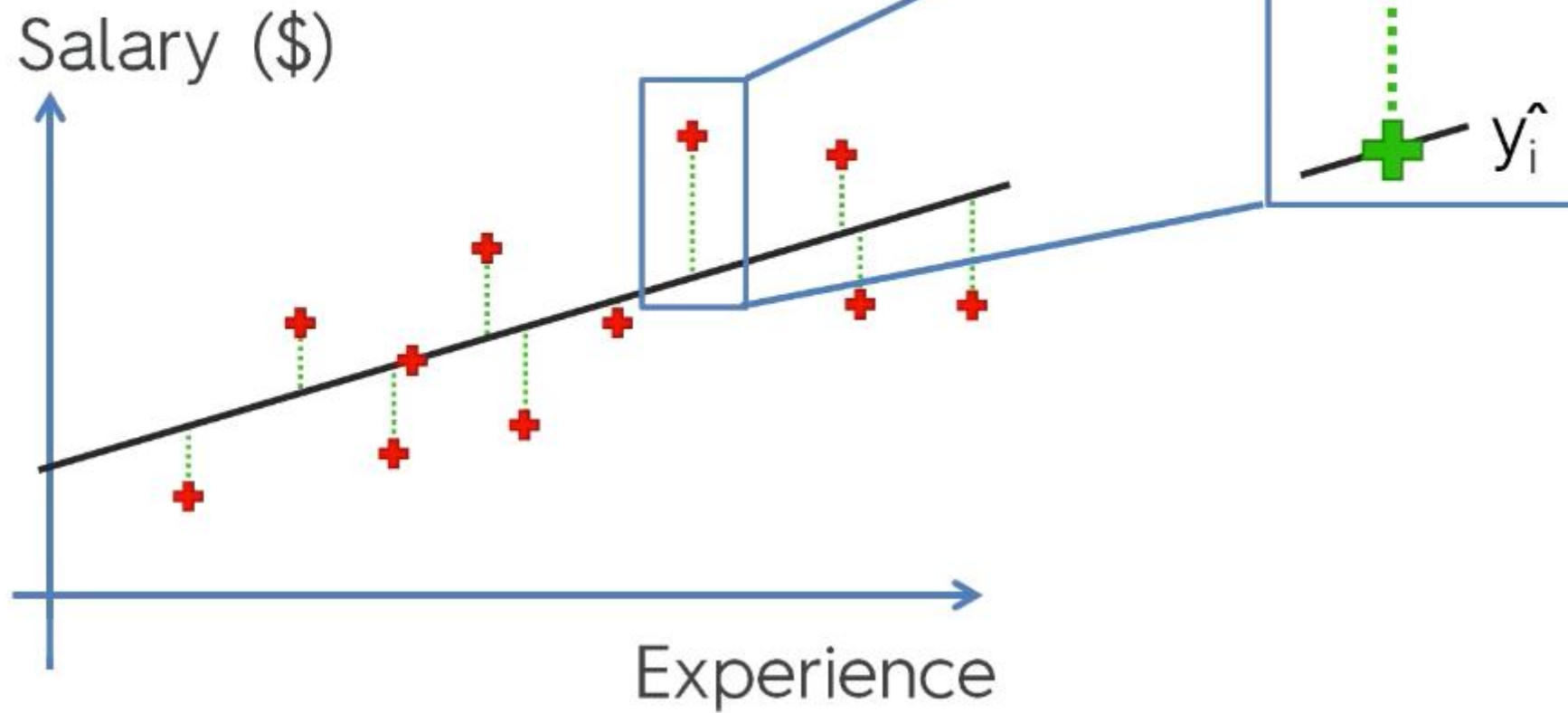
COST FUNCTION

Ordinary Least Squares

Simple Linear Regression:

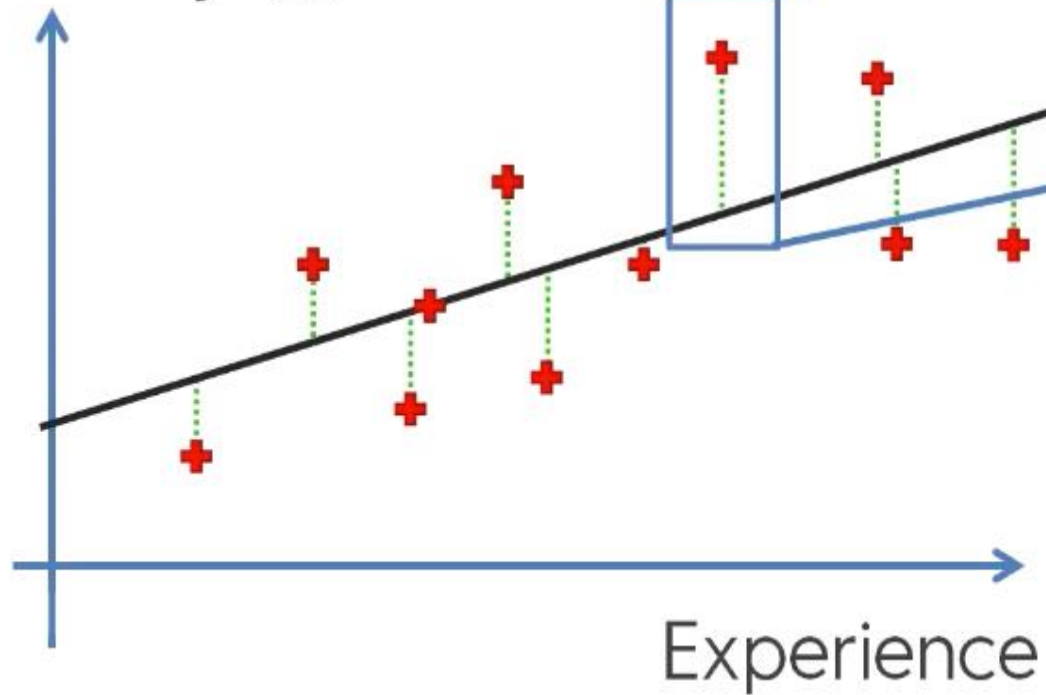


Simple Linear Regression:



Simple Linear Regression:

Salary (\$)



$$\text{SUM } (y - \hat{y})^2 \rightarrow \min$$

OBJECTIVE OF COST FUNCTION

- Quantify the errors in the prediction
- Minimise the cost function to find the optimal parameters
- Also used to monitor the model performance
- Only way to visualise the model in higher dimensions

GRADIENT DESCENT

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

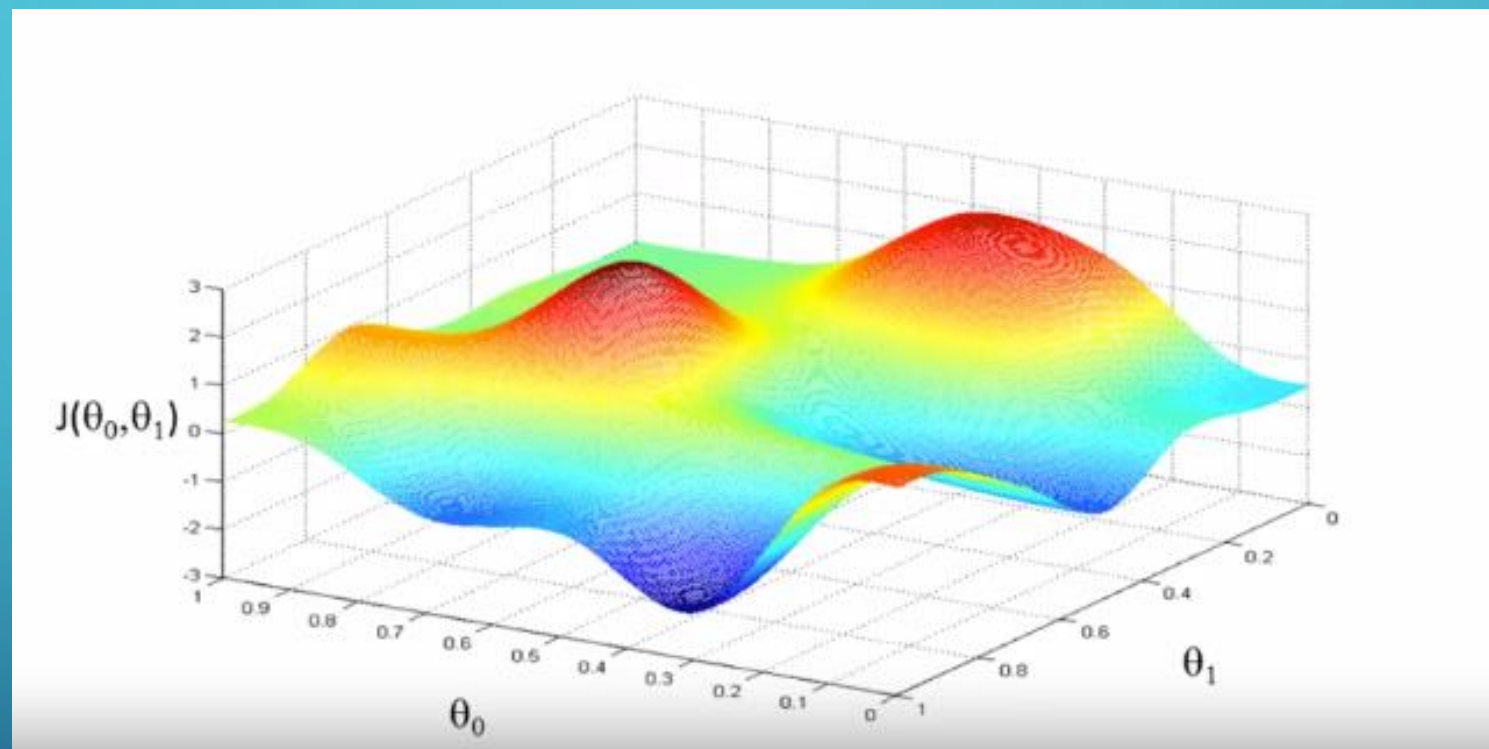
- Problem Statement:

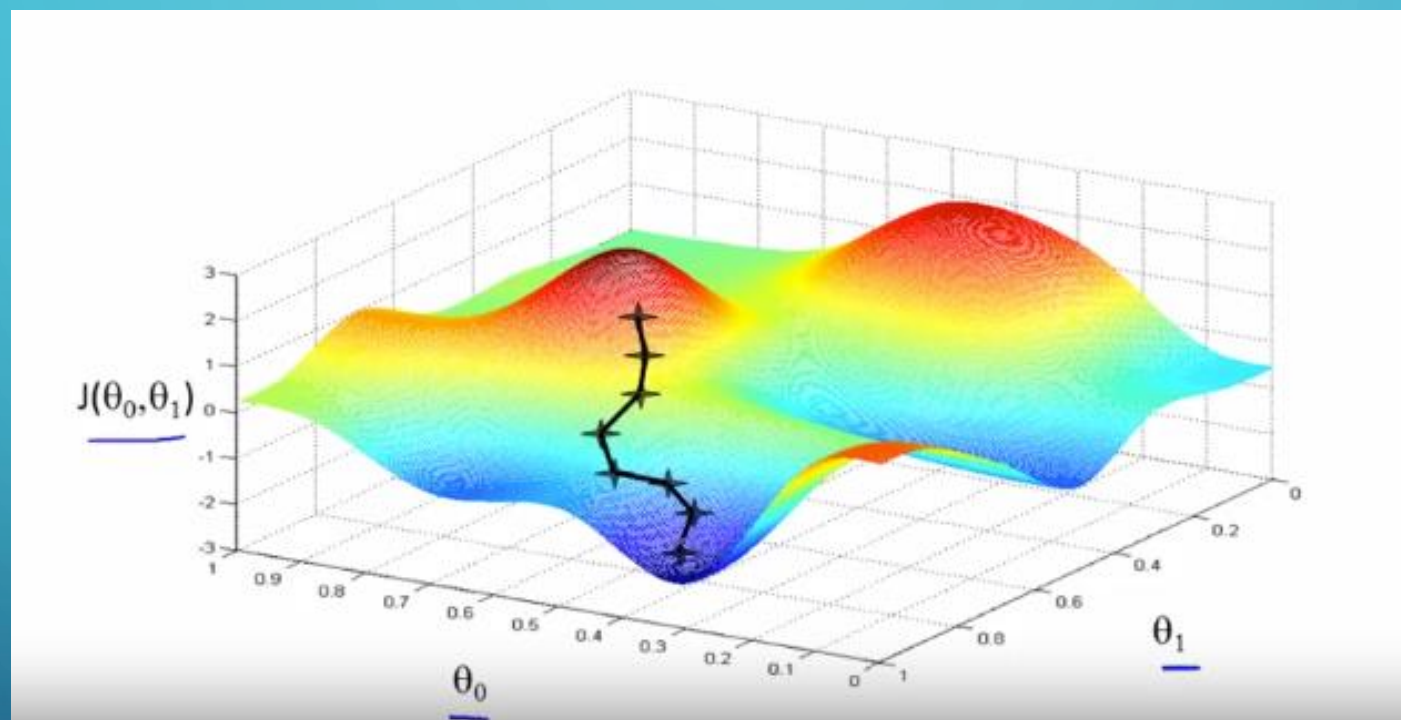
Have some function $J(\theta_0, \theta_1)$

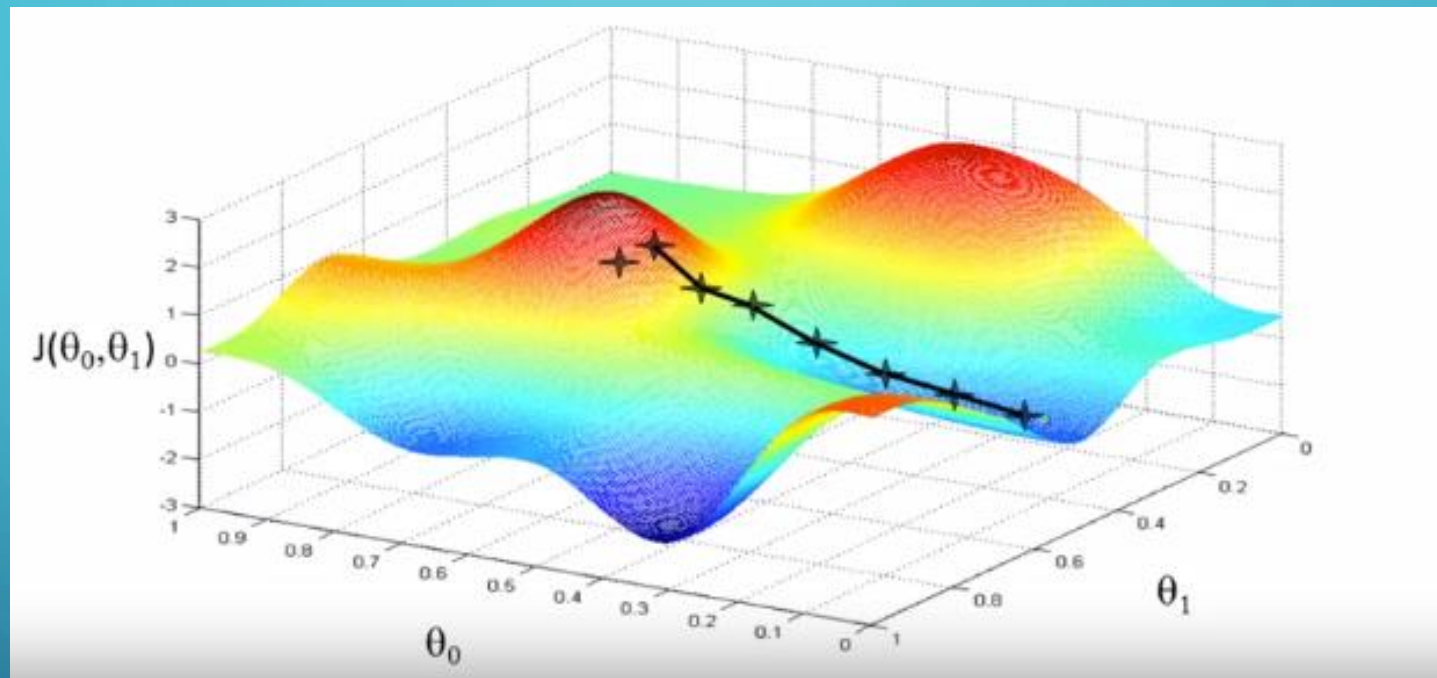
Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

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







Gradient descent algorithm

repeat until convergence {

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

}

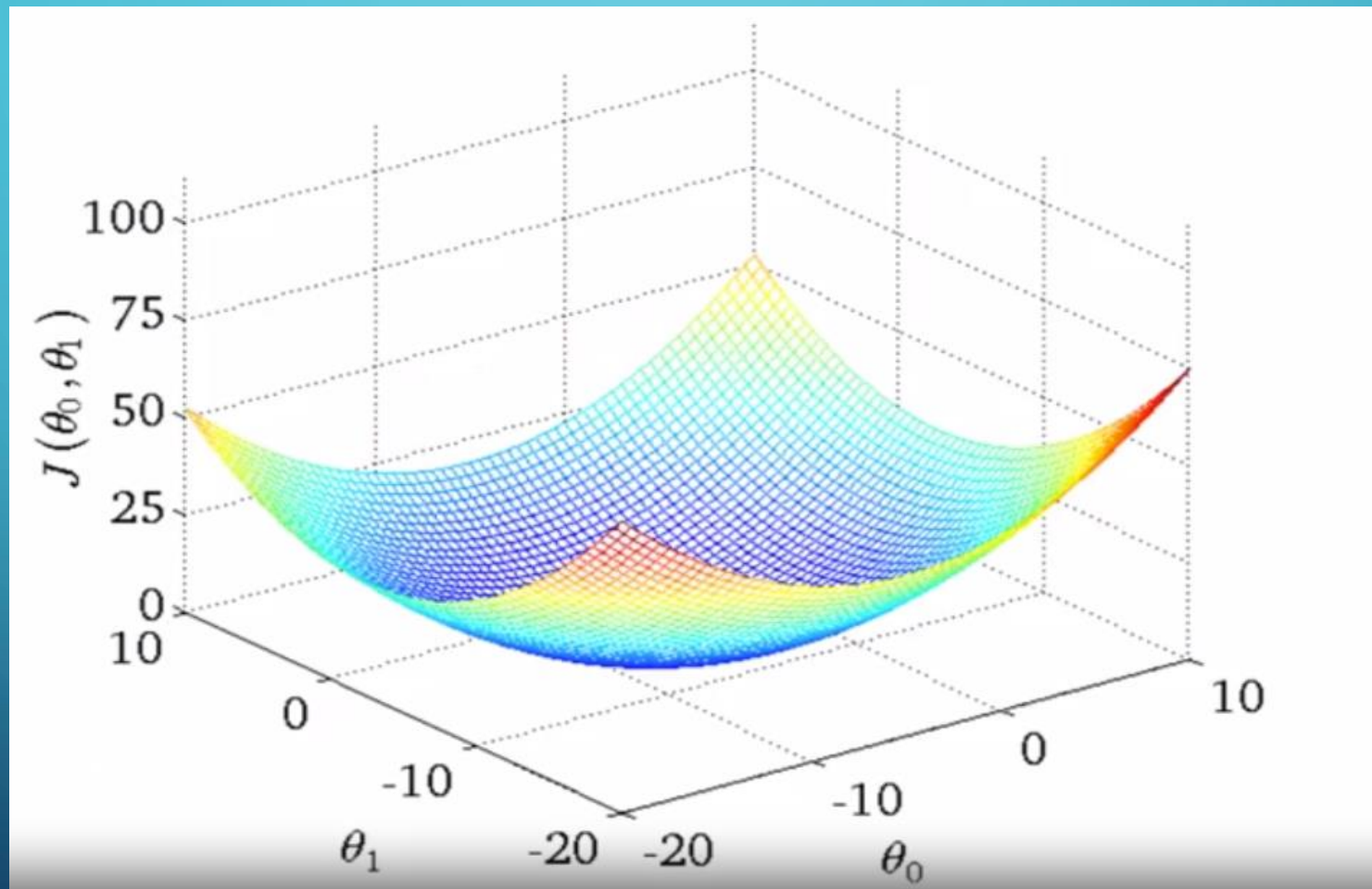
Correct: Simultaneous update

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

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

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

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



Gradient descent algorithm

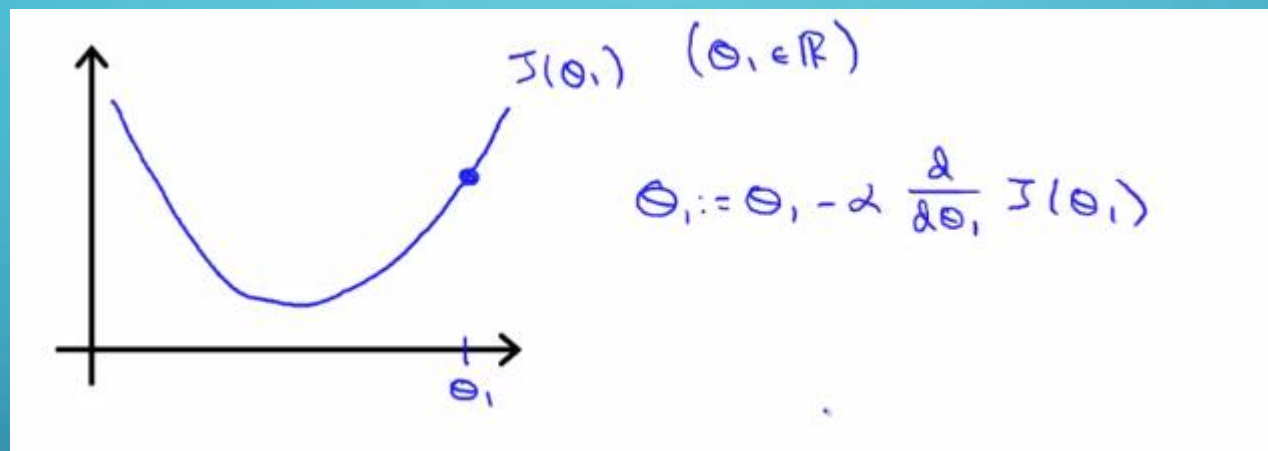
repeat until convergence {
 $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$
 (for $j = 1$ and $j = 0$)
}

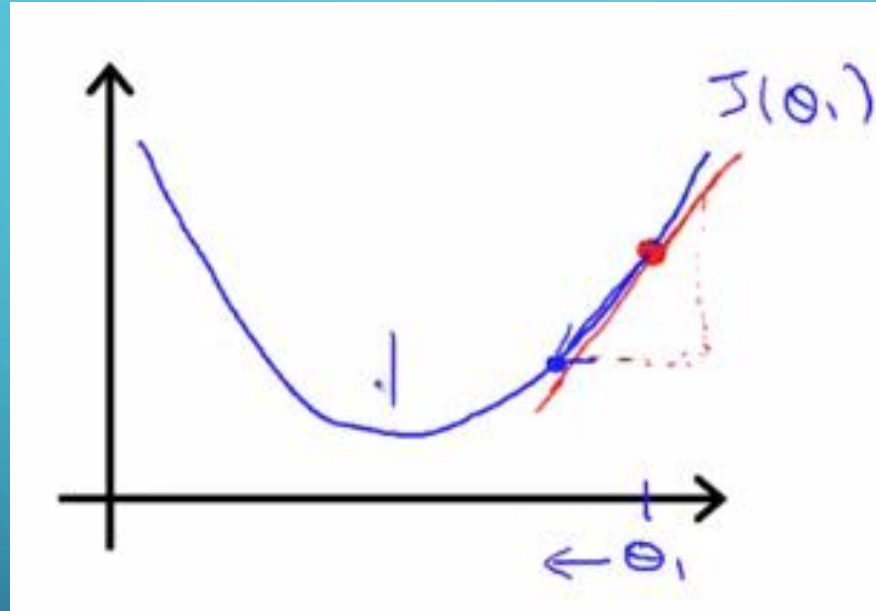
Linear Regression Model

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

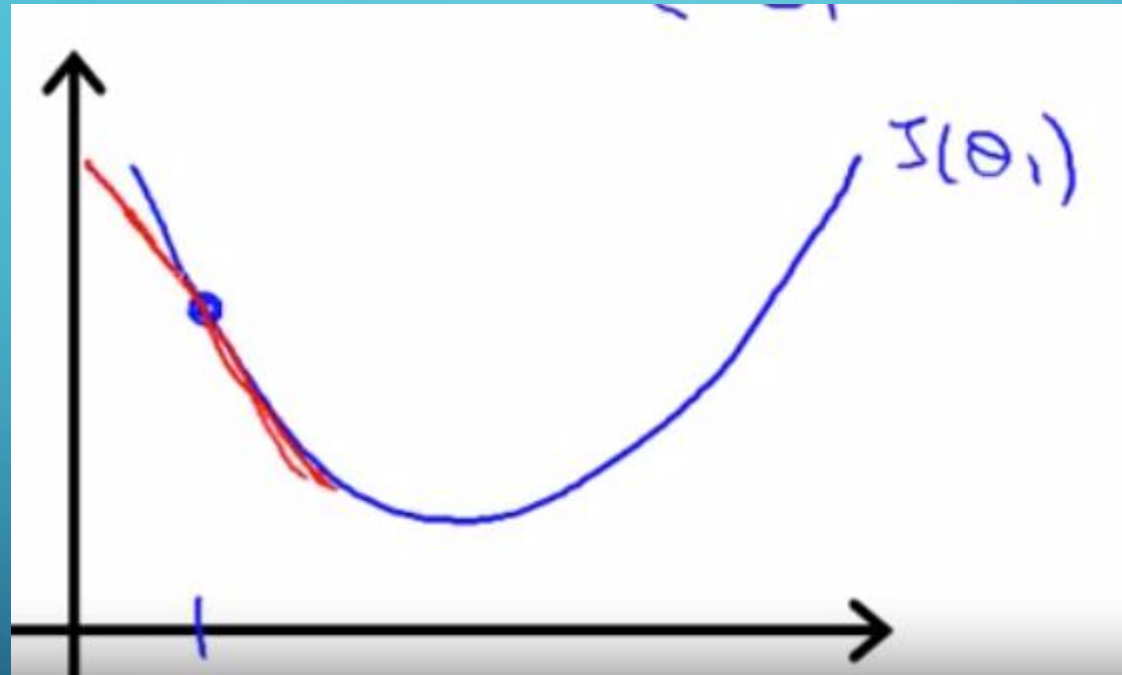
$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

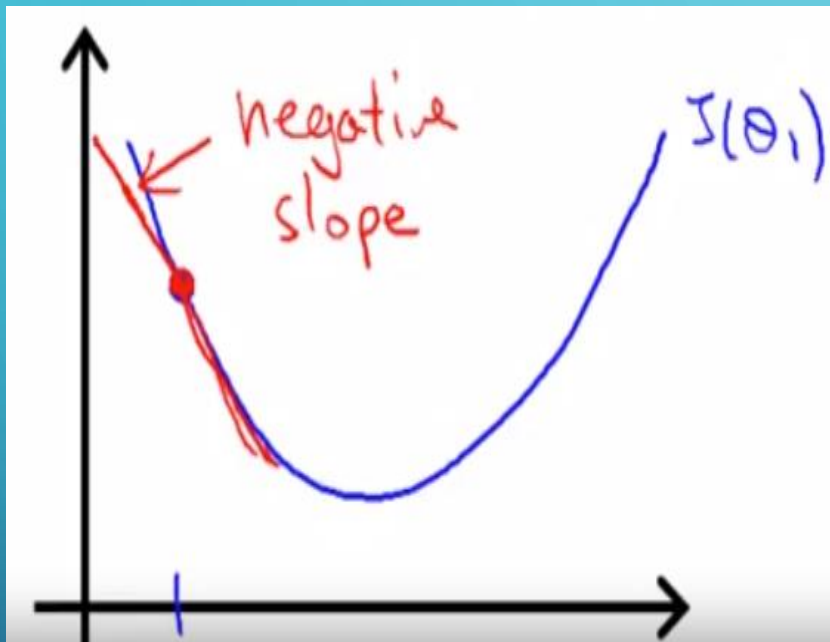
- Note: Slope is positive





- Slope is negative





$$\frac{\partial J(\theta_1)}{\partial \theta_1} \leq 0$$

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

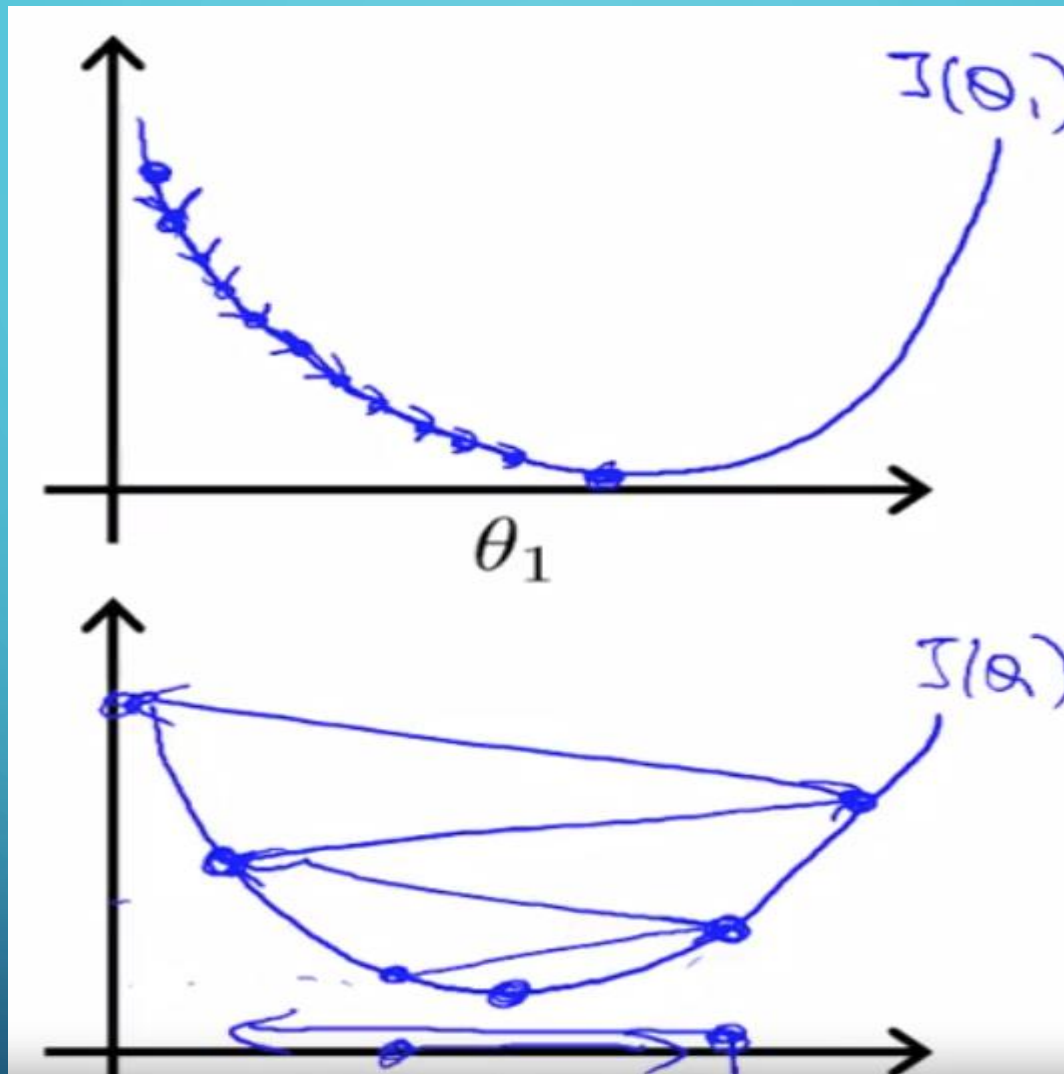
↑ ↑

- Alpha is the learning rate. It amplifies the jumps in the gradient steps

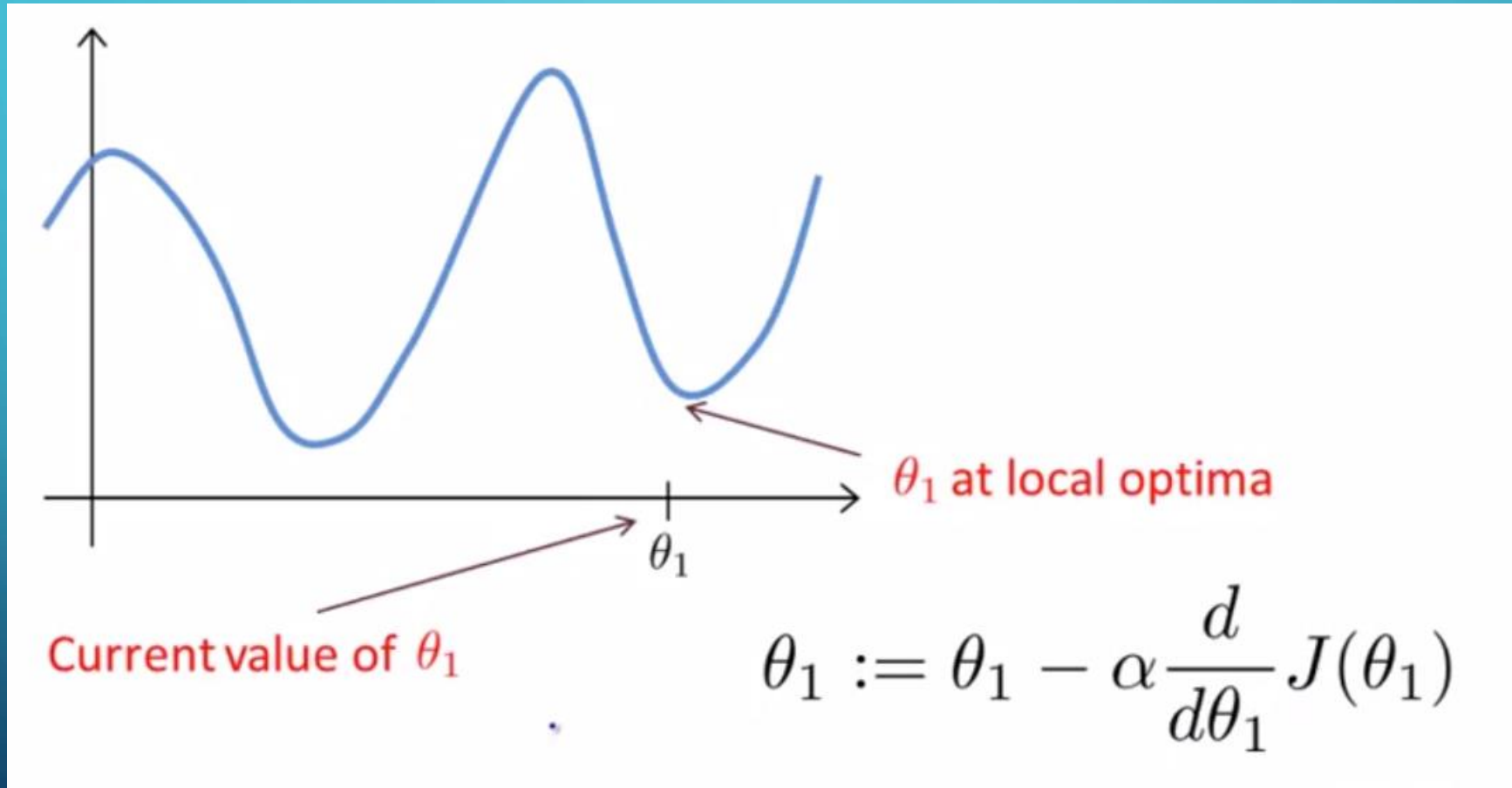
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

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



- Problem with Gradient Descent: Can get stuck at local minima



“Batch” Gradient Descent

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



THANK YOU