COURSE NAME: ARTIFICIAL INTELLIGENCE
COURSE CODE: CIS 412

SYED TANGIM PASHA

LECTURER,

DEPARTMENT OF COMPUTING AND INFORMATION SYSTEM (CIS)

DAFFODIL INTERNATIONAL UNIVERSITY (DIU)

DHAKA,BANGLADESH

LINEAR REGRESSION

1. Linear Regression is a Machine Learning algorithm based on supervised learning. It performs a regression task.

2. Linear Regression performs the task to predict a dependent variable value (Y) based on a given independent variable (X).

মেশিন লার্নিং অ্যালগরিদম

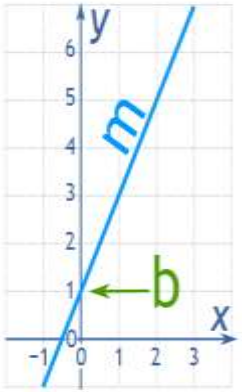
পিৎজার সাইজ (ইঞ্চিতে)	পিৎজার দাম (টাকায়)
6"	350/-
8"	775/-
12"	1150/-
14"	1395/-
18"	1675/-

LINEAR REGRESSION

$$y = mx + b$$

(or "y = mx + c" in the UK see below)

What does it stand for?



$$y = \underset{\text{Slope or Gradient}}{m}x + \underset{\text{y value when x=0 (see Y Intercept)}}{b}$$

y = how far up

x = how far along

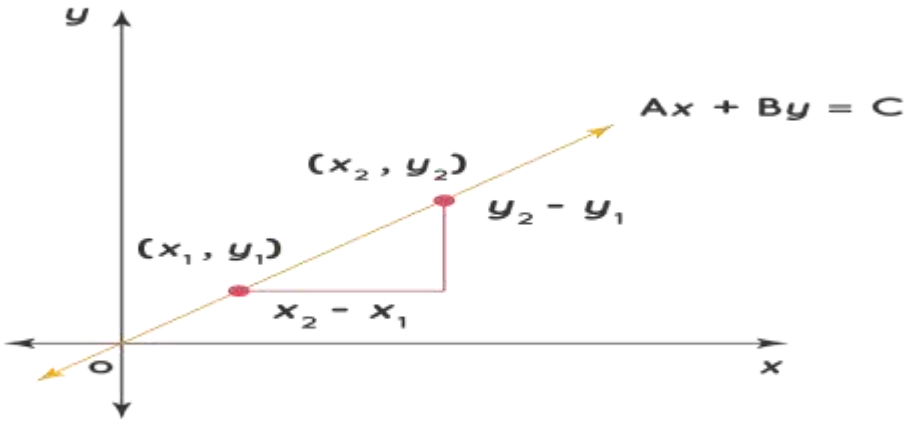
m = Slope or Gradient (how steep the line is)

Positive or Negative Slope?

Going from left-to-right, the cyclist has to **P**ush on a **P**ositive Slope:

Slope of a line

cuemath
THE MATH EXPERT



$$m = \frac{\text{change in } y}{\text{change in } x} = \frac{y_2 - y_1}{x_2 - x_1}$$

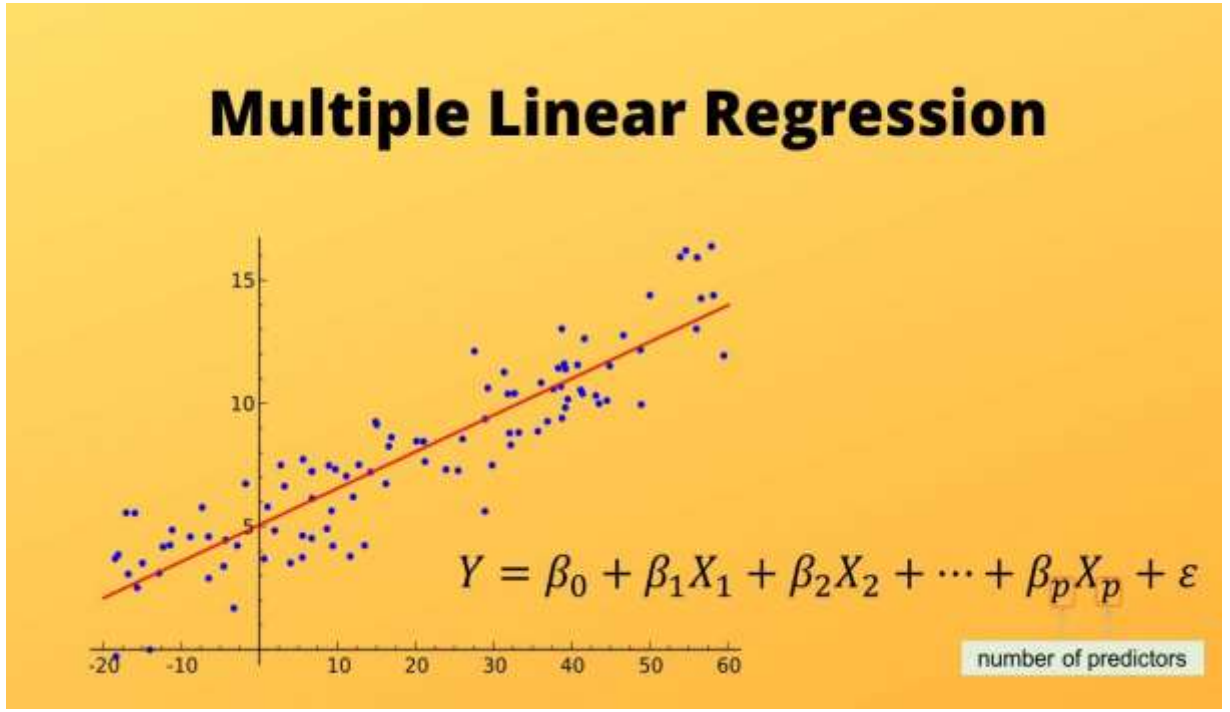
MULTIPLE LINEAR REGRESSION

- **Multiple Linear Regression:** Multiple Linear Regression is used to estimate the relationship between two or more independent variables (X) and one dependent variable (Y).

Multiple Linear Regression

$$y = b_0 + b_1*x_1 + b_2*x_2 + \ldots + b_n*x_n$$

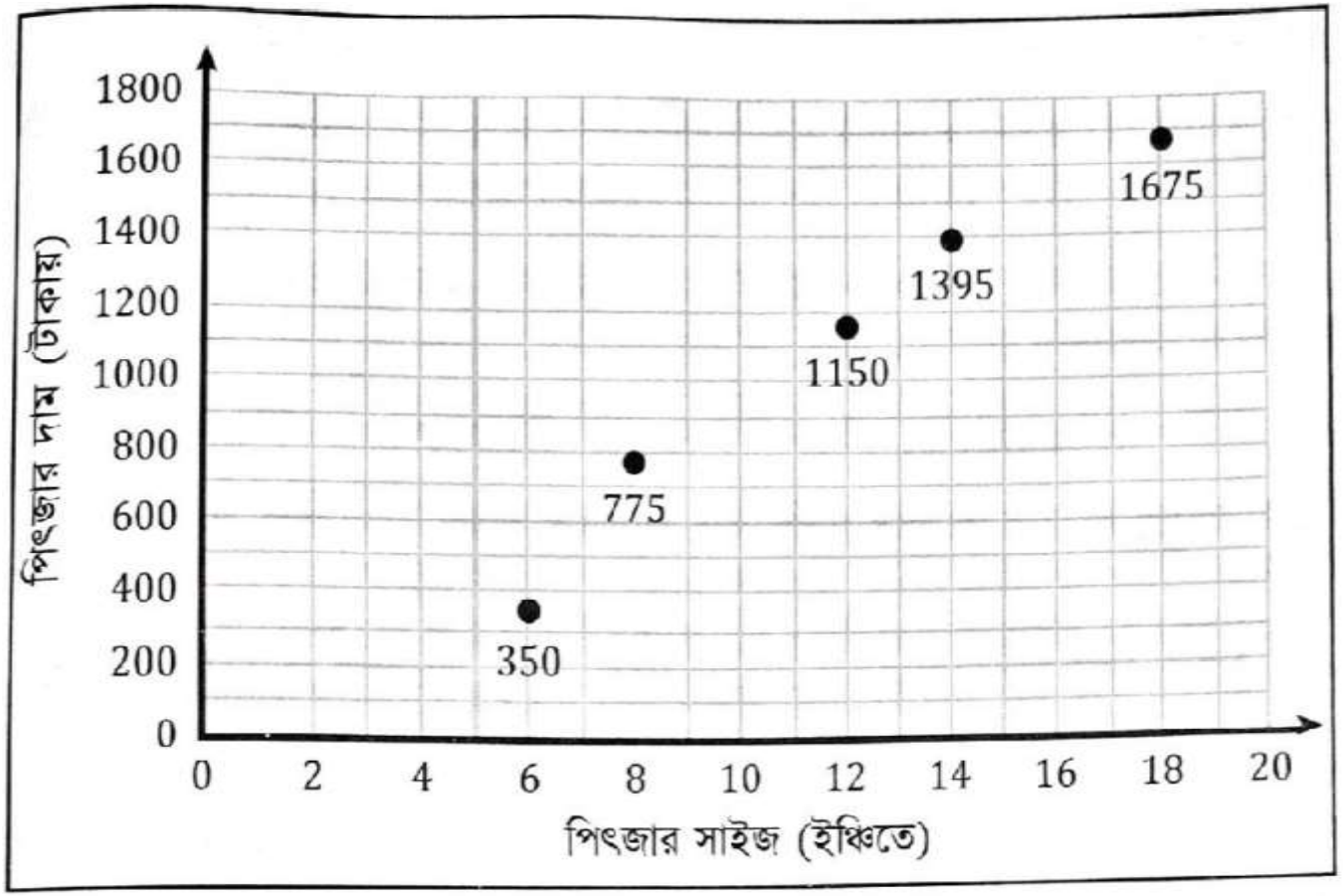
X_1	X_2	X_3	Y
11	68.028164	0	28.05442
12	69.086446	0	25.44146
13	84.806730	1	18.73395
14	19.011313	1	18.00611
15	63.046323	1	24.69284



LINEAR REGRESSION

মেশিন লার্নিং অ্যালগরিদম

পিৎজার সাইজ (ইঞ্চিতে)	পিৎজার দাম (টাকায়)
6"	350/-
8"	775/-
12"	1150/-
14"	1395/-
18"	1675/-



HYPOTHESIS FUNCTION

- **Hypothesis Function:** An example of a model that approximates the target function and performs mappings of inputs to outputs is called a hypothesis in machine learning.

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

COST FUNCTION

- **Cost Function:** By achieving the best fit regression line, the model aims to predict Y value such that the error difference between predicted value and true value is minimum. So, it is very important to update θo and θ1 values, to reach the best value that minimize the error between predicted Y value(pred) and true Y value (y).
- Cost function (J) of linear regression is the Root Mean Squared Error (RMSE) between predicted Y value (pred) and true Y value (y).

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

COST FUNCTION

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

Parameters: θ_0, θ_1

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

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

GRADIENT DESCENT

- **Gradient Descent:** To update θo and θ1 values in order to reduce Cost Function and achieving the best fit line the model uses Gradient Descent. The idea is to start with random θo and θ1 values and then iteratively updating the values, reaching minimum cost.

- Gradient descent is an iterative optimization algorithm for finding the local minimum of a function.

GRADIENT DESCENT

Gradient descent algorithm

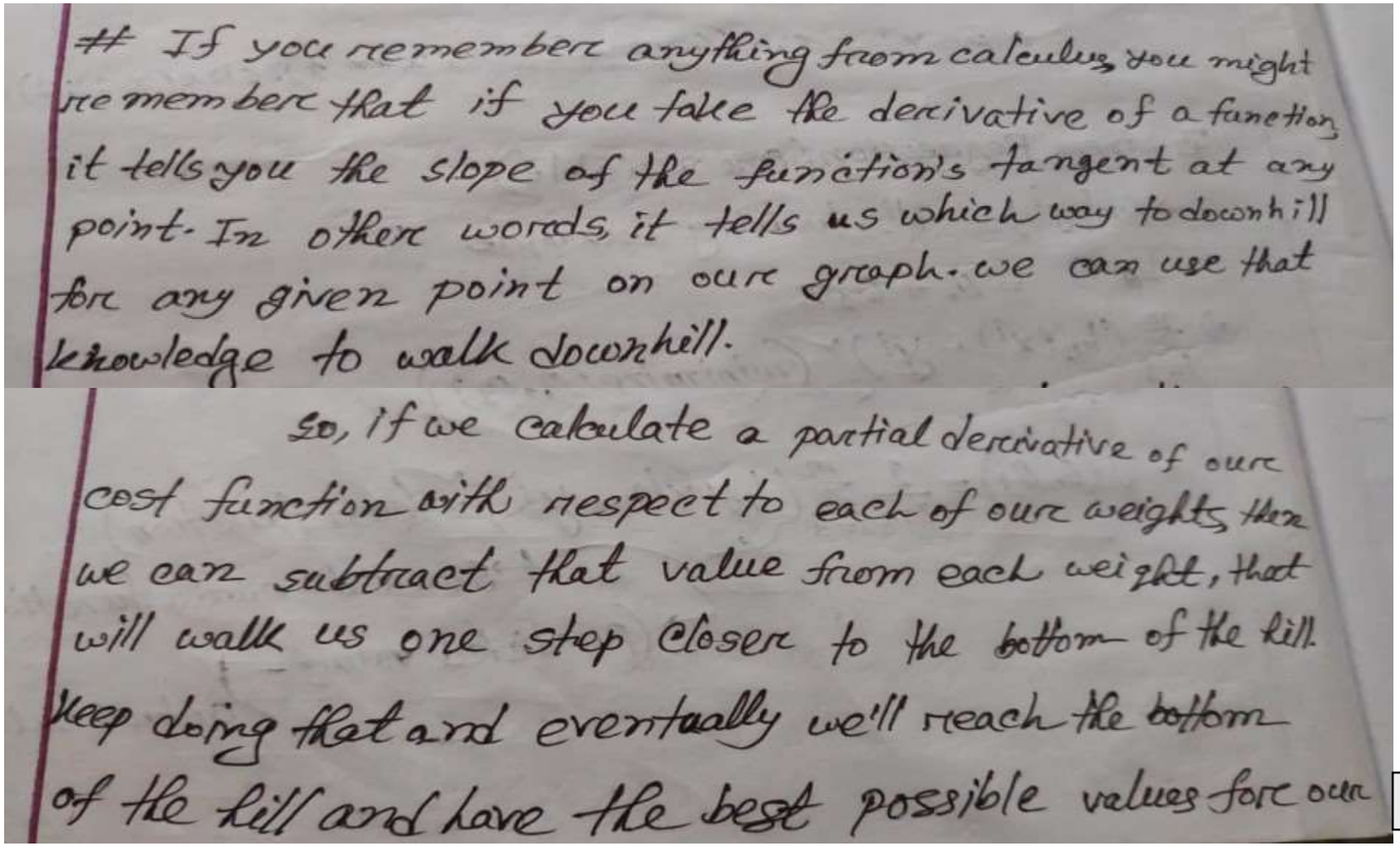
repeat until convergence {

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

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

}

GRADIENT DESCENT

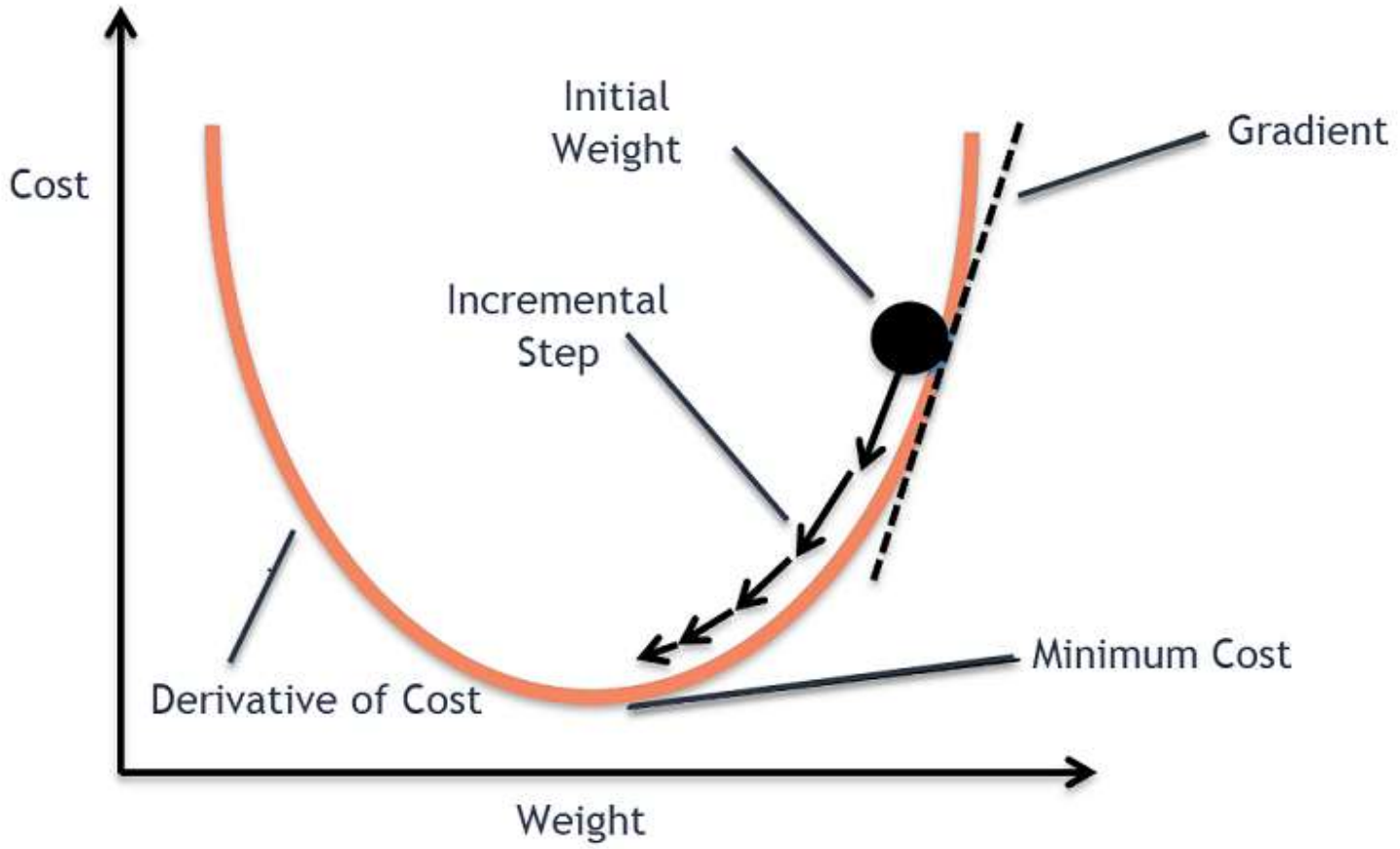


If you remember anything from calculus, you might remember that if you take the derivative of a function, it tells you the slope of the function's tangent at any point. In other words, it tells us which way to downhill for any given point on our graph. we can use that knowledge to walk downhill.

So, if we calculate a partial derivative of our cost function with respect to each of our weights, then we can subtract that value from each weight, that will walk us one step closer to the bottom of the hill. Keep doing that and eventually we'll reach the bottom of the hill and have the best possible values for our

weights

GRADIENT DESCENT



GRADIENT DESCENT CURVE

GRADIENT DESCENT

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Now,

$$\frac{\partial}{\partial \theta} J_\theta = \frac{\partial}{\partial \theta} \frac{1}{2m} \sum_{i=1}^{m} \left[h_\theta(x_i) - y_i \right]^2$$

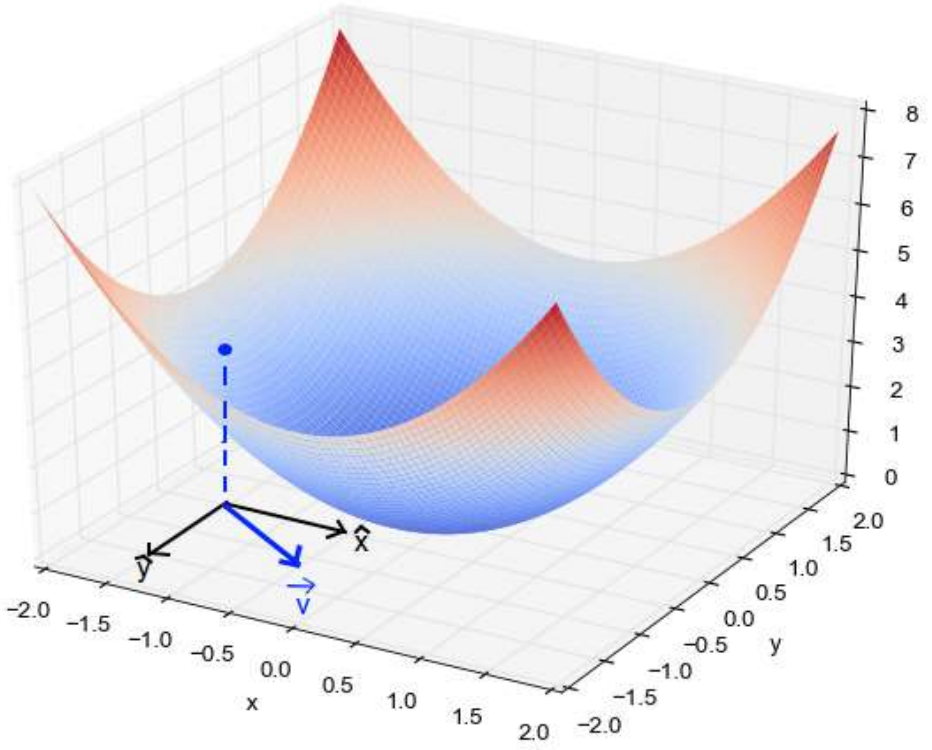
$$\frac{\partial}{\partial \theta} J_\theta = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i \right) \cdot \frac{\partial}{\partial \theta_j} \left(\theta x_i - y_i \right)$$

$$\frac{\partial}{\partial \theta} J_\theta = \frac{1}{m} \sum_{i=1}^{m} \left[\left(h_\theta(x_i) - y \right) x_i \right]$$

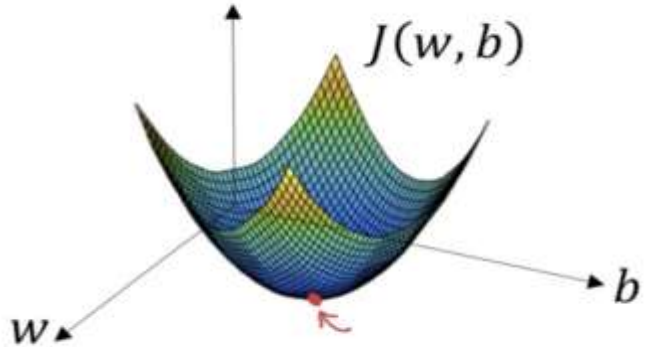
Therefore,

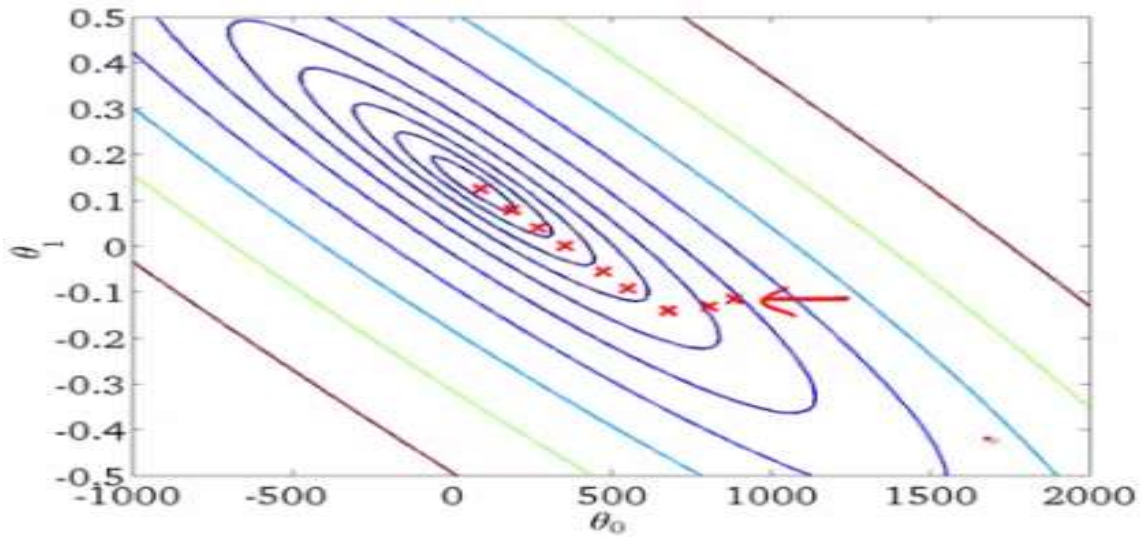
$$\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} \left[\left(h_\theta(x_i) - y_i \right) x_i \right]$$

GRADIENT DESCENT



3D GRAPH (X,Y,J)

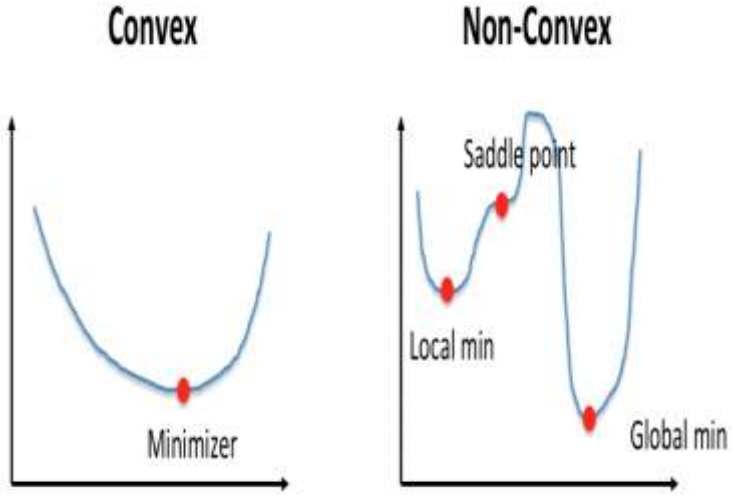




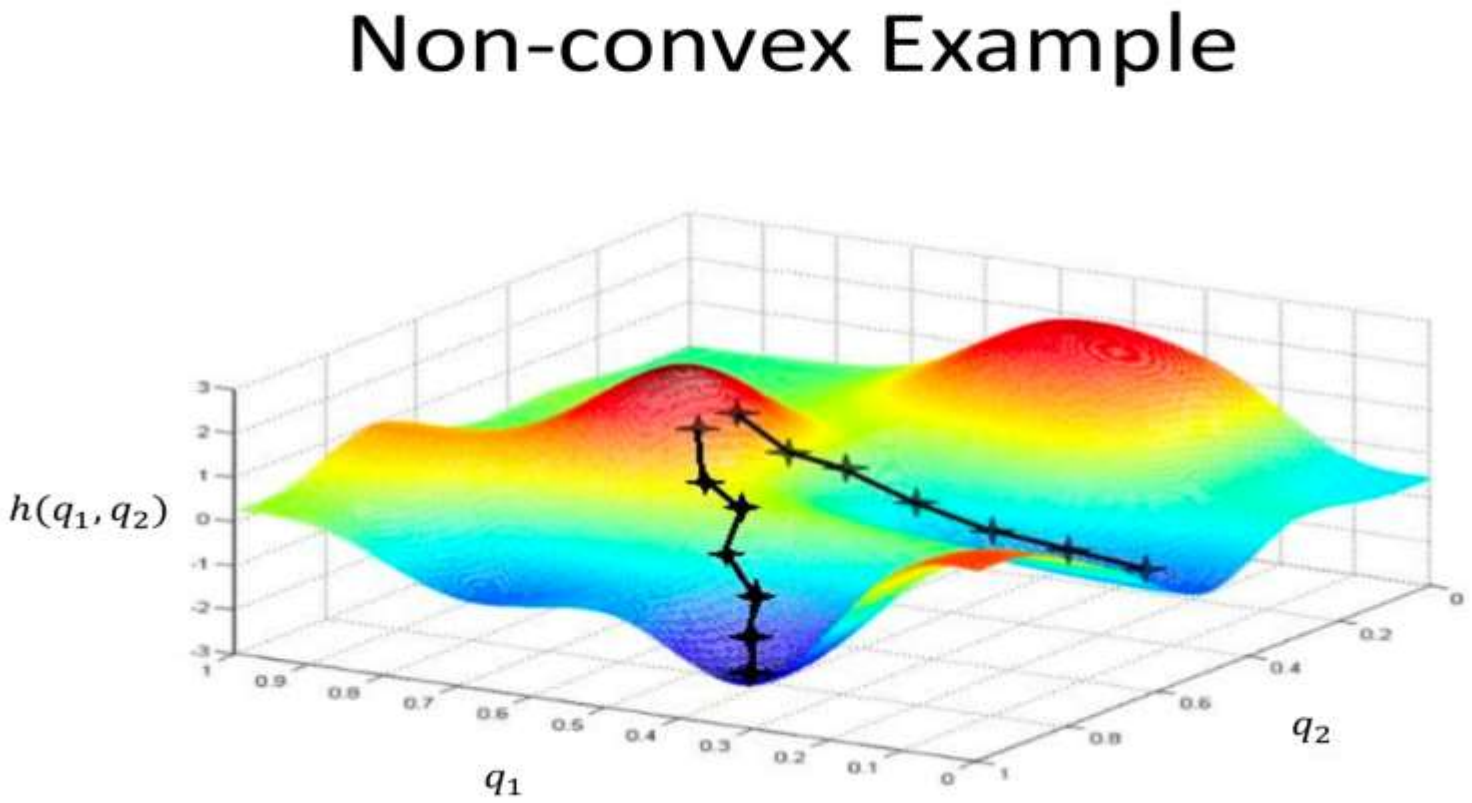
CONTOUR PLOT

GRADIENT DESCENT

- **Convex & NonConvex**: Convex and nonconvex are optimization problems. The basic difference between the two categories is that:
- **convex optimization** there can be only one optimal solution, which is globally optimal or you might prove that there is no feasible solution to the problem.
- While in **nonconvex optimization** may have multiple locally optimal points and it can take a lot of time to identify whether the problem has no solution or if the solution is global.

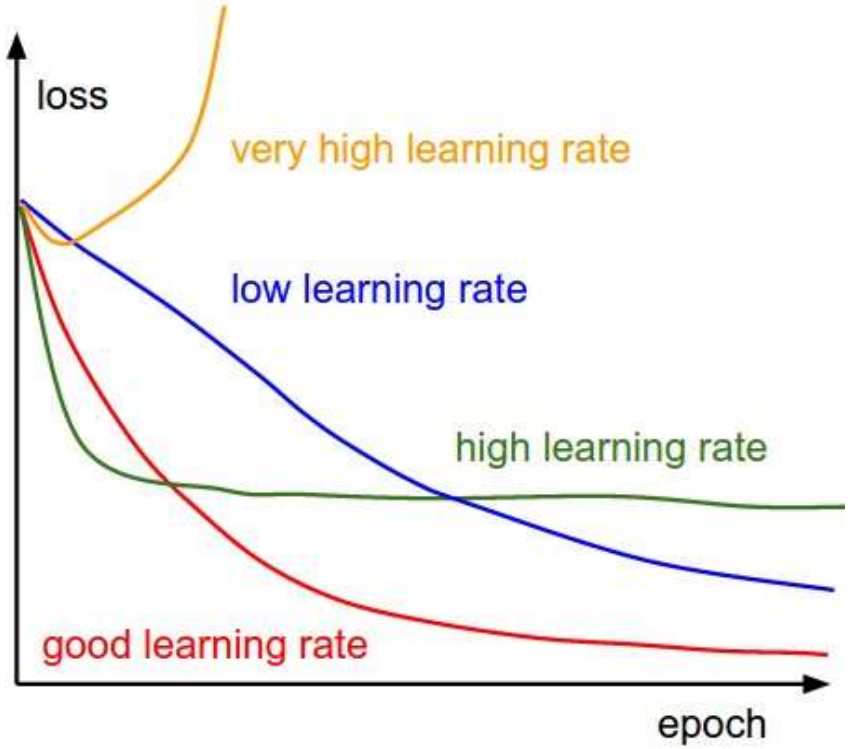


GRADIENT DESCENT



GRADIENT DESCENT

- **Learning Curves:** A good way to make sure gradient descent runs properly is by plotting the cost function as the optimization runs. Put the number of iterations on the x-axis and the value of the cost function on the y-axis. This helps you see the value of your cost function after each iteration of gradient descent and provides a way to easily spot how appropriate your learning rate is.



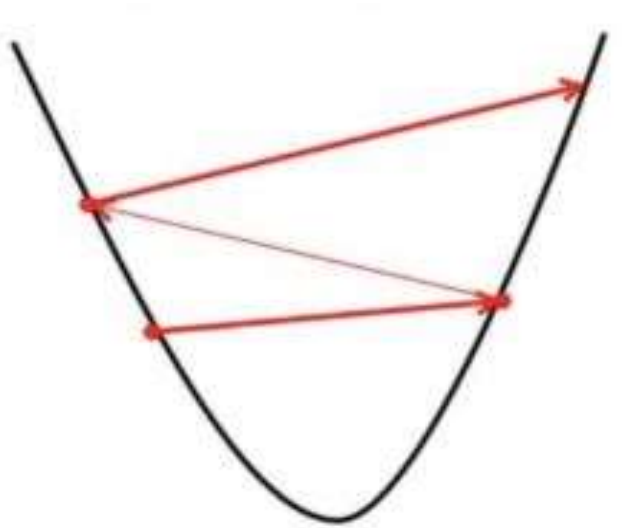
GRADIENT DESCENT LEARNING CURVES

LEARNING RATE- α

- **Learning Rate (α):** In Machine Learning and statistics, the learning rate is a tuning parameter in an optimization algorithm that determines the step size at each iteration while moving towards a minimum of a loss function.

LEARNING RATE- α

- **Learning Rate(α-High):** If we choose α to be very large, Gradient Descent can overshoot the minimum. It may fail to converge or even diverge.



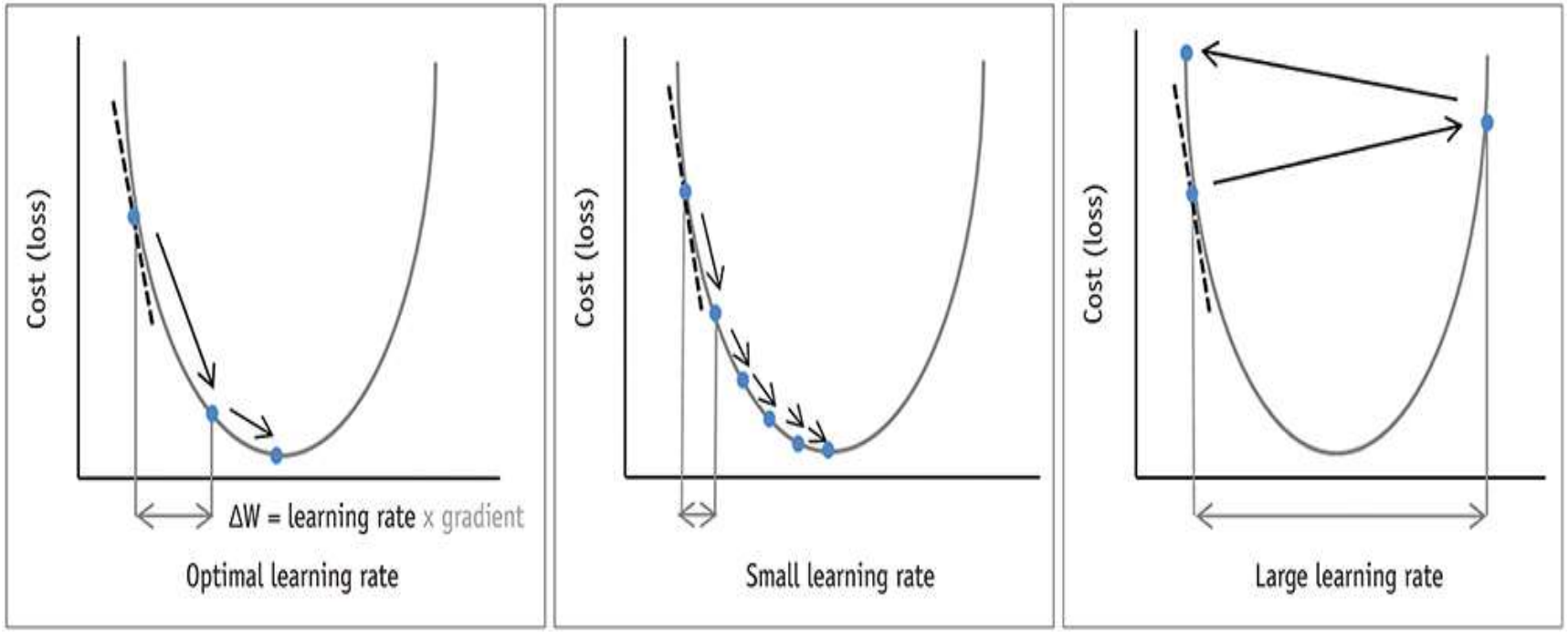
High Learning Rate (α)

LEARNING RATE- α

- **Learning Rate(α-Low):** If we choose α to be very small, Gradient Descent will take small steps to reach local minimum and will take a longer time to reach minimum.

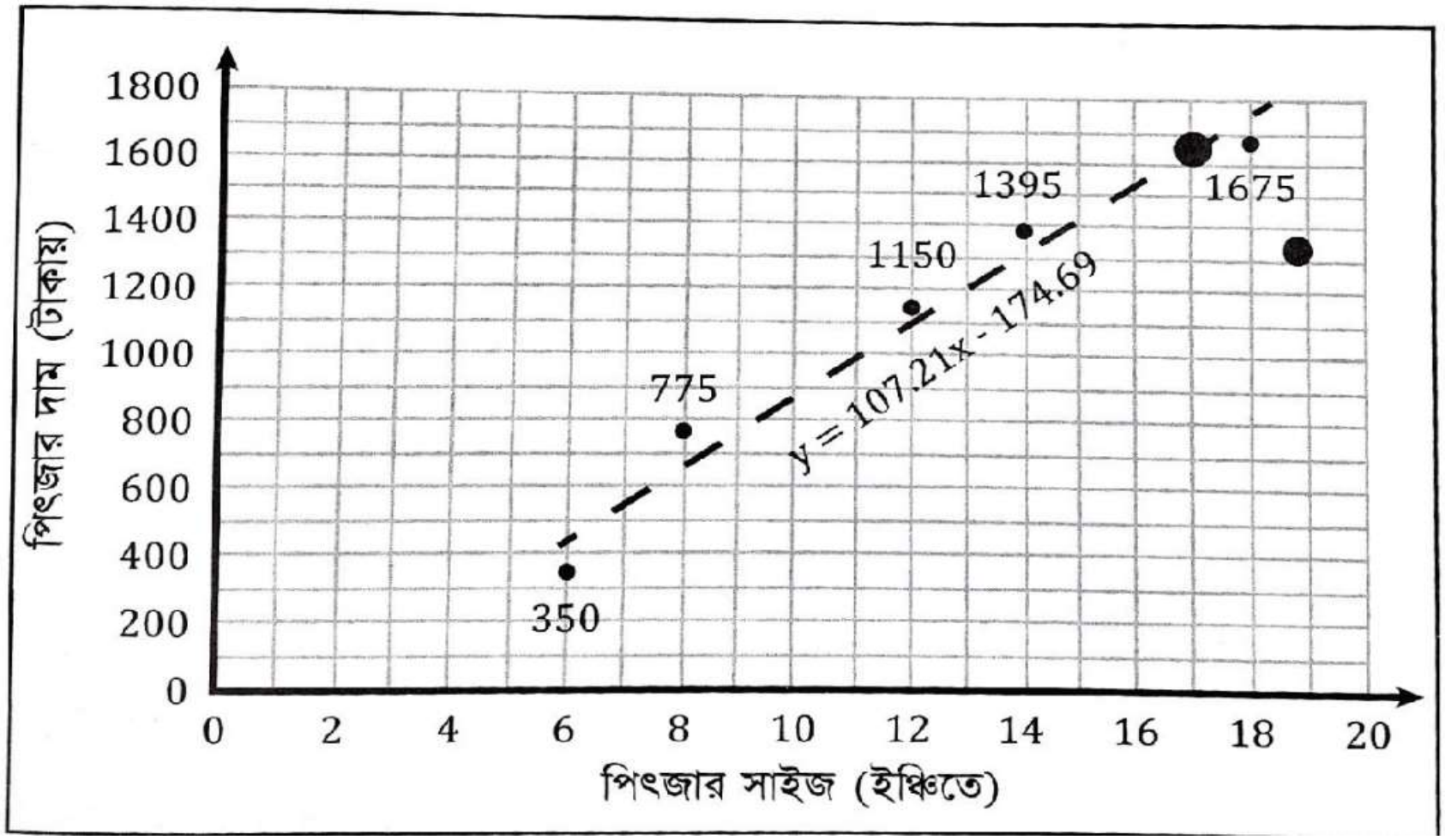
Low Learning Rate (α)

LEARNING RATE- α



- **Learning Rate(α-Good):** The range of value to consider for the learning rate is less than 1.0 and greater than 10^-6. A traditional default value for the learning rate is 0.1 or 0.01, and this may represent a good starting point on your problem.

RESULT


পিৎজার দাম (টাকায়)
1800
1600
1400
1200
1000
800
600
400
200
0
0
2
4
6
8
10
12
14
16
18
20
পিৎজার সাইজ (ইঞ্চিতে)
350
775
1150
1395
1675
y = 107.21x - 174.69

EXAMPLE

- Machine Learning Algorithm by Nafees Neehal-Bangla Book pdf version
- Page No.: 71-(3.8)