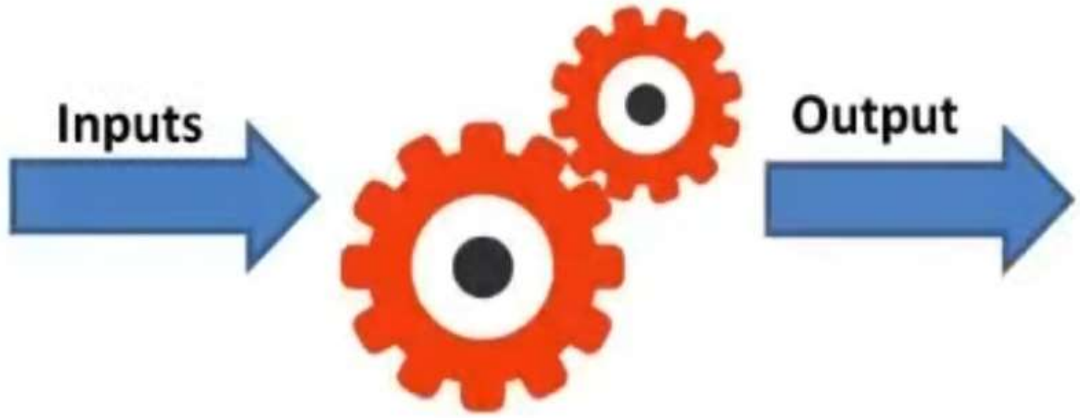
Linear Regression
Cost Function
Gradient Descent

What is Model ?

A mathematical representation of a real-world process in the form of input-output relationship.

Why make Model ?

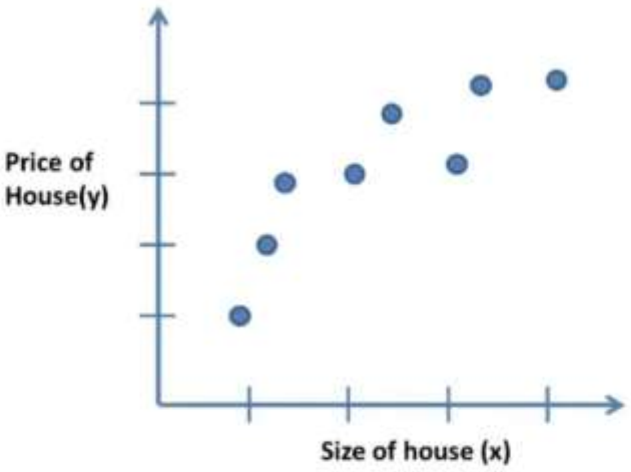
Not only models help us understand the nature of the process being modelled, they also enable us to predict the output based on input features

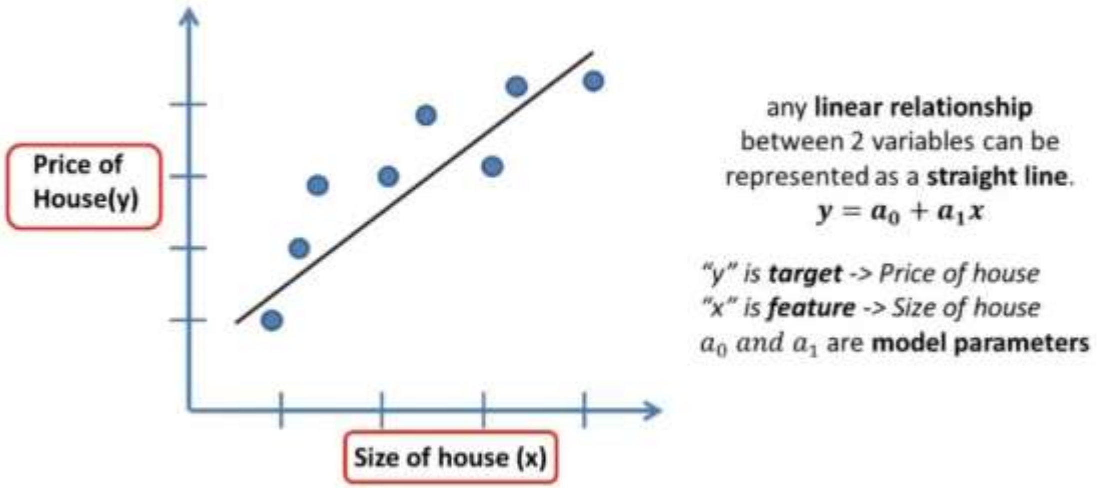


Let's see an Example !

Size of house(sqm.)		Price of house	
64		$	200,000
80		$	250,000
63		$	230,000
100		$	320,000
128		$	300,000
144		$	450,000
...		....	
...		....	
$\boxed{81}$		$\boxed{??}$	

..we want to know price(**output**) of a new house based on its Size(**input**).

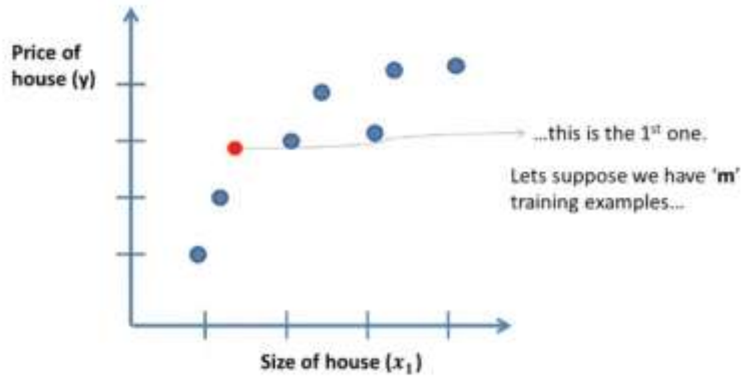


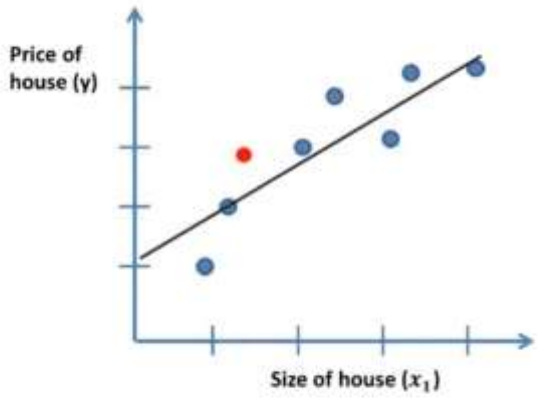


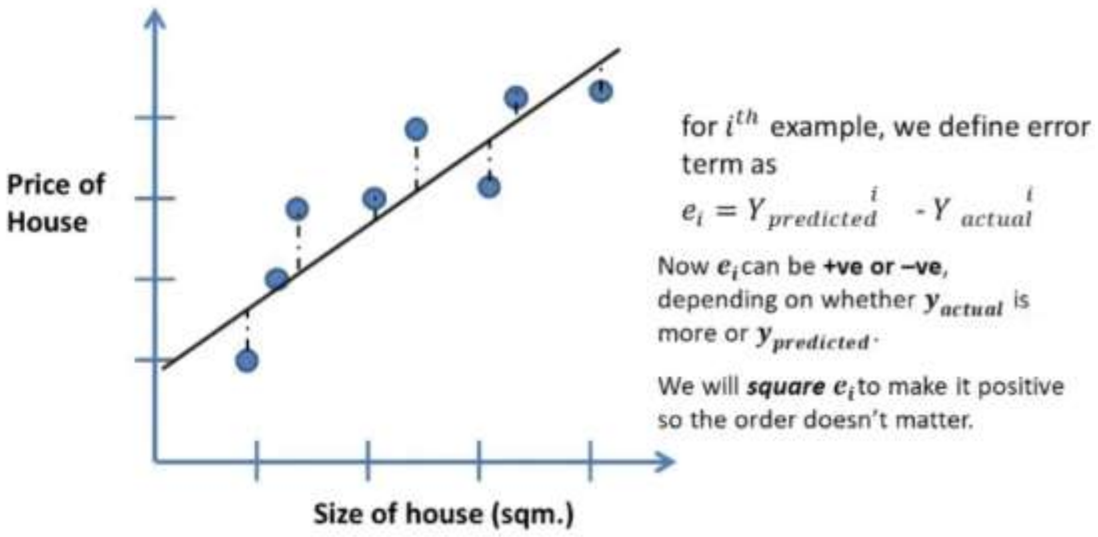
Linear Regression

Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables. It assumes that the relationship between the variables is linear, meaning it can be represented by a straight line.

Cost Function







Cost Function

A cost function, also known as a loss function or objective function, is a measure of how well a machine learning model performs. It quantifies the difference between the predicted values and the actual values for a given dataset.

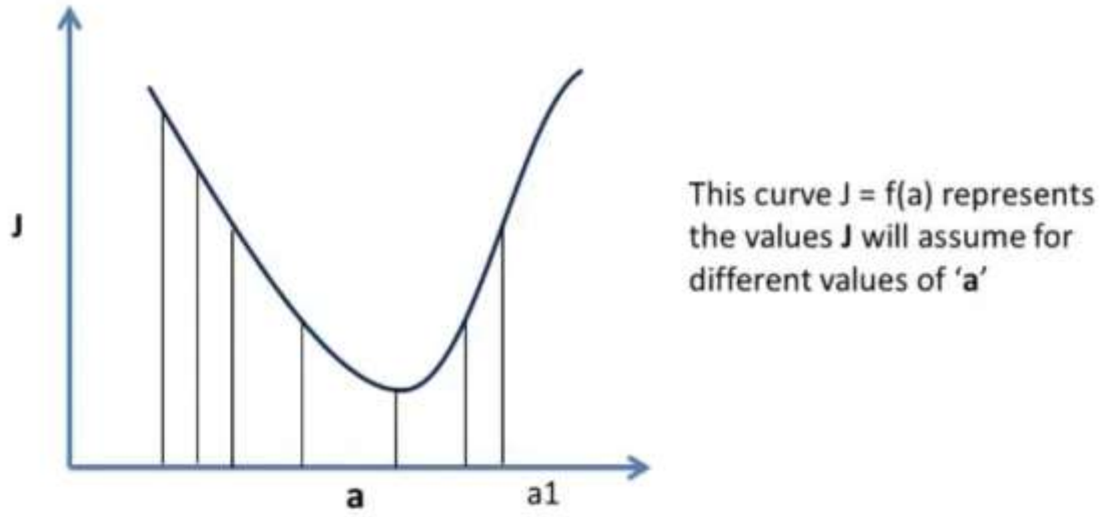
$$\text{Cost function } \mathbf{(J)} = \frac{1}{2m}\left(e_1^{\,2} + e_2^{\,2} + e_3^{\,2} + \ldots\ldots\ldots\ldots + e_m^{\,2}\right)$$

$$\mathbf{J(a)} = \frac{1}{2m} \sum_{i=1}^{m} \left(y_{i\,(pre)} - y_{i\,(act)}\right)^2$$

$$\boxed{\mathbf{J(a)} = \frac{1}{2m} \sum_{i=1}^{m} \left(a_0 + a_1\, x_1^{(i)} - y_{i\,(act)}\right)^2}$$

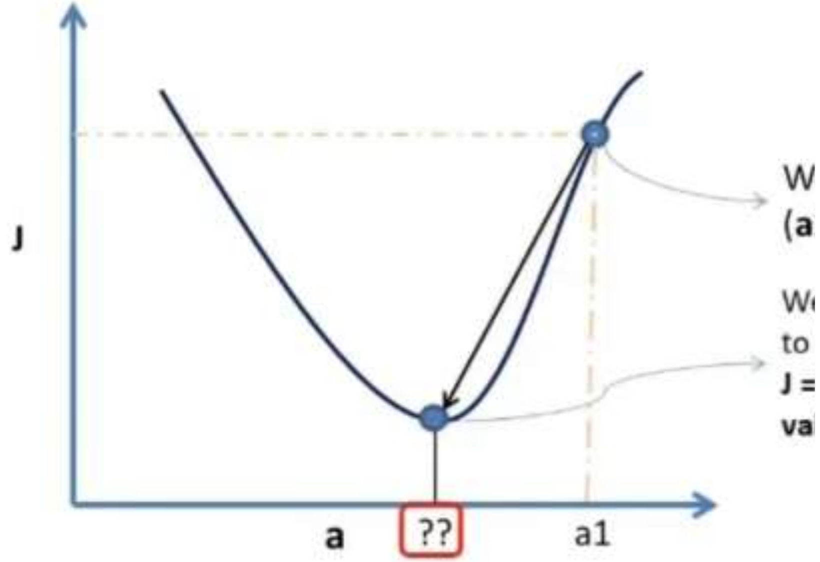
Clearly, **Cost function (J)** is a function of parameter space $\boldsymbol{a} = (\boldsymbol{a_0}, \boldsymbol{a_1})$.

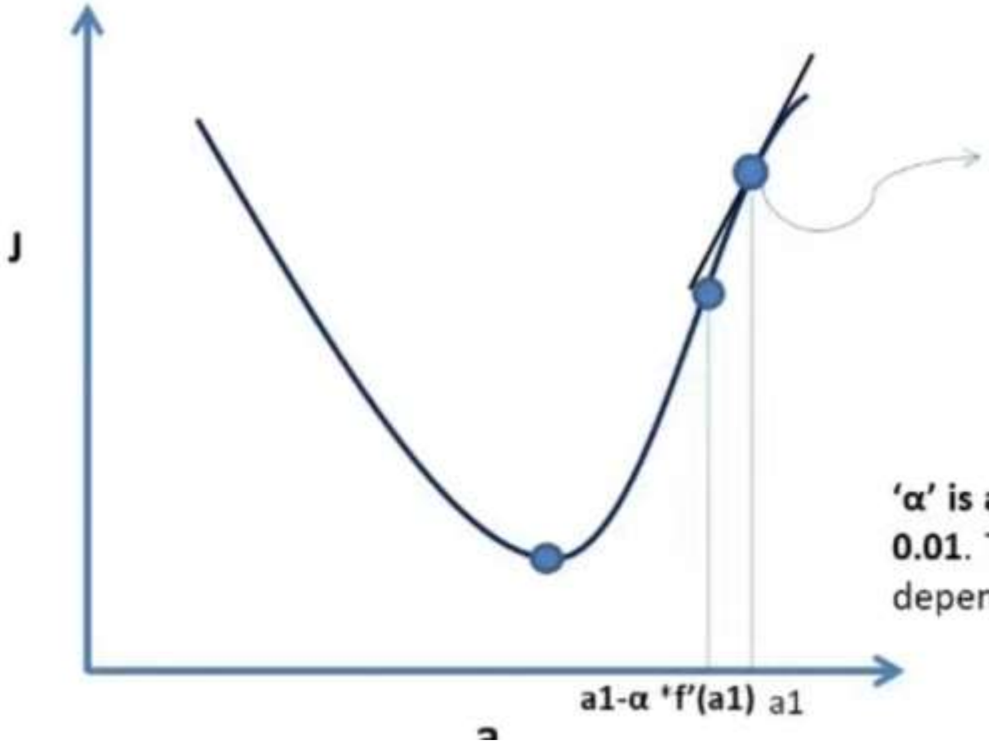
Gradient Descent



Gradient descent is an optimization algorithm used to minimize a function by iteratively moving towards the steepest descent, as defined by the negative of the gradient. It is widely used in machine learning and deep learning for optimizing model parameters

Gradient Descent





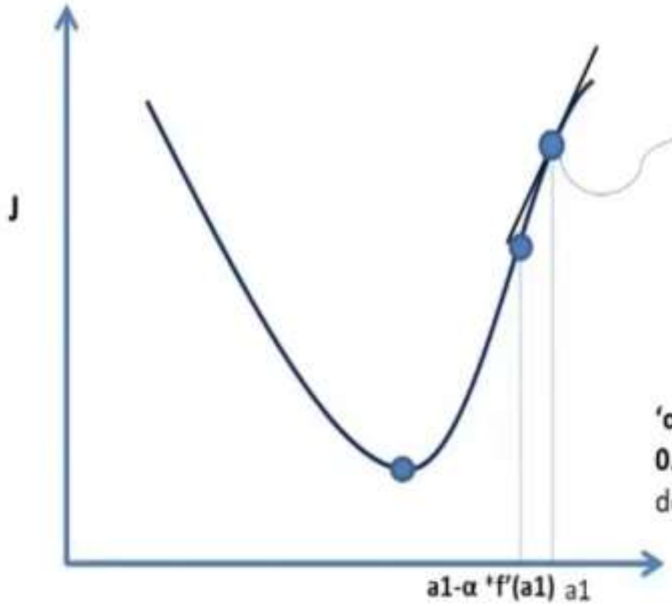
Lets calculate slope at this point.

It is $\frac{dJ}{da}$ **at a1** or f'(a1)

Lets move a step **'α'** in this direction to reach **a1- α *f'(a1)**

'α' is a small fixed quantity in the range of **0.01**. Therefore, the size of our steps is dependent on the slope **f'(a1)**.

Gradient Descent

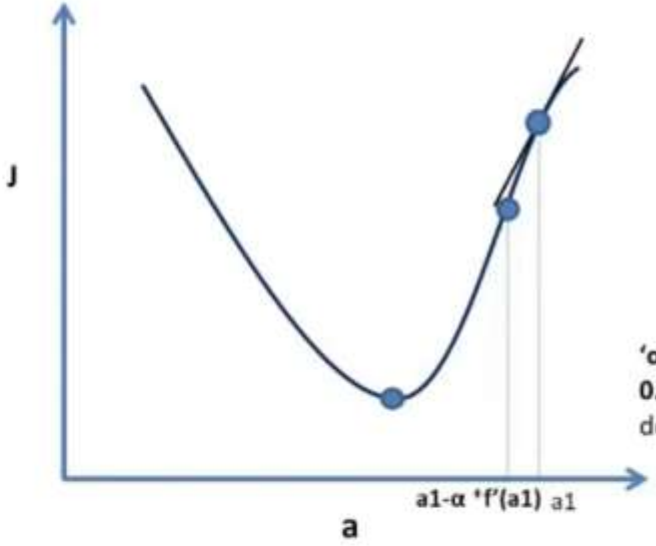


Lets calculate slope at this point.

It is $\frac{dJ}{da}$ **at a1** or f'(a1)

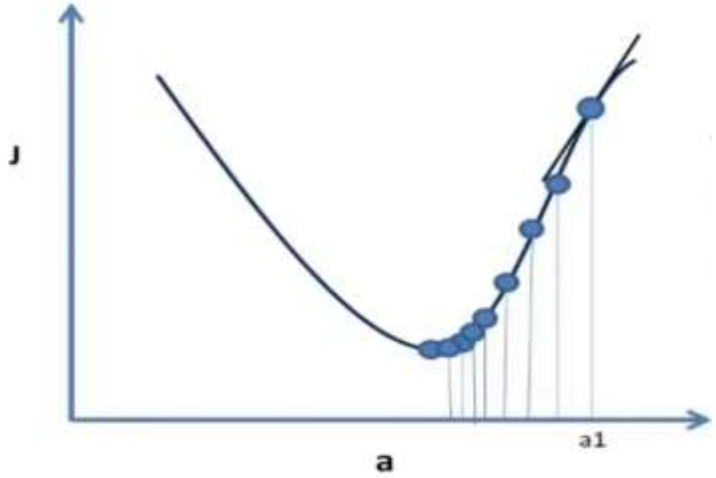
Lets move a step **'α'** in this direction to reach **a1- α *f'(a1)**

'α' is a small fixed quantity in the range of **0.01**. Therefore, the size of our steps is dependent on the slope **f'(a1)**.



Higher the value of slope (which occurs away from minima) **larger the steps and vice versa.**

'α' is a small fixed quantity in the range of **0.01**. Therefore, the size of our steps is dependent on the slope **f'(a1)**.



Steps of Gradient Descent

Step 1:

Calculate $\frac{\partial J}{\partial a_0}$ **(slope)** at the current value of parameter $\boldsymbol{a_0}$.

Calculate $\frac{\partial J}{\partial a_1}$ **(slope)** at the current value of parameter $\boldsymbol{a_1}$.

Step 2:

$$(\boldsymbol{new})\boldsymbol{a_0} = a_0 - \alpha\left(\frac{\partial J}{\partial a_0}\right)$$

$$(\boldsymbol{new})\boldsymbol{a_1} = a_1 - \alpha\left(\frac{\partial J}{\partial a_1}\right)$$

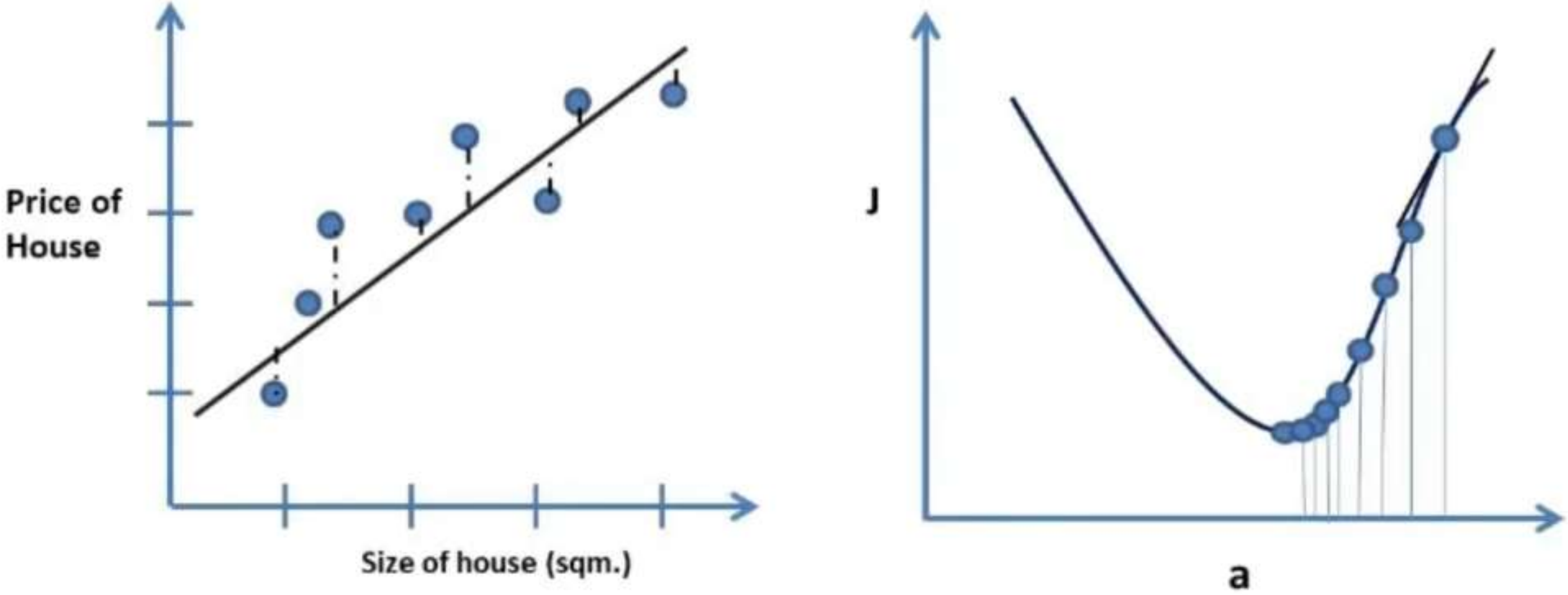
Step 3:

update Cost Function **J** with **new ($\boldsymbol{a_0}$ and $\boldsymbol{a_1}$)**

Repeat Step 1

Gradient Descent

Understanding **Cost Function** and **Gradient Descent Algorithm** was essential but as we shall see we can train a **Linear Regression model** with just **few lines of code** in Python.



There are **inbuilt packages** which implement all of this in an **optimized way** so that we don't have to **worry about all the maths behind it**.

Thank You