## Introduction

Linear Regression is a supervised machine learning algorithm where the predicted output is continuous and has a constant slope. It’s used to predict values within a continuous range, (e.g. sales, price) rather than trying to classify them into categories (e.g. cat, dog). There are two main types:

### Simple regression

Simple linear regression uses traditional slope-intercept form, where $m$ and $b$ are the variables our algorithm will try to “learn” to produce the most accurate predictions. $x$ represents our input data and $y$ represents our prediction.
$y$= mx+b

### Cost Function

Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number which is mean square error.
Given simple linear equation,
Y = mx +b
We can calculate MSE as

$$MSE = \frac{1}{N} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$$

- $N$ is the total number of observations (data points)
- $y_i$ is the actual value of an observation and $mx_i+b$ is our prediction

### GRADIENT DESCENT

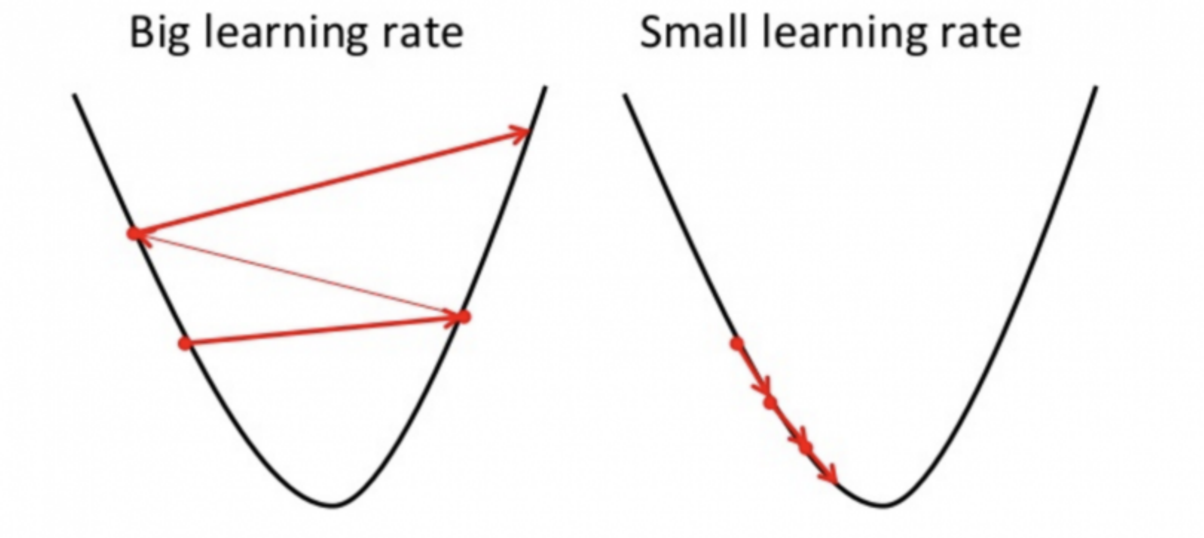
Gradient descent is by far the most popular optimization strategy used in machine learning and deep learning.
Gradient Descent is an optimization algorithm for finding a minimum of a differentiable function and is used to find the values of a function's parameters that minimize a cost function as far as possible.

How gradient descent  works?
The aim is to arrive at the optimal values for the slope m and the intercept b with minimal error which is called the global minima.
We start by defining the initial parameter's values and from there gradient descent uses calculus to iteratively adjust the values so they minimize the given cost-function

## IMPORTANCE OF THE LEARNING RATE

How big the steps are gradient descent takes into the direction of the local minimum are determined by the learning rate, which figures out how fast or slow we will move towards the optimal weights.

For gradient descent to reach the local minimum we must set the learning rate to an appropriate value, which is neither too low nor too high. This is important because if the steps it takes are too big, it may not reach the local minimum because it bounces back and forth between the convex function of gradient descent .If we set the learning rate to a very small value, gradient descent will eventually reach the local minimum but that may take a while .


Big learning rate
Small learning rate