LP-III Machine Learning (2024-25) **Assignment 1:** Implement Gradient Descent Algorithm to find the local minima of a function.	
Student Name:	**Roll No. :**
Batch:	**Division :**

Assignment 1: Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.

Title: Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.
Aim: Find local minima for given function.

Prerequisites: Gradient, Cost Function, Differential and Convex Function

Theory:

Gradient descent is an optimization algorithm that's used when training a machine learning model. It's based on a convex function and tweaks its parameters iteratively to minimize a given function to its local minimum.

A gradient simply measures the change in all weights with regard to the change in error. You can also think of a gradient as the slope of a function. The higher the gradient, the steeper the slope and the faster a model can learn. But if the slope is zero, the model stops learning. In mathematical terms, a gradient is a partial derivative with respect to its inputs.

Imagine you have a machine learning problem and want to train your algorithm with gradient descent to minimize your cost-function $J(w, b)$ and reach its local minimum by tweaking its parameters (w and b). The Figure 1 below shows the horizontal axes representing the parameters (w and b), while the cost function $J(w, b)$ is represented on the vertical axes. Gradient descent is a convex function.

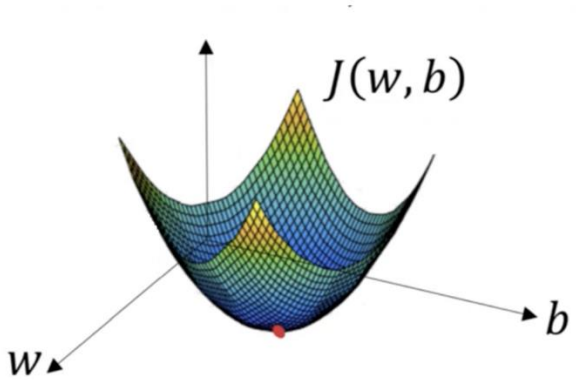

Figure 1: Cost Function J(w, b) mapped against w and b.
(image courtesy: https://builtin.com/data-science/gradient-descent)

We know we want to find the values of w and b that correspond to the minimum of the cost function. To start finding the right values we initialize w and b with some random numbers.

Gradient descent then starts at that point (somewhere around the top of our illustration), and it takes one step after another in the steepest downside direction (i.e., from the top to the bottom of the illustration) until it reaches the point where the cost function is as small as possible.

How big the steps the 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.

For given convex function, $(x+3)^2$ with start point as 2.
$(x+3)^2 = x^2 + 6x + 9$
Learning Rate = 0.1 or any other values
Number of Maximum Iteration = 100

Algorithm:

1. Choose a starting point (initialisation)

   start = 2

2. Calculate gradient at this point.

   diff = Learning Rate * Gradient(x)

   $x_{new} = x_{old}$ - diff

3. Make a scaled step in the opposite direction to the gradient (objective: minimise)
4. Repeat points 2 and 3 until one of the criteria is met:
   - maximum number of iterations reached
   - step size is smaller than the tolerance (due to scaling or a small gradient).

Conclusion: Using concept of Gradient Descent Algorithm, we have found local minima for $(x+3)^2$ with start point as 2.

Questions:

1. What is Local minima.

2. What is Gradient.

3. What is the effect of smaller and larger learning rate.

4. What is Stochastic Gradient Descent (SGD).

5. List out different optimizers?