Gradeint Descent

Zahoor Tanoli (PhD)

Topics covered

- What is gradient decent
- How it works
- Minimizing the Cost Function
- What are the pitfalls

- Note: You are supposed to know about derivatives

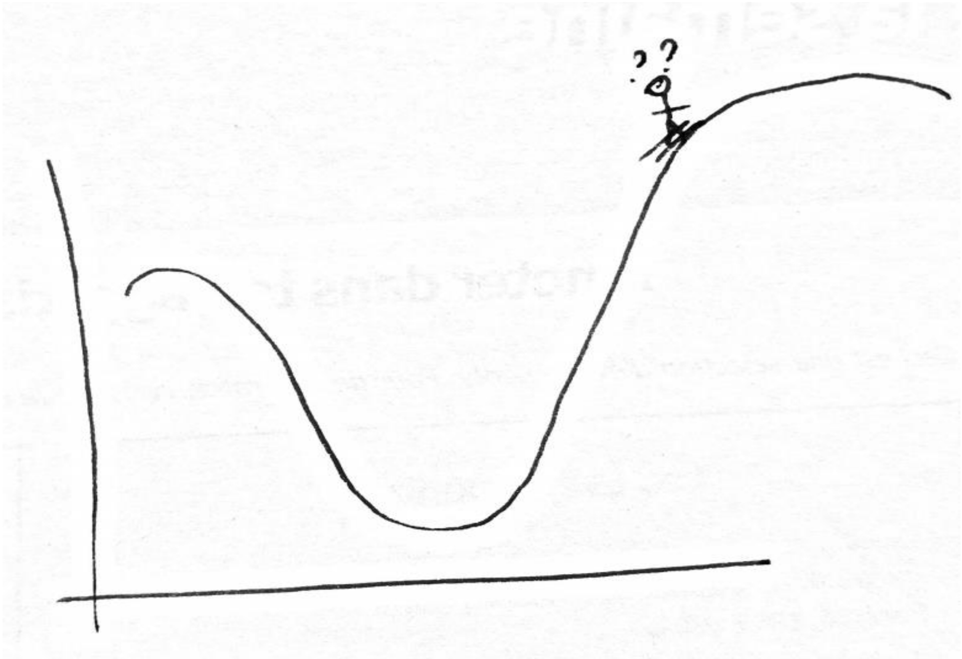
What is Gradient Decent

- Optimisation algorithm
- In derivatives we learn to find minimum of a function
  - Compute the first-order derivative
  - Solve the equation derivative = 0 to find the inflection points
  - Compute the second-order derivative in these points
- When it’s positive, it’s a minimum
- If it’s negative, it’s a maximum

When Used

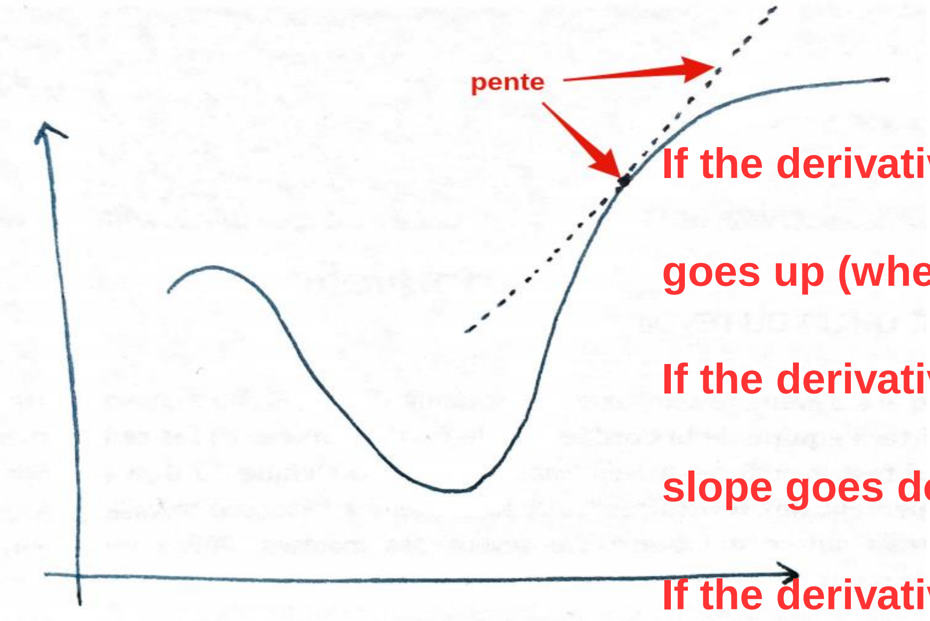
- When you want to train a model
- GD is applied which is:
  - Algorithmic
  - Iterative
  - Works rather well in most cases

How it Works



- Imagine you are a skier on a mountain
- Want to find the lowest point around

Looking Graphically



If the derivative is positive, it means the slope goes up (when going to the right!)

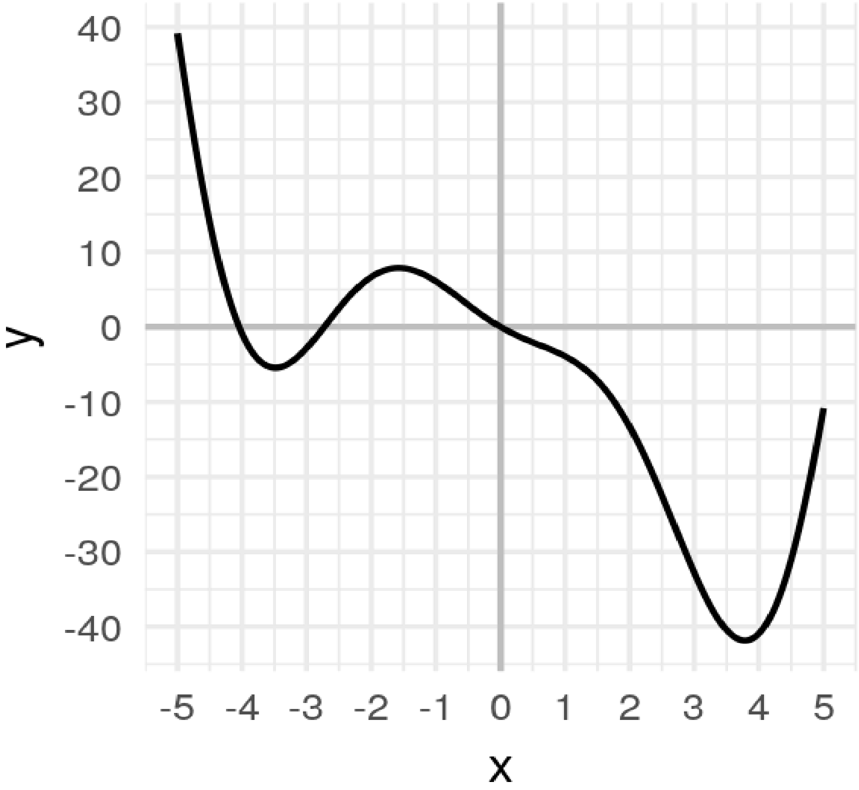
If the derivative is negative, it means the slope goes down

If the derivative is equal to 0 , it means it doesn’t go up or down

- Slope is the derevative
- The value of the derivative is the inclination of the slope at a specific point

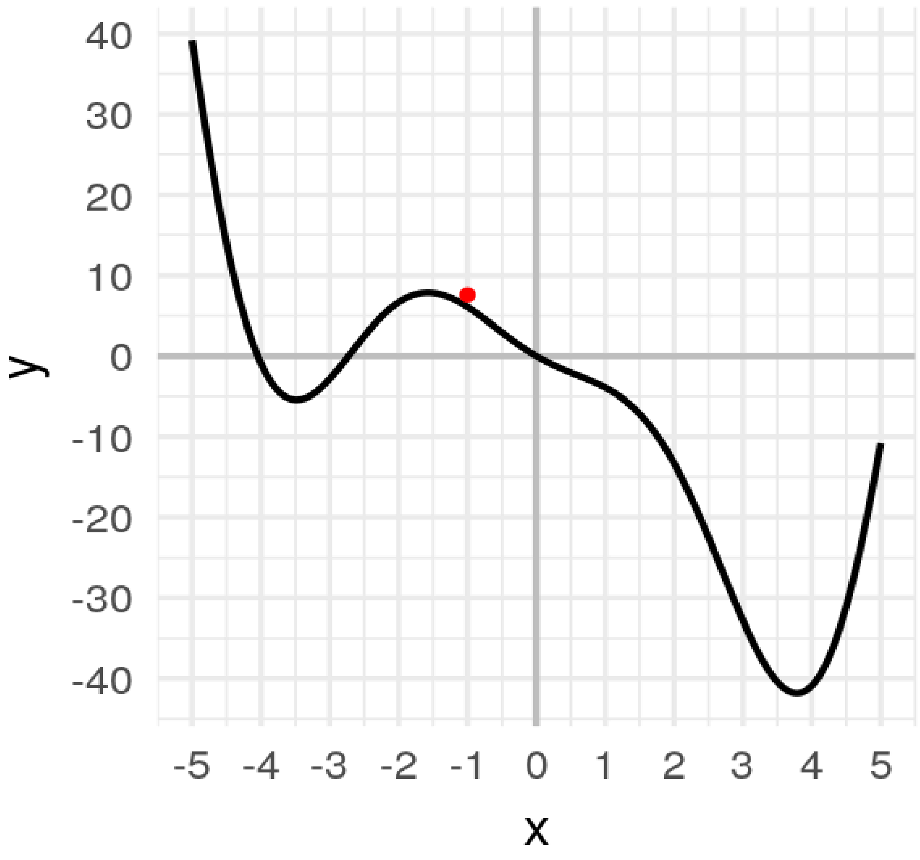
Continued...

- Consider the following function:
  - $f(x) = 2x^2 \cos(x) - 5x$
  - Study it on the [-5, 5] interval:
- Goal is to find the minimum
- There are 3 steps:
  - Take a random point x_0
  - Compute the value of the slope $f'(x0)$
  - Walk in the direction opposite to



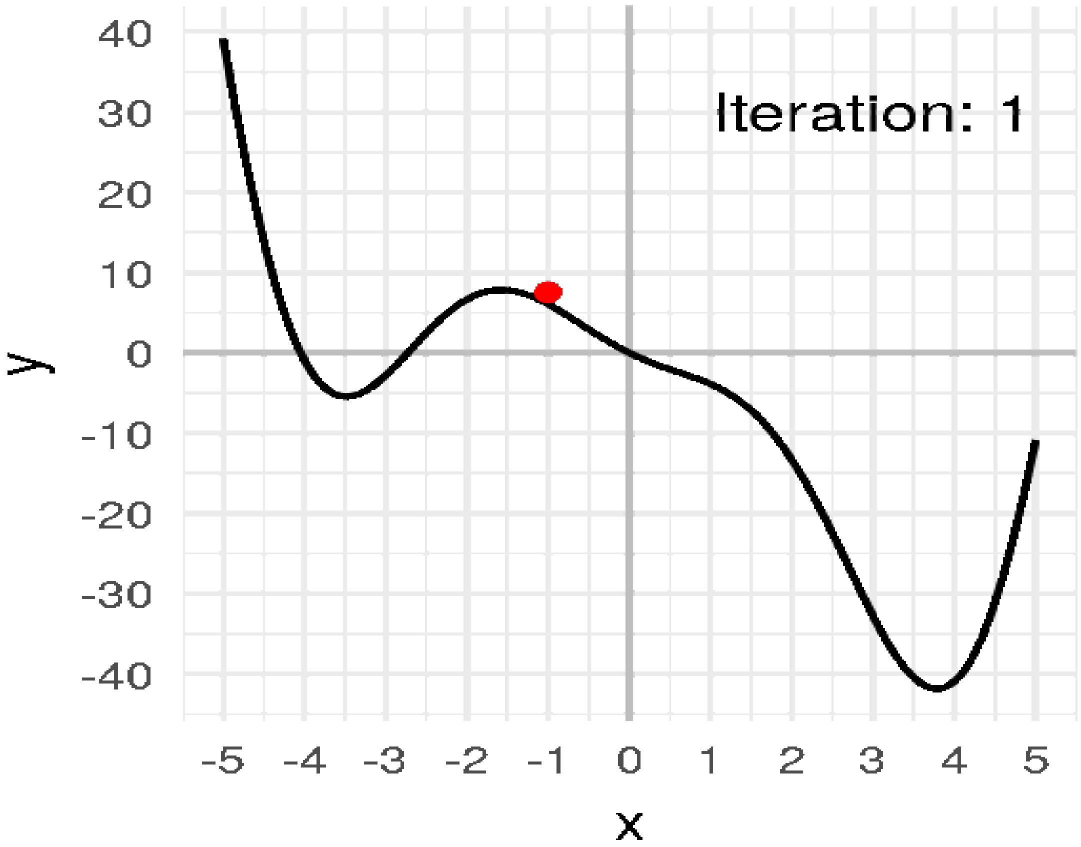
Step-wise process

- Step 1: Take a random point $x_0 = -1$ gives us $f(x_0) = 6.98$
- Step 2: Compute the slope
- Step 3: Walk in the opposite direction on the basis of the value of slope
  - But what will be the step size?



Step size = 0.05

- We're moving.. slowly!
- After a dozen iterations, we obtain convergence:
- Finally gets to the minimum



Cost Function

- In the equation, y = mX+b, ‘m’ and ‘b’ are its parameters
- During the training process, there will be a small change in their values
- Let that small change be denoted by δ
- parameters will be updated as m=m-δm and b=b-δb
- Aim is to find those values of m and b in y = mx+b , for which the error is minimum

Rewriting Cost Function

Parameters with small changes:

$$m = m - \delta m$$
$$b = b - \delta b$$

Given Cost Function for 'N' no of samples

$$Cost = \frac{1}{N}\sum_{i=1}^{N}(Y_i' - Y_i)^2$$

Cost function is denoted by J where J is a function of m and b

$$J_{m,b} = \frac{1}{N}\sum_{i=1}^{N}(Y_i' - Y_i)^2$$

Substituting the term Y'-Y with error for simplicity

$$J_{m,b} = \frac{1}{N}\sum_{i=1}^{N}(Error_i)^2$$

Calculating GD

$$J_{m,b} = \frac{1}{N}\sum_{i=1}^{N}(Error_i)^2$$

$$\frac{\partial J}{\partial m} = 2.Error.\frac{\partial}{\partial m}Erro$$

$$\frac{\partial J}{\partial b} = 2 \cdot Error.\frac{\partial}{\partial b}Erro$$

$$\frac{\partial}{\partial \mathrm{m}}\mathrm{Error} = \frac{\partial}{\partial \mathrm{m}}(\mathrm{Y}' - \mathrm{Y})$$

$$\frac{\partial}{\partial \mathrm{m}}\mathrm{Error} = \frac{\partial}{\partial \mathrm{m}}(\mathrm{mX} + \mathrm{b} - \mathrm{Y})$$

constants

$$\frac{\partial}{\partial \mathrm{m}}\mathrm{Error} = \mathrm{X}$$

$$\frac{\partial}{\partial \mathrm{b}}\mathrm{Error} = \frac{\partial}{\partial \mathrm{b}}(\mathrm{Y}' - \mathrm{Y})$$

$$\frac{\partial}{\partial \mathrm{b}}\mathrm{Error} = \frac{\partial}{\partial \mathrm{b}}(\mathrm{mX} + \mathrm{b} - \mathrm{Y})$$

constants

$$\frac{\partial}{\partial \mathrm{b}}\mathrm{Error} = 1$$

Continued...

- Plugging values in the cost function

$$\frac{\partial J}{\partial m} = 2.\,\text{Error} * X * \text{Learning Rate}$$

Determines the direction to minimize the Error

Determines how large a step to take

$$\frac{\partial J}{\partial b} = 2.\,\text{Error} * \text{Learning Rate}$$

2 in these equations isn't that significant since it just says that we have a learning rate twice as big or half as big

$$\frac{\partial J}{\partial m} = \text{Error} * X * \text{Learning Rate} \qquad \frac{\partial J}{\partial b} = \text{Error} * \text{Learning Rate}$$

$$\textit{Since } m = m - \delta m \qquad \textit{Since } b = b - \delta b$$

$$m^1 = m^0 - \text{Error} * X * \text{Learning Rate} \qquad b^1 = b^0 - \text{Error} * \text{Learning Rate}$$

m^1, b^1 = next position parameters;

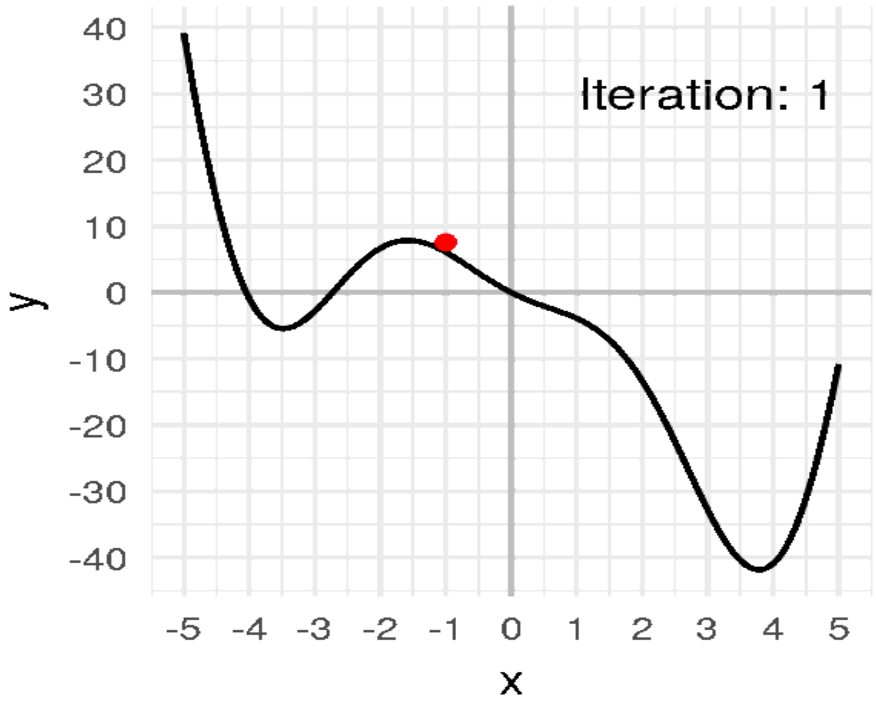
m^0, b^0 = current position parameters

Pitfalls of GD

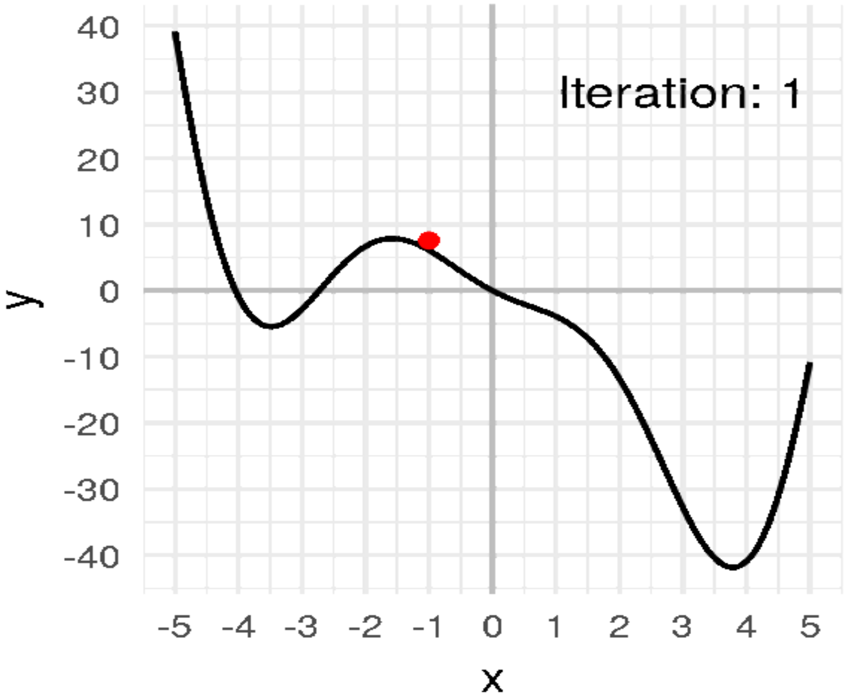
- Selecting good value of learning rate
- Vanishing gradient
- Local minima issue

Learning value selection

- See what happens if we take α=0.001



- See what happens if we take α=0.2



Local Minima

- Notice the final convergence point depends a lot on the initial point
- Sometimes it'll find the global minimum. Other times.. not.
- To avoid this problem, the best way is to run the algorithms multiple times