

kaggle

## Gradient Descent

$$\max_{\vec{w}, w_0} \sum_{i=1}^n y_i \frac{w^T x_i + w_0}{\|w\|} \quad \left. \vphantom{\sum_{i=1}^n} \right\} \text{Gain too}$$

maximisation problem

$$\begin{aligned} &1, 3, 6, 8 \\ &-1, -3, -6, -8 \end{aligned} \quad \begin{aligned} &\Rightarrow +8 \\ &* -1 * 8 \\ &- 8 \end{aligned}$$

$$\min_{\vec{w}, w_0} - \sum_{i=1}^n y_i \frac{w^T x_i + w_0}{\|w\|} \quad \left. \vphantom{\sum_{i=1}^n} \right\} \text{Loss function}$$

$$\min_{\vec{w}, w_0} f(w, w_0)$$

$$\min_{w_0, w_1, \dots, w_n} f(w_0, w_1, \dots, w_n)$$

$\Downarrow$   
 perform multivariate optimisation

## Multivariate Differentiation

$$y = f_1(x) \rightarrow \text{single var}$$

$$y = f_2(x_1, x_2, \dots, x_n) \rightarrow \text{multivariate}$$

$$\frac{d f(x)}{dx} = f'(x)$$

$$\frac{d f(x_1, x_2, \dots, x_n)}{dx} = ?$$

$$z = f(\underline{x}, y) \quad \text{find } \frac{dz}{dx}$$

$$y = g(x)$$

$$z = f(x, g(x))$$

Example

$$Z = f(x, y) = x^2 + y^2$$

$$Z = x^2 + [g(x)]^2$$

$$y = g(x)$$

$$\frac{dz}{dx} = 2x + 2g(x) \cdot g'(x)$$

$$z' = 2x + 2y \cdot y'$$

$$\frac{dz}{dx} = 0$$

$$\frac{dz}{dy} = 0$$

Partial derivative

$$\frac{\partial z}{\partial x} = \frac{df(x, y = \text{const})}{dx}$$

$$\frac{\partial z}{\partial y} = \frac{df(y, x = \text{const})}{dy}$$

$\xi_1$

$$f(x, y) = 2x^2y + 3y^3x^2 + 3y$$

$$\frac{\partial f}{\partial x} = 4xy + 6y^3x + 0$$

$$\frac{\partial f}{\partial y} = 2x^2 + 9y^2x^2 + 3$$

Gradient

Goal:

$$\min_{w_0, w_1, w_2, \dots, w_n} f(w_0, w_1, \dots, w_n)$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial w_0} \\ \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \end{bmatrix}$$

## Gradient Descent

$$y = x^2 - 30$$

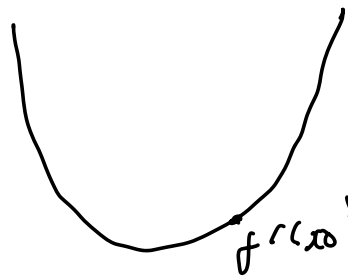
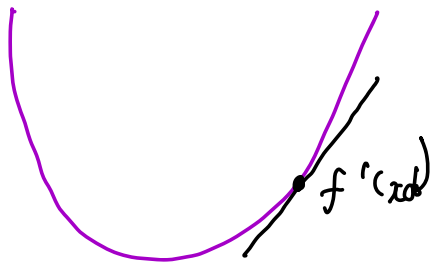
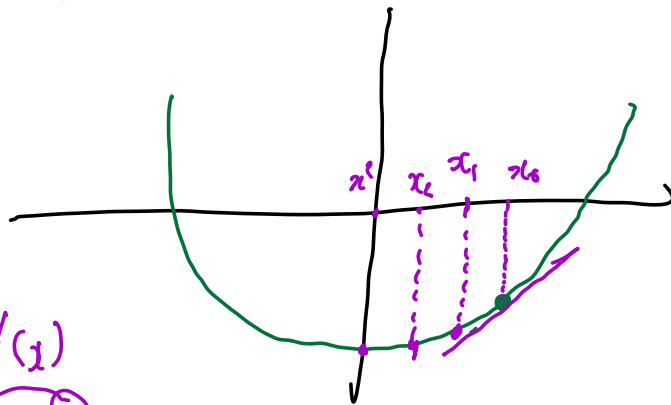
$$\frac{\partial y}{\partial x} = 2x$$

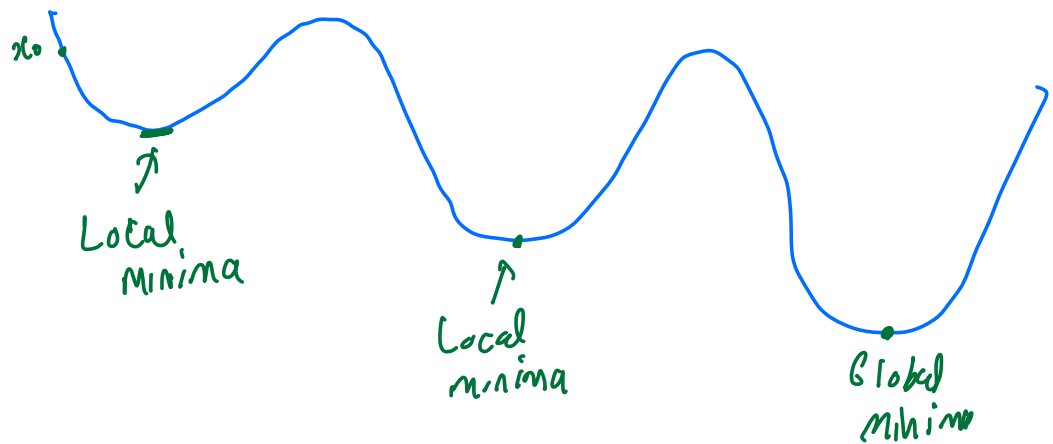
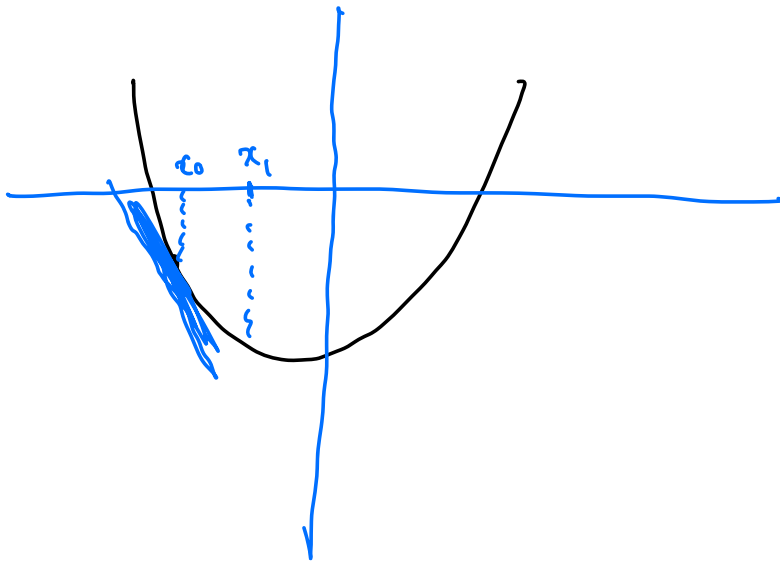
step 1

$$f(x_0) = x_0^2 - 30$$

$$f'(x_0) = 2x_0$$

$$x_1 = x_0 - \eta \underbrace{f'(x_0)}_{\substack{2x_0 \\ \text{+ve}}}$$

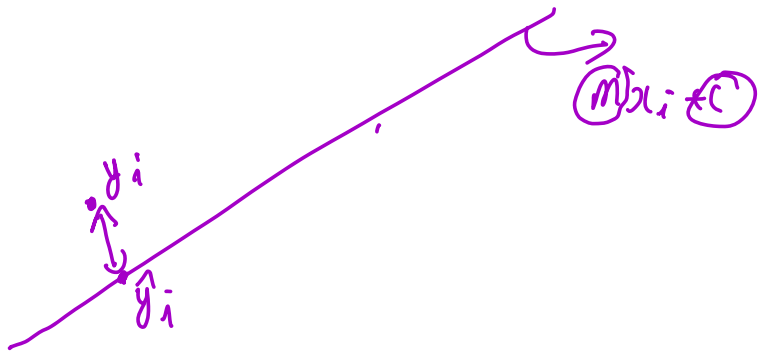




$$w_1 [next] = w_1 [current] - \eta \frac{\partial f}{\partial w_1}$$

$$\vdots$$

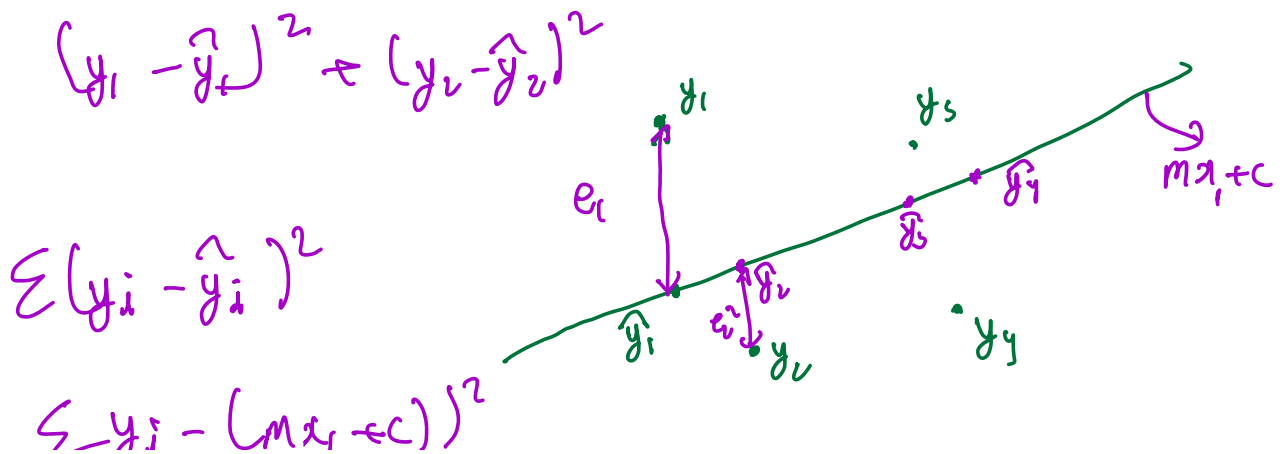
$$w_n [next] = w_n [current] - \eta \frac{\partial f}{\partial w_n}$$



$$J(m, c) = \sum_{i=1}^n (y_i - (m x_i + c))^2$$

$$\frac{\partial J}{\partial m} = 2 \sum_{i=1}^n (y_i - (m x_i + c)) (-x_i)$$

$$\frac{\partial J}{\partial c} = 2 \sum_{i=1}^n (y_i - (m x_i + c)) (-1)$$



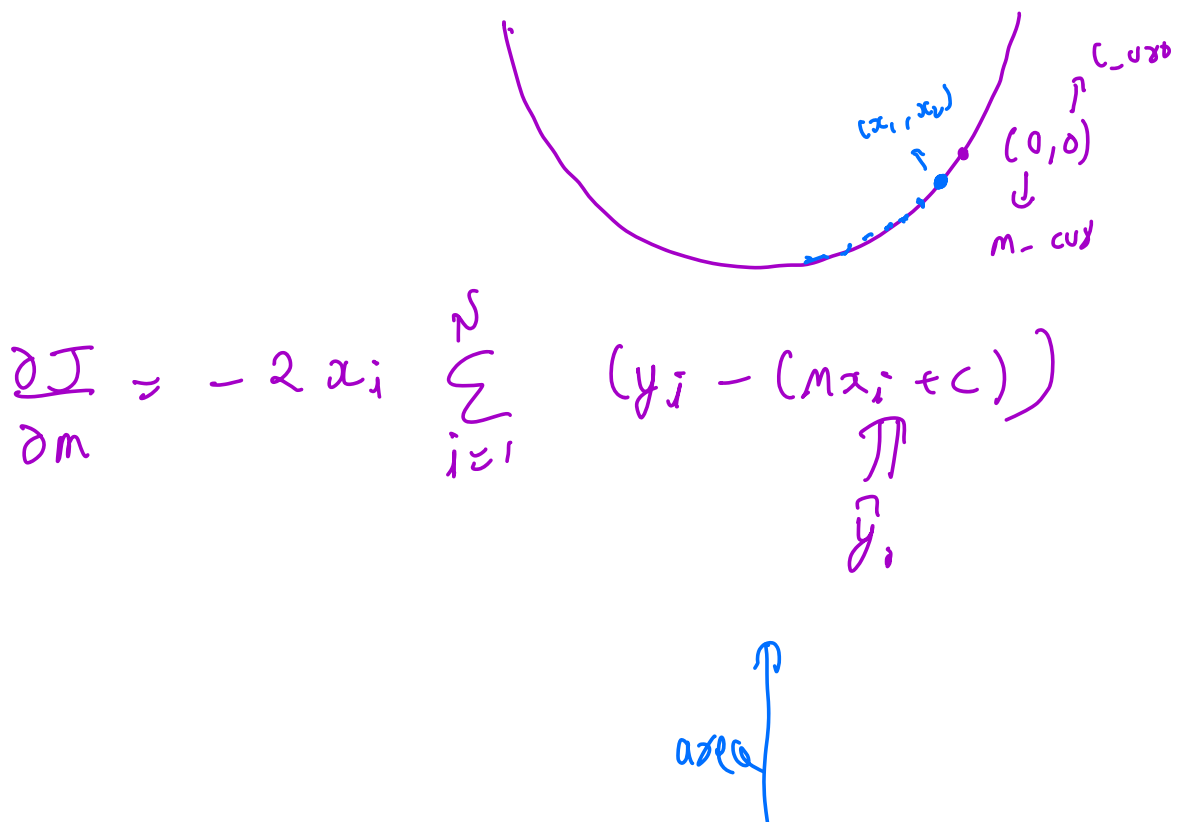
```

[74]: Implement gradient descent function
      Takes in X, y, current m and c (both initialised to 0), num_iterations,
      returns gradient at current m and c for each pair of m and c

def gradient(X, y, m_current=0, c_current=0, iters=1000, learning_rate=0.01):
    N = float(len(y))
    gd_df = pd.DataFrame( columns = ['m_current', 'c_current', 'cost'])
    for i in range(iters):
        y_current = (m_current * X) + c_current
        cost = sum([data**2 for data in (y-y_current)]) / N
        m_gradient = -(2/N) * sum(X * (y - y_current))
        c_gradient = -(2/N) * sum(y - y_current)
        m_current = m_current - (learning_rate * m_gradient)
        c_current = c_current - (learning_rate * c_gradient)
        gd_df.loc[i] = [m_current, c_current, cost]
    return(gd_df)

n [75]: # print gradients at multiple (m, c) pairs
        # notice that gradient decreased gradually towards 0
        # we have used 1000 iterations, can use more if needed
        gradients = gradient(X,y)

```





↓  $\psi_1$

$\frac{1}{Q_{051}}$

3, 5, 8

$\frac{1}{3}$ ,  $\frac{1}{5}$ ,  $\frac{1}{8}$

-3, -5,  $\boxed{-8}$