MATHEMATICAL FOUNDATIONS OF LINEAR REGRESSION USING GRADIENT DESCENT

1. LINEAR REGRESSION MODEL

We assume a **linear hypothesis function**, which proposes a linear relationship between the input feature and the predicted output:

$$\hat{y} = w \cdot x + b$$

Where:

- $\hat{y}$ = predicted output (dependent variable)
- x = input feature (independent variable)
- w = weight (representing the slope of the line)
- b = bias (representing the y-intercept of the line)

2. COST FUNCTION (MEAN SQUARED ERROR - MSE)

To quantify the accuracy of our model, we define a cost function that measures the average squared difference between the predicted and actual values. This is known as the Mean Squared Error (MSE):

$$J(w, b) = (1 / 2m) \Sigma (\hat{y}_i - y_i)^2$$

Expanding the predicted output ($\hat{y}_i$), the cost function becomes:

$$J(w, b) = (1 / 2m) \Sigma (w \cdot x_i + b - y_i)^2$$

Where:

- $J(w, b)$ = total cost for the given parameters *w* and *b*
- m = number of training examples in the dataset
- y_i = actual output for the i-th training example
- $\hat{y}_i$ = predicted output for the i-th training example

Note: The factor 1/2 is included for mathematical convenience. When we differentiate the cost function, the '2' from the squared term will cancel out this '1/2', simplifying the resulting gradient equations.

3. OBJECTIVE

The primary goal of linear regression using gradient descent is to **minimize J(w, b)**. This means finding the optimal values for **w** (weight) and **b** (bias) that result in the smallest possible error between our model's predictions and the actual data.

4. GRADIENT DESCENT ALGORITHM

Gradient Descent is an iterative optimization algorithm used to minimize the cost function. It works by adjusting the parameters **w** and **b** in the direction opposite to the gradient of the cost function, thereby moving towards the minimum.

The update rules for each parameter are:

`w := w - α * ∂J/∂w` `b := b - α * ∂J/∂b`

Where:

- **α (alpha)** = learning rate, a hyperparameter that controls the step size of each update. A well-chosen learning rate is crucial for efficient convergence.
- **∂J/∂w** = partial derivative of the cost function with respect to *w* (gradient for *w*)
- **∂J/∂b** = partial derivative of the cost function with respect to *b* (gradient for *b*)

5. DERIVATIVE OF COST W.R.T W

To find how the cost function changes with respect to the weight (w), we compute its partial derivative:

$$\partial J/\partial w = (1/m) \Sigma ((w \cdot x_i + b - y_i) \cdot x_i)$$

This derivative indicates the slope of the cost function surface in the direction of **w**. Moving in the negative direction of this slope will decrease the cost.

6. DERIVATIVE OF COST W.R.T B

Similarly, to understand how the cost function changes with respect to the bias (b), we compute its partial derivative:

$$\partial J/\partial b = (1/m) \Sigma (w \cdot x_i + b - y_i)$$

This derivative indicates the slope of the cost function surface in the direction of **b**. Moving in the negative direction of this slope will decrease the cost.

7. FINAL UPDATE RULES

Combining the gradient descent algorithm with the calculated derivatives, the parameter update rules for each iteration are:

$$w := w - \alpha * (1/m) \Sigma ((w \cdot x_i + b - y_i) \cdot x_i)$$

$$b := b - \alpha * (1/m) \Sigma (w \cdot x_i + b - y_i)$$

These equations are applied simultaneously in each step of the gradient descent process until convergence is achieved.

8. INTERPRETATION OF TERMS

Term	Meaning
w	Weight (slope) – determines the influence of the input feature on the output.
b	Bias (intercept) – represents the predicted output when all input features are zero.
α (alpha)	Learning rate – controls the magnitude of parameter updates. A higher value speeds up convergence but risks overshooting; a lower value ensures smoother convergence but can be very slow.
J(w, b)	Cost function (Mean Squared Error) – the metric we aim to minimize, representing the average error of our model.
∂J/∂w	Gradient with respect to weight – indicates the direction and steepness of the cost function's ascent relative to *w*.
∂J/∂b	Gradient with respect to bias – indicates the direction and steepness of the cost function's ascent relative to *b*.
m	Number of training examples – the total count of data points used for training the model.

9. CONVERGENCE

Gradient Descent iteratively updates **w** and **b** until one of the following conditions is met:

- The change in the cost function **J(w, b)** between iterations becomes very small (indicating that a minimum has been reached or approached).
- A predefined maximum number of iterations (epochs) is reached, preventing the algorithm from running indefinitely.

At convergence, the values of **w** and **b** are considered optimal, representing the best-fit line for the given training data.

10. SUMMARY

Gradient Descent is a fundamental optimization algorithm central to many machine learning models. In the context of linear regression, it provides an efficient method to find the optimal **weight (w)** and **bias (b)** values that minimize the **Mean Squared Error (J(w, b))**. By calculating the gradients (partial derivatives) of the cost function with respect to each parameter and iteratively adjusting the parameters in the direction of the negative gradient, Gradient Descent systematically steers the model towards the best possible linear fit for the data.