Regularization

Regularization is a technique used in machine learning to prevent overfitting, which otherwise causes models to perform poorly on unseen data. When training a machine learning model, the model can be easily overfitted or under fitted. To avoid this, we use regularization in machine learning to properly fit the model to our test set. Regularization techniques help reduce the possibility of overfitting and help us obtain an optimal model.

Regularization is implemented by modifying the loss function. In a typical machine learning model, the loss function measures how well the model's predictions match the actual data. Regularization adds an extra term to this loss function that penalizes large weights. General Form of Regularized Cost Function

Cost=Loss+λ×Penalty

- **Loss**: Error on training data
- **Penalty**: Function of model parameters
- **λ (lambda)**: Regularization parameter controlling strength
    - Large λ → simpler model
    - Small λ → more complex model

Loss = Original Loss + λ × Regularization Term

Original Loss: Measures the difference between the model's predictions and the actual values.

Regularization Term: Adds a penalty for larger weights.

λ (lambda): A hyperparameter that controls the strength of the penalty. A larger λ means more regularization.

1.L1 Regularization (Lasso Regression)

It adds the absolute value of the weights to the loss function.

$$\text{Loss} = \text{Original Loss} + \lambda \sum_i |W_i|$$

Imagine we have a dataset with many features (variables), but not all of them are important for predicting the output. Using L1 regularization can help the model focus on the most significant features by reducing the weights of less important ones to zero.

2.L2 Regularization (Ridge Regression)

It adds the square of the weights to the loss function.

$$\text{Loss} = \text{Original Loss} + \lambda \sum_i W_i^2$$

Suppose we're building a model to predict house prices based on various features like size, number of rooms, age, location, etc. L2 regularization helps ensure that the model doesn't assign too much importance to any one feature and considers all of them in a balanced way.

Example : House Price Prediction

Scenario: Imagine we're building a model to predict the price of a house based on various features like:

Size of the house (square feet)

Number of bedrooms

Location (urban, suburban, rural)

Age of the house

Presence of a swimming pool, garage, etc.

Without regularization, our model might give too much importance to some features, like the presence of a swimming pool or the age of the house, even if those features don't significantly influence the price. This could lead to overfitting, especially if the training data contains houses with unusual characteristics (outliers). For example, maybe one very expensive house has a large swimming pool, and the model might learn that "swimming pools" lead to a high price, which isn't true in general.