# Implementation of Perceptron and Delta Learning Rule

**Aim**: To implement the perceptron learning algorithm and delta learning rule for classification tasks using supervised learning, and to study their convergence behavior.

**Problem Statement**: In machine learning, classification tasks require models that can distinguish between different classes based on input features. The perceptron is one of the earliest and simplest neural network models used for binary classification. It adjusts its weights iteratively to correctly classify linearly separable data. However, for non-linearly separable data or cases requiring continuous learning, the delta learning rule (or Widrow-Hoff rule) is used as a generalized learning mechanism. Implement both learning rules and analyze their working, convergence, and limitations in learning classification boundaries.

**Theory**: Introduction to Perceptron: The perceptron is a fundamental unit of a neural network, functioning as a linear binary classifier. It takes multiple inputs, applies weights, sums them up, and passes the result through an activation function (usually a step function) to produce an output.

Mathematical Model: $y = f(\Sigma (w\ x\ ) + b)$ where $x$ represents the input, $w$ the weight, $b$ the bias, and $f$ is the activation function. The output is 1 if the weighted sum exceeds a threshold, and 0 otherwise

Perceptron Learning Rule: Weights are updated based on the error between the predicted and actual output as follows: $\Delta w = \eta (t - y) x$ where $\eta$ is the learning rate, $t$ is the target output, and $y$ is the predicted output.

Delta Learning Rule (Widrow-Hoff Rule): The delta rule extends the perceptron learning rule by minimizing the mean squared error between the predicted and actual outputs. Unlike the perceptron rule, which updates weights only for misclassified patterns, the delta rule continuously adjusts weights using gradient descent to minimize the loss function.

Mathematical Representation: $\Delta w = \eta (t - y) f'(net) x$
where $f'(net)$ is the derivative of the activation function and $(t - y)$ is the instantaneous error.

Implementation Approach:

- Initialize weights and bias to small random values.
- Present input-output training pairs to the network.
- Compute the output using the activation function.

- Update weights according to the perceptron or delta rule.
- Repeat until the error converges or maximum epochs are reached

Comparison:

• The perceptron rule works only for linearly separable data.
• The delta rule can handle continuous outputs and non-linear activation functions.
• The delta rule forms the foundation of backpropagation in multi-layer neural networks.

Applications:

• Pattern recognition
• Binary classification problems
• Adaptive signal processing
• Foundation for deep learning