

A PYTHON PROGRAM TO IMPLEMENT SINGLE LAYER PERCEPTRON

Aim:

To implement python program for the single layer perceptron.

Algorithm:

Step 1: Import Necessary Libraries:

- Import numpy for numerical operations.

Step 2: Initialize the Perceptron:

- Define the number of input features (input_dim).
- Initialize weights (W) and bias (b) to zero or small random values.

Step 3: Define Activation Function:

- Choose an activation function (e.g., step function, sigmoid, or ReLU).
- User Defined function - sigmoid_func(x):
 - o Compute $1/(1+\text{np.exp}(-x))$ and return the value.
- User Defined function - der(x):
 - o Compute the product of value of sigmoid_func(x) and $(1 - \text{sigmoid_func}(x))$ and return the value.

Step 4; Define Training Data:

- Define input features (X) and corresponding target labels (y).

Step 5: Define Learning Rate and Number of Epochs:

- Choose a learning rate (alpha) and the number of training epochs.

Step 6: Training the Perceptron:

- For each epoch:
 - o For each input sample in the training data:
 - o Compute the weighted sum of inputs (z) as the dot product of input features and weights plus bias ($z = \text{np.dot}(X[i], W) + b$).

- o Apply the activation function to get the predicted output (y_{pred}).
- o Compute the error ($\text{error} = y[i] - y_{\text{pred}}$).
- o Update the weights and bias using the learning rate and error ($W += \alpha * \text{error} * X[i]$; $b += \alpha * \text{error}$).

Step 7: Prediction:

- Use the trained perceptron to predict the output for new input data.

Step 8: Evaluate the Model:

- Measure the performance of the model using metrics such as accuracy, precision, recall, etc.