LAB-5

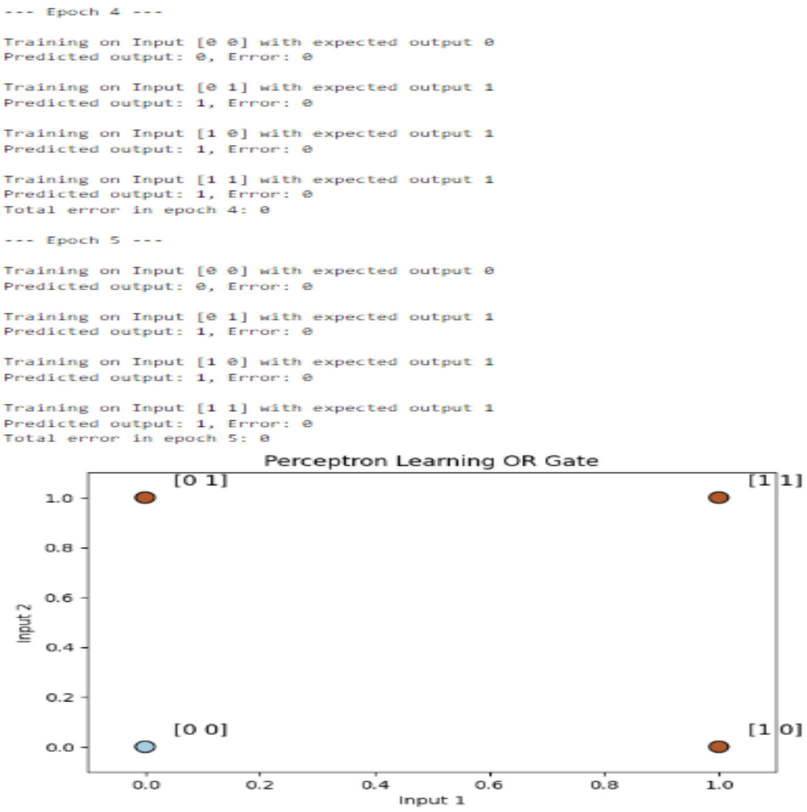
Aim: Write a program to implement Perceptron Neural Network

SOURCE CODE:

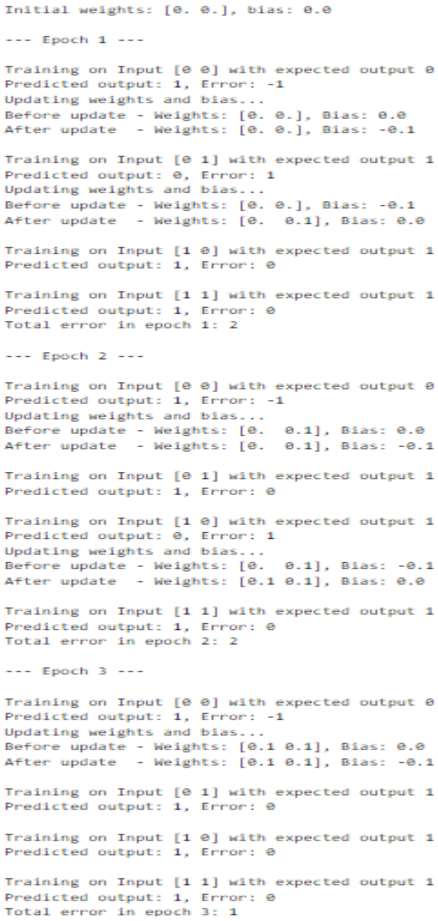
```
import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation


def step_function(x):

  return np.where(x >= 0, 1, 0)


class Perceptron:

  def _init_(self, input_size, learning_rate=0.1, epochs=5):

    self.weights = np.zeros(input_size + 1)

    self.learning_rate = learning_rate

    self.epochs = epochs

    self.history = []


  def predict(self, x):

    z = np.dot(x, self.weights[1:]) + self.weights[0]

    return step_function(z)


  def train(self, X, y):

    print(f"Initial weights: {self.weights[1:]}, bias: {self.weights[0]}")

    for epoch in range(self.epochs):

      total_error = 0

      print(f"\n--- Epoch {epoch + 1} ---")

      for i in range(X.shape[0]):

        prediction = self.predict(X[i])

        error = y[i] - prediction

        total_error += abs(error)

        print(f"\nTraining on Input {X[i]} with expected output {y[i]}")

        print(f"Predicted output: {prediction}, Error: {error}")

        if error != 0:

          print(f"Updating weights and bias...")
```

```
                    print(f"Before update - Weights: {self.weights[1:]}, Bias: {self.weights[0]}")
                self.weights[1:] += self.learning_rate * error * X[i]
                self.weights[0] += self.learning_rate * error
                if error != 0:
                    print(f"After update  - Weights: {self.weights[1:]}, Bias: {self.weights[0]}")
            self.history.append((self.weights.copy(), total_error))
            print(f"Total error in epoch {epoch + 1}: {total_error}")

X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
y = np.array([0, 1, 1, 1])

perceptron = Perceptron(input_size=2, learning_rate=0.1, epochs=5)
perceptron.train(X, y)

def plot_decision_boundary_evolution(X, y, model):
    fig, ax = plt.subplots()
    ax.scatter(X[:, 0], X[:, 1], c=y, s=100, edgecolors='k', cmap=plt.cm.Paired)
    for i, txt in enumerate(X):
        ax.annotate(f"{X[i]}", (X[i][0]+0.05, X[i][1]+0.05), fontsize=12)
    ax.set_xlim([-0.1, 1.1])
    ax.set_ylim([-0.1, 1.1])
    ax.set_xlabel('Input 1')
    ax.set_ylabel('Input 2')
    ax.set_title('Perceptron Learning OR Gate')

    def update(frame):
        ax.collections.clear()
        ax.contourf(xx, yy, Z_history[frame], alpha=0.8, cmap=plt.cm.Paired)
        ax.scatter(X[:, 0], X[:, 1], c=y, s=100, edgecolors='k', cmap=plt.cm.Paired)
        ax.annotate(f"Epoch: {frame + 1}", (0.7, 0.9), fontsize=15, color='red')

    x_min, x_max = -0.1, 1.1
    y_min, y_max = -0.1, 1.1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01),
                         np.arange(y_min, y_max, 0.01))

    Z_history = []
    for weights, _ in model.history:
        Z = step_function(np.dot(np.c_[xx.ravel(), yy.ravel()], weights[1:]) + weights[0])

        Z = Z.reshape(xx.shape)

        Z_history.append(Z)


    ani = FuncAnimation(fig, update, frames=len(model.history), interval=500, repeat=False)

    plt.show()


plot_decision_boundary_evolution(X, y, perceptron)


errors = [error for _, error in perceptron.history]

plt.plot(range(1, len(errors) + 1), errors, marker='o')

plt.title('Perceptron Error over Epochs (OR Gate)')

plt.xlabel('Epochs')

plt.ylabel('Total Error')

plt.grid(True)

plt.show()
```

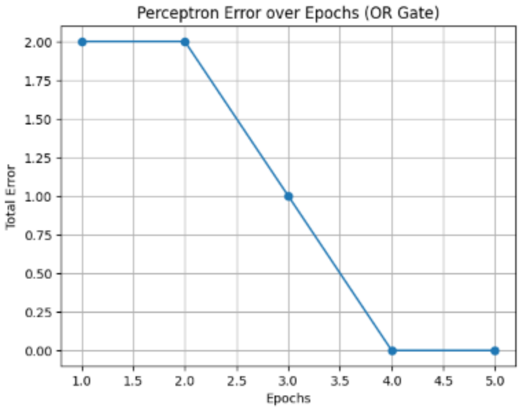
Output:



```
--- Epoch 4 ---

Training on Input [0 0] with expected output 0
Predicted output: 0, Error: 0

Training on Input [0 1] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 0] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 1] with expected output 1
Predicted output: 1, Error: 0
Total error in epoch 4: 0

--- Epoch 5 ---

Training on Input [0 0] with expected output 0
Predicted output: 0, Error: 0

Training on Input [0 1] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 0] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 1] with expected output 1
Predicted output: 1, Error: 0
Total error in epoch 5: 0
```



```
Initial weights: [0. 0.], bias: 0.0

--- Epoch 1 ---

Training on Input [0 0] with expected output 0
Predicted output: 1, Error: -1
Updating weights and bias...
Before update - Weights: [0. 0.], Bias: 0.0
After update  - Weights: [0. 0.], Bias: -0.1

Training on Input [0 1] with expected output 1
Predicted output: 0, Error: 1
Updating weights and bias...
Before update - Weights: [0. 0.], Bias: -0.1
After update  - Weights: [0.  0.1], Bias: 0.0

Training on Input [1 0] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 1] with expected output 1
Predicted output: 1, Error: 0
Total error in epoch 1: 2

--- Epoch 2 ---

Training on Input [0 0] with expected output 0
Predicted output: 1, Error: -1
Updating weights and bias...
Before update - Weights: [0.  0.1], Bias: 0.0
After update  - Weights: [0.  0.1], Bias: -0.1

Training on Input [0 1] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 0] with expected output 1
Predicted output: 0, Error: 1
Updating weights and bias...
Before update - Weights: [0.  0.1], Bias: -0.1
After update  - Weights: [0.1 0.1], Bias: 0.0

Training on Input [1 1] with expected output 1
Predicted output: 1, Error: 0
Total error in epoch 2: 2

--- Epoch 3 ---

Training on Input [0 0] with expected output 0
Predicted output: 1, Error: -1
Updating weights and bias...
Before update - Weights: [0.1 0.1], Bias: 0.0
After update  - Weights: [0.1 0.1], Bias: -0.1

Training on Input [0 1] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 0] with expected output 1
Predicted output: 1, Error: 0

Training on Input [1 1] with expected output 1
Predicted output: 1, Error: 0
Total error in epoch 3: 1
```


Perceptron Error over Epochs (OR Gate)
Total Error
Epochs