Koushik Sahu
118CS0597
Soft Computing Lab – 2
23rd January 2022

Code:

```
import numpy as np
import numpy.typing as npt
from typing import Tuple, List
import matplotlib.pyplot as plt

"""
    Author: Koushik Sahu
    Created: 2022-01-17 15:31 IST
"""


def get_data(gate: str) -> Tuple[npt.NDArray, npt.NDArray]:
    X: List[List[int]] = list()
    y: List[int] = list()

    i: int
    j: int
    for i in range(2):
        for j in range(2):
            X.append([i, j, 1])

    for i, j, _ in X:
        if gate == 'and':
            y.append(i&j)
        elif gate == 'or':
            y.append(i|j)
        elif gate == 'xor':
            y.append(i^j)

    for smp in X:
        for idx, val in enumerate(smp):
            if val == 0:
                smp[idx] = -1

    for idx, val in enumerate(y):
        if val == 0:
            y[idx] = -1
```

```
    return np.array(X, dtype=np.int64), np.array(y, dtype=np.int64)


class Hebbian:
    def __init__(self, gate: str):
        self.X: npt.NDArray
        self.y: npt.NDArray
        self.X, self.y = get_data(gate)
        self.gate = gate

        self.n: int = self.X.shape[1]
        self.wt: npt.NDArray[np.float64] = np.zeros(shape=(1, self.n), dtype=np.float64)
        self.b: np.float64 = np.float64(0)


    def fit(self):
        print(f'{self.gate.upper()} gate')
        print('**************')
        print('Weight updates')
        i: npt.NDArray[np.float64]
        j: np.int64
        for i, j in zip(self.X, self.y):
            self.wt += i*j
            self.b += j
            print(self.wt)
        print()

    def plot_decision_boundary(self):
        for i, j, _ in self.X:
            plt.scatter(i, j)
        x_val: npt.NDArray[np.float64] = np.linspace(-10, 10, num=10000, dtype=np.float64)
        y_val: npt.NDArray[np.float64] = (self.wt[0][0]*x_val + self.wt[0][2])
        if self.gate != 'xor': y_val /= (-1*self.wt[0][1]);
        plt.plot(x_val, y_val)
        plt.xlabel('x1')
        plt.ylabel('x2')
        plt.title(f'{self.gate.upper()} gate')
        plt.show()


if __name__ == '__main__':
    for gate in ['and', 'or', 'xor']:
        hb: Hebbian = Hebbian(gate)
        hb.fit()
```

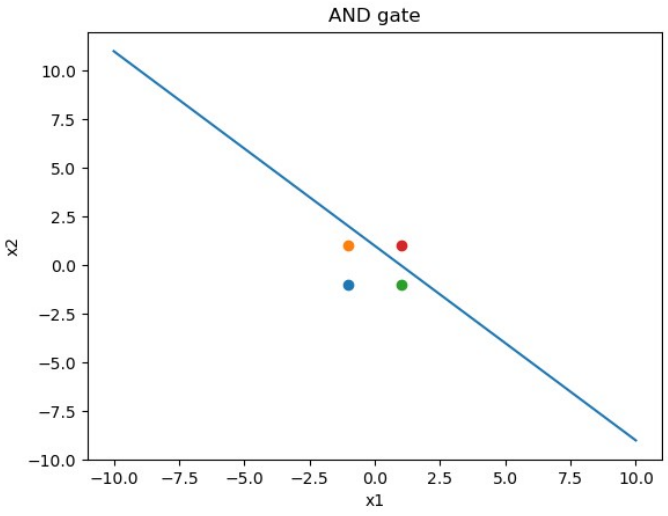
```
hb.plot_decision_boundary()
```

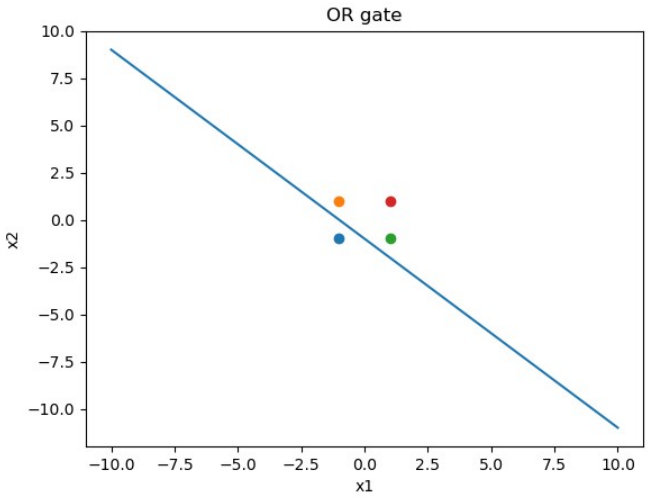
Output:

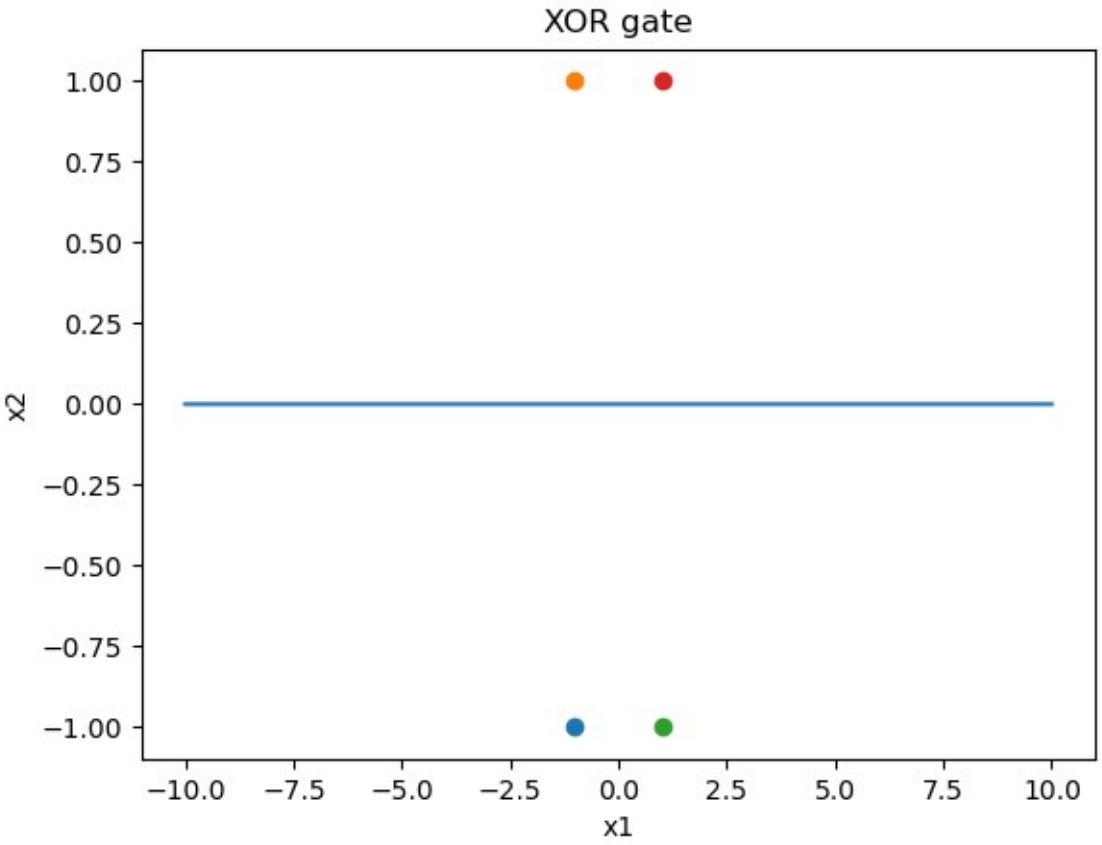
```
~/nitr/soft-computing-lab  master ?1 ❯ python3 hebbian.py
AND gate
*************
Weight updates
[[ 1.  1. -1.]]
[[ 2.  0. -2.]]
[[ 1.  1. -3.]]
[[ 2.  2. -2.]]

OR gate
*************
Weight updates
[[ 1.  1. -1.]]
[[0. 2. 0.]]
[[1. 1. 1.]]
[[2. 2. 2.]]

XOR gate
*************
Weight updates
[[ 1.  1. -1.]]
[[0. 2. 0.]]
[[1. 1. 1.]]
[[0. 0. 0.]]
```






XOR gate
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
x1