ML Experiment 5: Hebbian Learning

```
def hebbian_learning(samples):
     print(f'{"INPUT":^8} {"TARGET":^16}{"WEIGHT CHANGES":^15}{"WEIGHTS":^28}')
     w1, w2, b = 0, 0, 0
     print(' ' * 48, f'({w1:2}, {w2:2}, {b:2})')
     for x1, x2, y in samples:
         w1 = w1 + x1 * y
         w2 = w2 + x2 * y
         b = b + y
         print(f'({x1:2}, {x2:2})\t {y:2}\t ({x1*y:2}, {x2*y:2}, {y:2})\t\t ({w1:2}, {w2:2}, {b:2})')


AND_samples = {
    'binary_input_binary_output': [
        [1, 1, 1],
        [1, 0, 0],
        [0, 1, 0],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, 1],
        [1, 0, -1],
        [0, 1, -1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, 1],
        [ 1, -1, -1],
        [-1, 1, -1],
        [-1, -1, -1]
    ]
}
OR_samples = {
    'binary_input_binary_output': [
        [1, 1, 1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, 1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, 1],
        [ 1, -1, 1],
        [-1, 1, 1],
        [-1, -1, -1]
    ]
}
XOR_samples = {
    'binary_input_binary_output': [
        [1, 1, 0],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, 0]
    ],
    'binary_input_bipolar_output': [
        [1, 1, -1],
        [1, 0, 1],
        [0, 1, 1],
        [0, 0, -1]
    ],
    'bipolar_input_bipolar_output': [
        [ 1, 1, -1],
        [ 1, -1, 1],
        [-1, 1, 1],
        [-1, -1, -1]
    ]
}
```

AND

```
print('-'*20, 'HEBBIAN LEARNING', '-'*20)
print('AND with Binary Input and Binary Output')
hebbian_learning(AND_samples['binary_input_binary_output'])
print('AND with Binary Input and Bipolar Output')
hebbian_learning(AND_samples['binary_input_bipolar_output'])
print('AND with Bipolar Input and Bipolar Output')
hebbian_learning(AND_samples['bipolar_input_bipolar_output'])
```

```
-------------------- HEBBIAN LEARNING --------------------
AND with Binary Input and Binary Output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1,  0)          0      ( 0,  0,  0)           ( 1,  1,  1)
( 0,  1)          0      ( 0,  0,  0)           ( 1,  1,  1)
( 0,  0)          0      ( 0,  0,  0)           ( 1,  1,  1)
AND with Binary Input and Bipolar Output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1,  0)         -1      (-1,  0, -1)           ( 0,  1,  0)
( 0,  1)         -1      ( 0, -1, -1)           ( 0,  0, -1)
( 0,  0)         -1      ( 0,  0, -1)           ( 0,  0, -2)
AND with Bipolar Input and Bipolar Output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1, -1)         -1      (-1,  1, -1)           ( 0,  2,  0)
(-1,  1)         -1      ( 1, -1, -1)           ( 1,  1, -1)
(-1, -1)         -1      ( 1,  1, -1)           ( 2,  2, -2)
```

OR

```
print('-'*20, 'HEBBIAN LEARNING', '-'*20)
print('OR with binary input and binary output')
hebbian_learning(OR_samples['binary_input_binary_output'])
print('OR with binary input and bipolar output')
hebbian_learning(OR_samples['binary_input_bipolar_output'])
print('OR with bipolar input and bipolar output')
hebbian_learning(OR_samples['bipolar_input_bipolar_output'])
```

```
-------------------- HEBBIAN LEARNING --------------------
OR with binary input and binary output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1,  0)          1      ( 1,  0,  1)           ( 2,  1,  2)
( 0,  1)          1      ( 0,  1,  1)           ( 2,  2,  3)
( 0,  0)          0      ( 0,  0,  0)           ( 2,  2,  3)
OR with binary input and bipolar output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1,  0)          1      ( 1,  0,  1)           ( 2,  1,  2)
( 0,  1)          1      ( 0,  1,  1)           ( 2,  2,  3)
( 0,  0)         -1      ( 0,  0, -1)           ( 2,  2,  2)
OR with bipolar input and bipolar output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                ( 0,  0,  0)
( 1,  1)          1      ( 1,  1,  1)           ( 1,  1,  1)
( 1, -1)          1      ( 1, -1,  1)           ( 2,  0,  2)
(-1,  1)          1      (-1,  1,  1)           ( 1,  1,  3)
(-1, -1)         -1      ( 1,  1, -1)           ( 2,  2,  2)
```

XOR

```
print('-'*20, 'HEBBIAN LEARNING', '-'*20)
print('XOR with binary input and binary output')
hebbian_learning(XOR_samples['binary_input_binary_output'])
print('XOR with binary input and bipolar output')
hebbian_learning(XOR_samples['binary_input_bipolar_output'])
```

```
print('XOR with bipolar input and bipolar output')
hebbian_learning(XOR_samples['bipolar_input_bipolar_output'])
```

```
-------------------- HEBBIAN LEARNING --------------------
XOR with binary input and binary output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                 ( 0,  0,  0)
( 1,  1)          0      ( 0,  0,  0)           ( 0,  0,  0)
( 1,  0)          1      ( 1,  0,  1)           ( 1,  0,  1)
( 0,  1)          1      ( 0,  1,  1)           ( 1,  1,  2)
( 0,  0)          0      ( 0,  0,  0)           ( 1,  1,  2)
XOR with binary input and bipolar output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                 ( 0,  0,  0)
( 1,  1)         -1      (-1, -1, -1)           (-1, -1, -1)
( 1,  0)          1      ( 1,  0,  1)           ( 0, -1,  0)
( 0,  1)          1      ( 0,  1,  1)           ( 0,  0,  1)
( 0,  0)         -1      ( 0,  0, -1)           ( 0,  0,  0)
XOR with bipolar input and bipolar output
 INPUT        TARGET     WEIGHT CHANGES           WEIGHTS
                                                 ( 0,  0,  0)
( 1,  1)         -1      (-1, -1, -1)           (-1, -1, -1)
( 1, -1)          1      ( 1, -1,  1)           ( 0, -2,  0)
(-1,  1)          1      (-1,  1,  1)           (-1, -1,  1)
(-1, -1)         -1      ( 1,  1, -1)           ( 0,  0,  0)
```