

single layer perceptron

```
import numpy as np
X = np.array([[0, 0],
              [0, 1],
              [1, 0],
              [1, 1]])

Y = np.array([0, 0, 0, 1])

weights = np.array([0.3, -0.2])
bias = -0.4
learning_rate = 0.2

def activation(net):
    return 1 if net >= 0 else 0

epoch = 0
converged = False

print("Initial weights:", weights)
print("Initial bias:", bias)
print("Learning rate:", learning_rate)
print("\nTraining started...\n")

while not converged:
    epoch += 1
    print(f"Epoch {epoch}")
    converged = True

    for i in range(len(X)):
        x_input = X[i]
        desired = Y[i]

        net = np.dot(weights, x_input) + bias
        output = activation(net)

        error = desired - output

        if error != 0:
            weights = weights + learning_rate * error * x_input
            bias = bias + learning_rate * error
            converged = False
```

```

        print(f"Input: {x_input}, Desired: {desired}, Output:
{output}, Error: {error}")
        print(f"Updated weights: {weights}, Updated bias: {bias}")
    else:
        print(f"Input: {x_input}, Desired: {desired}, Output:
{output}, Error: {error} → No change")

    print("-----")

```

```

print("\nTraining completed!")
print("Final weights:", weights)
print("Final bias:", bias)

```

```

print("\nTesting AND function:")
for i in range(len(X)):
    net = np.dot(weights, X[i]) + bias
    output = activation(net)
    print(f"Input: {X[i]} → Output: {output}")

```

```

Initial weights: [ 0.3 -0.2]
Initial bias: -0.4
Learning rate: 0.2

```

Training started...

```

Epoch 1
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 1], Desired: 1, Output: 0, Error: 1
Updated weights: [0.5 0. ], Updated bias: -0.2
-----

```

```

Epoch 2
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 0], Desired: 0, Output: 1, Error: -1
Updated weights: [0.3 0. ], Updated bias: -0.4
Input: [1 1], Desired: 1, Output: 0, Error: 1
Updated weights: [0.5 0.2], Updated bias: -0.2
-----

```

```

Epoch 3
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 1, Error: -1
Updated weights: [0.5 0. ], Updated bias: -0.4
Input: [1 0], Desired: 0, Output: 1, Error: -1
Updated weights: [0.3 0. ], Updated bias: -0.6000000000000001
Input: [1 1], Desired: 1, Output: 0, Error: 1
Updated weights: [0.5 0.2], Updated bias: -0.4000000000000001
-----

```

Epoch 4

```
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 0], Desired: 0, Output: 1, Error: -1
Updated weights: [0.3 0.2], Updated bias: -0.6000000000000001
Input: [1 1], Desired: 1, Output: 0, Error: 1
Updated weights: [0.5 0.4], Updated bias: -0.4000000000000001
-----
```

Epoch 5

```
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 0], Desired: 0, Output: 1, Error: -1
Updated weights: [0.3 0.4], Updated bias: -0.6000000000000001
Input: [1 1], Desired: 1, Output: 1, Error: 0 → No change
-----
```

Epoch 6

```
Input: [0 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [0 1], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 0], Desired: 0, Output: 0, Error: 0 → No change
Input: [1 1], Desired: 1, Output: 1, Error: 0 → No change
-----
```

Training completed!

Final weights: [0.3 0.4]

Final bias: -0.6000000000000001

Testing AND function:

```
Input: [0 0] → Output: 0
Input: [0 1] → Output: 0
Input: [1 0] → Output: 0
Input: [1 1] → Output: 1
```