```
import numpy as np
```

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

```
w = np.array([0.3, -0.2])
theta = 0.4
bias = -theta
lr = 0.2
```

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

```
epochs = 10
for epoch in range(epochs):
    print(f"Epoch {epoch+1}")
    error_count = 0
    for i in range(len(X)):
        x_in = np.dot(X[i], w) + bias
        y_pred = activation(x_in)
        error = Y[i] - y_pred
        if error != 0:
            w = w + lr * error * X[i]
            bias = bias + lr * error
            print(f" Updated weights: {w}, bias: {bias}")
        else:
            print(" No change")
        error_count += abs(error)
    if error_count == 0:
        break
```

```
Epoch 1
 No change
 No change
 No change
 Updated weights: [0.5 0. ], bias: -0.2
Epoch 2
 No change
 No change
 Updated weights: [0.3 0. ], bias: -0.4
 Updated weights: [0.5 0.2], bias: -0.2
Epoch 3
 No change
 Updated weights: [0.5 0. ], bias: -0.4
 Updated weights: [0.3 0. ], bias: -0.6000000000000001
 Updated weights: [0.5 0.2], bias: -0.4000000000000001
Epoch 4
 No change
 No change
 Updated weights: [0.3 0.2], bias: -0.6000000000000001
 Updated weights: [0.5 0.4], bias: -0.4000000000000001
Epoch 5
 No change
 No change
 Updated weights: [0.3 0.4], bias: -0.6000000000000001
 No change
Epoch 6
 No change
 No change
 No change
 No change
```

```
print("\nFinal Weights:", w)
print("Final Bias:", bias)
print("Testing AND Function:")
for i in range(len(X)):
    x_in = np.dot(X[i], w) + bias
    print(X[i], "=>", activation(x_in))
```

```
Final Weights: [0.3 0.4]
Final Bias: -0.6000000000000001
Testing AND Function:
[0 0] => 0
[0 1] => 0
[1 0] => 0
[1 1] => 1
```