[1]:
```
#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])
```

[2]:
```
w = np.array([0.3, -0.2])
bias = -0.4
```

[3]:
```
eta=0.2
```

[4]:
```
def activation(net):
    return 1 if net >= 0 else 0
```

[5]:
```
error_count = 0
```

[6]:
```
for i in range(len(X)):
    x = X[i]
    y = Y[i]
    net = np.dot(x, w) + bias
    y_pred = activation(net)
    error = y - y_pred
```

[7]:
```
if error != 0:
    w = w + eta * error * x
    bias = bias + eta * error
    error_count += 1
    print(f"Update: Input={x}, Target={y}, Pred={y_pred}, Error={error}, New weight={w}, New Bias={bias}")
else:
    print(f"No change: Input={x}, Target={y}, Pred={y_pred}")
```

```
Update: Input=[1 1], Target=1, Pred=0, Error=1, New weight=[0.5 0. ], New Bias=-0.2
```

[8]:
```
if error_count == 0:
    print("\nTraining converged.")
print("\nFinal Results:")
for i in range(len(X)):
    net = np.dot(X[i], w) + bias
    y_pred = activation(net)
    print(f"Input: {X[i]} → Output: {y_pred}")
```

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