MACHINE LEARNING – 2CS501

PRACTICAL 8

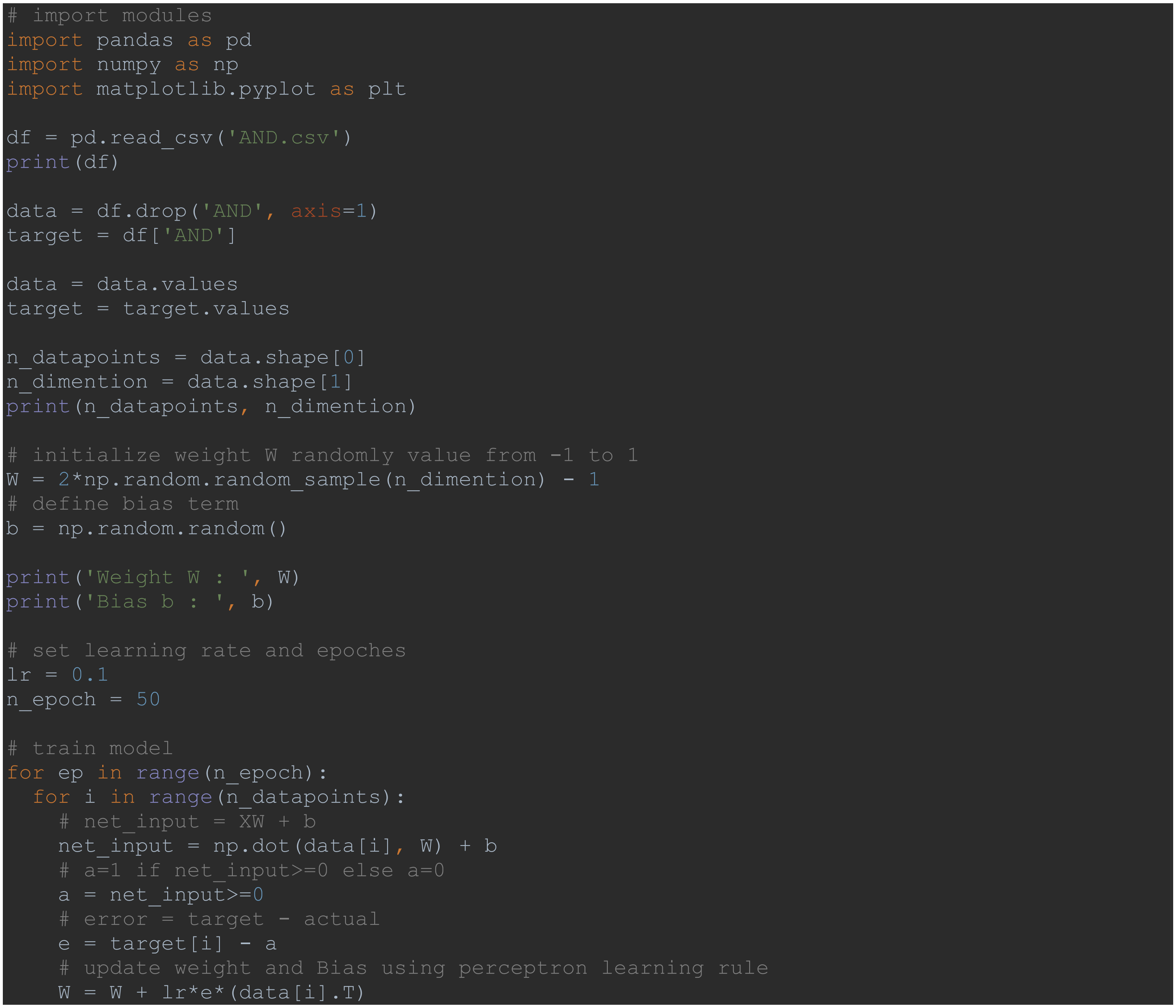
Name: Bhanderi Mihir

Roll No.: 19BCE023

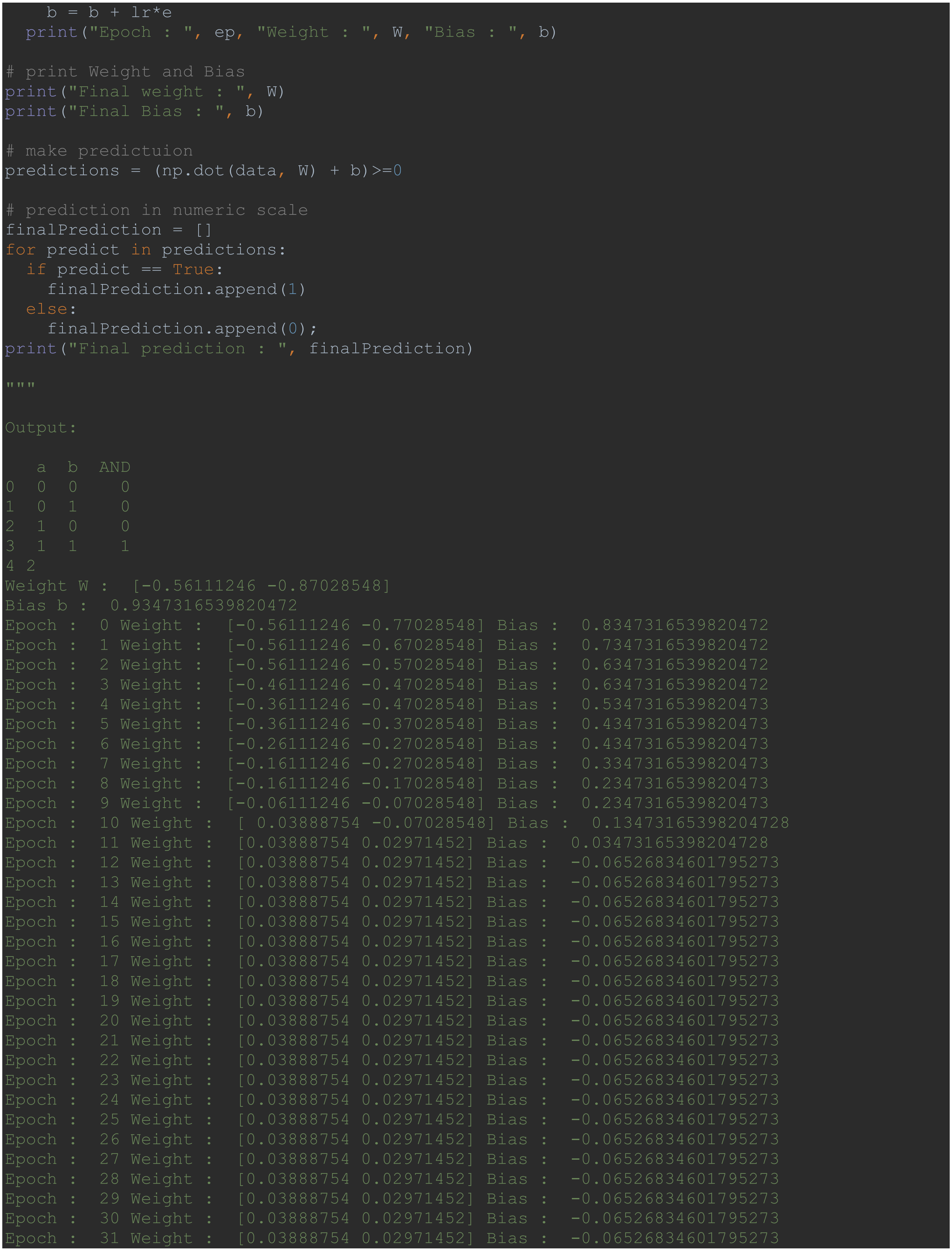
Batch No.: A-1

1) AND

Code:



```
# import modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv('AND.csv')
print(df)

data = df.drop('AND', axis=1)
target = df['AND']

data = data.values
target = target.values

n_datapoints = data.shape[0]
n_dimention = data.shape[1]
print(n_datapoints, n_dimention)

# initialize weight W randomly value from -1 to 1
W = 2*np.random.random_sample(n_dimention) - 1
# define bias term
b = np.random.random()

print('Weight W : ', W)
print('Bias b : ', b)

# set learning rate and epoches
lr = 0.1
n_epoch = 50

# train model
for ep in range(n_epoch):
  for i in range(n_datapoints):
    # net_input = XW + b
    net_input = np.dot(data[i], W) + b
    # a=1 if net_input>=0 else a=0
    a = net_input>=0
    # error = target - actual
    e = target[i] - a
    # update weight and Bias using perceptron learning rule
    W = W + lr*e*(data[i].T)
```



```
    b = b + lr*e
  print("Epoch : ", ep, "Weight : ", W, "Bias : ", b)

# print Weight and Bias
print("Final weight : ", W)
print("Final Bias : ", b)

# make predictuion
predictions = (np.dot(data, W) + b)>=0

# prediction in numeric scale
finalPrediction = []
for predict in predictions:
  if predict == True:
    finalPrediction.append(1)
  else:
    finalPrediction.append(0);
print("Final prediction : ", finalPrediction)

"""

Output:

   a  b  AND
0  0  0    0
1  0  1    0
2  1  0    0
3  1  1    1
4 2
Weight W :  [-0.56111246 -0.87028548]
Bias b :  0.9347316539820472
Epoch :  0 Weight :  [-0.56111246 -0.77028548] Bias :  0.8347316539820472
Epoch :  1 Weight :  [-0.56111246 -0.67028548] Bias :  0.7347316539820472
Epoch :  2 Weight :  [-0.56111246 -0.57028548] Bias :  0.6347316539820472
Epoch :  3 Weight :  [-0.46111246 -0.47028548] Bias :  0.6347316539820472
Epoch :  4 Weight :  [-0.36111246 -0.47028548] Bias :  0.5347316539820473
Epoch :  5 Weight :  [-0.36111246 -0.37028548] Bias :  0.4347316539820473
Epoch :  6 Weight :  [-0.26111246 -0.27028548] Bias :  0.4347316539820473
Epoch :  7 Weight :  [-0.16111246 -0.27028548] Bias :  0.3347316539820473
Epoch :  8 Weight :  [-0.16111246 -0.17028548] Bias :  0.2347316539820473
Epoch :  9 Weight :  [-0.06111246 -0.07028548] Bias :  0.2347316539820473
Epoch :  10 Weight :  [ 0.03888754 -0.07028548] Bias :  0.13473165398204728
Epoch :  11 Weight :  [0.03888754 0.02971452] Bias :  0.03473165398204728
Epoch :  12 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  13 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  14 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  15 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  16 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  17 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  18 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  19 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  20 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  21 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  22 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  23 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  24 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  25 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  26 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  27 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  28 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  29 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  30 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  31 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  32 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  33 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  34 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  35 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  36 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  37 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  38 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  39 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  40 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  41 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  42 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  43 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  44 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  45 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  46 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  47 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  48 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Epoch :  49 Weight :  [0.03888754 0.02971452] Bias :  -0.06526834601795273
Final weight :  [0.03888754 0.02971452]
Final Bias :  -0.06526834601795273
Final prediction :  [0, 0, 0, 1]

"""
```

2) OR

Code:

```python
# import modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv('Or.csv')
print(df)

data = df.drop('OR', axis=1)
target = df['OR']

data = data.values
target = target.values

n_datapoints = data.shape[0]
n_dimention = data.shape[1]
print(n_datapoints, n_dimention)

# initialize weight W randomly value from -1 to 1
W = 2 * np.random.random_sample(n_dimention) - 1
# define bias term
b = np.random.random()

print('Weight W : ', W)
print('Bias b : ', b)

# set learning rate and epoches
lr = 0.1
```

```
n_epoch = 50

# train model
for ep in range(n_epoch):
    for i in range(n_datapoints):
        # net_input = XW + b
        net_input = np.dot(data[i], W) + b
        # a=1 if net_input>=0 else a=0
        a = net_input >= 0
        # error = target - actual
        e = target[i] - a
        # update weight and Bias using perceptron learning rule
        W = W + lr * e * (data[i].T)
        b = b + lr * e
    print("Epoch : ", ep, "Weight : ", W, "Bias : ", b)

# print Weight and Bias
print("Final weight : ", W)
print("Final Bias : ", b)

# make predictuion
predictions = (np.dot(data, W) + b) >= 0

# prediction in numeric scale
finalPrediction = []
for predict in predictions:
    if predict == True:
        finalPrediction.append(1)
    else:
        finalPrediction.append(0);
print("Final prediction : ", finalPrediction)

"""

Output:

   a  b  OR
0  0  0   0
1  0  1   1
2  1  0   1
3  1  1   1
4 2
Weight W :  [ 0.29377335 -0.97520962]
Bias b :  0.3490816949051555
Epoch :  0 Weight :  [ 0.39377335 -0.77520962] Bias :  0.4490816949051555
Epoch :  1 Weight :  [ 0.39377335 -0.67520962] Bias :  0.4490816949051555
Epoch :  2 Weight :  [ 0.39377335 -0.57520962] Bias :  0.4490816949051555
Epoch :  3 Weight :  [ 0.39377335 -0.47520962] Bias :  0.4490816949051555
Epoch :  4 Weight :  [ 0.39377335 -0.37520962] Bias :  0.4490816949051555
Epoch :  5 Weight :  [ 0.39377335 -0.27520962] Bias :  0.4490816949051555
Epoch :  6 Weight :  [ 0.39377335 -0.27520962] Bias :  0.3490816949051555
Epoch :  7 Weight :  [ 0.39377335 -0.17520962] Bias :  0.3490816949051555
Epoch :  8 Weight :  [ 0.39377335 -0.17520962] Bias :  0.2490816949051555
Epoch :  9 Weight :  [ 0.39377335 -0.07520962] Bias :  0.2490816949051555
Epoch :  10 Weight :  [ 0.39377335 -0.07520962] Bias :  0.1490816949051555
Epoch :  11 Weight :  [0.39377335 0.02479038] Bias :  0.1490816949051555
Epoch :  12 Weight :  [0.39377335 0.02479038] Bias :  0.04908169490515549
Epoch :  13 Weight :  [0.39377335 0.12479038] Bias :  0.04908169490515549
Epoch :  14 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  15 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  16 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  17 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  18 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  19 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  20 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  21 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  22 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  23 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  24 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  25 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  26 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  27 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  28 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  29 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  30 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  31 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  32 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  33 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  34 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  35 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  36 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  37 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  38 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  39 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  40 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  41 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  42 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  43 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  44 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  45 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  46 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  47 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  48 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Epoch :  49 Weight :  [0.39377335 0.12479038] Bias :  -0.05091830509484452
Final weight :  [0.39377335 0.12479038]
Final Bias :  -0.05091830509484452
Final prediction :  [0, 1, 1, 1]


"""
```