Experiment 2

```
import tensorflow as tf
import numpy as np
X = np.array([[0,0], [0,1], [1,0], [1,1]], dtype=np.float32)
y = np.array([[0], [1], [1], [0]], dtype=np.float32)
model = tf.keras.Sequential([
    tf.keras.layers.Dense(2, input_dim=2, activation='sigmoid'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model.fit(X, y, epochs=25, verbose=2)
print("Output:", model.predict(X).round())
```

Output

```
Epoch 1/25
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an
`Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
1/1 - 1s - 1s/step - accuracy: 0.5000 - loss: 0.7936
Epoch 2/25
1/1 - 0s - 56ms/step - accuracy: 0.5000 - loss: 0.7930
Epoch 3/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7924
Epoch 4/25
1/1 - 0s - 47ms/step - accuracy: 0.5000 - loss: 0.7918
Epoch 5/25
1/1 - 0s - 61ms/step - accuracy: 0.5000 - loss: 0.7912
Epoch 6/25
1/1 - 0s - 59ms/step - accuracy: 0.5000 - loss: 0.7906
Epoch 7/25
1/1 - 0s - 57ms/step - accuracy: 0.5000 - loss: 0.7900
Epoch 8/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7894
Epoch 9/25
1/1 - 0s - 45ms/step - accuracy: 0.5000 - loss: 0.7889
Epoch 10/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7883
Epoch 11/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7877
Epoch 12/25
1/1 - 0s - 49ms/step - accuracy: 0.5000 - loss: 0.7871
Epoch 13/25
1/1 - 0s - 45ms/step - accuracy: 0.5000 - loss: 0.7866
Epoch 14/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7860
Epoch 15/25
1/1 - 0s - 44ms/step - accuracy: 0.5000 - loss: 0.7854
Epoch 16/25
1/1 - 0s - 44ms/step - accuracy: 0.5000 - loss: 0.7848
Epoch 17/25
1/1 - 0s - 59ms/step - accuracy: 0.5000 - loss: 0.7843
Epoch 18/25
1/1 - 0s - 45ms/step - accuracy: 0.5000 - loss: 0.7837
Epoch 19/25
1/1 - 0s - 44ms/step - accuracy: 0.5000 - loss: 0.7832
Epoch 20/25
1/1 - 0s - 58ms/step - accuracy: 0.5000 - loss: 0.7826
Epoch 21/25
1/1 - 0s - 46ms/step - accuracy: 0.5000 - loss: 0.7820
Epoch 22/25
1/1 - 0s - 46ms/step - accuracy: 0.5000 - loss: 0.7815
Epoch 23/25
1/1 - 0s - 46ms/step - accuracy: 0.5000 - loss: 0.7809
Epoch 24/25
1/1 - 0s - 61ms/step - accuracy: 0.5000 - loss: 0.7804
Epoch 25/25
1/1 - 0s - 55ms/step - accuracy: 0.5000 - loss: 0.7798
1/1 ---------- 0s 67ms/step
Output: [[0.]
 [0.]
 [0.]
 [0.]]
```

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# XOR Input and Output
X = np.array([[0, 0],
              [0, 1],
              [1, 0],
              [1, 1]])
y = np.array([[0], [1], [1], [0]])

# Build the model
model = Sequential()
model.add(Dense(2, input_dim=2, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train the model
model.fit(X, y, epochs=100, verbose=0)

# Predict
output = model.predict(X)
print("Output:\n", np.round(output))
```

Output

```
1/1 ---------- 0s 72ms/step
 Output:
[[1.]
[1.]
[1.]
[1.]]
```

```
import torch
import torch.nn as nn
import torch.optim as optim

# XOR Input and Output
X = torch.tensor([[0,0],[0,1],[1,0],[1,1]], dtype=torch.float32)
y = torch.tensor([[0],[1],[1],[0]], dtype=torch.float32)

# Model definition
model = nn.Sequential(
    nn.Linear(2, 2),
    nn.Sigmoid(),
    nn.Linear(2, 1),
    nn.Sigmoid()
)

# Loss and optimizer
criterion = nn.BCELoss()
optimizer = optim.Adam(model.parameters(), lr=0.1)

# Training loop
for epoch in range(5000):
    optimizer.zero_grad()
    output = model(X)
    loss = criterion(output, y)
    loss.backward()
    optimizer.step()

# Final Output
print("Output:", torch.round(model(X)))
```

OUTPUT

```
Output:
tensor([[0.],
[1.],
 [1.],
[0.]],
grad_fn=<RoundBackward0>)
```