**Practical No : 10**

**Problem Statement:**

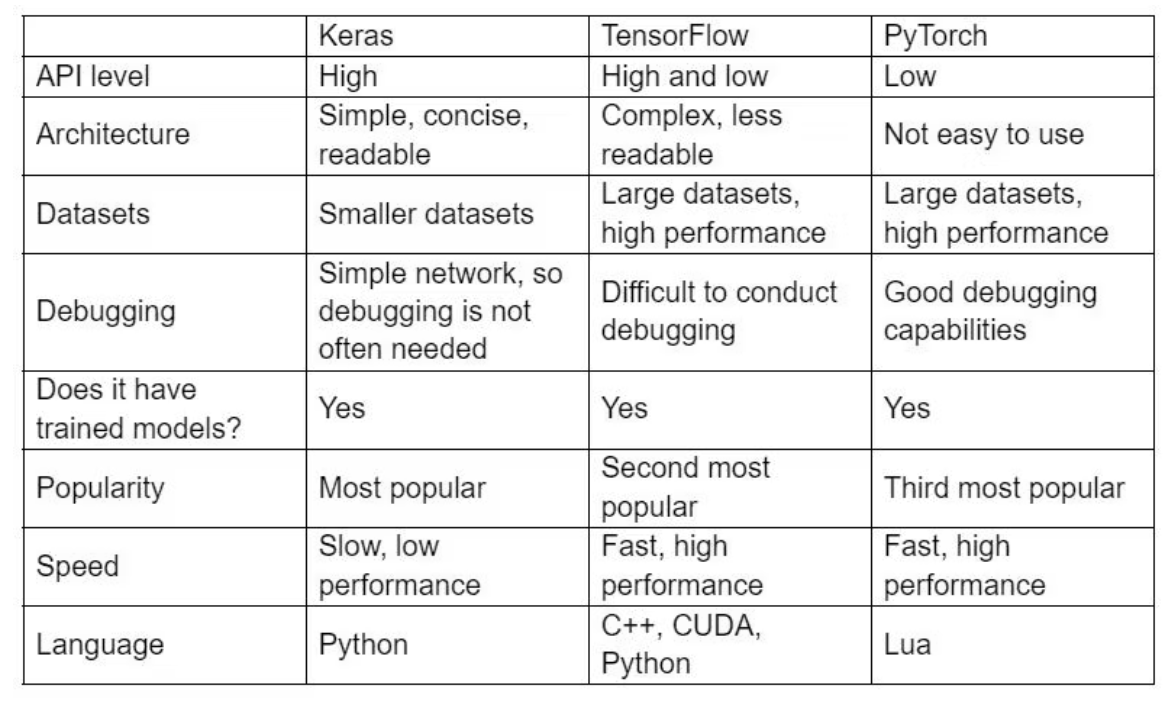
1. Understand the difference between Keras and Tensorflow for ANNs.

2. Explore https://playground.tensorflow.org/ - You may experiment with various 1. no. of layers, 2. activation functions, 3. learning rate 4. Problem type. Submit screenshot for assignment.

3. Create a simple artificial neural network in Python for AND gate and XOR Gate.

(You may refer to the following page:https://broutonlab.com/blog/tutorial-create-simple-neural-network)

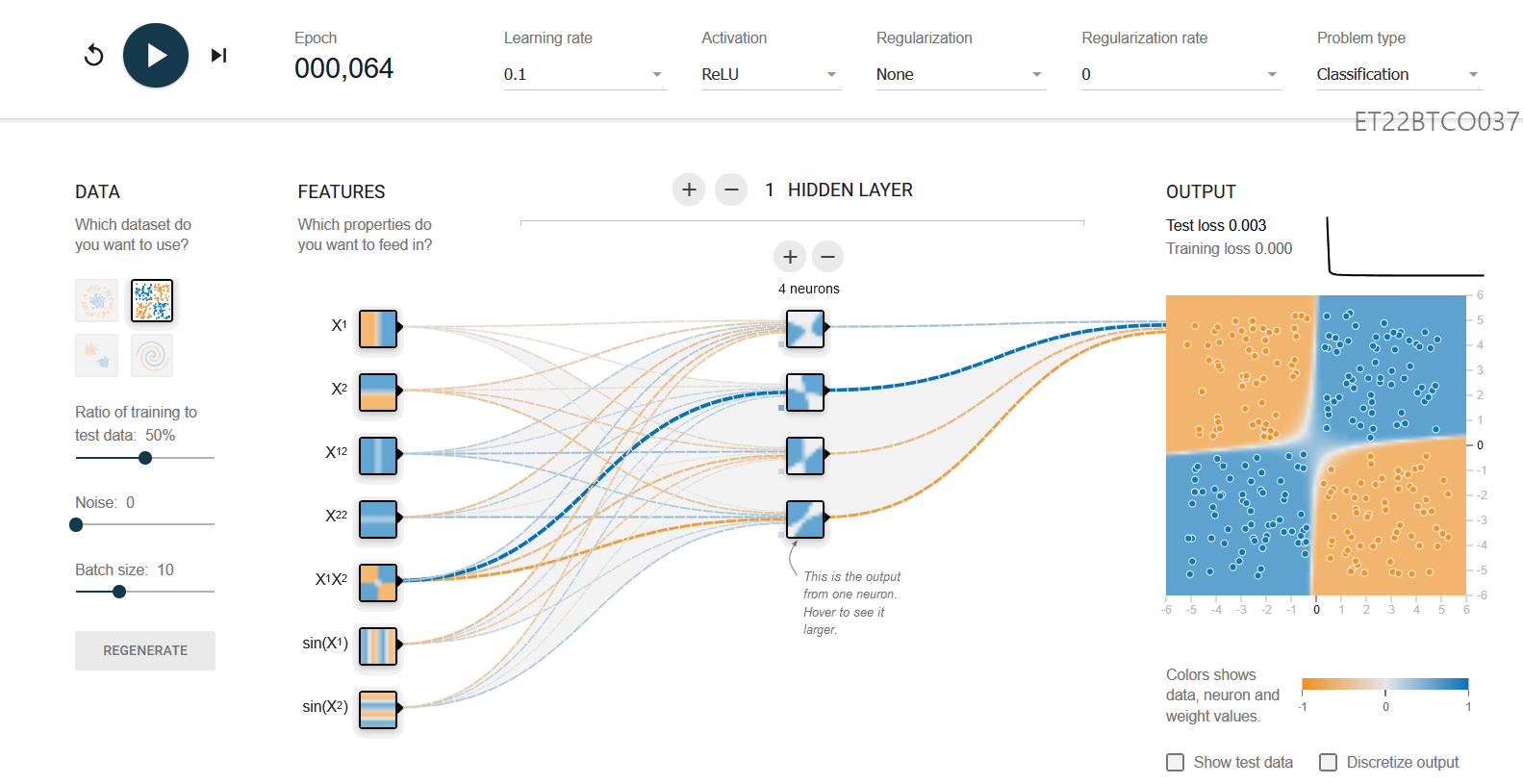
**1. Understand the difference between Keras and Tensorflow for ANNs.**

****

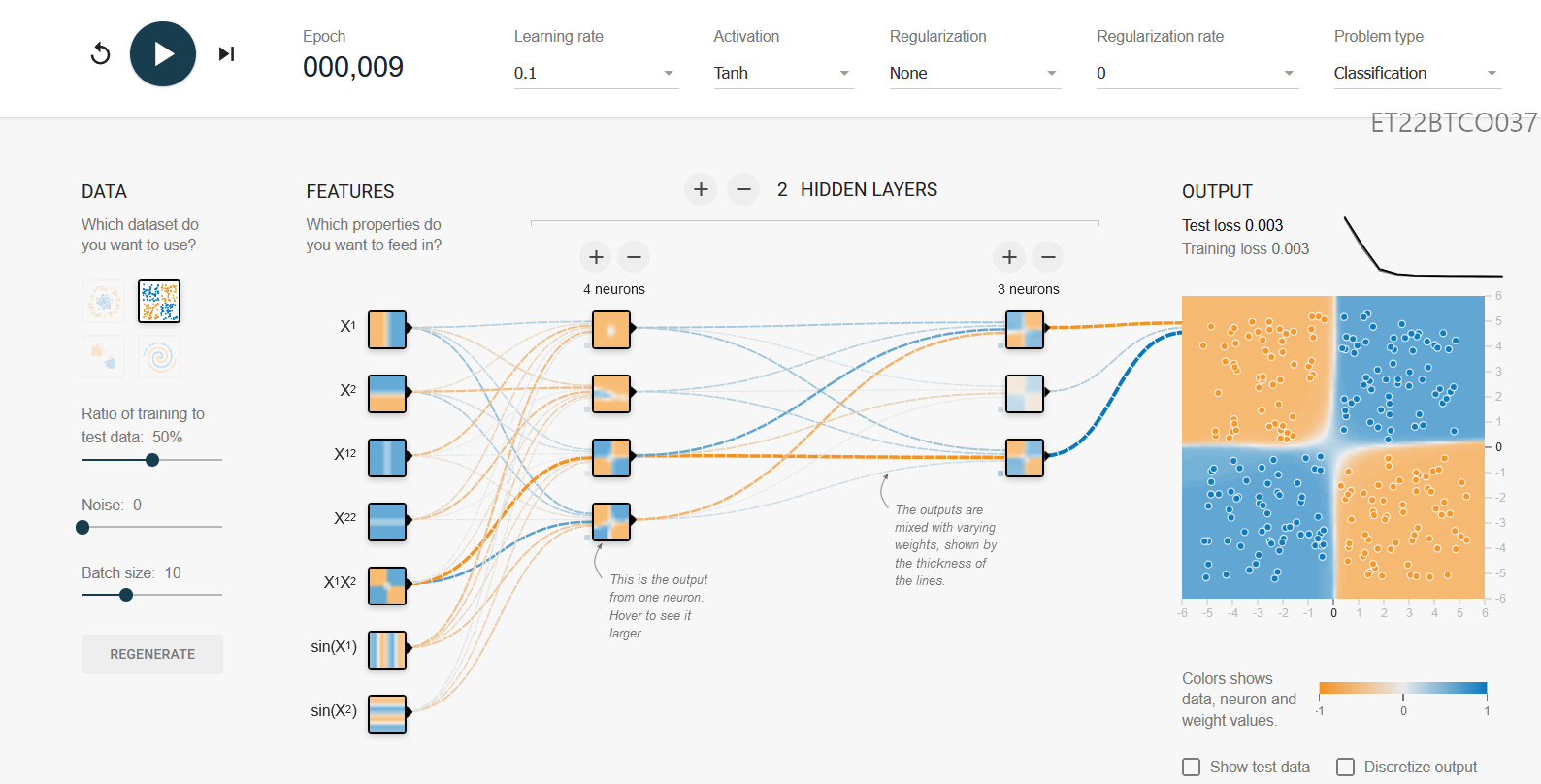
**2. Explore https://playground.tensorflow.org/ - You may experiment with various 1. no. of layers, 2. activation functions, 3. learning rate 4. Problem type. Submit screenshot for assignment.**

**Output:**

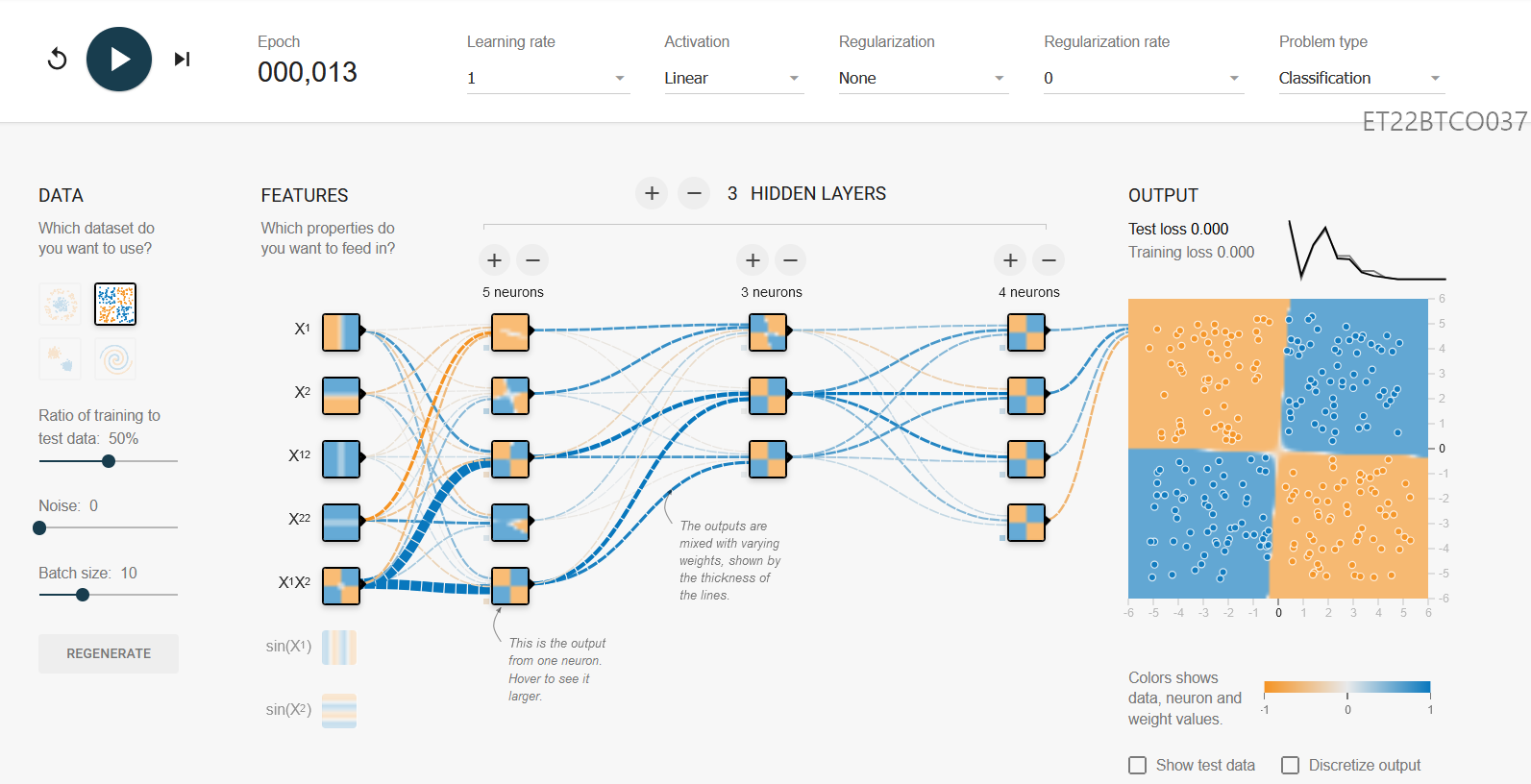
1. **Classification**
2. ReLU

****

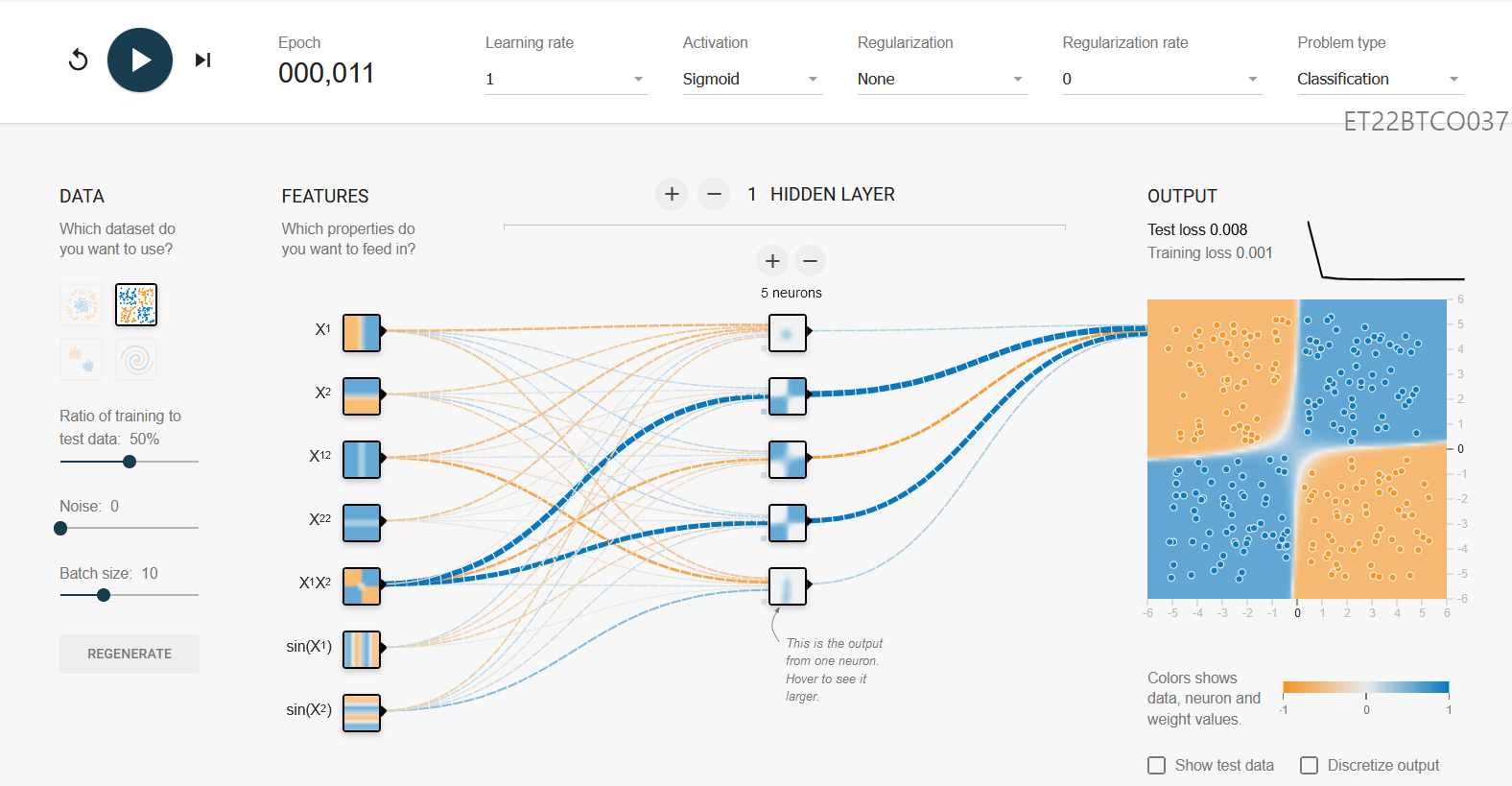
1. Tanh

****

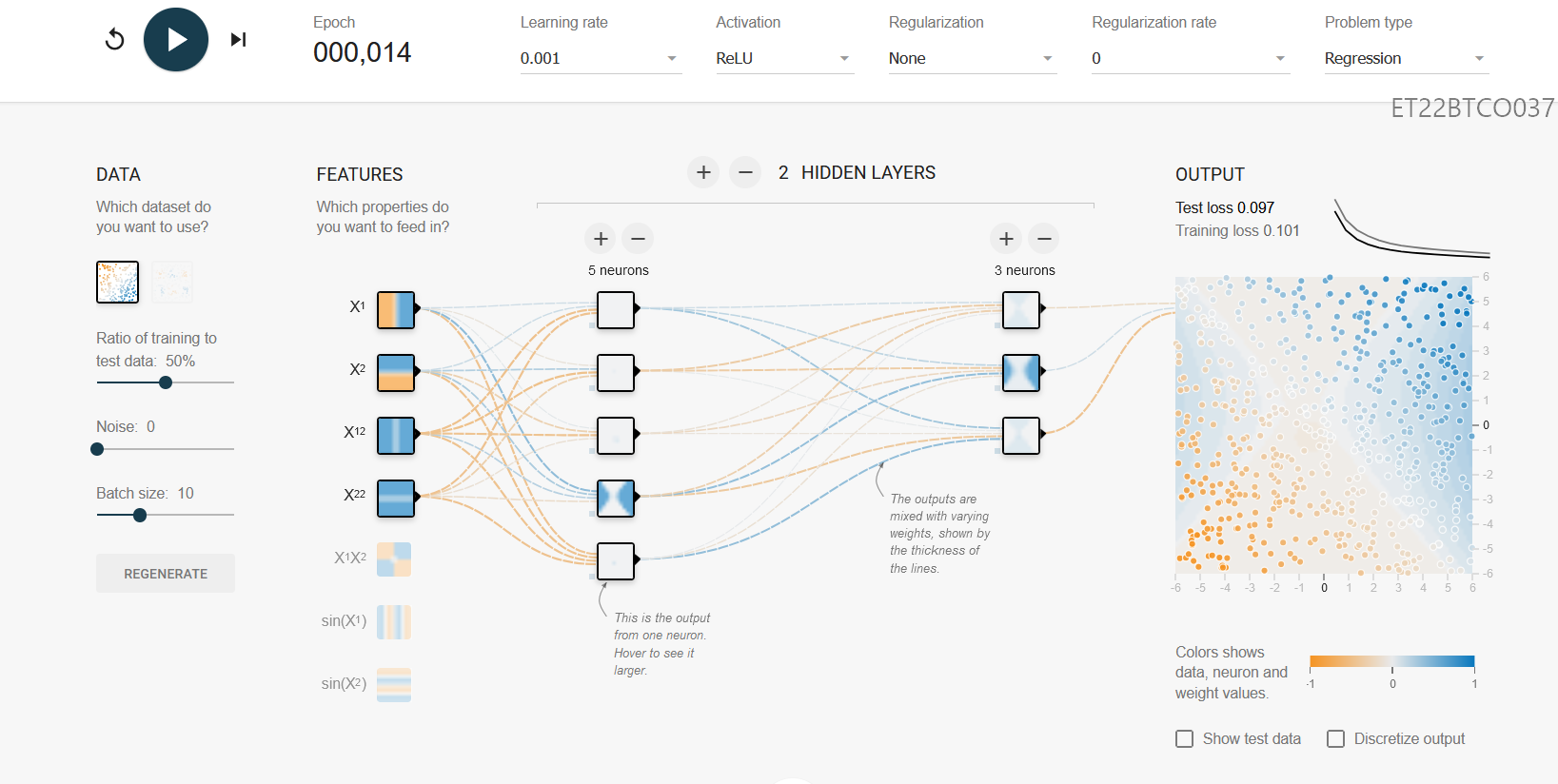
1. Linear

****

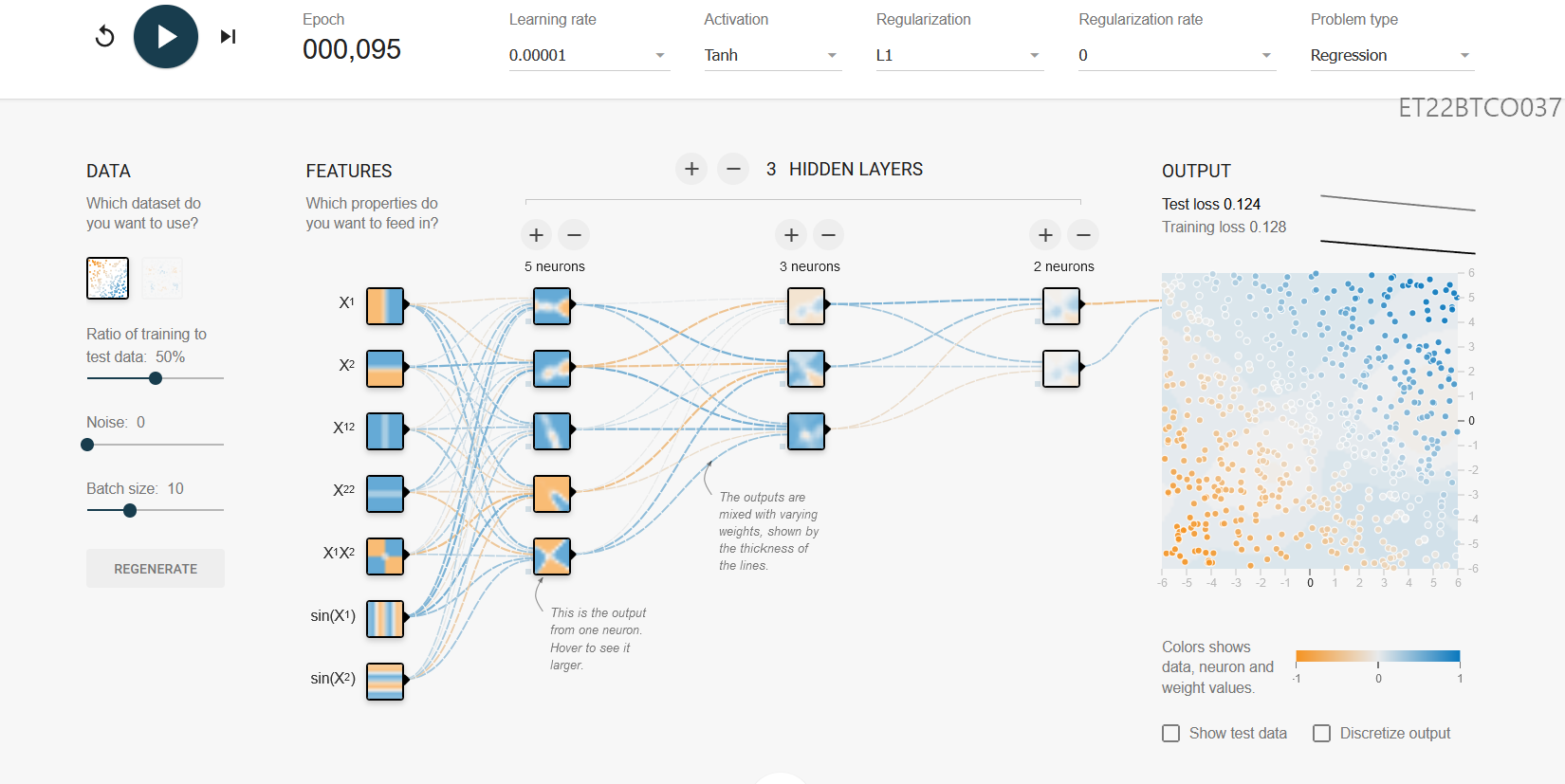
1. Sigmoid

****

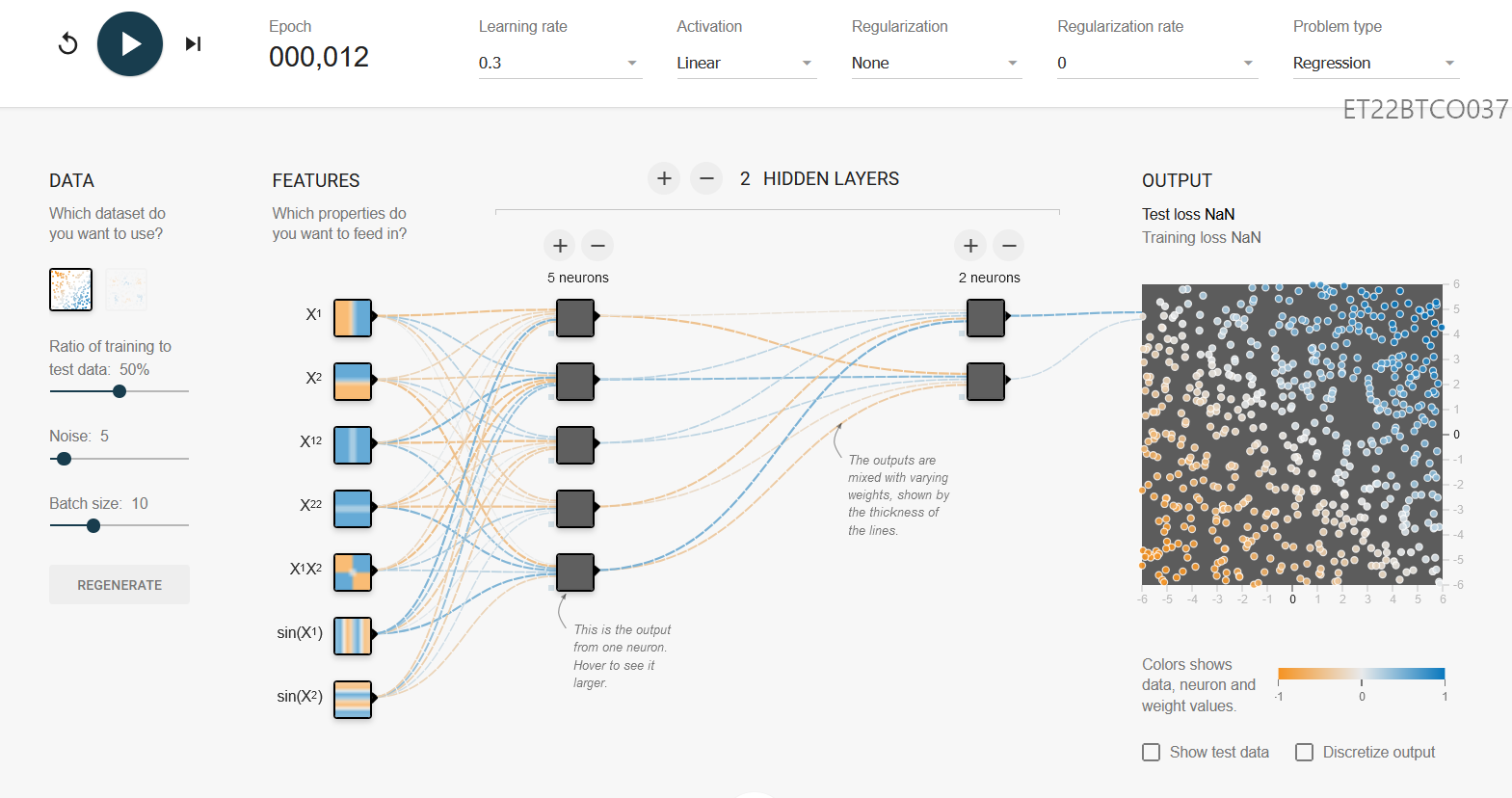
1. **Regression**
2. ReLU

****

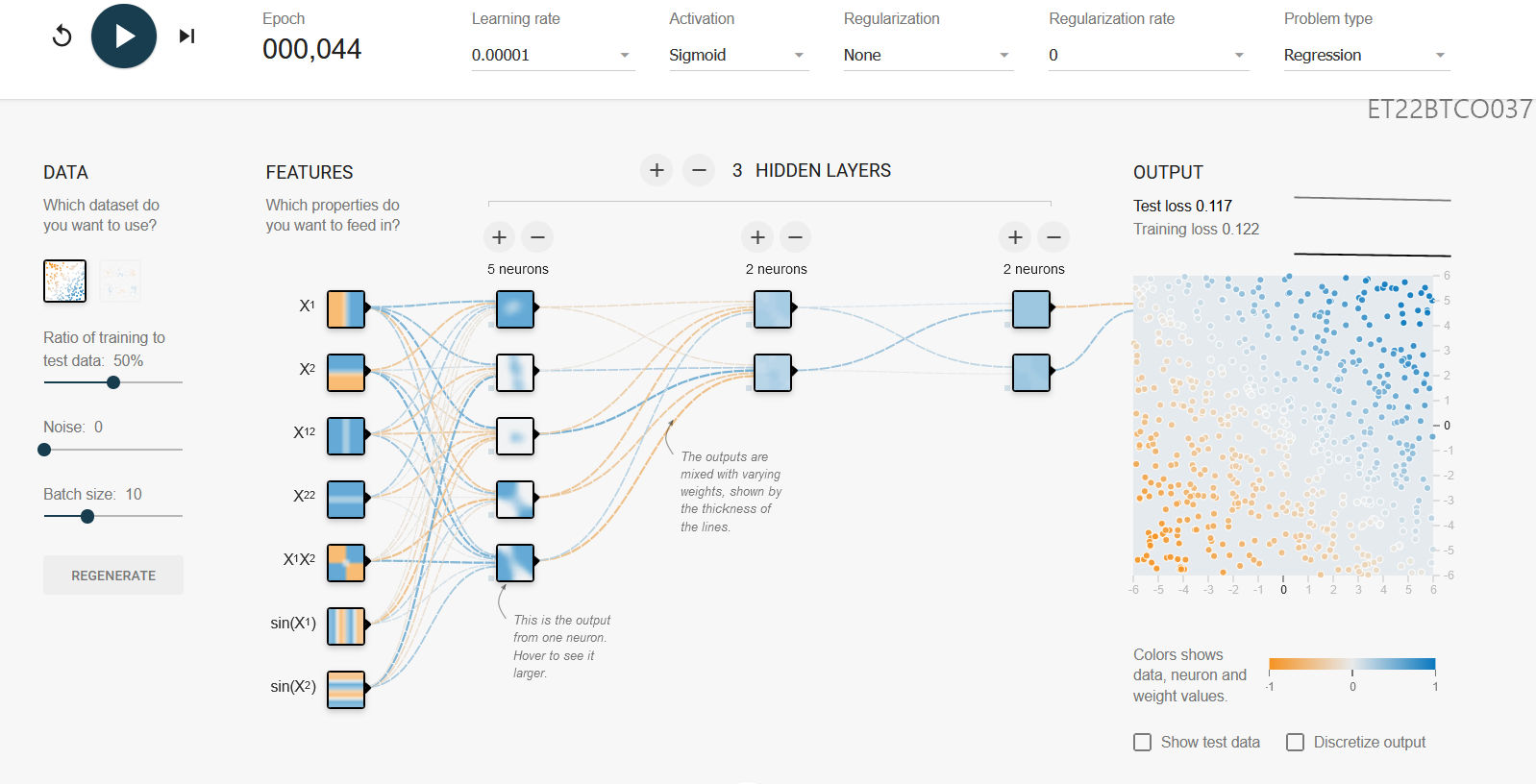
1. Tanh

****

1. Linear

****

1. Sigmoid

****

**3. Create a simple artificial neural network in Python for AND gate and XOR Gate.**

**a. AND gate**

**Code:**

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

training\_data = np.array([[0,0],[0,1],[1,0],[1,1]], "float32")

target\_data = np.array([[0],[0],[0],[1]], "float32")

model = Sequential()

model.add(Dense(16, input\_dim=2, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(loss='mean\_squared\_error', optimizer='adam', metrics=['binary\_accuracy'])

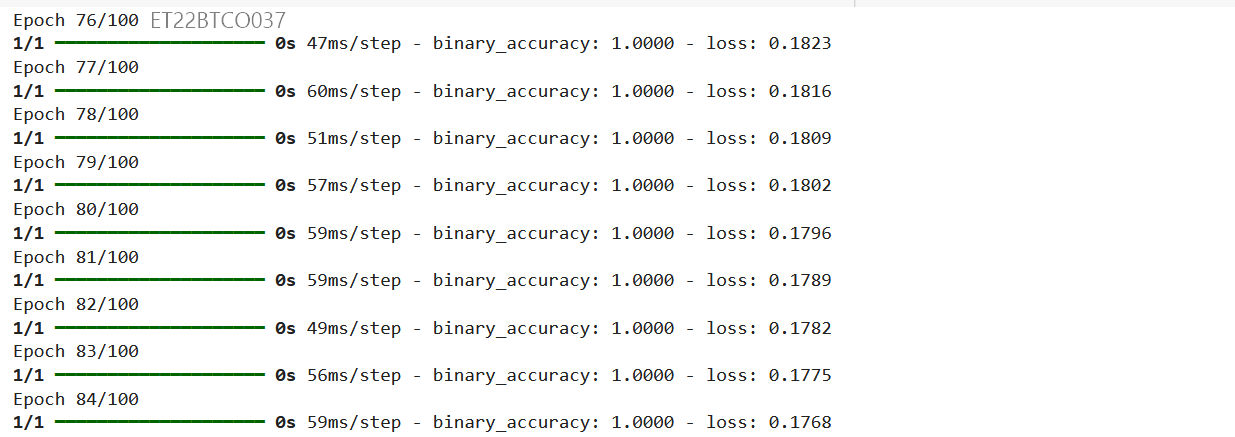
model.fit(training\_data, target\_data, epochs=100)

scores = model.evaluate(training\_data, target\_data)

print(f"\n{model.metrics\_names[1]}: {scores[1]\*100}")

print(model.predict(training\_data).round())

**Output:**

****

**b. XOR gate**

**Code:**

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

training\_data = np.array([[0,0],[0,1],[1,0],[1,1]], "float32")

target\_data = np.array([[0],[1],[1],[0]], "float32")

model = Sequential()

model.add(Dense(16, input\_dim=2, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(loss='mean\_squared\_error', optimizer='adam', metrics=['binary\_accuracy'])

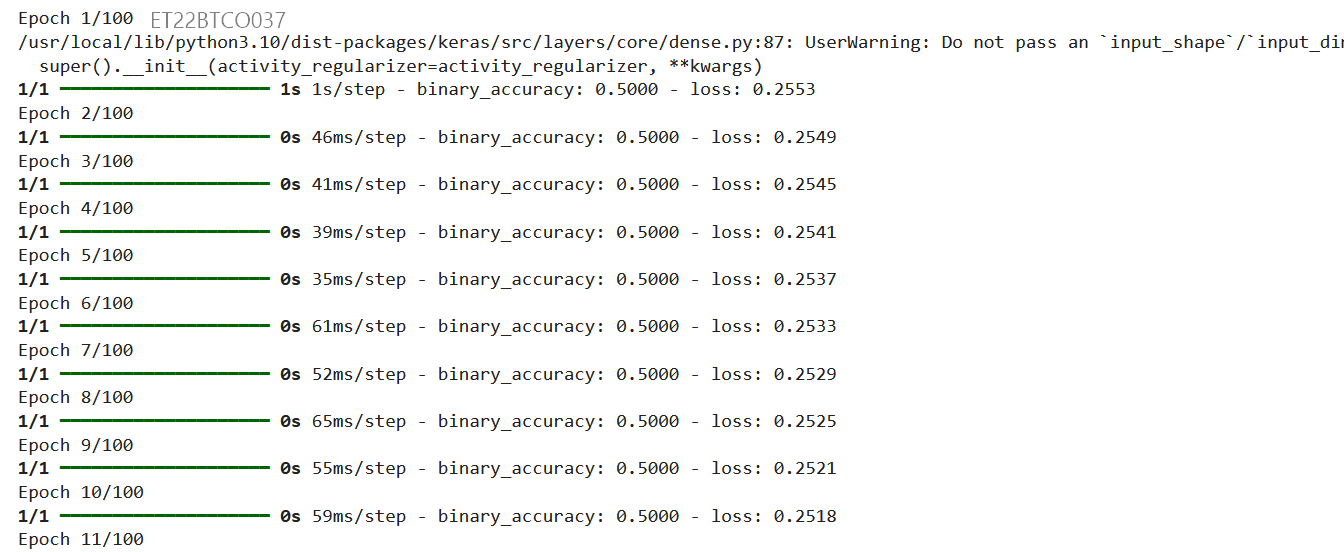
model.fit(training\_data, target\_data, epochs=100)

scores = model.evaluate(training\_data, target\_data)

print(f"\n{model.metrics\_names[1]}: {scores[1]\*100}")

print(model.predict(training\_data).round())

**Output:**

****

****