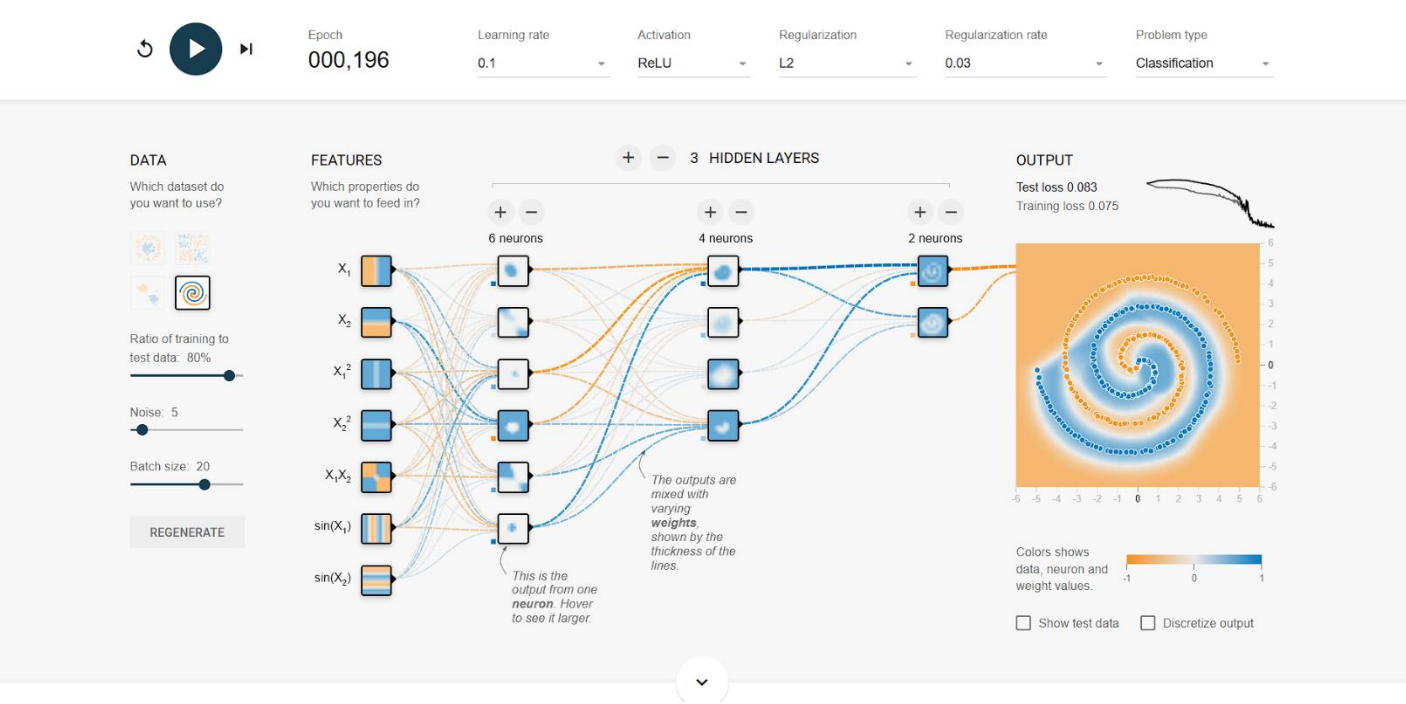
MULTIPLE LAYERED NEURAL NETWORK USING TENSORFLOW PLAYGROUND

Introduction:

Multiple layered neural networks, also known as multi-layer perceptrons (MLPs), are a type of artificial neural network that consists of multiple layers of neurons, each layer fully connected to the next one. These networks can learn complex patterns and relationships in data due to their depth and structure. Here's an overview of how they work:

Multiple Layered Neural Network using Tensorflow Playground:



Description:

The image presents a user interface for configuring a multi-layered neural network model. Here we have used one input layer, 3 hidden layers and one output layer. In first hidden layer 6 neurons are used, in second hidden layer 4 neurons are used and at the third hidden layer 2 neurons are used. At the top, the epoch count is displayed as 000,196, indicating that the model has completed 196 passes through the training dataset. The learning rate is set to 0.1, which is relatively high, allowing for significant weight updates during training but potentially risking overshooting the optimal solution. The activation function is configured to ReLU (Rectified Linear Unit), a popular choice that introduces non-linearity by allowing only positive values to pass through while outputting zero for negative inputs. The model employs L2 regularization to help prevent overfitting by adding a penalty to the loss function based on the square of the magnitude of the weights. The regularization rate is set to 0.03, indicating a moderate penalty. The problem type is classified as a classification task, meaning the model is designed to predict discrete categories based on the input features. Additionally, users can select datasets and adjust parameters such as the ratio of training to test data, the amount of noise in the data, and the batch size used during training, allowing for a customized approach to training the neural network.

Formulas Used:

- **Summation Formula (Z):**

$$Z^{(l)} = W^{(l)} A^{(l-1)} + b^{(l)}$$

Where:

- $Z^{(l)}$ is the weighted sum for layer l.
- $W^{(l)}$ is the weight matrix for layer l.
- $A^{(l-1)}$ is the activation from the previous layer.
- $b^{(l)}$ is the bias vector for layer l.

- **Activation Function (A)**:

$$A^{(l)} = f(Z^{(l)})$$

Where f can be any activation function, such as:

- ReLU: $f(Z) = \max(0, Z)$
- Sigmoid: $f(Z) = \frac{1}{1+e^{-Z}}$
- Tanh: $f(Z) = \frac{e^{Z}-e^{-Z}}{e^{Z}+e^{-Z}}$

- **Mean Squared Error (MSE)** for regression tasks:

$$\text{Loss} = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2$$

Where:

- $y^{(i)}$ is the true label.
- $\hat{y}^{(i)}$ is the predicted output.
- n is the number of samples.

- **Weight Update Rule**:

$$W^{(l)} = W^{(l)} - \eta \frac{\partial \text{Loss}}{\partial W^{(l)}}$$

References:

- Tensorflow Playground : A Neural Network Playground (tensorflow.org)
- Github Repository (Amruth-Sai-Mudivarthi):Amruth-Sai-Mudivarthi/Tensorflow_Playground (github.com)