

## ASSIGNMENT NO.1

**TITLE-** Tinker with Neural Network Visualization

**Problem Definition-** Visualize neural network architecture on <https://playground.tensorflow.org> website

**Objectives:** 1) Understanding of technical terms involved in neural network

2) Visualizing neural network using tensorflow playground website

**Outcome-** After completion of this assignment students are able to visualize the working of neural network

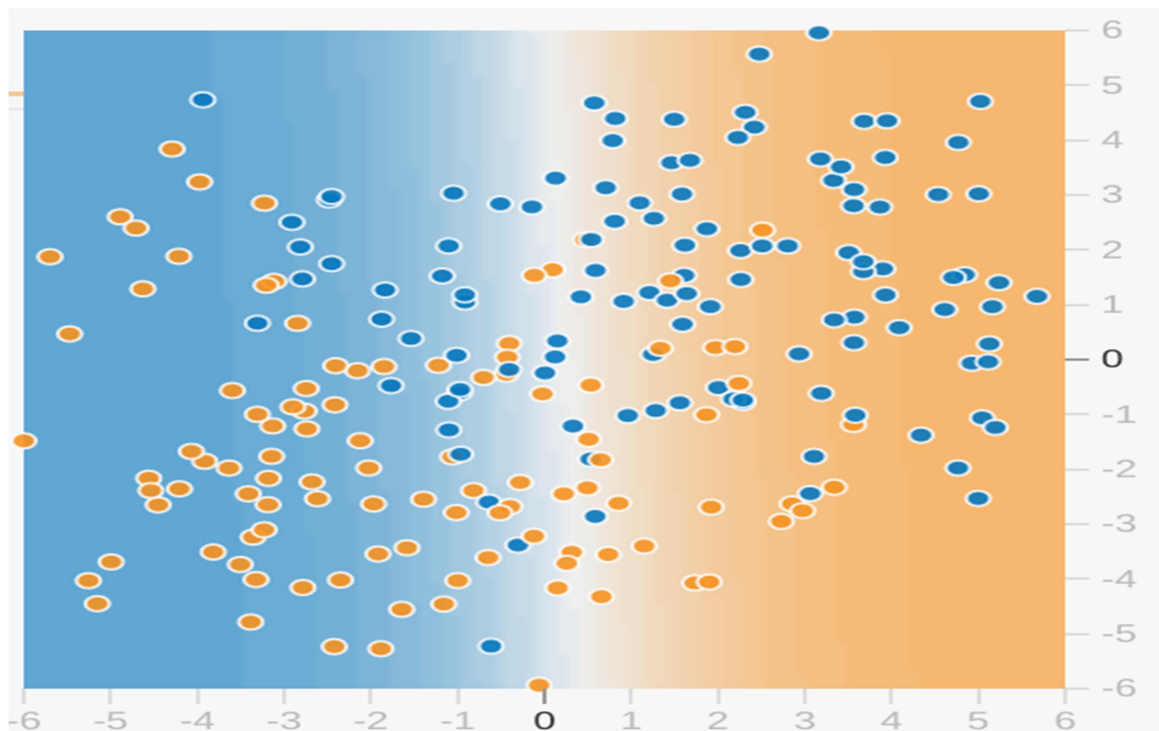
### **Theory-**

Playground-

Playground, Google's open-source work that has been very well-known for explaining how neural network in the interactive way.

Each Playground exercise generates a dataset. The label for this dataset has two possible values.

Each Playground exercise displays a visualization of the current state of the model. For example, here's a visualization:



- Each axis represents a specific feature. In the case of spam vs. not spam, the features could be the word count and the number of recipients of the email.
- Each dot plots the feature values for one example of the data. For example, the blue dots can represent non-spam emails while the orange dots can represent spam emails.
- The background color represents the model's prediction of where examples of that color should be found. A blue background around a blue dot means that the model is correctly predicting that example. Conversely, an orange background around a blue dot means that the model is incorrectly predicting that example.

- The background blues and oranges are scaled. For example, the left side of the visualization is solid blue but gradually fades to white in the center of the visualization. You can think of the color strength as suggesting the model's confidence in its guess. So solid blue means that the model is very confident about its guess and light blue means that the model is less confident

## Important Deep Learning Glossary

In brief, the neural network (also the “perceptron”) consists of, at least, three layers of nodes: **(1) an input layer, (2) a hidden layer and (3) an output layer**. If the neural network has multiple hidden layers, we call it the ‘**deep neural network**’ Some must-know key terms include:

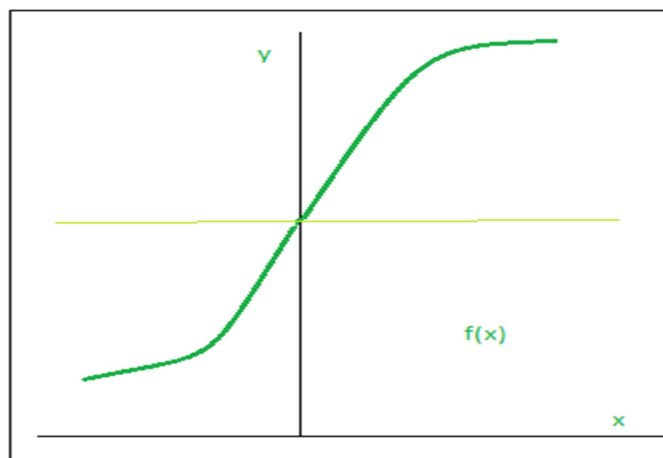
- **Epoch:** generally defined as “one pass over the entire dataset”. In each epoch we take our input data and run it through entire neural network. Then we find our prediction and calculate the error namely how far we are from actual labels and finally we backpropagate this error in order to update the weights.
- **Learning rate-**“a scalar used to train a model via gradient descent. During each iteration, the [gradient descent](#) algorithm multiplies the learning rate by the gradient. The resulting product is called the **gradient step**. In Tensorflow playground, the learning rate ranges from 0.00001 to 10.
- **Activation function-**The activation function is a non-linear transformation that we do over the input before sending it to the next layer of neurons or finalizing it as output. The function is attached to each neuron in the network, and determines whether it should be activated (“fired”) or not, based on whether each neuron’s input is relevant for the model’s prediction. Activation functions also help normalize the output of each neuron to a range between 1 and 0 or between -1 and 1.

Types of Activation Function-

1.Sigmoid Function: Sigmoid function is a widely used activation function. It is defined as:

$$A = 1/(1 + e^{-x})$$

Graphically,

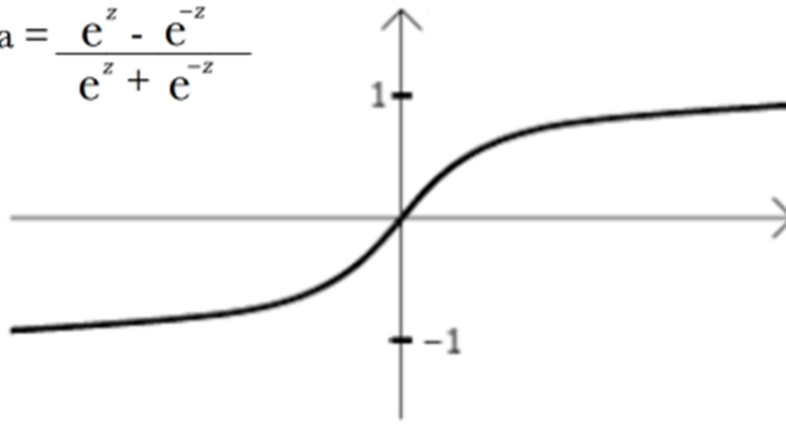


This is a smooth function and is continuously differentiable.

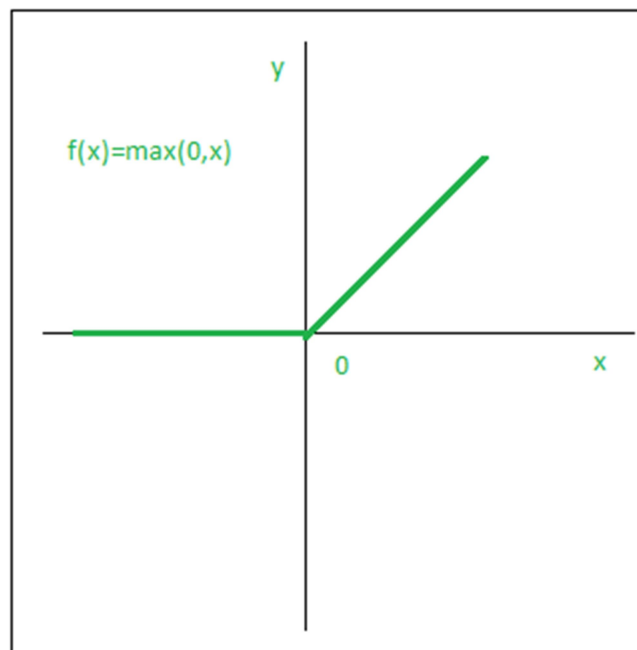
2.Tanh function: The activation that works almost always better than sigmoid function is Tanh function also known as **Tangent Hyperbolic function**. The range of the tanh function is from (-1 to 1)

### Tanh Function

$$a = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$



3)ReLU function: The ReLU function is the Rectified linear unit. It is the most widely used activation function. Graphically,




The main advantage of using the ReLU function over other activation functions is that it does not activate all the neurons at the same time. If the input is negative it will convert it to zero and the neuron does not get activated.

- **Regularization** — a hyperparameter to prevent overfitting. Available values are L1 and L2. L1 computes the sum of the weights, whereas L2 takes the sum of the square of the weights.
- **Regularization Rate** — a scalar used to specify the rate at which the model applies the regularization, ranging from 0 to 10.
- **Problem Type** — classification (categorical output) vs. regression (numerical output)

- **Ratio of the Training and Testing Sets** — the proportion of a subset to train a model and a subset to test a model. We usually set it to 80/20. In the visualization, note the following distinction:
  - ✓ The training examples have a white outline.
  - ✓ The test examples have a black outline.
- **Noise** — a distortion in data that is construed to be extraneous to the original data.
- **Batch Size** — “a small, randomly selected subset of the entire batch of examples run together in a single iteration of training or inference. The batch size of a mini-batch is usually between 10 and 1,000.” If we say we have 100 data points we split them into 5 batches of 20 points. We take points in first batch & run them through neural network, calculate error & its gradient & backpropagate to update weights. We repeat this for all batches
- **Features** — represents an input layer to feed in.
- **Hidden Layer** — a layer in between input layers and output layers, where artificial neurons take in a set of weighted inputs and produce an output through an activation function. In this context, you can specify as many as you want, but bear in mind that the more hidden layer you add, the more complex the model becomes.
- **Output** — an output layer in the neural network, often involving the loss evaluation. Loss function (or a cost function) is a method of evaluating how well the neural network performs in the given data. If predictions deviates too much from actual results, loss function will be high. We often evaluate the losses both on training and testing sets.

#### Steps to perform visualization:

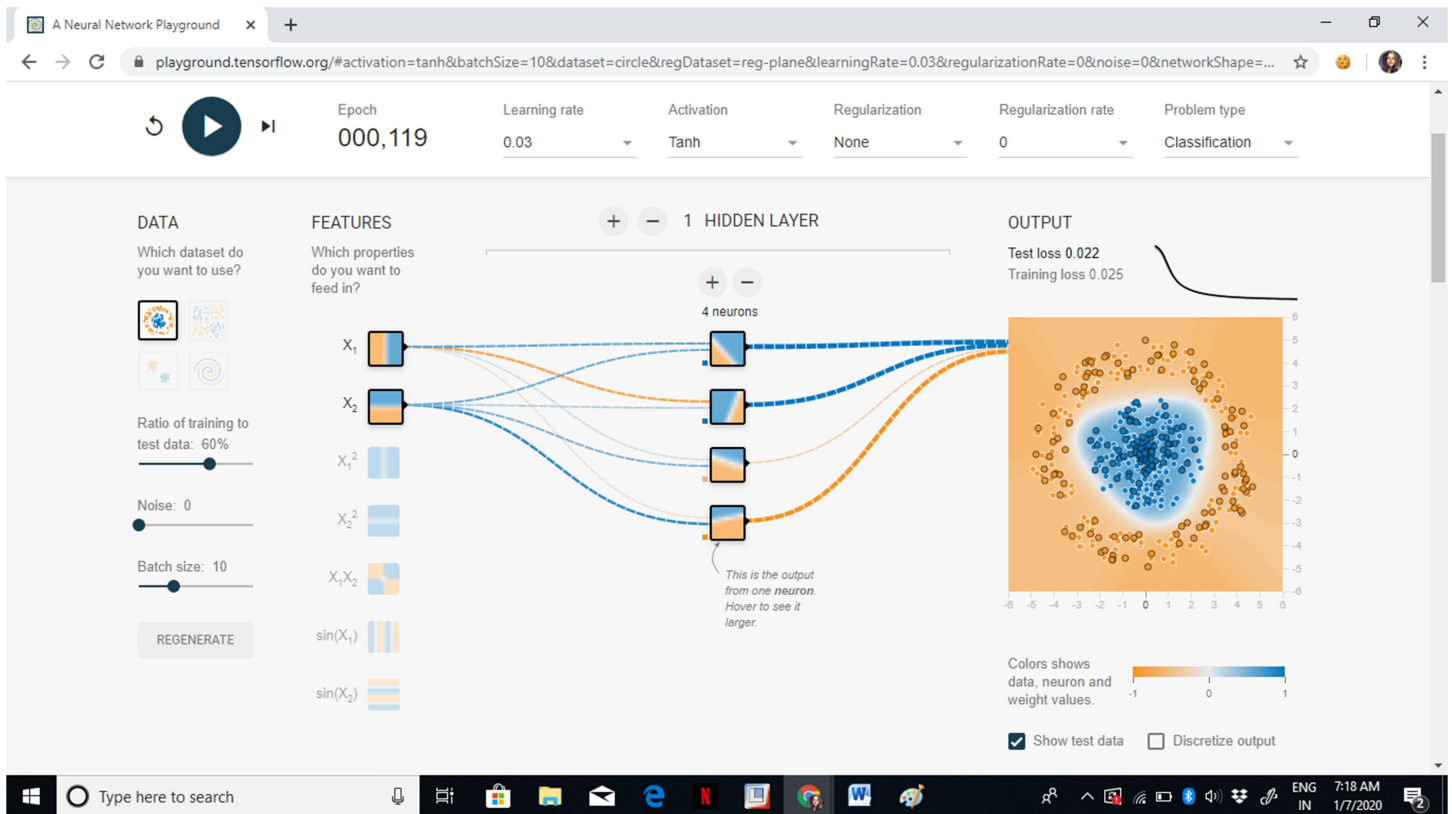
1. Click the Run/Pause button: 
2. Watch the Test loss and Training loss values change.
3. When the Test loss and Training loss values stop changing or only change once in a while, press the Run/Pause button again to pause Playground.

Note the delta between the Test loss and Training loss

#### **Technical Demo:**

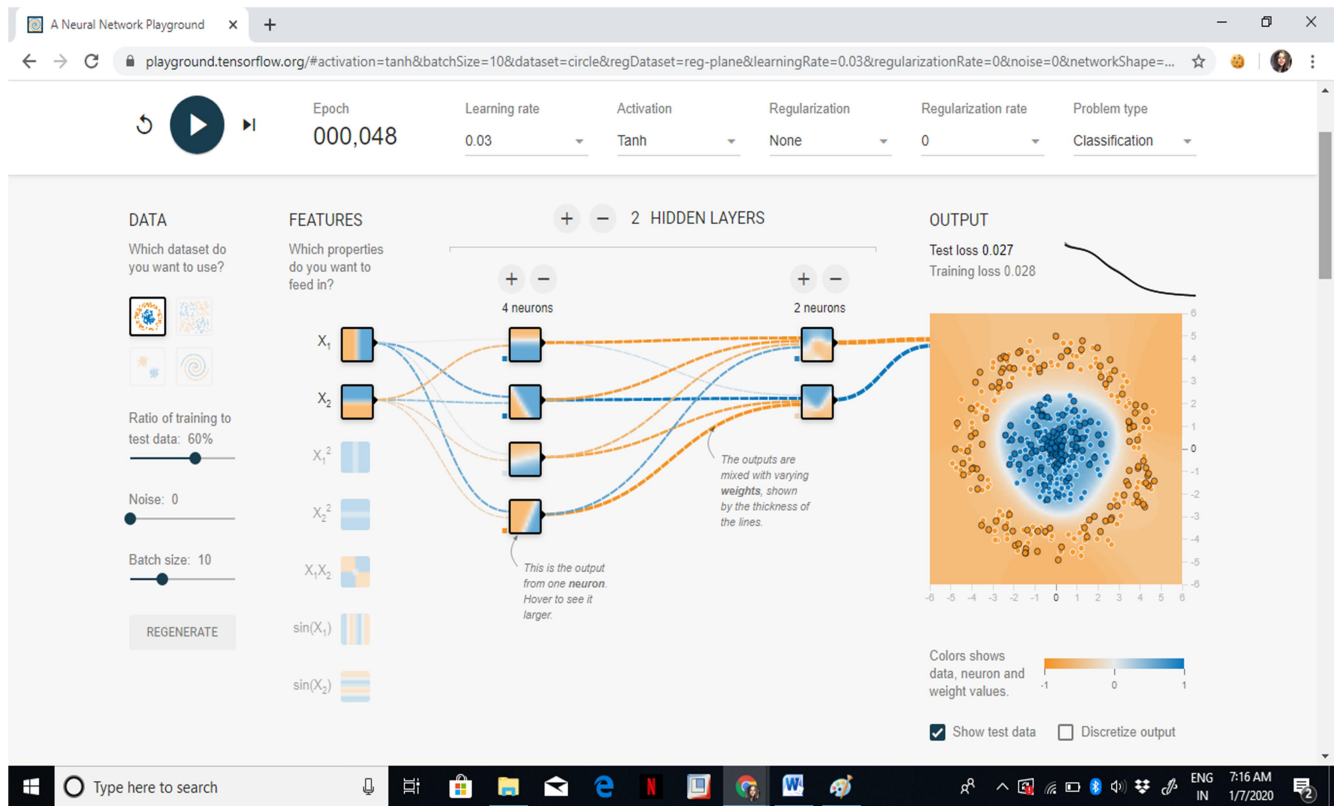
#### **Experiment #1: Tanh Activation Function**

- Learning rate: 0.003
- Activation function: Tanh
- Regularization: None
- Regularization rate: 0
- Problem type: Classification
- 1 Hidden Layers, one with 4 neurons



## Experiment #2: Tanh Activation Function- Added Hidden Layers

- Learning rate: 0.003
- Activation function: Tanh
- Regularization: None
- Regularization rate: 0
- Problem type: Classification
- 2 Hidden Layers, one with 4 neurons the other with 2 neurons



Observe that we are able to train model in less epochs.

Conclusion: Through this assignment we have studied the technical terms of neural network architecture and also performed visualization of neural network on interactive Tensorflow Playground