Faiza Abdullah

A06 TensorFlow Playground Presentation

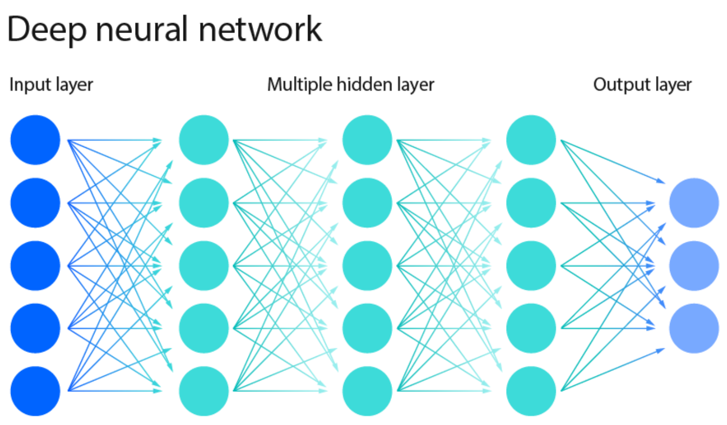
ITAI 1378 Comp Vision-Artificial Intelligence

Professor: Anna Devarakonda

Unveiling the Art and Science of Neural Networks in TensorFlow Playground

INTRODUCTION TO NEURAL NETWORKS AND THEIR COMPONENTS:

Neural networks are computational models inspired by the human brain, designed to recognize patterns and solve complex problems. They consist of interconnected nodes called neurons, organized into layers: an input layer, one or more hidden layers, and an output layer.



Key components include:

- Neurons: Basic units that process input data by applying weights, biases, and activation functions.
- Layers: The input layer receives data, hidden layers perform intermediate computations, and the output layer produces the final result.
- Activation Functions: Nonlinear functions that determine whether a neuron "fires," enabling the network to model complex relationships.
- Learning Rate: A hyperparameter controlling the step size of weight updates during training.
- Dataset: The data used to train and evaluate the network's performance.

Neural networks are significant in tasks like classification, regression, and pattern recognition, widely applied in fields like image processing and natural language understanding. This report explores how these components influence network behavior using TensorFlow Playground, an interactive tool available at *https://playground.tensorflow.org*.

EXPLORATION PHASE:

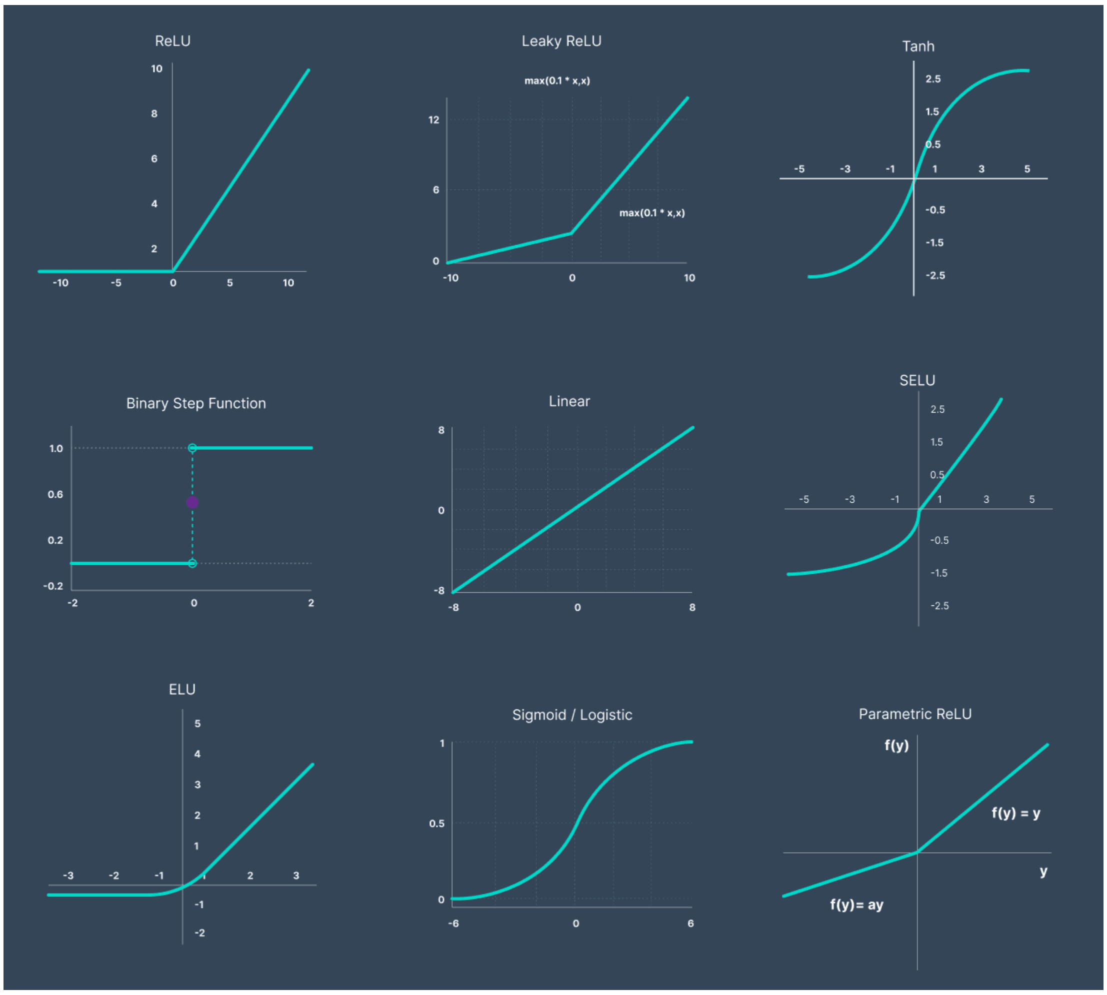
Using TensorFlow Playground, I conducted experiments to analyze how activation functions, neuron counts, learning rates, data noise, and datasets affect neural network performance. Below are the tasks, observations, and explanations.

Task 1 - Activation Functions

What Are Activation Functions?

Activation functions introduce nonlinearity into neural networks, allowing them to model complex patterns. Without nonlinearity, a network would behave like a linear regression model, regardless of depth. Common functions include:

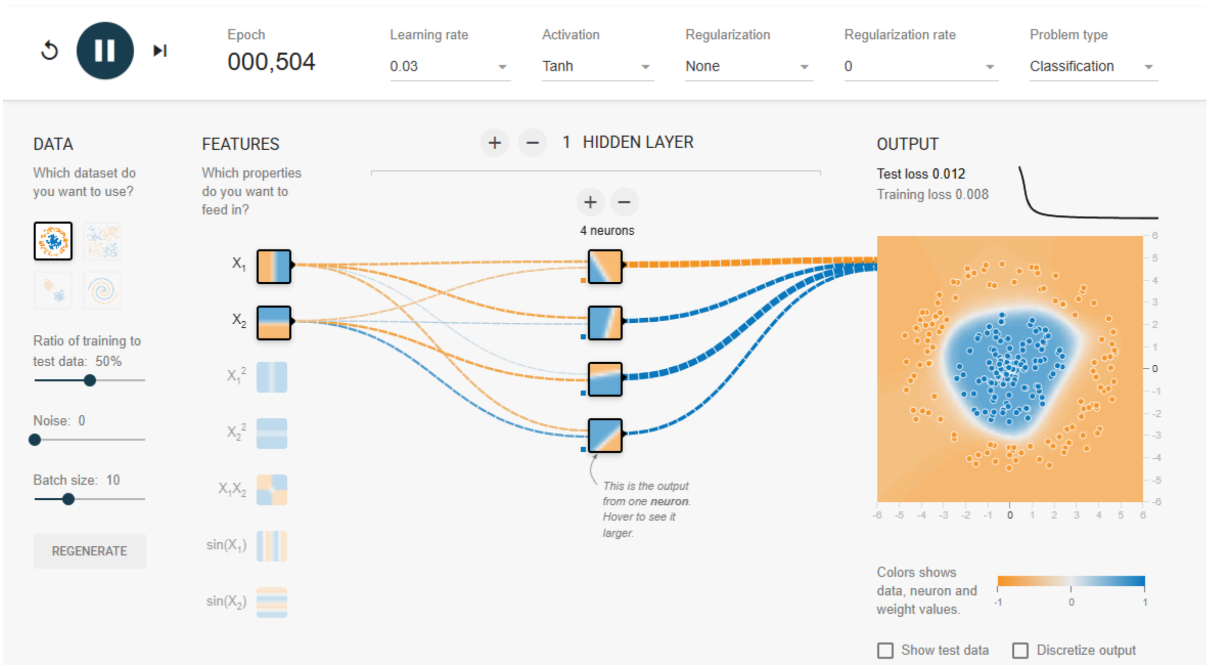
- Tanh: Maps inputs to (-1, 1), useful for centered data.
- ReLU (Rectified Linear Unit): Outputs max(0, x), promoting sparsity and faster convergence.
- Sigmoid: Maps inputs to (0, 1), often used in binary classification but prone to vanishing gradients.
- Linear: Outputs the input directly, limiting the network to linear transformations.



Experiment & Observations

I created a neural network with one hidden layer and tested four activation functions: Tanh, ReLU, Sigmoid, and Linear. The setup included the default dataset (e.g., spiral) and a fixed learning rate.

- Tanh: The network converged moderately well, capturing the spiral pattern with smooth boundaries.


Epoch 000,504

Learning rate 0.03

Activation Tanh

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

FEATURES

Which properties do you want to feed in?

1 HIDDEN LAYER

4 neurons

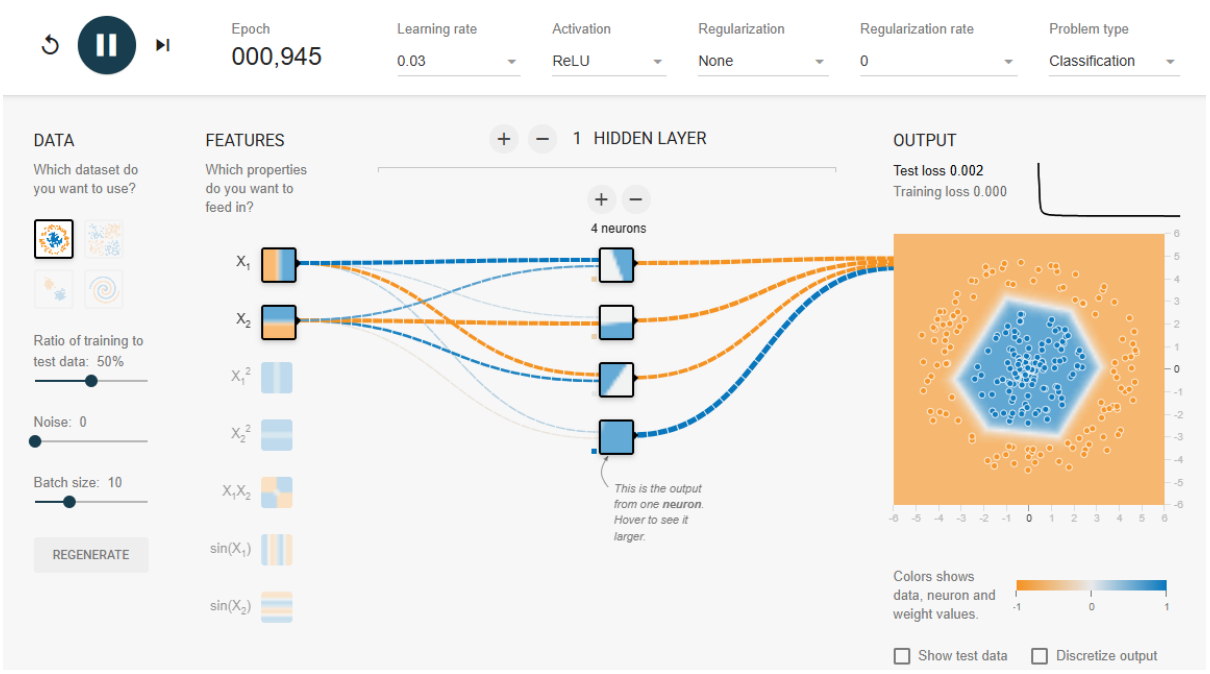
OUTPUT

Test loss 0.012

Training loss 0.008

REGENERATE
X1
X2
X1^2
X2^2
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
Colors shows data, neuron and weight values.
Show test data
Discretize output

- ReLU: Convergence was faster, with sharper decision boundaries, though it sometimes struggled with noisy data.


Epoch 000,945

Learning rate 0.03

Activation ReLU

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

FEATURES

Which properties do you want to feed in?

1 HIDDEN LAYER

4 neurons

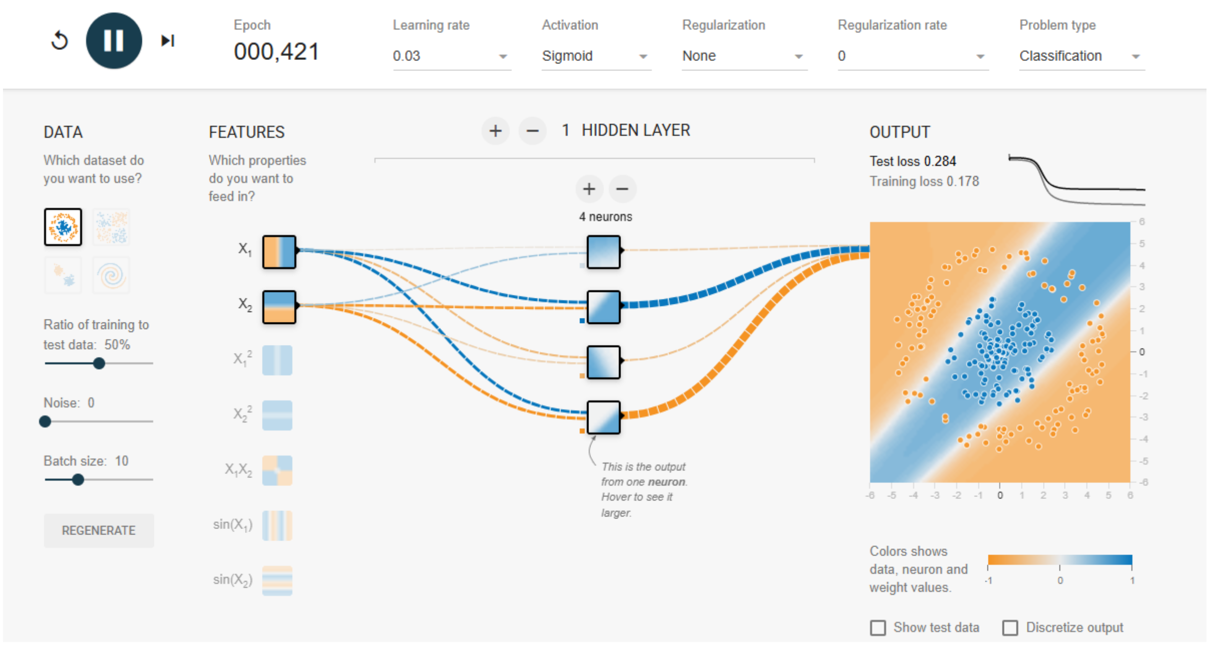
OUTPUT

Test loss 0.002

Training loss 0.000

REGENERATE
X1
X2
X1^2
X2^2
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
Colors shows data, neuron and weight values.
Show test data
Discretize output

- Sigmoid: Slower convergence due to vanishing gradients, with less defined boundaries. (Insert screenshot: "One hidden layer – Sigmoid")


Epoch 000,421

Learning rate 0.03

Activation Sigmoid

Regularization None

Regularization rate 0

Problem type Classification

DATA
Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

FEATURES
Which properties do you want to feed in?

1 HIDDEN LAYER

4 neurons

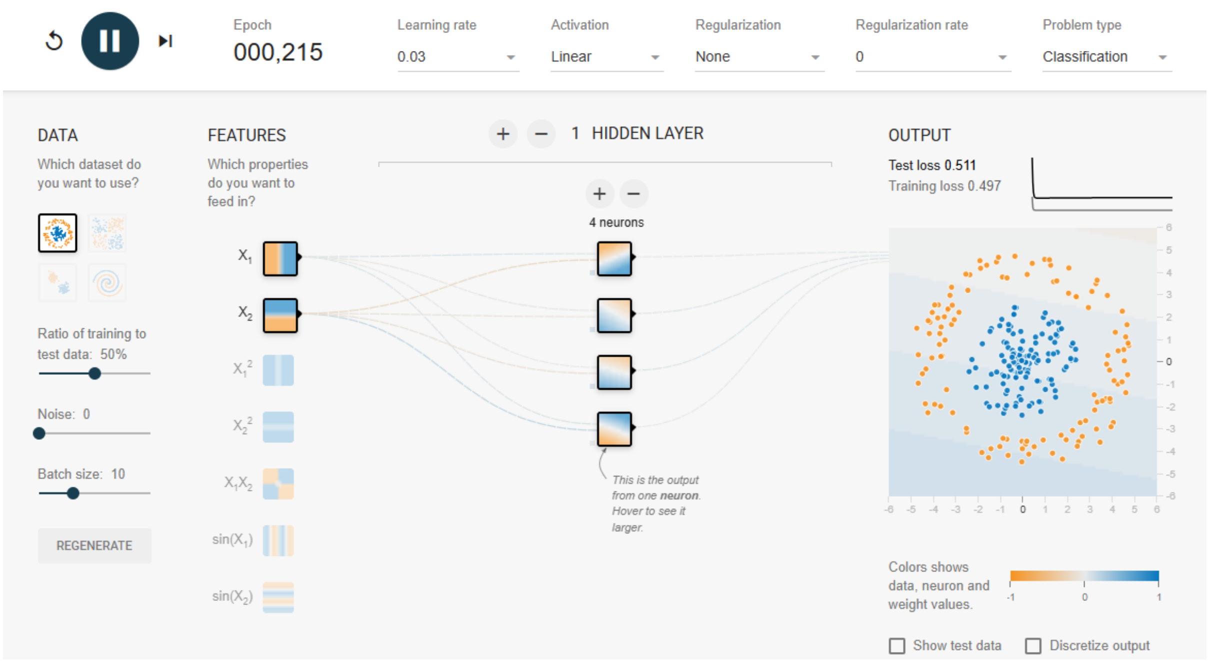
OUTPUT

Test loss 0.284

Training loss 0.178

REGENERATE
This is the output from one neuron. Hover to see it larger.
Colors shows data, neuron and weight values.
Show test data
Discretize output

- Linear: Poor performance; the network failed to capture nonlinear patterns, resulting in high error. (Insert screenshot: "One hidden layer – Linear")


Epoch 000,215

Learning rate 0.03

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

DATA
Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

FEATURES
Which properties do you want to feed in?

1 HIDDEN LAYER

4 neurons

OUTPUT

Test loss 0.511

Training loss 0.497

REGENERATE
This is the output from one neuron. Hover to see it larger.
Colors shows data, neuron and weight values.
Show test data
Discretize output

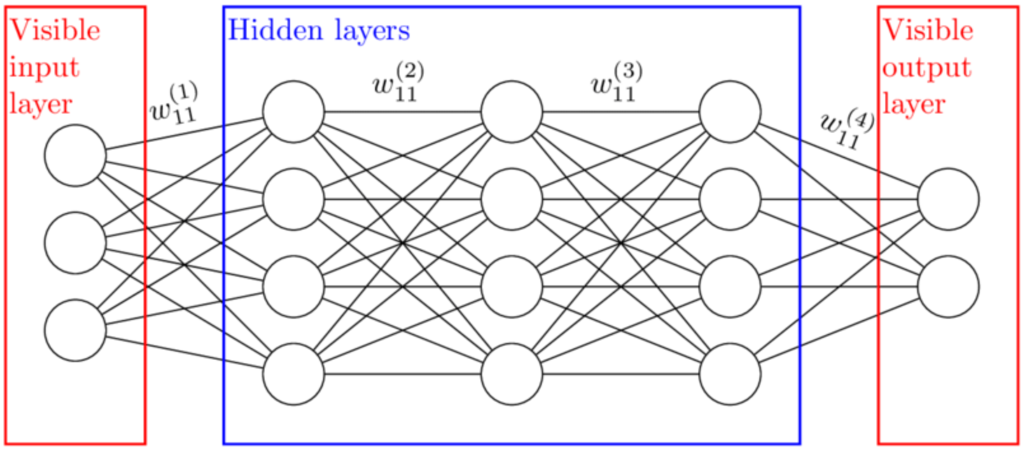
Explanation

ReLU's sparsity and gradient properties often lead to faster training, while Sigmoid's saturation slows learning. Tanh balances between the two, and Linear lacks the nonlinearity needed for complex tasks.

Task 2 - Hidden Layer Neurons

Role of Neurons and Hidden Layers

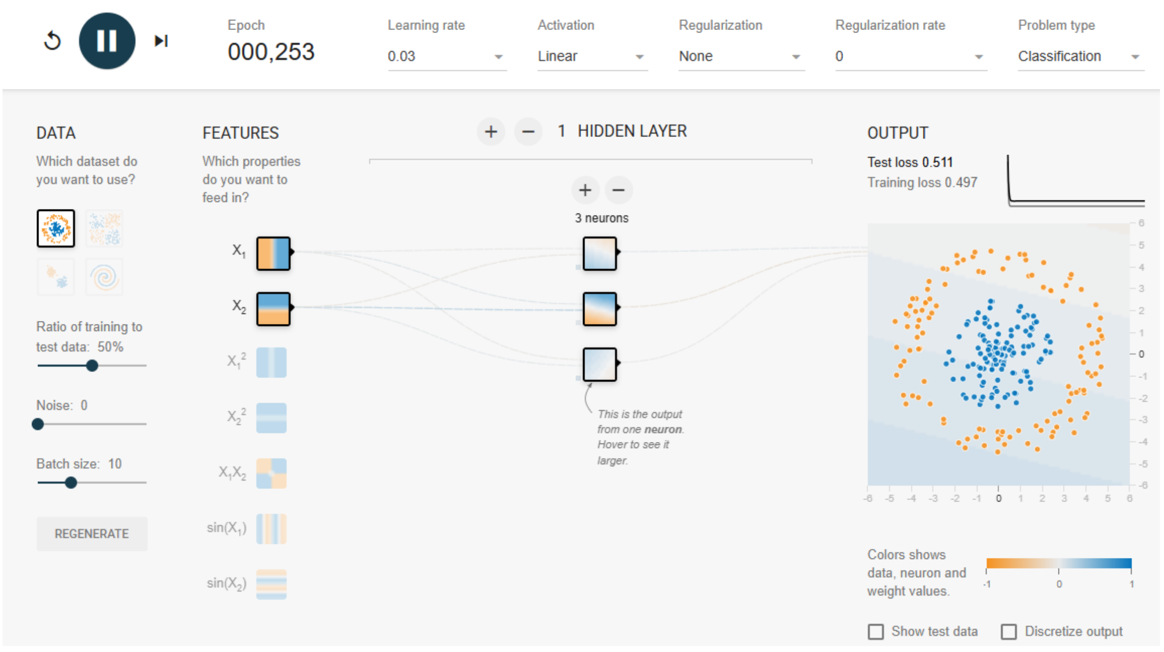
Neurons in hidden layers compute weighted sums of inputs, applying activation functions to extract features. More neurons increase model capacity, while additional layers enable hierarchical feature learning (e.g., edges to shapes).



Experiment & Observations

I varied the number of hidden layers and neurons and observed:

- 1 Layer, 3 Neurons: Limited capacity; struggled with complex patterns like spirals.



Epoch 000,253
Learning rate 0.03
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

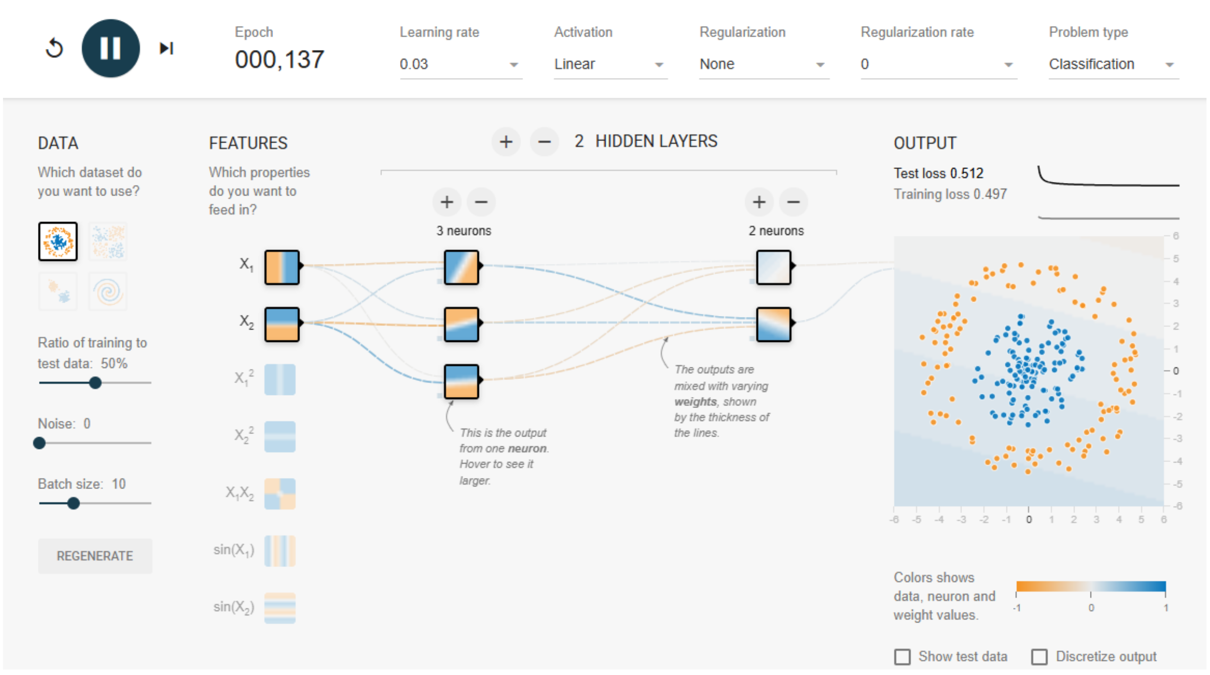
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?
X_1, X_2, X_1^2, X_2^2, X_1X_2, $\sin(X_1)$, $\sin(X_2)$

1 HIDDEN LAYER
3 neurons
This is the output from one neuron. Hover to see it larger.

OUTPUT
Test loss 0.511
Training loss 0.497
Colors shows data, neuron and weight values. -1 0 1
Show test data
Discretize output

- 2 Layers, 3 Neurons: Improved performance, capturing more intricate boundaries.


Epoch 000,137
Learning rate 0.03
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

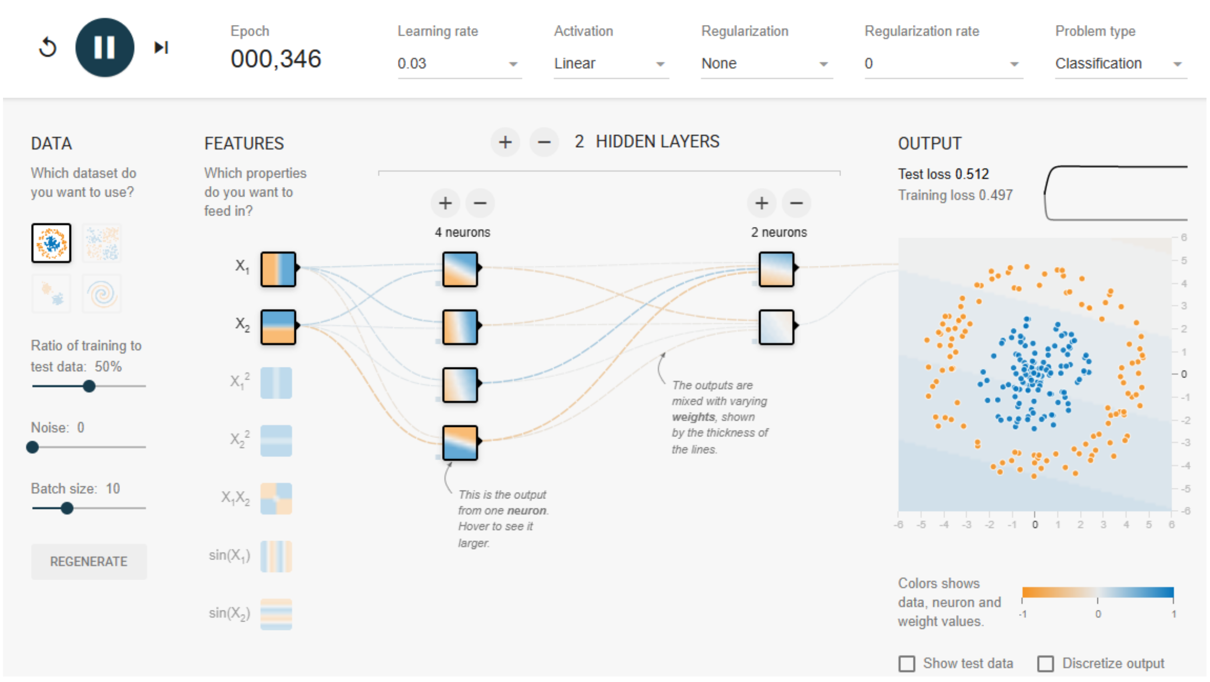
FEATURES
Which properties do you want to feed in?

2 HIDDEN LAYERS
3 neurons
2 neurons
This is the output from one **neuron**. Hover to see it larger.
The outputs are mixed with varying **weights**, shown by the thickness of the lines.

OUTPUT
Test loss 0.512
Training loss 0.497
Colors shows data, neuron and weight values.

Show test data
Discretize output

- 2 Layers, 4 Neurons: Further refinement in accuracy and boundary definition.


Epoch 000,346
Learning rate 0.03
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

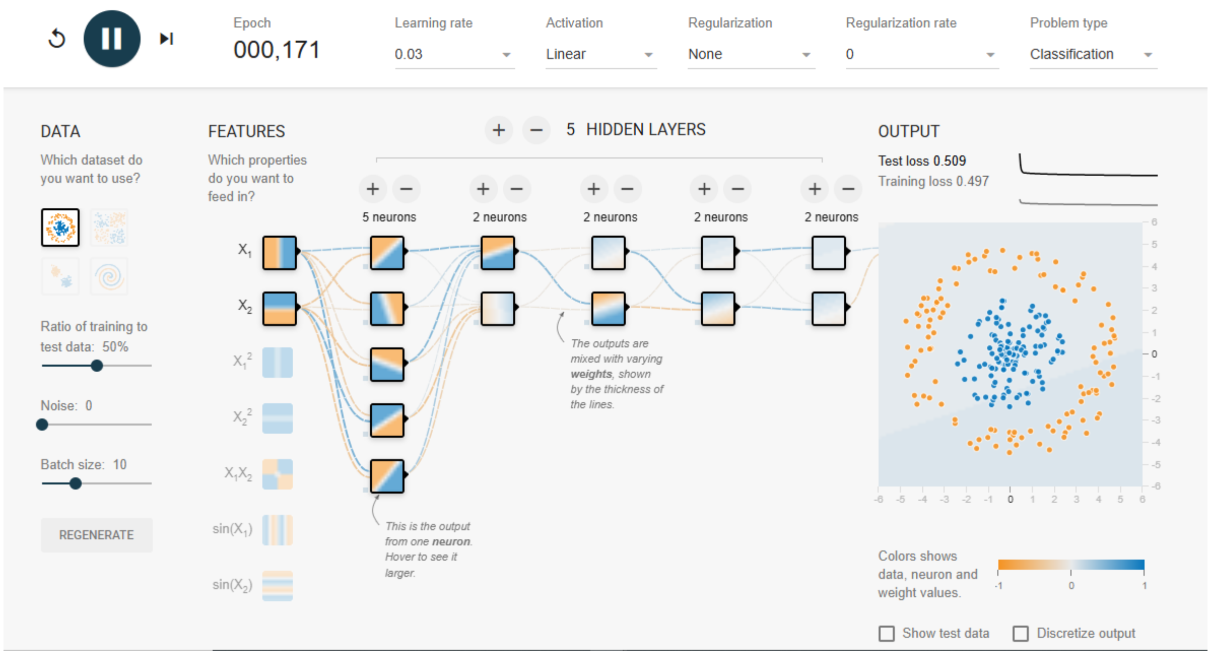
FEATURES
Which properties do you want to feed in?

2 HIDDEN LAYERS
4 neurons
2 neurons
This is the output from one **neuron**. Hover to see it larger.
The outputs are mixed with varying **weights**, shown by the thickness of the lines.

OUTPUT
Test loss 0.512
Training loss 0.497
Colors shows data, neuron and weight values.

Show test data
Discretize output

- 5 Layers, 5 Neurons: High capacity but risked overfitting; performance varied with dataset complexity.



Epoch 000,171

Learning rate 0.03

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

REGENERATE

FEATURES

Which properties do you want to feed in?

5 HIDDEN LAYERS

5 neurons 2 neurons 2 neurons 2 neurons 2 neurons

The outputs are mixed with varying *weights*, shown by the thickness of the lines.

This is the output from one *neuron*. Hover to see it larger.

OUTPUT

Test loss 0.509

Training loss 0.497

Colors shows data, neuron and weight values.

Show test data Discretize output

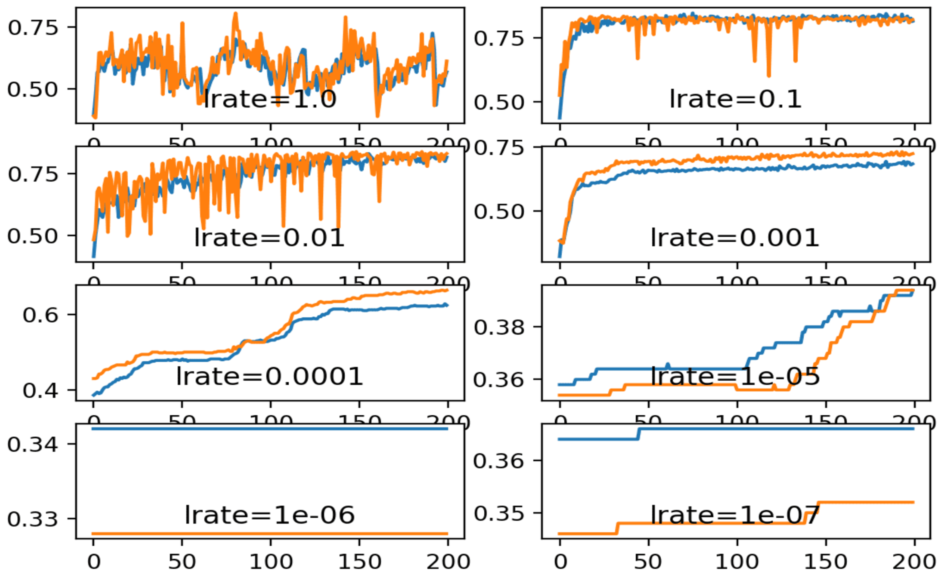
Explanation

Increasing neurons and layers enhances the network’s ability to model nonlinearity, but too many can overfitting, especially with small datasets.

Task 3 - Learning Rate

What Is Learning Rate?

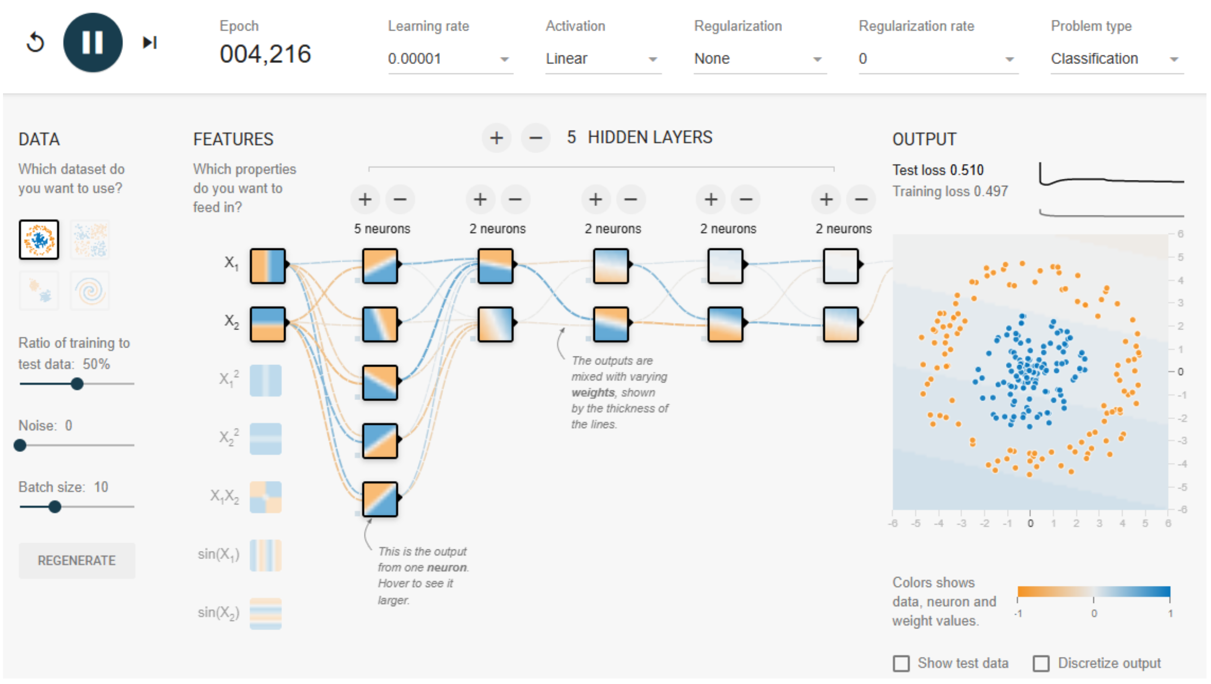
The learning rate determines the step size of weight updates during gradient descent. Too small a rate slows convergence; too large a rate causes instability or divergence.



Experiment & Observations

I tested learning rates of 0.00001, 3, and 10, observing convergence speed and accuracy.

- Learning Rate: 0.00001: Extremely slow convergence; error decreased minimally after 10 epochs.



Epoch 004,216
Learning rate 0.00001
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?

5 HIDDEN LAYERS
5 neurons 2 neurons 2 neurons 2 neurons 2 neurons

The outputs are mixed with varying **weights**, shown by the thickness of the lines.

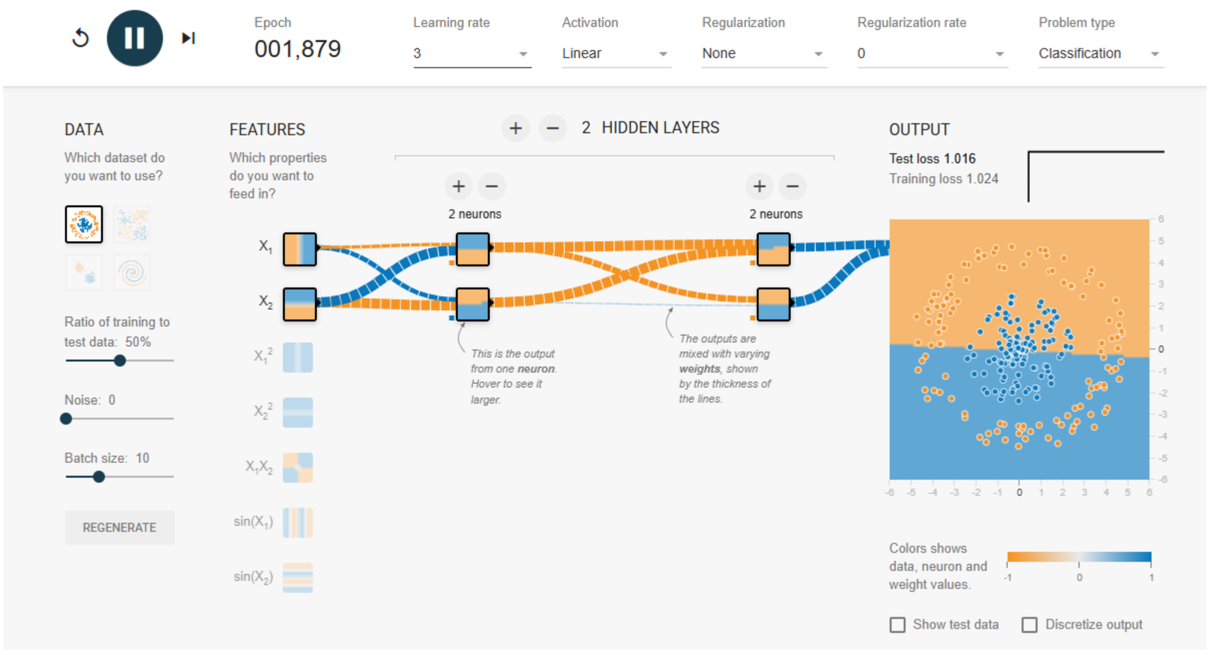
This is the output from one **neuron**. Hover to see it larger.

OUTPUT
Test loss 0.510
Training loss 0.497

Colors shows data, neuron and weight values.

Show test data
Discretize output

- Learning Rate: 3: Overshot optimal weights, leading to oscillation or divergence.



Epoch 001,879
Learning rate 3
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?

2 HIDDEN LAYERS
2 neurons 2 neurons

This is the output from one **neuron**. Hover to see it larger.

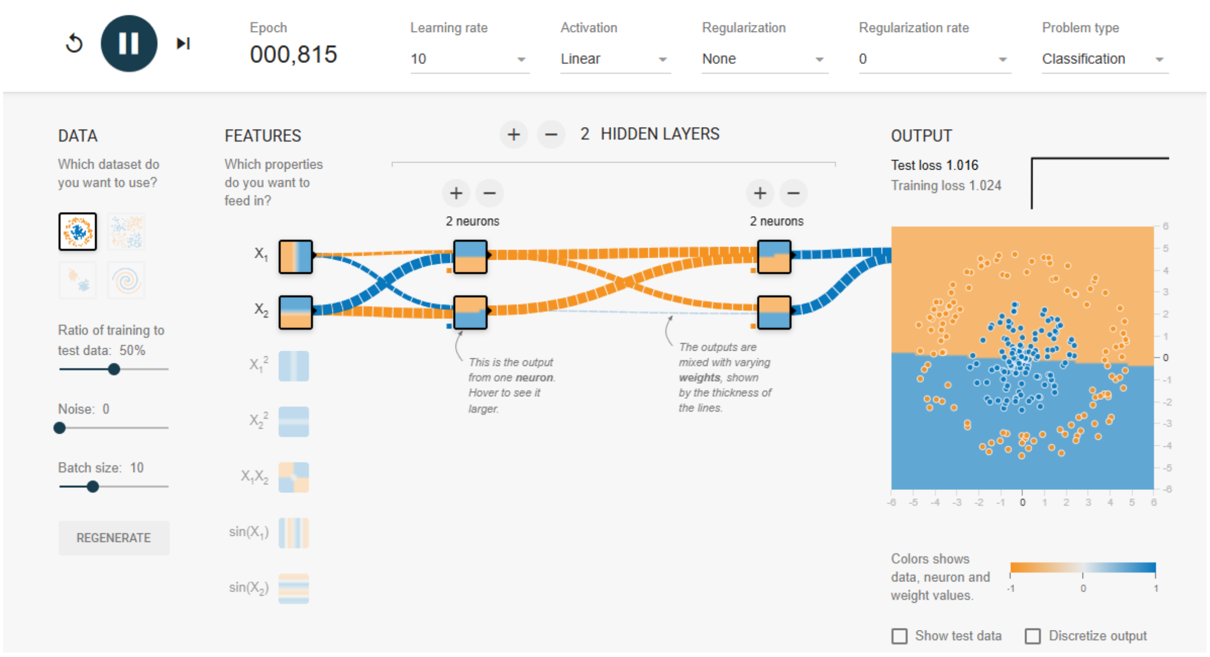
The outputs are mixed with varying **weights**, shown by the thickness of the lines.

OUTPUT
Test loss 1.016
Training loss 1.024

Colors shows data, neuron and weight values.

Show test data
Discretize output

- Learning Rate: 10: Highly unstable; the network failed to converge, with error rates fluctuating wildly or increasing over time.



Epoch 000,815

Learning rate 10

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

Which properties do you want to feed in?

2 HIDDEN LAYERS

2 neurons

2 neurons

This is the output from one **neuron**. Hover to see it larger.

The outputs are mixed with varying **weights**, shown by the thickness of the lines.

Test loss 1.016

Training loss 1.024

Colors shows data, neuron and weight values.

Explanation

A low learning rate like 0.00001 ensures stability but requires more epochs, while a high rate like 3 risks missing the optimum. A learning rate of 10 is excessively large for most datasets in TensorFlow Playground (e.g., spiral or circle), causing the model to overshoot repeatedly and fail to settle on a solution. Optimal rates typically lie in a moderate range (e.g., 0.01–1), depending on the problem complexity.

Task 4 - Data Noise

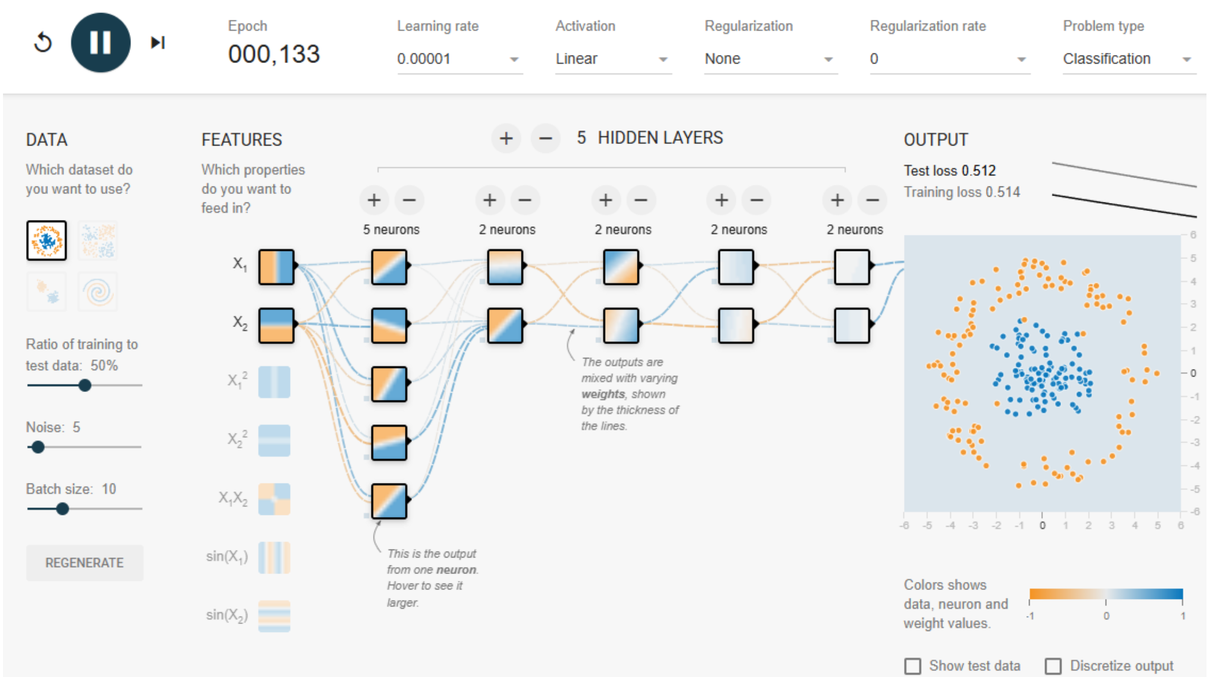
What Is Data Noise?

Data noise refers to random variations or errors in the dataset, simulating real-world imperfections. It challenges the network's ability to generalize beyond training data.

Experiment & Observations

I adjusted the noise slider to 5, 15, and 30, observing generalization accordingly:

- Noise: 5: Minimal impact; the network maintained good accuracy.


Epoch 000,133

Learning rate 0.00001

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 5

Batch size: 10

FEATURES

Which properties do you want to feed in?

5 HIDDEN LAYERS

5 neurons, 2 neurons, 2 neurons, 2 neurons, 2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT

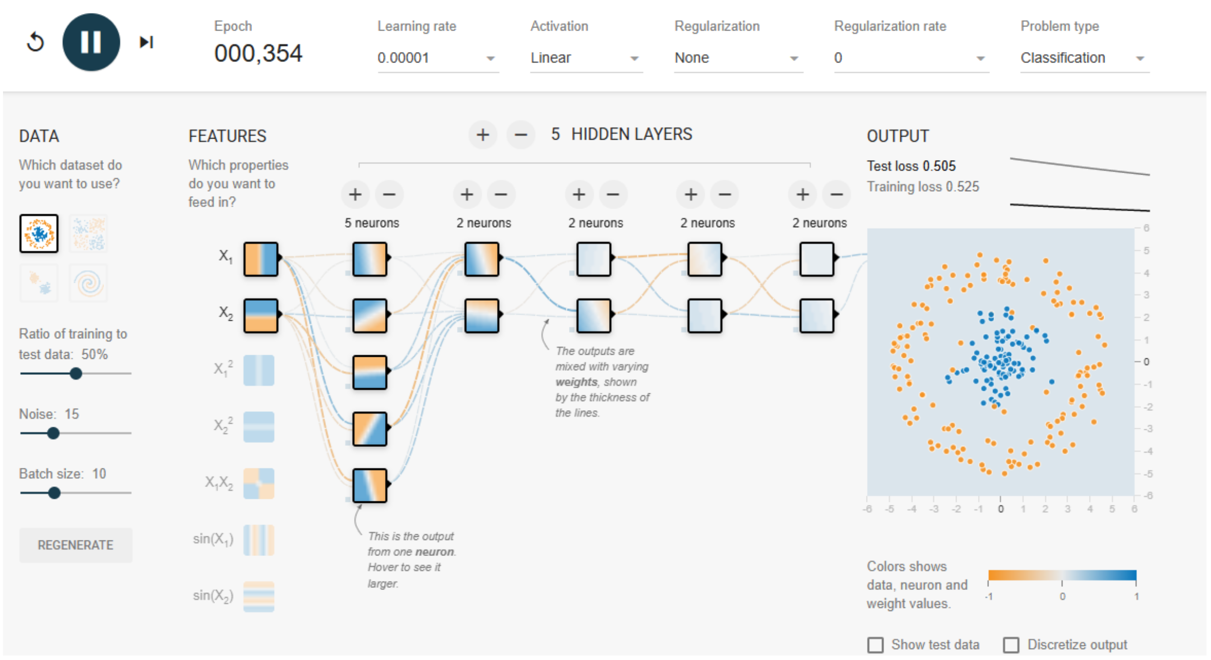
Test loss 0.512

Training loss 0.514

Colors shows data, neuron and weight values.

Show test data Discretize output

- Noise: 15: Reduced accuracy; boundaries became less precise.


Epoch 000,354

Learning rate 0.00001

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 15

Batch size: 10

FEATURES

Which properties do you want to feed in?

5 HIDDEN LAYERS

5 neurons, 2 neurons, 2 neurons, 2 neurons, 2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT

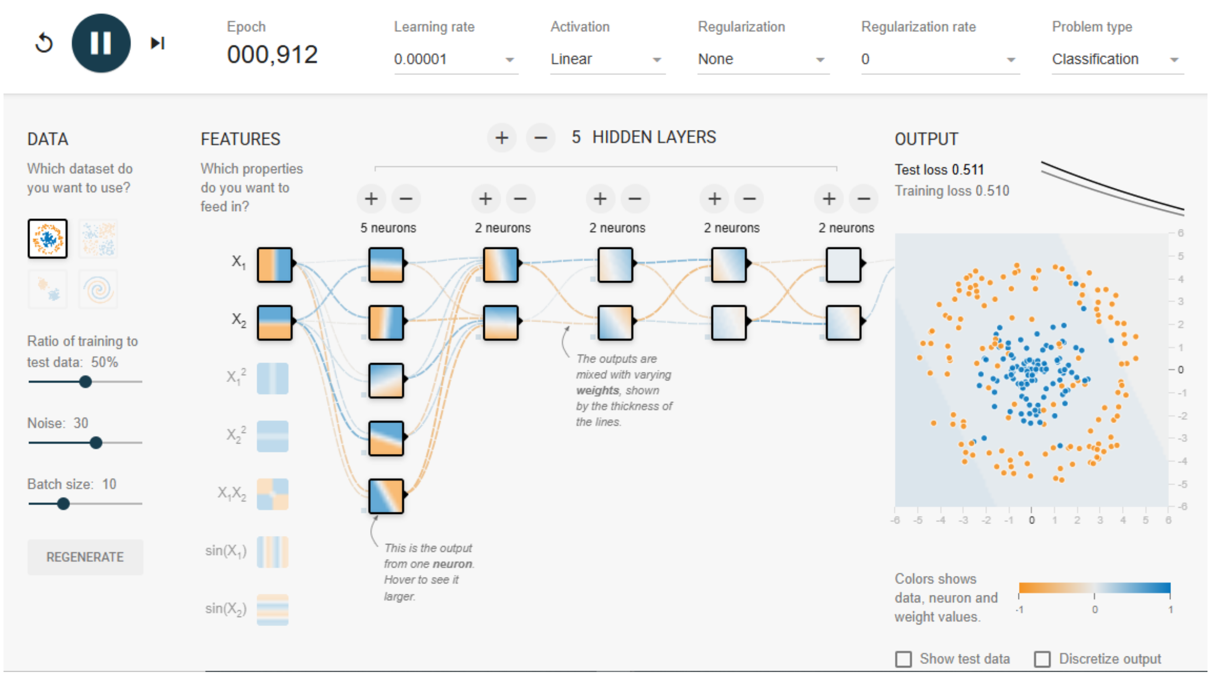
Test loss 0.505

Training loss 0.525

Colors shows data, neuron and weight values.

Show test data Discretize output

- Noise: 30: Significant degradation; the network overfit or failed to generalize.


DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output

Epoch 000,912

Learning rate 0.00001

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

Ratio of training to test data: 50%

Noise: 30

Batch size: 10

5 HIDDEN LAYERS: 5 neurons, 2 neurons, 2 neurons, 2 neurons, 2 neurons

Test loss 0.511

Training loss 0.510

Explanation

Moderate noise tests robustness, while high noise obscures patterns, requiring regularization (e.g., dropout) for better generalization.

Task 5 - Dataset Exploration

Importance of Dataset Selection

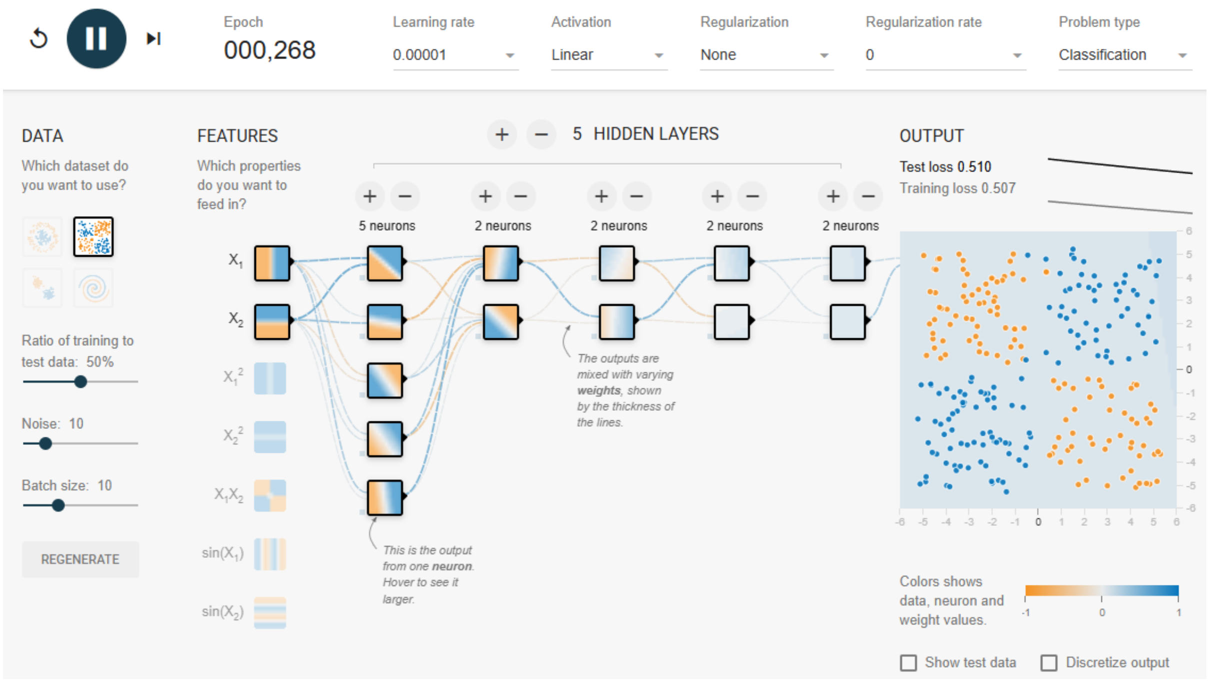
Datasets define the problem's complexity and influence architecture choice. Simple datasets (e.g., circle) require fewer layers, while complex ones (e.g., spiral) need deeper networks.

Experiment & Observations

I tested Datasets 1 (circle), 2 (circle with overlap), 3 (exclusive-or), and 4 (spiral), analyzing performance.

Throughout this assignment I have used Dataset 2 – circle with overlap hence here only showing other three.

- Dataset 2 (Circle): Easily separated with one layer; ReLU performed well.


Epoch 000,268 | Learning rate 0.00001 | Activation Linear | Regularization None | Regularization rate 0 | Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 10
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?
X_1, X_2, X_1^2, X_2^2, X_1X_2, $\sin(X_1)$, $\sin(X_2)$

5 HIDDEN LAYERS
5 neurons | 2 neurons | 2 neurons | 2 neurons | 2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

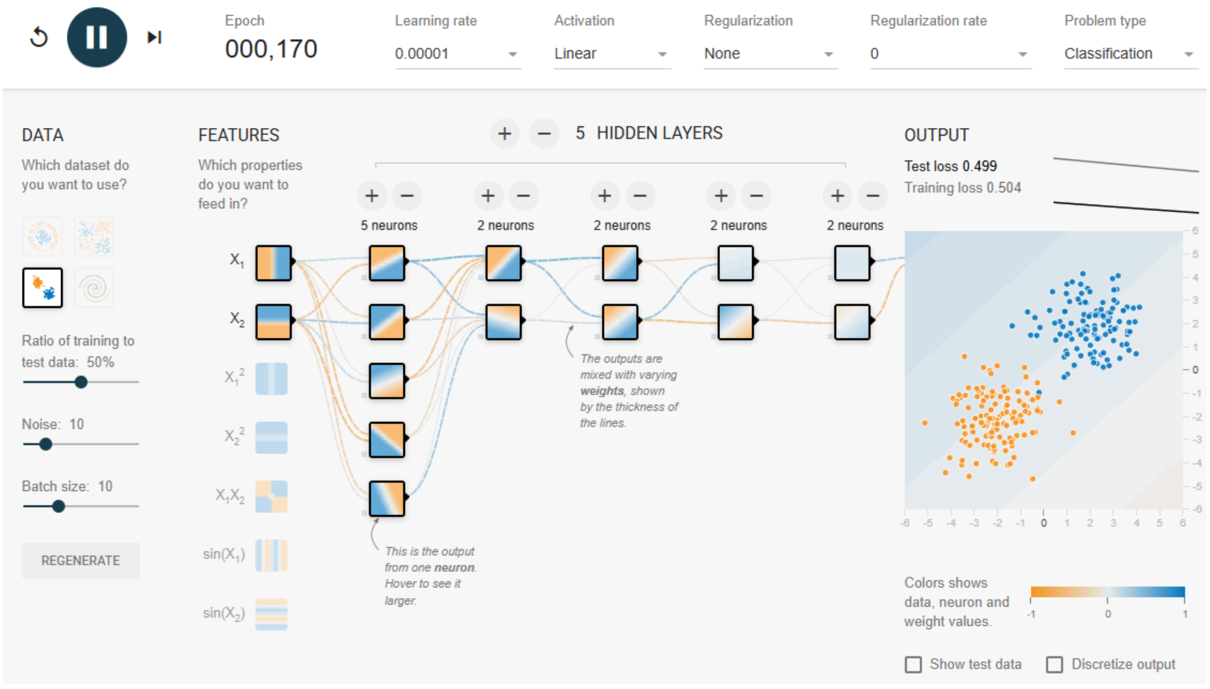
*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT
Test loss 0.510
Training loss 0.507

Colors shows data, neuron and weight values.

Show test data   Discretize output

- Dataset 3 (XOR): Required at least two layers for nonlinearity.


Epoch 000,170 | Learning rate 0.00001 | Activation Linear | Regularization None | Regularization rate 0 | Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 10
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?
X_1, X_2, X_1^2, X_2^2, X_1X_2, $\sin(X_1)$, $\sin(X_2)$

5 HIDDEN LAYERS
5 neurons | 2 neurons | 2 neurons | 2 neurons | 2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

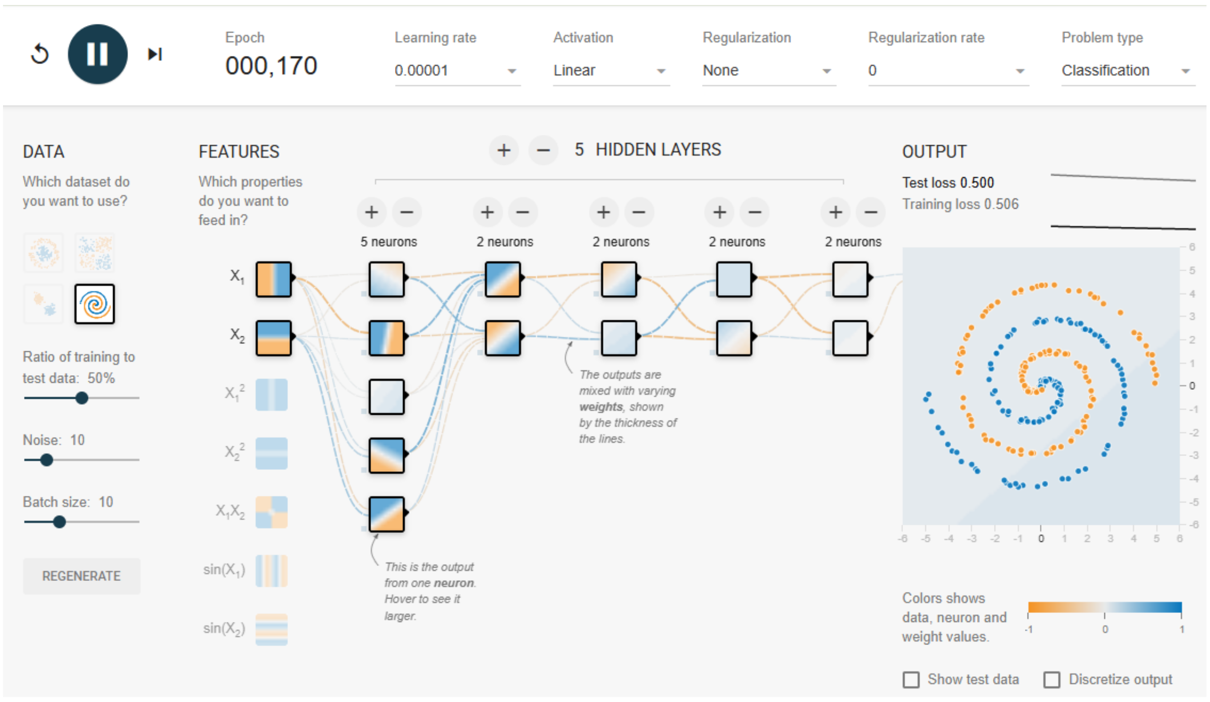
*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT
Test loss 0.499
Training loss 0.504

Colors shows data, neuron and weight values.

Show test data   Discretize output

- Dataset 4 (Spiral): Best with multiple layers and neurons; sensitive to noise.



Epoch 000,170
Learning rate 0.00001
Activation Linear
Regularization None
Regularization rate 0
Problem type Classification

Ratio of training to test data: 50%
Noise: 10
Batch size: 10

This is the output from one **neuron**. Hover to see it larger.

The outputs are mixed with varying **weights**, shown by the thickness of the lines.

Test loss 0.500
Training loss 0.506

Colors shows data, neuron and weight values.

Explanation

- Dataset 1 (Circle): This dataset represents a linearly separable problem with two distinct classes forming concentric circles. A simple network with one hidden layer and a nonlinear activation like ReLU can model it effectively, as the boundary is straightforward.
- Dataset 2 (Circle with Overlap): Adds complexity with overlapping points, testing the network's ability to generalize. More neurons or layers improve accuracy by refining the decision boundary.
- Dataset 3 (XOR): A classic nonlinear problem requiring at least two layers, as a single layer cannot represent the XOR function due to its non-monotonic nature.
- Dataset 4 (Spiral): Highly nonlinear and intricate, demanding deeper architectures and careful parameter tuning to trace the spiral arms accurately.

Dataset complexity dictates network depth and capacity, highlighting the need for tailored architectures. For instance, Dataset 1's simplicity contrasts with Dataset 4's intricacy, showing how problem structure drives design choices.

PRACTICAL IMPLICATIONS

These findings have real-world applications:

- Activation Functions: ReLU is preferred in deep networks for speed, but Tanh suits smaller models.
- Neurons/Layers: Balance capacity and overfitting in tasks like image recognition.
- Learning Rate: Tune for efficiency in time-sensitive applications (e.g., autonomous driving).
- Noise: Robustness to noise is critical in noisy environments (e.g., sensor data).
- Datasets: Dataset choice informs model design in domains like medical diagnostics.
- Understanding these parameters ensures effective neural network development.

CONCLUSION

This hands-on exploration revealed how neural network parameters shape performance. Challenges included balancing complexity and overfitting, overcome by iterative testing. I learned the interplay of activation functions, architecture, and hyperparameters, foundational to designing practical AI systems.

CITATIONS:

https://eagleonline.hccs.edu/courses/278598/files/70587251?module_item_id=18927460

https://playground.tensorflow.org

https://www.deeplearningbook.org/

https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/

https://pyimagesearch.com/2021/05/06/introduction-to-neural-networks/

https://www.ibm.com/think/topics/neural-networks

https://www.geeksforgeeks.org/activation-functions-neural-networks/

https://www.v7labs.com/blog/neural-networks-activation-functions

https://www.geeksforgeeks.org/activation-functions/

http://neuralnetworksanddeeplearning.com/

https://developers.google.com/machine-learning/crash-course/neural-networks/nodes-hidden-layers

https://www.coursera.org/articles/hidden-layer-neural-network

https://stats.stackexchange.com/questions/63152/what-does-the-hidden-layer-in-a-neural-network-compute

https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

https://www.geeksforgeeks.org/impact-of-learning-rate-on-a-model/

https://www.rudderstack.com/learn/machine-learning/generalization-in-machine-learning/

https://cloud.google.com/blog/products/ai-machine-learning/understanding-neural-networks-with-tensorflow-playground