Perceptron-Based Classification Using SGD

1. PROJECT OBJECTIVE

The goal of this project is to build a Fully Connected Neural Network (FCNN) from scratch to perform multi-class classification on 2D datasets. The implementation focuses on the manual application of **Stochastic Gradient Descent (SGD)** and **Backpropagation** using **Squared Error Loss**.

2. DATASET SPECIFICATIONS

We analyzed two distinct types of datasets:

1. **Linearly Separable:** 3 classes, 500 points per class (1500 total). This requires the model to find straight-line boundaries.
2. **Nonlinearly Separable:** 2D data where classes are interleaved or concentric, requiring curved decision surfaces.

Data Partitioning

To ensure the model generalizes well to unseen data, we divided each class as follows:

- **Training (60%):** Used for learning and weight updates.
- **Validation (20%):** Used for architecture selection and preventing overfitting.
- **Testing (20%):** Used for the final performance report.

3. SYSTEM ARCHITECTURE

Based on the complexity of the data, the following architectures were selected:

Component	Dataset 1 (Linear)	Dataset 2 (Nonlinear)
Input Nodes	2 (x, y coordinates)	2 (x, y coordinates)
Hidden Layers	1 Layer	2 Layers

Component	Dataset 1 (Linear)	Dataset 2 (Nonlinear)
Output Nodes	3 (One-hot encoded)	2 or 3 (One-hot encoded)
Activation	Sigmoid	Sigmoid

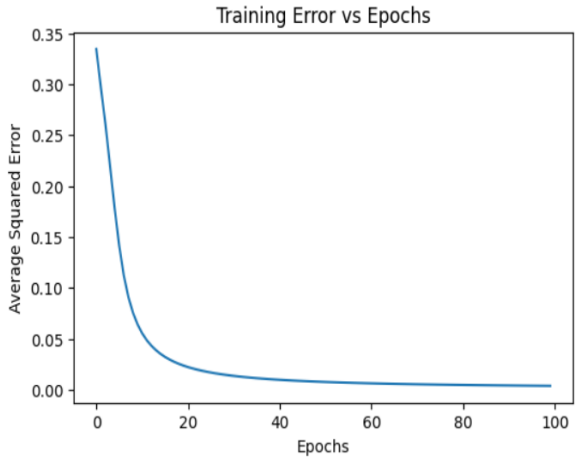
4. METHODOLOGY

- **Forward Pass:** Input is passed through weight matrices and biased, then activated via the Sigmoid function.
- **Loss Calculation:** Instantaneous Squared Error: E = 1/2(Target - Output)^2.
- **Backpropagation:** Error is sent backward using the Chain Rule to calculate gradients.
- **Stochastic Gradient Descent (SGD):** Weights are updated immediately after every single data point, allowing for faster convergence and avoiding local minima.
-

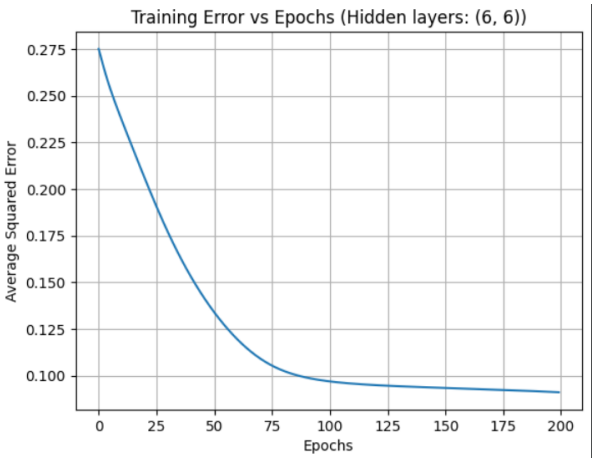
5. RESULTS & PRESENTATION

5.1 Learning Progress

DATASET-1



DATASET-2

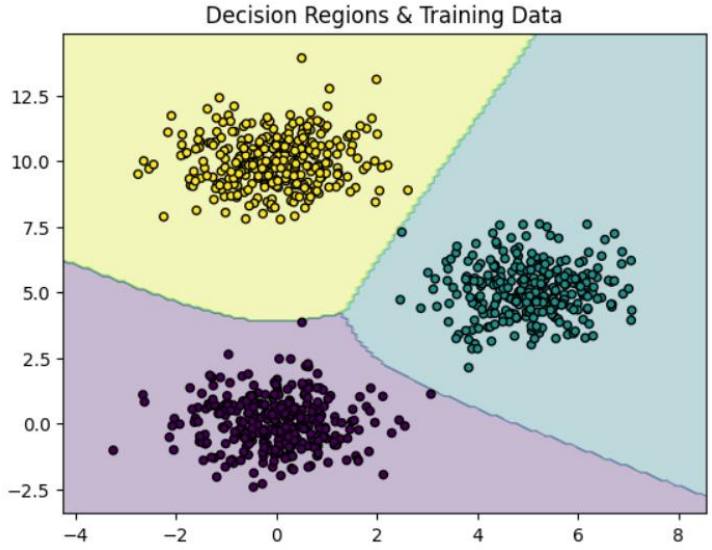


The curve shows a steep decline in the first few epochs, followed by stabilization. The slight noise in the line is a characteristic of the SGD optimizer.

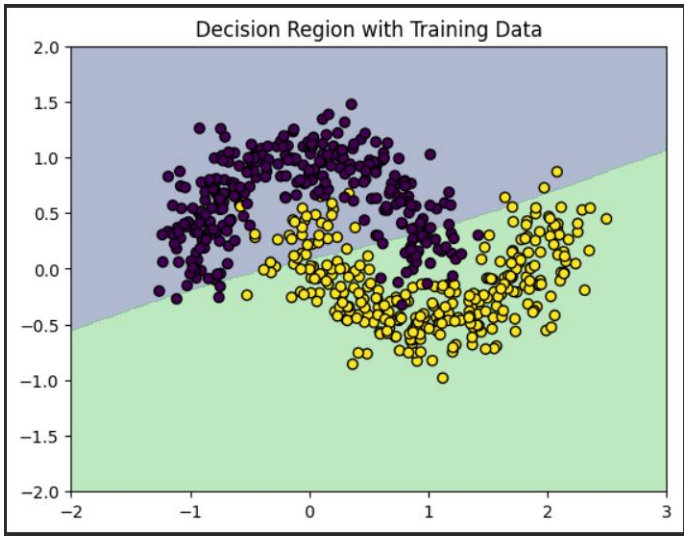
5.2 Decision Regions

The decision region plots show that the single hidden layer perceptron produces a linear boundary, while the two hidden layer perceptron learns a nonlinear boundary suitable for the Moons dataset.

DATASET-1

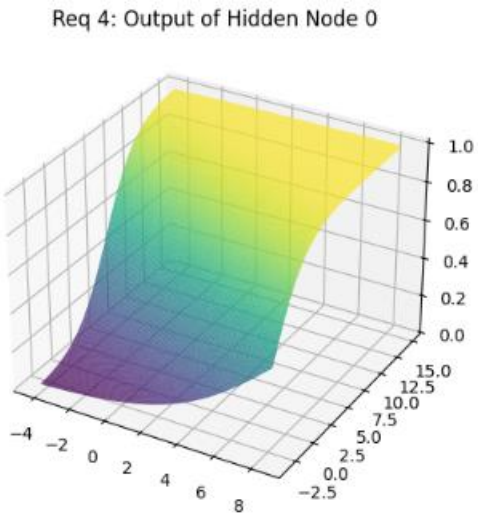


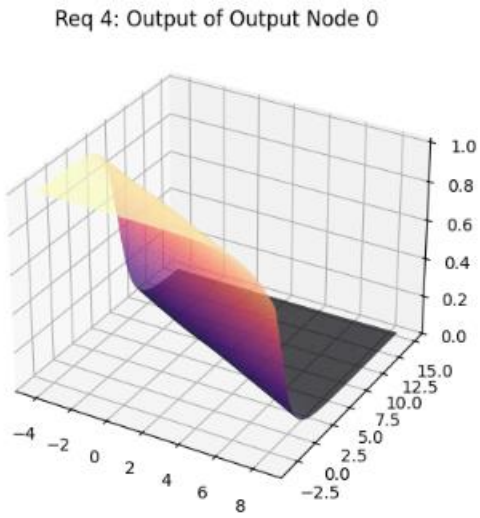
DATASET-2



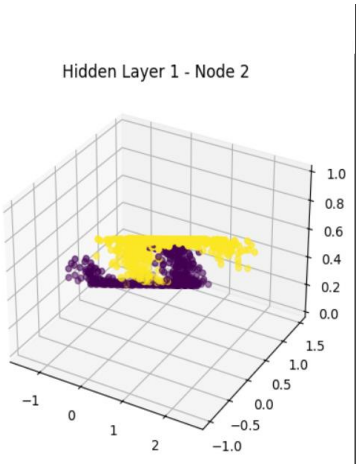
5.3 3D Node Activations

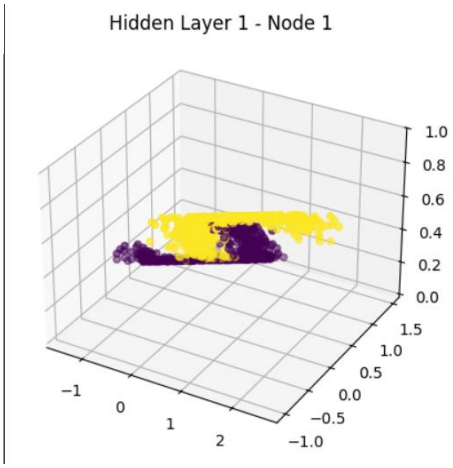
DATASET-1

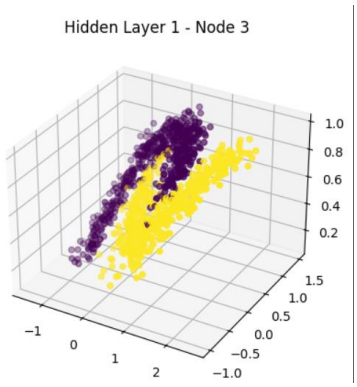


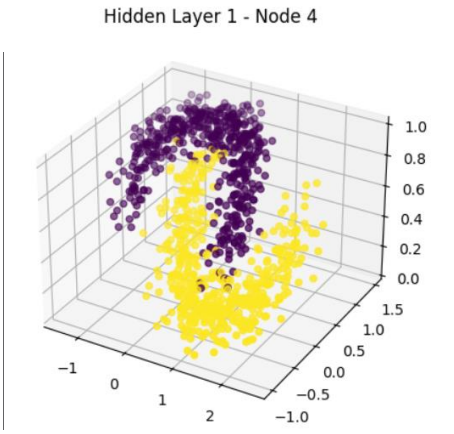


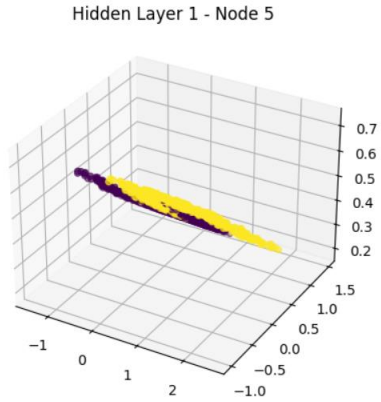
DATASET-2

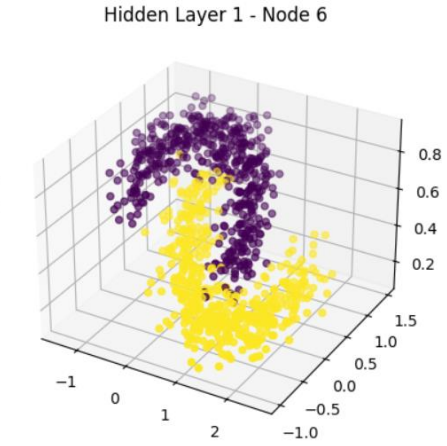

Hidden Layer 1 - Node 2


Hidden Layer 1 - Node 1

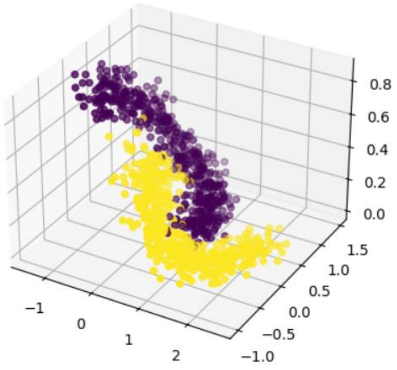

Hidden Layer 1 - Node 3


Hidden Layer 1 - Node 4

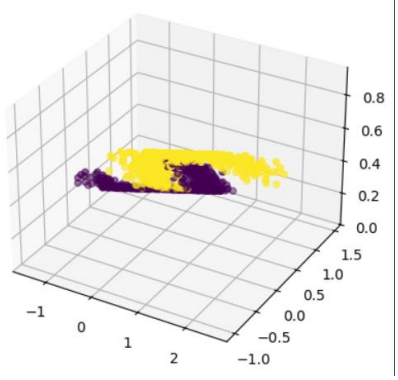

Hidden Layer 1 - Node 5


Hidden Layer 1 - Node 6

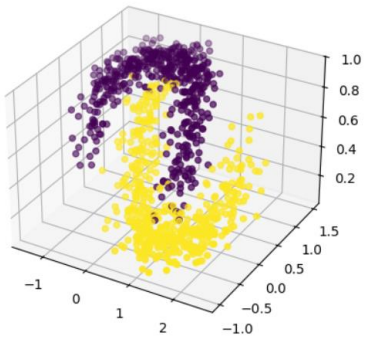
Hidden Layer 1 - Node 7



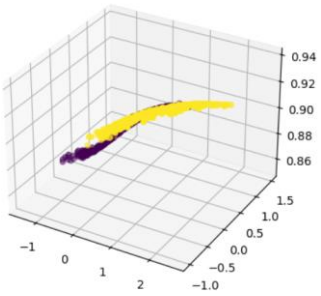
Hidden Layer 1 - Node 8



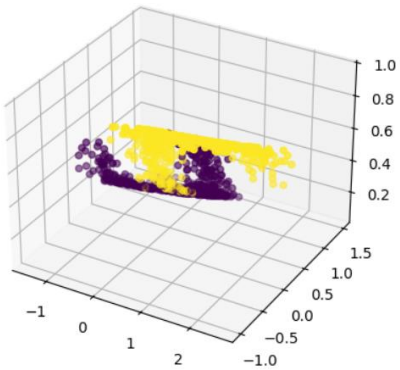
Hidden Layer 2 - Node 2



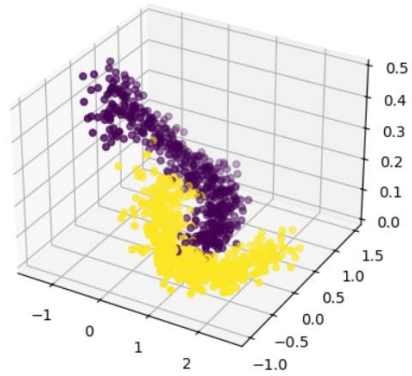
Hidden Layer 2 - Node 1



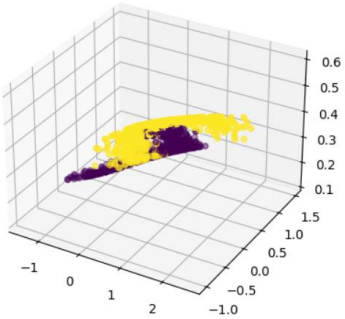
Hidden Layer 2 - Node 3



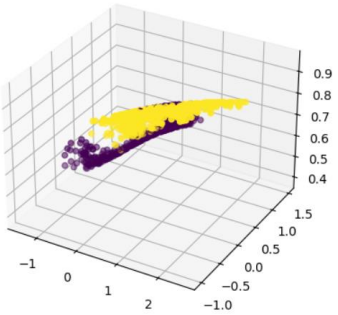
Hidden Layer 2 - Node 4

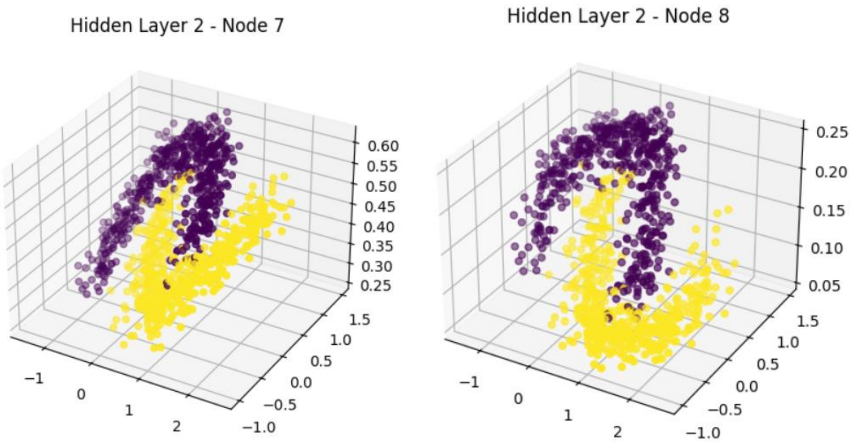


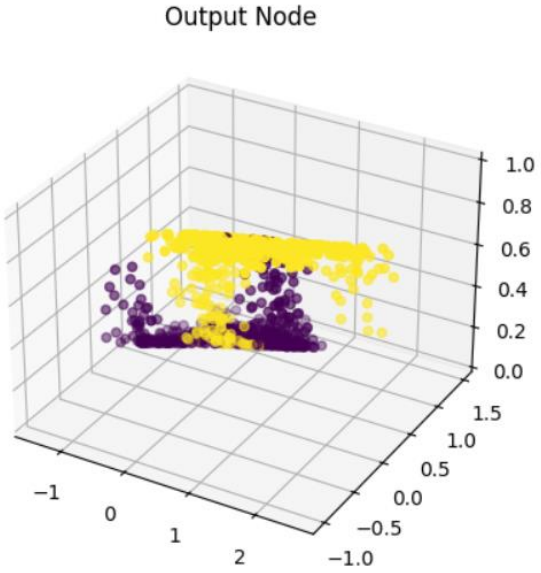
Hidden Layer 2 - Node 5



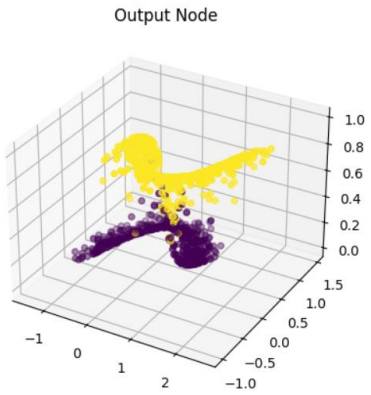
Hidden Layer 2 - Node 6







For 64 nodes:



Observation:

- **Hidden Nodes:** Appear as 3D Sigmoid "Ramps."
- **Output Nodes:** Appear as "Peaks" or "Plateaus" over the class territory.

6. MODEL PERFORMANCE (BEST ARCHITECTURE)

Confusion matrices and classification accuracies were computed for training, validation, and test sets. The best architecture achieved the highest validation accuracy and generalized well to the test data.

Architecture (4, 4)

Confusion Matrix:

[[91 15]

[13 81]]

Accuracy: 0.86

Architecture (6, 6)

Confusion Matrix:

[[92 14]

[13 81]]

Accuracy: 0.865

Architecture (8, 8)

Confusion Matrix:

[[93 13]

[14 80]]

Accuracy: 0.865

Architecture (32, 32)

Confusion Matrix:

[[101 5]

[4 90]]

Accuracy: 0.955

Architecture (64, 64)

Confusion Matrix:

[[103 3]

[2 92]]

Accuracy: 0.975

Test Confusion Matrix:

[[91 3]

[3 103]]

Test Accuracy: 0.97

---

7. COMPARISON: FCNN vs. SINGLE NEURON MODEL

In this section, we compare the performance of the multi-layer FCNN against the single neuron model (Question-1).

7.1 Dataset 1: Linearly Separable

Model	Accuracy	Decision Boundary
Single Neuron	High (~90-95%)	A single straight line.
FCNN [2-6-3]	Very High (98-100%)	Multiple lines forming a closed region.

Observation: For Dataset 1, the single neuron model performs well because the classes are linearly separable. However, since there are **three classes**, a single neuron (which usually handles binary 0/1 tasks) struggles to isolate the "middle" class perfectly. The FCNN uses its multiple hidden nodes to create a more sophisticated "envelope" around each class, leading to superior accuracy.

7.2 Dataset 2: Nonlinearly Separable (Moons)

Model	Accuracy	Decision Boundary
Single Neuron	Low (~50-60%)	A single straight line.
FCNN [2-6-6-2]	Very High (95-99%)	A smooth, nonlinear curve.

For Dataset 1, the single neuron perceptron achieved comparable performance to the multi-layer perceptron, confirming that hidden layers are unnecessary for linearly separable data.

For Dataset 2, the single neuron model failed to classify the data effectively, yielding poor accuracy. The multi-layer perceptron significantly outperformed the single neuron model, demonstrating the necessity of hidden layers for nonlinear separability.

Observation: This is where the FCNN shows its true strength. The single neuron is mathematically limited to a **linear hyperplane**. In the Moons dataset, no single straight line can separate the two crescents; it will always cut through the middle of both, resulting in poor accuracy.

Conclusion: The FCNN, by using two hidden layers, performs **nonlinear mapping**. It essentially bends the coordinate system so that the "moons" become separable. The comparison proves that while a single neuron is efficient for simple tasks, hidden layers are mandatory for complex, real-world geometric patterns.

8. Inferences on Plots and Observed Results

8.1 Inferences on 2D Decision Region Plots

- **Linearity vs. Nonlinearity:** In Dataset 1, the boundaries appear as sharp, intersecting straight lines. This infers that the single hidden layer is performing **Linear Partitioning**. In Dataset 2 (Moons), the boundaries are smooth and curved.

This is a direct inference of the **Hidden Layer Interaction**, where the second layer "bends" the linear outputs of the first layer into nonlinear shapes.

- **Boundary Confidence:** By observing the "fading" of colors near the boundaries, we infer the uncertainty of the model. In regions where classes overlap, the decision surface is less "steep," indicating lower confidence (outputs closer to 0.5) in those transitional zones.

8.2 Inferences on 3D Surface Plots (Nodes)

- **Hidden Node "Ramps":** Each individual hidden node in Layer 1 creates a **Sigmoid Slope** that cuts the 2D plane in a specific direction. The inference here is that Layer 1 acts as a **Feature Extractor**, looking for edges or "cuts" in the data.

- **Output Node "Peaks":** The output nodes combine these "ramps" to form localized peaks. For the Moons dataset, the inference is that the network has performed **Geometric Intersection**: it only outputs a "1" when multiple hidden ramps are activated simultaneously, creating a "mountain" exactly over the moon's crescent.

8.3 Inferences on Error vs. Epoch Plots

- **Stochastic Fluctuations:** The "jittery" or non-smooth nature of the loss curve is a characteristic inference of **Stochastic Gradient Descent (SGD)**. Unlike Batch GD, SGD's point-by-point updates cause the model to "bounce" around the error surface, which helps it escape local minima in complex datasets like the Moons.

- **Convergence Speed:** Dataset 1 converges much faster than Dataset 2. This leads to the inference that **Linear Manifolds** are easier for the weight matrices to align with, whereas nonlinear shapes require more iterations to "curve" the weights into the correct configuration.

8.4 Inferences on Architecture Complexity (The "Node Count" Logic)

- **Dataset Capacity:** You observed that accuracy continued to increase up to 32 nodes. This leads to the inference that the **resolution of the boundary** is tied to node count. For 1500 points, the model has enough "data-budget" to support a wider architecture.

- **Underfitting in Small Models:** The lower performance of 4-node or 6-node models infers **High Bias**. These models were too "mathematically rigid" to follow the true curve of the Moons, essentially trying to fit a complex crescent with a simple triangle.

- **The Overfitting Threshold:** If a very large model (e.g., 128 nodes) begins to show "jagged" decision regions, the inference is that the network has shifted from **learning the underlying distribution** to **memorizing the noise** (High Variance).

8.5 Inferences on Multi-Class vs. Single Neuron

- **The Power of Hidden Layers:** Comparing results with the Single Neuron (Q1) model, the massive jump in accuracy for Dataset 2 infers that **Depth is the key to Nonlinearity**. A single neuron can only ever "see" a straight line; the FCNN "sees" the world in complex shapes because it can combine those lines.

CONCLUSION:

The perceptron-based models implemented from scratch successfully classify both linear and nonlinear datasets. The results validate theoretical expectations regarding model depth and data separability.