Project Summary: Solving the XOR Problem with a Neural Network

The objective of this project was to construct and train a two-layer neural network from scratch using NumPy to solve the **XOR (Exclusive OR) classification problem**. This task serves as a fundamental benchmark in machine learning because the XOR function is **not linearly separable**, meaning its output classes cannot be divided by a single straight line.

The theoretical solution lies in the network's ability to create a non-linear decision boundary. This is achieved through the use of a **hidden layer** combined with a **non-linear activation function**, such as the sigmoid function. The hidden layer acts as a data transformer, mapping the 2D input data into a higher-dimensional feature space where the data points become linearly separable. The final layer can then easily classify these transformed points with a simple linear boundary (a hyperplane).

The network was trained through an iterative process involving two key components:

1. **Forward Propagation:** Input data is passed through the network, layer by layer, to produce a prediction. The process involves a weighted sum of inputs and biases, followed by the application of the sigmoid activation function at each neuron.
2. **Backpropagation:** The error between the prediction and the actual output is calculated. This error is then propagated backward through the network to compute the **gradient** of the loss function with respect to each weight and bias. This gradient indicates the direction and magnitude of the error.
3. **Gradient Descent:** The calculated gradients are used to update the weights and biases, effectively minimizing the loss and improving the network's accuracy over thousands of training epochs.

The project successfully demonstrated that a neural network with a hidden layer can learn to model complex, non-linear relationships, thus solving a problem that is intractable for simpler linear models.