|  |  |
| --- | --- |
| **Ex No: 2**  **Date:** | **Planar Data Classification using a Shallow Neural Network** |

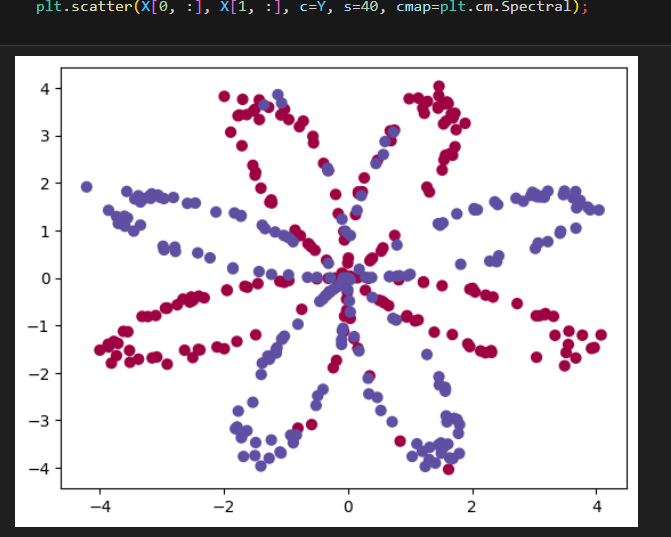
**Objective:**

To build a binary class classification neural network with a single hidden layer which uses activation functions tanh and sigmoid respectively. The Loss function used is the Cross-entropy loss.

**Descriptions:**

**Introduction to Binary Classification**

Binary classification involves categorizing data points into one of two classes. In this context, we will classify a "flower" dataset, which contains data points that belong to one of two classes. The data points are described by features X, and the output is represented by Y, where Y=0 or Y=1.

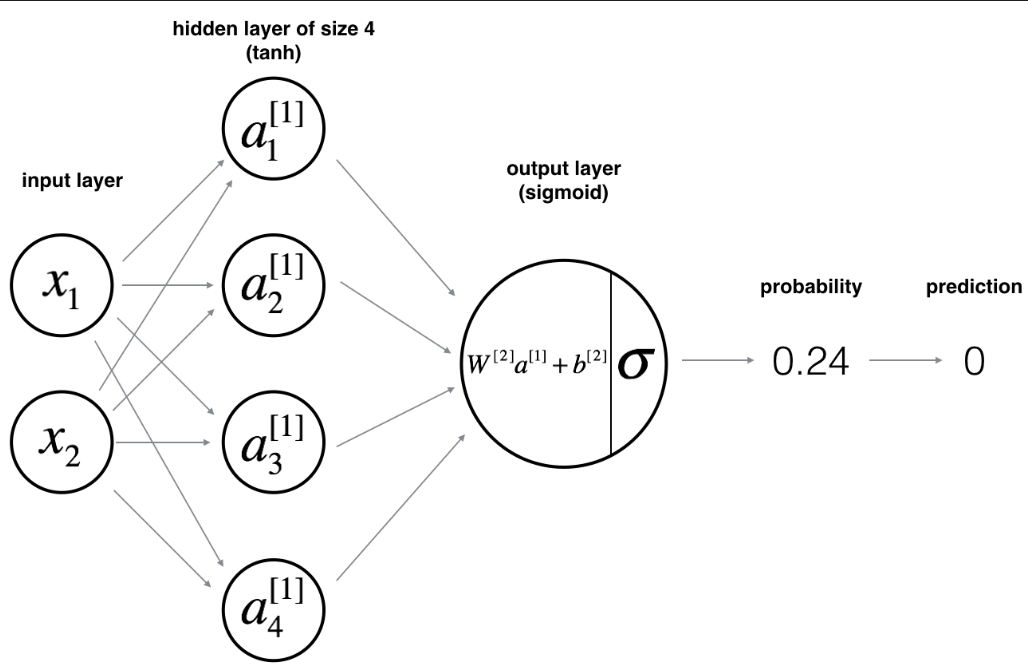


**Neural Network Architecture**

In this experiment, we design a neural network with one hidden layer. This hidden layer allows the network to capture more complex patterns in the data compared to logistic regression, which lacks a hidden layer.

* **Input Layer:** Receives the features XXX.
* **Hidden Layer:** Applies a non-linear activation function (tanh) to capture complex relationships.
* **Output Layer:** Uses the sigmoid function to predict probabilities for each class.

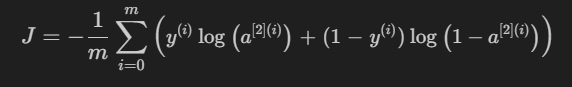
The model parameters (weights and biases) are initialized randomly and updated during training using gradient descent.



**Model Implementation**

The main steps for building this neural network are outlined below:

1. **Model Structure Definition**
   * Number of input features: The dimension of the feature vector X.
   * Number of hidden units: A hyperparameter that can be tuned for better performance.
   * Activation functions: We use tanh for the hidden layer and sigmoid for the output layer.
2. **Parameter Initialization**
   * The weights and biases are initialized randomly.
   * Initialization is crucial for ensuring that the network learns effectively during training.
3. **Forward Propagation**
   * Compute the linear combination of inputs and weights.
   * Apply the activation functions to introduce non-linearity.
   * Calculate the predicted output.
4. **Loss Calculation**
   * The cross-entropy loss function measures the discrepancy between the predicted and actual labels.
   * This loss guides the model on how to adjust its parameters.



1. **Backward Propagation**
   * Calculate the gradients of the loss function with respect to the parameters.
   * Use these gradients to update the parameters in the direction that reduces the loss.
2. **Parameter Update**
   * Gradient descent is employed to minimize the loss function, iteratively improving the model's performance.
3. **Building a Network using “*nn\_model()*”**
   * Integrating all the defined functions, namely forward propagation, cost computation, backward propagation, and parameter updating, into a model, thus adjusting the parameters and making predictions.

**Results and Discussion**

The experiment involves training the neural network on a planar dataset. The model's performance is evaluated by visualizing the decision boundary and comparing the predicted outputs with the actual labels. A well-trained model should accurately classify the data points, demonstrating the effectiveness of using a hidden layer for binary classification.

The developed shallow Neural Network resulted in an accuracy of 91%

**GitHub Link: https://github.com/gowrigalgali/Deep\_Learning-/blob/main/Lab2/Planar\_data%20DISTRI.ipynb**