| **Ex No: 2**  **Date: 14-08-24** | **Planar Data Classification with One Hidden Layer** |
| --- | --- |

**Objective:**

To build a neural network classifier with one hidden layer for classifying planar data into two classes.

**Descriptions:**

Planar data classification is a task where the goal is to classify data points into two distinct classes based on their features. This task is often visualized in a 2D space, making it easier to understand the decision boundary formed by the classifier.

Neural networks can be used for such binary classification tasks. A neural network with one hidden layer is particularly effective as it can capture complex patterns in the data, unlike logistic regression, which might struggle with non-linear boundaries. The network consists of an input layer, one hidden layer with a specified number of neurons, and an output layer with a single neuron to produce the binary classification result.

The training process involves forward propagation to calculate predictions, loss computation to measure the difference between predicted and actual outcomes, and backpropagation to update the weights using Gradient Descent, minimizing the loss over time.

**MODAL:**

**Building the Parts of the Neural Network:**

**The following are the main steps to build a neural network with one hidden layer:**

1. Define the Model Structure:
   * Input layer: Takes in the features from the planar data.
   * Hidden layer: Contains neurons that apply an activation function to capture non-linearities.
   * Output layer: Produces the final binary output (class 0 or 1).
2. Initialize the Model’s Parameters:
   * Initialize weights and biases for both the hidden and output layers.
3. Loop:
   * Forward Propagation:
     + Calculate the output of the hidden layer using the activation function.
     + Calculate the output of the network (prediction) using the activation function applied to the hidden layer output.
   * Loss Computation:
     + Measure the loss using a suitable loss function (e.g., cross-entropy).
   * Backward Propagation:
     + Compute the gradients of the loss with respect to the network’s parameters.
   * Parameter Update:
     + Update the weights and biases using Gradient Descent to minimize the loss.

USN NUMBER: 1RVU22BSC110

NAME:YASHWANT RAJ

**GitHub Link:**

[**https://github.com/Yashr22/Lab-2---Planar.git**](https://github.com/Yashr22/Lab-2---Planar.git)