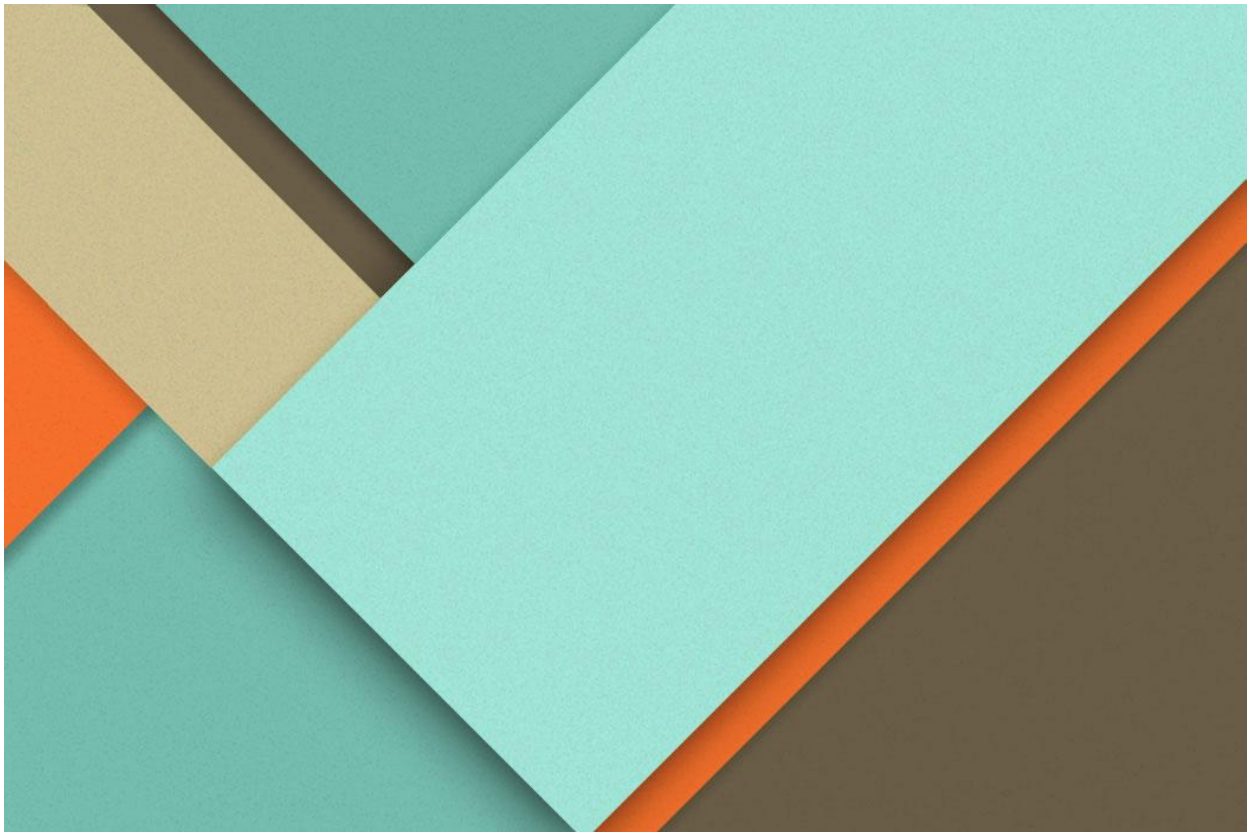

AI-LAB

ROLL NO. BSDSF21M016

13-12-2023

TOPIC : PERCEPTRONS

QUESTION 1

Search the internet for different activation functions(at least 4) that are used in Single/Multi Layer Perceptrons. Enlist them and also mention what they are mostly used for in a neural network/multilayer perceptron.

ANSWER

It is used to determine the output of neural network like yes or no. It maps the resulting values in between 0 to 1 or -1 to 1 etc. (depending upon the function).

The Activation Functions can be basically divided into 2 types-

- Linear Activation Function
- Non-linear Activation Functions

Activation functions are :

- Sigmoid or Logistic Activation Function
- Tanh or hyperbolic tangent Activation Function
- ReLU (Rectified Linear Unit) Activation Function
- Leaky ReLU

Sigmoid or Logistic Activation Function

Range : (0,1)

Formula :

$$S(x) = \frac{1}{1 + e^{-x}}$$

Used for : Historically used in the output layer for binary classification problems, but less common in hidden layers due to the vanishing gradient problem. Also, used in the output layer of binary classification problems, where the goal is to produce a probability output.

Hyperbolic Tangent Activation Function

Range : (-1,1)

Formula :

F(x) = (e^x – e^-x) / (e^x + e^-x)

Used for : Similar to the sigmoid function but with a range from -1 to 1. It is often used in hidden layers of neural networks.

ReLU (Rectified Linear Unit) Activation Function

Range : [0,+∞)

Formula :

f(x) = max(0,x)

Used for : One of the most popular choices for hidden layers due to its simplicity and effectiveness. It helps with the vanishing gradient problem and accelerates convergence.

Leaky Rectified Linear Unit Activation Function

Range : (-∞,+∞)

Formula :

f(x)=max(0.01*x , x)

Used for : Addresses the "dying ReLU" problem by allowing a small, non-zero gradient for negative inputs. It is a variation of the ReLU and aims to improve learning performance.A variant of ReLU that allows a small, non-zero gradient when the input is negative, helping to mitigate the dying ReLU problem.

QUESTION 2

Implement different perceptrons in Python using the activation functions that you have found in the above question. Use the same data to train each of these and then check which perceptron gives you the highest accuracy.

ANSWER

Sigmoid (Logistic) Activation Function shows highest accuracy which is **0.928.**

QUESTION 3

Research and describe in your own words the following topics

ANSWER

1. **Epochs**
   In the context of machine learning, an epoch refers to a complete pass through the entire training dataset during the training phase of a model. During each epoch, the model's parameters (weights and biases) are adjusted in an attempt to minimize the error or loss function. Multiple epochs are often required to train a model effectively, allowing it to learn patterns in the data.

2. Forward Propagation

Forward propagation is the process of processing input data through a neural network layer by layer in order to generate predictions or outputs. Each layer of the network in this procedure conducts a linear transformation (weighted sum of inputs) followed by a non-linear activation function. The output of one layer becomes the input for the next layer, and so on until the network's output is produced by the last layer. Forward propagation is required for neural network prediction during both the training and testing phases.

3. Backward Propagation

Backward propagation, or backpropagation, is the process of updating the model's parameters based on the computed error in the forward propagation phase. It involves calculating the gradient of the loss function with respect to the model parameters using the chain rule of calculus. These gradients are then used to adjust the parameters in the direction that minimizes the error, improving the model's performance over time.

4. Bias in a Perceptron

Bias is an additional parameter that is added to the weighted sum of input characteristics in the setting of a perceptron. It enables the model to take into account instances in which all input features are zero. The bias term essentially alters the decision boundary, allowing the model to more correctly fit the data. It determines the offset or y-intercept of the decision boundary in a geometric sense. Including a bias component in a perceptron increases its ability to simulate a broader set of data interactions.

5. Implementation of XOR Using a Single Layer Perceptron

Because of its linear decision boundary, a singlelayer perceptron (SLP) cannot properly learn the XOR (exclusive OR) function. The binary operation XOR produces true only when the number of true inputs is odd. Because XOR is not linearly separable, it cannot be correctly modeled by a single-layer perceptron, which generates only linear decision boundaries. A multi-layer perceptron (MLP) or neural network with at least one hidden layer is required to achieve this. The extra layer enables the network to learn complicated, non-linear relationships, allowing for successful modeling of XOR and other non-linear functions.