Response

Write python functions to perform the following tasks in a neural network .

1. Feed forward Algorithm.

**def feed\_forward():**

2. Sigmoid Activation Function.

**def sig\_activation():**

3. Output Function that calls Functions 1 and 2 .

**def call\_output():**

Paste your code in the answer box below

**Note :**

**In each of the above functions accept necessary parameters. Use the given Function names while writing the functions.**

**Do not import any advanced packages for this task. Use only primary packages.**

def feed\_forward(self, inActs): """ Propagate input vector forward to calculate outputs. Args: inActs (list<float>): the input to the NN (an example) Returns: list<list<float/int>> A list of lists. The first list is the input list, and the others are lists of the output values 0f all perceptrons in each layer. """ returnList = [] returnList.append(inActs) currentInput = inActs for currLayer in self.layers: pList = [] for perceptor in currLayer: pList.append(perceptor.sig\_activation(currentInput)) currentInput = pList #updating the currentInput List after each layer returnList.append(pList) return returnList import math def sig\_activation(self, inActs): """ Inputs: inActs (list<float/int>): input values, not including bias Returns: int, The rounded value of the sigmoid of the weighted input """ inActs.insert(0, 1.0) #this is the bias input added to the front of activationList weightedSum = self.getWeightedSum(inActs) #helper method to compute overall weightedSum del inActs[0] #getting rid of bias from the list return round(self.sigmoid(weightedSum)) #calling sigmoid fucntion on it as helper method def call\_output(): output = self.feed\_forward(example[0]) # sig\_activation function has been called in feed\_forward function