**College of Computer Science and Engineering**

**Department of Computer Science and Artificial Intelligence**

**CCAI-321: Artificial Neural Networks**

**Lab#6 Implementing Multilayer Networks using Python**

# Student ID: 1948080

# Marks Obtained = / 15 PLO = S1 - AI

**Marks:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Questions** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **Total** |
| **Allocated** | **2** | **1** | **4** | **2** | **2** | **2** | **2** | **15** |
| **Obtained** | **2** | **1** | **3** | **2** | **2** | **2** | **2** | **14** |
|  |  |  |  |  |  |  |  |  |
| **Allocated** |  |  |  |  |  |  |  |  |
| **Marks** |  |  |  |  |  |  |  |  |

**Weighted Marks:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Allocated** |  |  |  |
| **Obtained** |  |  |  |

(Q3 Report the values you got by hand and from your python code in the table are a2-(by hand) values are missing)

# Objectives

* Implement a 2-layer network in python
* Experiment with the response of the network when setting different parameters in python

# Lab Tool(s)

[Download Python | Python.org](https://www.python.org/downloads/)

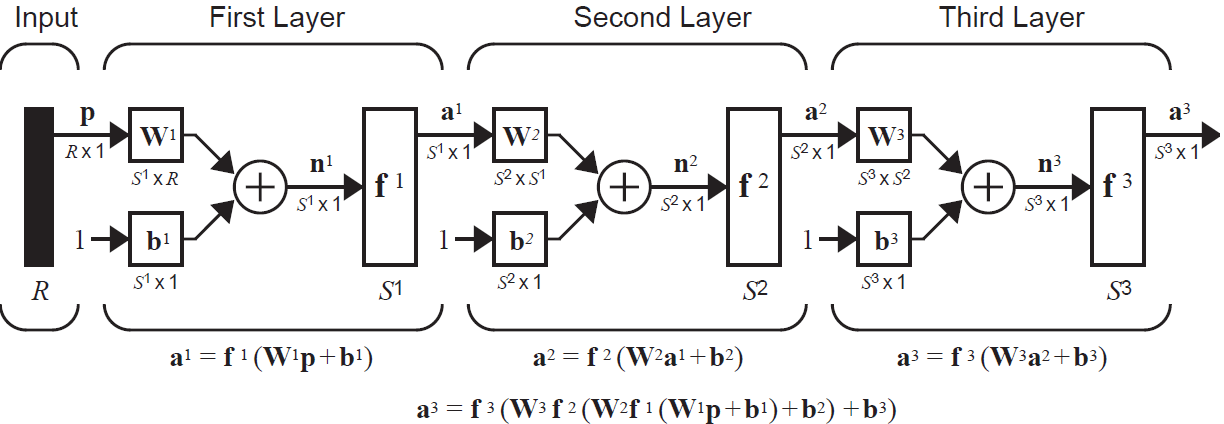
[Anaconda | Individual Edition](https://www.anaconda.com/products/individual)

# Lab Deliverables

Submit a pdf document on Blackboard containing your solution to the lab assessment at the end of this document.

# What is a Multilayer Network?

A network with more than one layer is called a multilayer network.

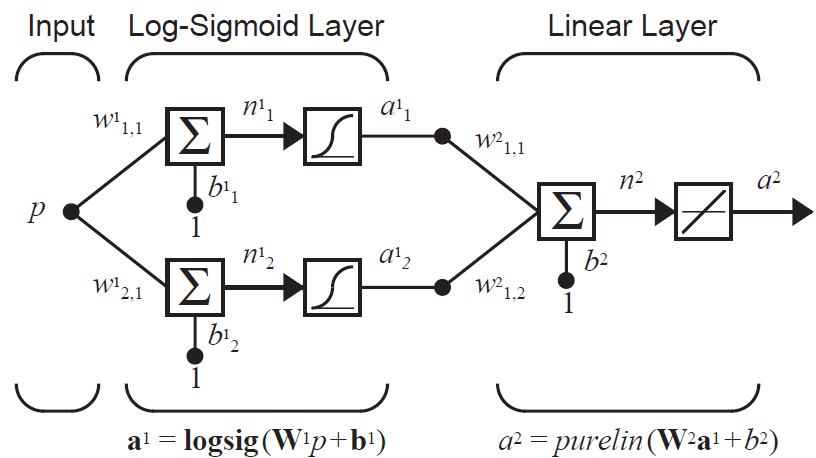


In a multilayer network, the first layer has R inputs and S1 neurons. Therefore, W1 is a (S1 x R), because we have R inputs for each neuron. The bias b1 is a (S1 x 1) vector, because we have 1 bias per neuron. n1 and a1 are also (S1 x 1) vectors, because we have 1 output for each neuron.

The second layer has S1 inputs and S2 neurons. Therefore, W2 is a (S2 x S1), because we have S1 inputs for each neuron. The bias b2 is a (S2 x 1) vector, because we have 1 bias per neuron. n2 and a2 are also (S2 x 1) vectors, because we have 1 output for each neuron.

The third layer has X inputs and X neurons. Therefore, W3 is a (X), because we have X inputs for each neuron. The bias b3 is a (X) vector, because we have 1 bias per neuron. n3 and a3 are also (X) vectors, because we have 1 output for each neuron.

In this lab, we will implement a two layer network, depicted in the figure below. We will start by implementing each layer separately then visualizing the network output when we change the values of W and b.

****

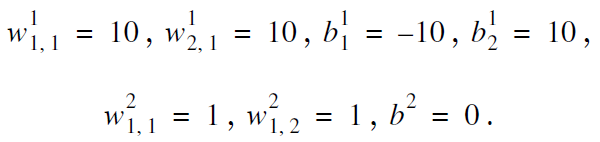
**Q1. Implement the first layer and name it “logsiglayer”. The function takes the following parameters: W**1**,** p **and b**1**. W is the weight matrix, p is the input and b is the bias. The layer computes the output, given the input, using the following formula. [2 marks]**

**a**1 **= logsig(W**1p**+b**1**)**

**Q2. Implement the second layer and name it “linearlayer”. The function takes the following parameters: W**2**, a**1 **and b**2**. W is the weight matrix, a is the input and b is the bias. The layer computes the output, given the input, using the following formula. [1 marks]**

**a**2 **= purelin(W**2 **a**1**+b**2**)**

**Q3. Let the network has the following values.**

****

**Plot the output of the network. To do so, follow the steps below:**

1. **Set the weight and bias values as given above**
2. **Let p (the input) be between -2 and 2.**
3. **Pass the input p to layer 1, to get a1 (the output of the first layer).**
4. **Pass a1 to layer 2, to get a2 (the output of the second layer).**
5. **Plot p vs a2. [2 marks]**

**2. To do so, follow the steps below:**

1. **Use the weight and bias values as given previously**
2. **For each value of p, compute by hand the following:** 
   1. **a**1 **= logsig(W**1p**+b**1**)**
   2. **a**2 **= purelin(W**2 **a**1**+b**2**)**
3. **Report the values you got by hand and from your python code in the table below. [2 marks]**

|  |  |  |
| --- | --- | --- |
| **p** | **a2 (from code)** | **a2 (by hand)** |
| **-2** | **0** | **?** |
| **-1** | **0.5** | **?** |
| **0** | **1** | **?** |
| **1** | **1.5** | **?** |
| **2** | **2** | **?** |

**Q4. Test the effect of changing , and plot the results. To do so, follow the steps below:**

1. **Set the weight and bias values as given previously**
2. **Let be between**
3. **For each value of , repeat:** 
   1. **Let p (the input) be between -2 and 2.**
   2. **Pass the input p to layer 1, to get a1 (the output of the first layer).**
   3. **Pass a1 to layer 2, to get a2 (the output of the second layer).**
   4. **Plot p vs a2. [2 marks]**
4. **What changes does have on the network output?**

**Q5. Repeat the same for . What changes does have on the network output? [2 marks]**

**Q6. Test the effect of changing , and plot the results. To do so, follow the steps below:**

1. **Set the weight and bias values as given previously**
2. **Let be between**
3. **For each value of , repeat:** 
   1. **Let p (the input) be between -2 and 2.**
   2. **Pass the input p to layer 1, to get a1 (the output of the first layer).**
   3. **Pass a1 to layer 2, to get a2 (the output of the second layer).**
   4. **Plot p vs a2. [2 marks]**
4. **What changes does have on the network output?**

**Q7. Repeat the same for . What changes does have on the network output? [2 marks]**