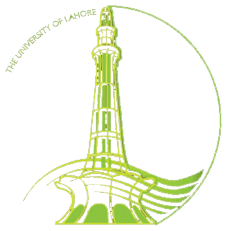

THE UNIVERSITY OF LAHORE
ISLAMABAD CAMPUS

Artificial Intelligence
(CS13217)

Lab Report

Name:	Sameea Naeem
Registration #:	CSU-XS16-139
Lab Report #:	09
Submitted To:	Sir Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 9
Implementation of single layer perceptrons.

Objective
Implementation of single layer perceptrons.

Software Tool
1. Python
2. Sublime text
3. Windows 10
4. Latex

1 Theory

A single-layer perceptron network consists of one or more artificial neurons in parallel. The neurons may be of the same type we've seen in the Artificial Neuron Applet. Each neuron in the layer provides one network output, and is usually connected to all of the external (or environmental) inputs.
1. The applet in this tutorial is an example of a single-neuron, single-layer perceptron network, with just two inputs.
2. The perceptron learning rule, which we study next, provides a simple algorithm for training a perceptron neural network. However, as we will see, single-layer perceptron networks cannot learn everything: they are not computationally complete. As mentioned in the introduction, two-input networks cannot approximate the XOR (or XNOR) functions. Of the (22)n or 16 possible functions, a two-input perceptron can only perform 14 functions. As the number of inputs, n, increases, the proportion of functions that can be computed decreases rapidly.

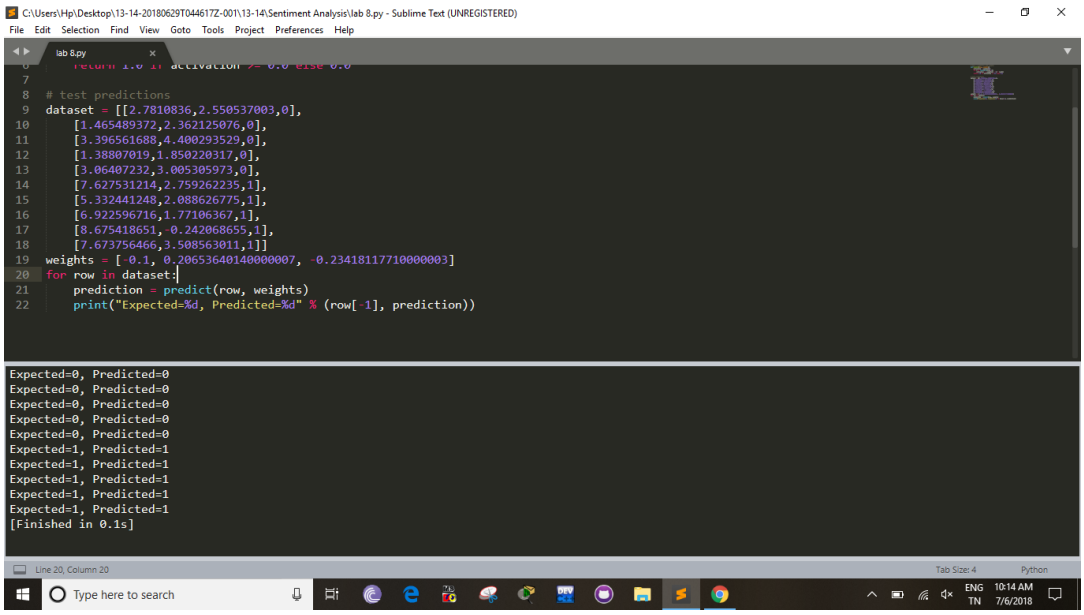

Figure 1: output 1

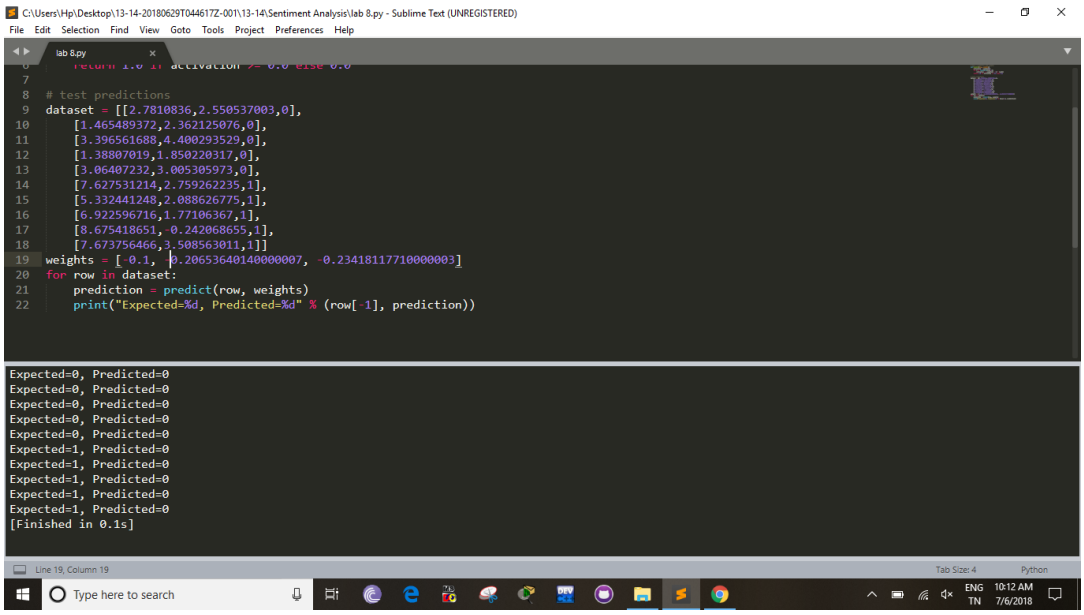

Figure 2: output 2

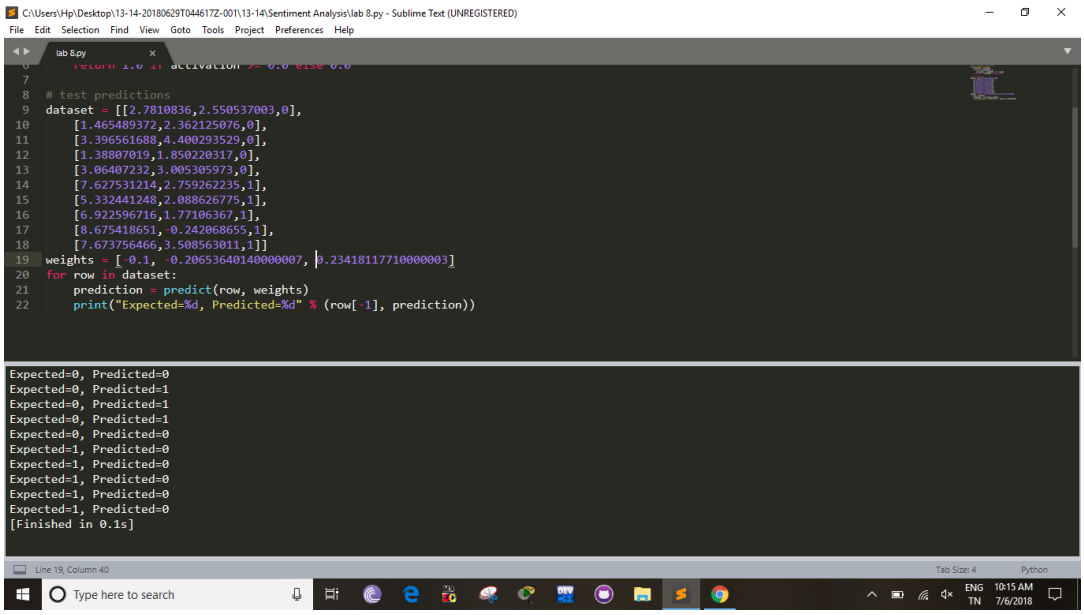

Figure 3: output 3

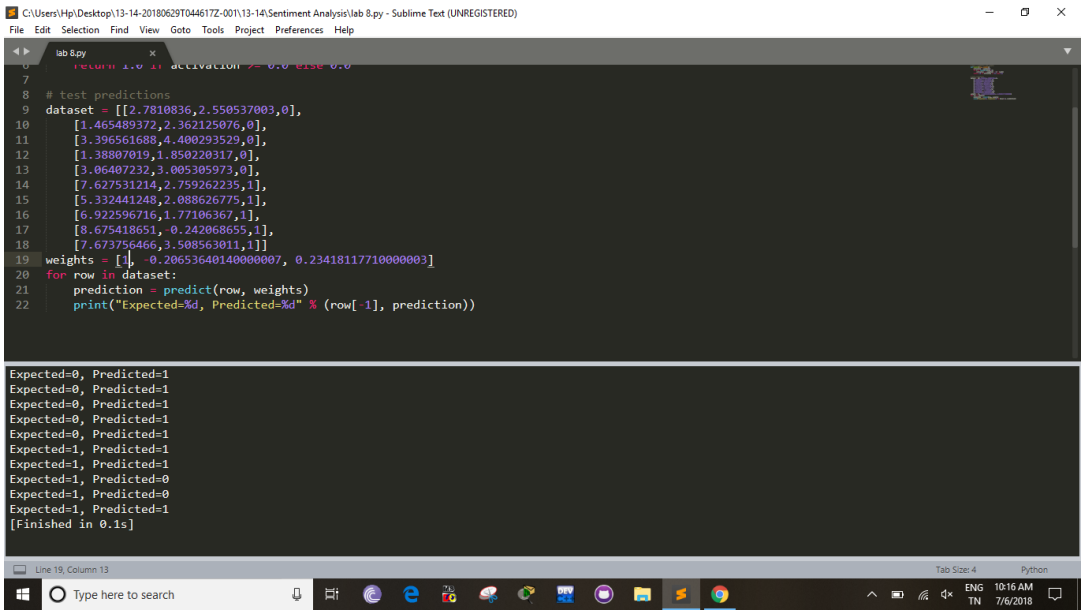

Figure 4: output 4

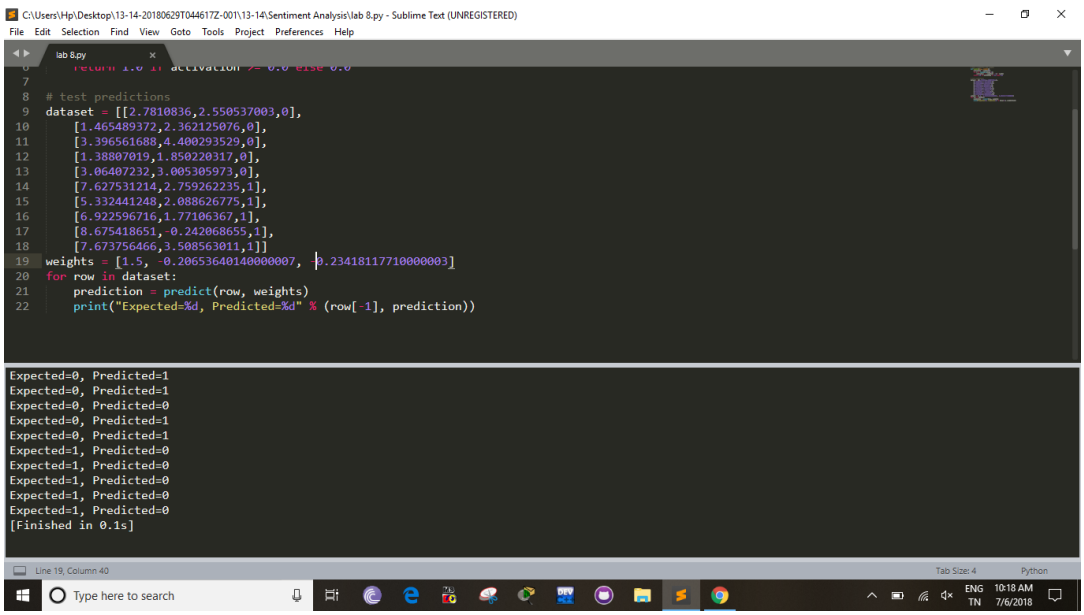

Figure 5: output 5

2 Procedure: Task 1

2.1 Procedure: Task 2

```
# Make a prediction with weights
def predict(row, weights):
        activation = weights[0]
        for i in range(len(row)-1):
                activation += weights[i + 1] * row[i]
        return 1.0 if activation >= 0.0 else 0.0

# test predictions
dataset = [[2.7810836,2.550537003,0],
        [1.465489372,2.362125076,0],
        [3.396561688,4.400293529,0],
        [1.38807019,1.850220317,0],
        [3.06407232,3.005305973,0],
        [7.627531214,2.759262235,1],
        [5.332441248,2.088626775,1],
```

```
        [6.922596716,1.77106367,1],
        [8.675418651,-0.242068655,1],
        [7.673756466,3.508563011,1]]
weights = [1.7, -0.20653640140000007, -0.23418117710000003]
for row in dataset:
        prediction = predict(row, weights)
        print("Expected=%d, Predicted=%d" % (row[-1], prediction))
```

3 Conclusion

Outputs shows different single layer perceptron results.