ASSIGNMENT NO. 2

PARADIGMS OF

Artificial Intelligence

NAME: SHAHERYAR ALAM KHAN

ROLL: 364823

SUBMITTED TO: Brig Dr. Nasir Rashid

DATE: 09-NOV-2022

Simple Neuron in Python

Outlook & Features of Simple Neuron:

- The program is a simple perceptron neural network that takes in input from the user to choose the activation function, the number of neural inputs, the threshold, and the associated weights. The output is in 0 or 1, for example, the AND and OR gates.
- All things are taken as input from the user, not predefined. The program first asks the user to select an activation function from the given list. Then, the user is asked to enter the inputs for the perceptron. The inputs are then multiplied by the respective weights and passed through the chosen activation function. The output is then printed to the console.
- The threshold function, signum function, hyperbolic tangent function, and sigmoid function are the activation functions used in this software.
- To translate the input values into a range between 0 and 1, the sigmoid function is utilized. Using the hyperbolic tangent function, the input values are transformed into a range between -1 and 1. simply returns the supplied value's sign using the signum function (positive if the input is positive, negative if the input is negative, and 0 if the input is 0). When the input value exceeds or equals 0.5, the threshold function returns 1, otherwise it returns 0.
- The user has the flexibility to choose any combination of activation functions and inputs. The program is designed to work with any number of inputs and any number of activation functions. This makes it a very versatile tool that can be used for a variety of applications.

Python Code:

```
import math

def sigmoid(x):
  return 1 / (1 + math.exp(-x))

def tanh(x):
  return (2 / (1 + math.exp(-2*x))) - 1

def signum(x):
  if x>0:
    return 1
  elif x<0:
    return -1
  else:
    return 0

def threshold(x):
  if x>=0.5:
    return 1
  else:
    return 0

def main():
```

```
print("\nPlease select an activation function: ")
print("1. Sigmoid")
print("2. Tanh")
print("3. Signum")
print("4. Threshold")

act_func = int(input("\nYour selection: "))

print("\nPlease enter the inputs: ")
x1 = float(input("x1: "))
x2 = float(input("x2: "))
w1 = float(input("w1: "))
w2 = float(input("w2: "))
threshold = float(input("threshold: "))

if act_func == 1:
  threshold = sigmoid(x1*w1 + x2*w2)

if act_func == 2:
  threshold = tanh(x1*w1 + x2*w2)

if act_func == 3:
  threshold = signum(x1*w1 + x2*w2)

if act_func == 4:
  threshold = threshold(x1*w1 + x2*w2)
```

```
  if threshold >= 0.5:
    print("Output: 1")
  else:
    print("Output: 0")

main()
```

OUTPUT:

Output using different activation functions & different values of neural inputs, weights and threshold

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold
Your selection: 2
Please enter the inputs:
x1: 2
x2: 5
w1: 6
w2: 2
threshold: 4
Output: 1
>
```

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold

Your selection: 1
Please enter the inputs:
x1: 9
x2: 10
w1: 7
w2: 13
threshold: 10
Output: 1
>
```

```
Please select an activation function: 1. Sigmoid
2. Tanh
3. Signum
4. Threshold
Your selection: 2
Please enter the inputs:
x1: 1
x2: 0
w1: 0
w2: 0
threshold: 1
Output: 0
>
```

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold

Your selection: 3
Please enter the inputs:
x1: 1
x2: 0
w1: 0
w2: 0
threshold: 1
Output: 0
>
```

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold
Your selection: 3
Please enter the inputs:
x1: 0
x2: 1
w1: 1
w2: 0
threshold: 1
Output: 0
>
```

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold

Your selection: 3
Please enter the inputs:
x1: 0
x2: 1
w1: 1
w2: 1
threshold: 0
Output: 1
>
```

```
Please select an activation function:
1. Sigmoid
2. Tanh
3. Signum
4. Threshold
Your selection: 1
Please enter the inputs:
x1: 0
x2: 0
w1: 0
w2: 0
threshold: 0
Output: 1
>
```

Explanation & Conclusion:

- One application of this program is to create a simple binary classifier. This can be done by using the signum function as the activation function and two input values. The first input value can

be the value to be classified and the second input value can be a threshold. If the first input is greater than the threshold, the output will be 1, indicating that the value is in the first class. If the first input is less than the threshold, the output will be 0, indicating that the value is in the second class.

- Another application of this program is to create a simple AND gate. This can be done by using the threshold function as the activation function and two input values. The first input value can be 0 or 1 and the second input value can be 0 or 1. If both input values are 1, the output will be 1. Otherwise, the output will be 0.
- Similarly, a simple OR gate can be created by using the threshold function as the activation function and two input values. The first input value can be 0 or 1 and the second input value can be 0 or 1. If either input value is 1, the output will be 1. Otherwise, the output will be 0.
- This program can be extended to create more complex neural networks by adding more layers and more neurons. This would allow the program to solve more complex problems.
- In conclusion, this program is a simple perceptron neural network that can be used for a variety of applications