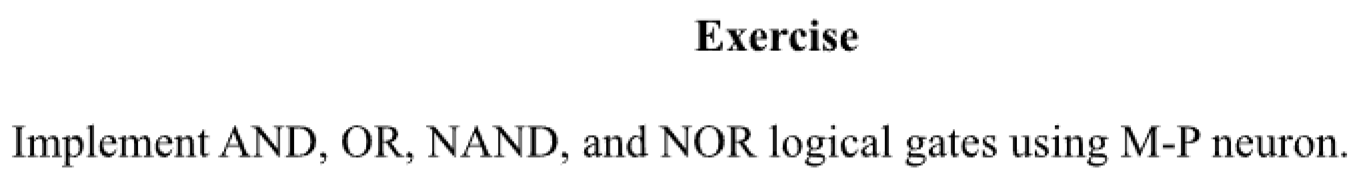
Assignment 2

Name: : Chukka Chanakya Devendra

Roll Number: : AP21110011577

Submission Date: : 12-08-2024

Question



Exercise

Implement AND, OR, NAND, and NOR logical gates using M-P neuron.

In [ ]:
```
import pandas as pd
import numpy as np
```

In [ ]:
```
class Neuron(object):
    def __init__(self,w=[1,1],t=1) -> None:
        self.weights=np.array(w)
        self.threshold=t

    def output(self,val):
        x=val
        sum=np.inner(self.weights,x)
        if sum>=self.threshold:
            return 1
        else:
            return 0

    def truthtable(self,in_s,in_labels,out_label):
        table=pd.DataFrame(in_s,columns=in_labels)
        out_signal=[]
        for r in in_s:

            sig=self.output(val=r)
            out_signal.append(sig)
        table[out_label]=pd.Series(out_signal)
        return table
```

AND Gate Truth Table and Implementation using MCP Neuron

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

In [ ]:
```
in_signals = np.array([[0,0], [0,1], [1,0], [1,1]])
in_labels = ['x1','x2']
out_label = 'y'

AND = Neuron(w = [1,1], t = 2)
AND_table = AND.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Result Main Table ")
print(AND_table)

inp=input("Enter the input to check separated by spaces : ")
in_signals = np.array([list(map(int, inp.split()))])
AND_table = AND.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Result for User Input : ")
print(AND_table)
```

```
Result Main Table
   x1  x2  y
0   0   0  0
1   0   1  0
2   1   0  0
3   1   1  1
Result for User Input :
   x1  x2  y
0   1   1  1
```

OR Gate Truth Table and Implementation using MCP Neuron

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

In [ ]:

```
## For main table :
in_signals = np.array([[0,0], [0,1], [1,0], [1,1]])


OR = Neuron(w = [1,1], t = 1)

OR_table = OR.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Output for Input Table : ")
print(OR_table)

# For User Input :
inp=input("Enter x1 and x2 separated by spaces : ")
in_signals = np.array([list(map(int, inp.split()))])
in_labels = ['x1','x2']
out_label = 'y'


OR_table = OR.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Output for User Input :")
print(OR_table)
```

```
Output for Input Table :
   x1  x2  y
0   0   0  0
1   0   1  1
2   1   0  1
3   1   1  1
Output for User Input :
   x1  x2  y
0   1   1  1
```

NAND Gate Truth Table and Implementation using MCP Neuron

x1	x2	y
0	0	1
0	1	1
1	0	1
1	1	0

In [ ]:

```
# Main Table Results
in_signals = np.array([[0,0], [0,1], [1,0], [1,1]])
NAND = Neuron(w = [-1,-1], t = -1)
NAND_table = NAND.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Truth Table Results : ")
print(NAND_table)

# For User Input :
inp=input("Enter x1 and x2 separated by spaces : ")
in_signals = np.array([list(map(int, inp.split()))])
NAND_table = NAND.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("User Input Results : ")
print(NAND_table)
```

```
Truth Table Results :
   x1  x2  y
0   0   0  1
1   0   1  1
2   1   0  1
3   1   1  0
User Input Results :
   x1  x2  y
0   1   1  0
```

NOR Gate Truth Table and Implementation using MCP Neuron

x1	x2	y
0	0	1
0	1	0
1	0	0
1	1	0

In [ ]:

```
in_signals = np.array([[0,0], [0,1], [1,0], [1,1]])
NOR = Neuron(w = [-1,-1], t = 0)
NOR_table = NOR.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("Truth Table Results : ")
print(NOR_table)

inp=input("Enter x1 and x2 separated by spaces : ")
in_signals = np.array([list(map(int, inp.split()))])
NOR_table = NOR.truthtable(in_signals, in_labels = in_labels, out_label = out_label)
print("User Input Results : ")
print(NOR_table)
```

```
Truth Table Results :
   x1  x2  y
0   0   0  1
1   0   1  0
2   1   0  0
3   1   1  0
User Input Results :
   x1  x2  y
0   1   1  0
```