module3

September 25, 2022

0.1 Logic Gates Implementation for AND, NAND, OR, NOR operations

X1	X2	AND	NAND	OR	NOR
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	0	1	0

[1]:
```
#import libraries
import numpy as np
```

[2]:
```
#Create Truth table
xs = [(0,0), (0,1), (1,0), (1,1)]
print(xs)
```

```
[(0, 0), (0, 1), (1, 0), (1, 1)]
```

[3]:
```
#Function to perform AND gate and bias denoted as b
def do_and(x1,x2):
    x = np.array([x1,x2])
    w = np.array([0.5,0.5])
    b = -0.8
    y = np.sum(x*w) + b
    return 1 if y>0 else 0
```

[4]:
```
# calculate y for AND operation of all values in truth table
for x in xs:
    res = do_and(x[0], x[1])
    print("AND operation with {} and {}, we will have {}".format(x[0],x[1],res))
```

```
AND operation with 0 and 0, we will have 0
AND operation with 0 and 1, we will have 0
AND operation with 1 and 0, we will have 0
AND operation with 1 and 1, we will have 1
```

[5]:
```
#Function to perform Nand gate and bias denoted as b

def do_nand(x1,x2):
```

```
    x = np.array([x1,x2])
    w = np.array([0.5,0.5])
    b = -0.8
    y = np.sum(x*w) + b
    return 0 if y >= 0 else 1
```

[6]:
```
# calculate y for NAND operation of all values in truth table
for x in xs:
    ans = do_nand(x[0], x[1])
    print("NAND operation with {} and {}, we will have {}".
 ↪format(x[0],x[1],ans))
```

```
NAND operation with 0 and 0, we will have 1
NAND operation with 0 and 1, we will have 1
NAND operation with 1 and 0, we will have 1
NAND operation with 1 and 1, we will have 0
```

[7]:
```
#Function to perform OR gate and bias denoted as b
def do_or(x1,x2):
    x = np.array([x1,x2])
    w = np.array([0.5,0.5])
    b = -0.2
    y = np.sum(x*w) + b
    return 1 if y > 0 else 0
```

[8]:
```
# calculate y for OR operation of all values in truth table

for x in xs:
    result = do_or(x[0], x[1])
    print("OR operation with {} and {}, we will have {}".
 ↪format(x[0],x[1],result))
```

```
OR operation with 0 and 0, we will have 0
OR operation with 0 and 1, we will have 1
OR operation with 1 and 0, we will have 1
OR operation with 1 and 1, we will have 1
```

[9]:
```
#Function to perform NOR gate and bias denoted as b
def do_nor(x1,x2):
    x = np.array([x1,x2])
    w = np.array([0.5,0.5])
    b = -0.2
    y = np.sum(x*w) + b
    return 0 if y > 0 else 1
```

[10]:
```
# calculate y for NOR operation of all values in truth table
```

```
for x in xs:
    results = do_nor(x[0], x[1])
    print("NOR operation with {} and {}, we will have {}".
 ↪format(x[0],x[1],results))
```

```
NOR operation with 0 and 0, we will have 1
NOR operation with 0 and 1, we will have 0
NOR operation with 1 and 0, we will have 0
NOR operation with 1 and 1, we will have 0
```