```
class Neuron:
    theta = 0
    w1 = 0
    w2 = 0
    possible_w1_vals = [-1, 1]
    possible_w2_vals = [-1, 1]
    possible_thresh_vals = [-2, -1, 0, 1, 2]
    def __init__(self, input_matrix):
        '''
        Example input matrix for AND gate
        | x1 | x2 | y |
        | -1 | -1 | 0 |
        | -1 | +1 | 0 |
        | +1 | -1 | 0 |
        | +1 | +1 | 1 |
        '''
        self.input_matrix = input_matrix
    def iterate_all(self):
        for w1 in self.possible_w1_vals:
            self.w1 = w1
            for w2 in self.possible_w2_vals:
                self.w2 = w2
                for theta in self.possible_thresh_vals:
                    self.theta = theta
                    if self.check_combination():
                        return True
        return False
    def check_combination(self):
        valid = True
        for (x1, x2, y) in self.input_matrix:
              if not self.compare_target(x1, x2, y):
                  valid = False
        return valid
    def compare_target(self, x1, x2, target):
        if self.activate(x1, x2) == target:
            return True
        else:
             return False
    def activate(self, x1, x2):
        output = self.w1*x1 + self.w2*x2
        if output >= self.theta:
            return 1
        else:
            return 0


def neuron_calculate(mp):
    if mp.iterate_all():
        print("Weights are : {}, {}".format(mp.w1, mp.w2))
        print("Theta is {}".format(mp.theta))
    else:
        print("Not linearly separable.")
print()




if __name__=="__main__":
    AND_Matrix = [
        [-1, -1, 0],
        [-1, 1, 0],
        [ 1, -1, 0],
        [ 1, 1, 1],
```

```
]
OR_Matrix = [
    [-1, -1, 0],
    [-1, 1, 1],
    [ 1, -1, 1],
    [ 1, 1, 1],
]
NAND_Matrix = [
    [-1, -1, 1],
    [-1, 1, 1],
    [ 1, -1, 1],
    [ 1, 1, 0],
]
XOR_Matrix = [
    [-1, -1, 0],
    [-1, 1, 1],
    [ 1, -1, 1],
    [ 1, 1, 0],
]
print("AND Gate")
mp_AND = Neuron(AND_Matrix)
neuron_calculate(mp_AND)
print("OR Gate")
mp_OR = Neuron(OR_Matrix)
neuron_calculate(mp_OR)
print("NAND Gate")
mp_NAND = Neuron(NAND_Matrix)
neuron_calculate(mp_NAND)
print("XOR Gate")
mp_XOR = Neuron(XOR_Matrix)
neuron_calculate(mp_XOR)
```

```
AND Gate
Weights are : 1, 1
Theta is 1
OR Gate
Weights are : 1, 1
Theta is -1
NAND Gate
Weights are : -1, -1
Theta is -1
XOR Gate
Not linearly separable.
```

Colab paid products - Cancel contracts here