Indian Institute of Technology, Indore
Computer Science & Engineering
CS 354N: Assignment 4- ANN

ANIKEIT SETHI

190001003

In [12]:

```
import numpy as np
class Model:
    def __init__(self, no_inp):
        self.wts = np.zeros(no_inp)
        self.n = no_inp
        self.Threshold = 0
        self.epochs=0
    def predict(self, inputs):
        sum = np.dot(inputs, self.wts[:])
        if sum>self.Threshold:
            prediction = 1
        else:
            prediction = 0
        return prediction
# createsall possible wts
    def create_weights_list(self):
        Weights=[]
        for i in range(0,2**self.n):
            wt=[]
            for pos in range(0,self.n):
                if (1<<pos)&(i):
                    wt.append(1)
                else:
                    wt.append(-1)
            Weights.append(wt)
        return Weights
    def fit(self, inputs, label):
        self.epochs
        # creating all possible wts
        Weights= self.create_weights_list()

    # run for all weight combinations
        for wt in Weights:
    # Assume required weight is current weight
            self.wts= np.array(wt)
            self.epochs=self.epochs+1
    # We find if any possible threshold value satisfies it
            for threshold in range(-self.n,self.n+1):
                self.Threshold= threshold
                i=0
                for input in inputs:
                    prediction = self.predict(input)
                    if prediction != label[i]:
                        break
                    i= i + 1
    # if all inputs give right prediction then our Model is trained
                if i==len(inputs):
                    return
```

Q1: 2 Input

In [13]:

```
# AND Gate
import numpy as np
train_inputs= np.array([
    [1,1], # bot yes
    [1,0], # one yes, one no
    [0,1], # one no, one yes
    [0,0]  # both no
])
labels= np.array([1,0,0,0])
mcp = Model(2)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
```

```
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1] 1
[1 0] 0
[0 1] 0
[0 0] 0
Threshold 1
Weights [1 1]
Epochs 4
```

In [14]:

```
# OR Gate
import numpy as np
train_inputs=  np.array([
    [1,1], # bot yes
    [1,0], # one yes, one no
    [0,1], # one no, one yes
    [0,0]  # both no
])
labels= np.array([1,1,1,0])
mcp = Model(2)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1] 1
[1 0] 1
[0 1] 1
[0 0] 0
Threshold 0
Weights [1 1]
Epochs 4
```

In [15]:

```
# NAND Gate
import numpy as np
train_inputs=  np.array([
    [1,1], # bot yes
    [1,0], # one yes, one no
    [0,1], # one no, one yes
    [0,0]  # both no
])
labels= np.array([0,1,1,1])
mcp = Model(2)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1] 0
[1 0] 1
[0 1] 1
[0 0] 1
Threshold -2
Weights [-1 -1]
Epochs 1
```

In [16]:

```
# NOR Gate
import numpy as np
train_inputs=  np.array([
    [1,1], # bot yes
```

```
    [1,0], # one yes, one no
    [0,1], # one no, one yes
    [0,0]  # both no
])
labels= np.array([0,0,0,1])
mcp = Model(2)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1] 0
[1 0] 0
[0 1] 0
[0 0] 1
Threshold -1
Weights [-1 -1]
Epochs 1
```

Q2 : 3 Input

In [17]:

```
# AND Gate
import numpy as np
train_inputs= np.array([
  [1,1,1],
  [1,1,0],
  [1,0,1],
  [1,0,0],
  [0,1,1],
  [0,1,0],
  [0,0,1],
  [0,0,0]
  ])
labels= np.array([1,0,0,0,0,0,0,0])
mcp = Model(3)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1 1] 1
[1 1 0] 0
[1 0 1] 0
[1 0 0] 0
[0 1 1] 0
[0 1 0] 0
[0 0 1] 0
[0 0 0] 0
Threshold 2
Weights [1 1 1]
Epochs 8
```

In [18]:

```
# OR Gate
import numpy as np
train_inputs= np.array([
  [1,1,1],
  [1,1,0],
  [1,0,1],
  [1,0,0],
  [0,1,1],
  [0,1,0],
  [0,0,1],
  [0,0,0]
```

```
  ])
labels= np.array([1,1,1,1,1,1,1,0])
mcp = Model(3)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1 1] 1
[1 1 0] 1
[1 0 1] 1
[1 0 0] 1
[0 1 1] 1
[0 1 0] 1
[0 0 1] 1
[0 0 0] 0
Threshold 0
Weights [1 1 1]
Epochs 8
```

In [19]:

```
# NAND Gate
import numpy as np
train_inputs= np.array([
  [1,1,1],
  [1,1,0],
  [1,0,1],
  [1,0,0],
  [0,1,1],
  [0,1,0],
  [0,0,1],
  [0,0,0]
  ])
labels= np.array([0,1,1,1,1,1,1,1])
mcp = Model(3)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1 1] 0
[1 1 0] 1
[1 0 1] 1
[1 0 0] 1
[0 1 1] 1
[0 1 0] 1
[0 0 1] 1
[0 0 0] 1
Threshold -3
Weights [-1 -1 -1]
Epochs 1
```

In [20]:

```
# NOR Gate
import numpy as np
train_inputs= np.array([
  [1,1,1],
  [1,1,0],
  [1,0,1],
  [1,0,0],
  [0,1,1],
  [0,1,0],
  [0,0,1],
  [0,0,0]
  ])
labels= np.array([0,0,0,0,0,0,0,1])
```

```
mcp = Model(3)
mcp.fit(train_inputs,labels)
for inputs in train_inputs:
  print(inputs, mcp.predict(inputs))
print("Threshold",mcp.Threshold)
print("Weights",mcp.wts)
print("Epochs",mcp.epochs)
```

```
[1 1 1] 0
[1 1 0] 0
[1 0 1] 0
[1 0 0] 0
[0 1 1] 0
[0 1 0] 0
[0 0 1] 0
[0 0 0] 1
Threshold -1
Weights [-1 -1 -1]
Epochs 1
```