Perceptron

When we add a value to the weights of a NP neuron it becomes a perceptron. In the NP neuron the value of weight is always 1.

In perceptron, we will make the neuron learn the weights.

image.png

Weight Learning OR Gate

pred function here is the TLU.

While learning weights we add a learning rate as well

```
import numpy as np

def pred(x,w):
  return 1 if (x*w).sum() > 0 else 0

def learning(x,y,w):
  print("Initial weight: ", w)
  for k in range(10):
    print("------ iterations", k+1)
    for i,j in zip(x,y):
      if j[0] - pred(i,w) > 0:
        w = w + 0.1*i
      elif j[0] - pred(i,w) < 0:
        w = w- 0.1*i
  return w

x = np.array([[0,0], [0,1], [1,0], [1,1]])
y = np.array([[0],[1],[1],[1]])
w = np.random.randn((len(x[0])))
w = learning(x,y,w)
# print(w)
# pred(x[1],w)

#test
print("test ---- ")
for i in x:
  print(pred(i,w))

print("updated wight: ", w)
print(y)
```

```
Initial weight:  [-0.49013666 -1.05214049]
------ iterations 1
------ iterations 2
------ iterations 3
------ iterations 4
------ iterations 5
------ iterations 6
------ iterations 7
------ iterations 8
------ iterations 9
------ iterations 10
test ----
0
1
1
1
updated wight:  [0.20986334 0.04785951]
[[0]
 [1]
 [1]
 [1]]
```

Weight Learing AND Gate

```
import numpy as np

def pred(x,w):
  return 1 if (x*w).sum() > 0 else 0

def learning(x,y,w):
  print("Initial weight: ", w)
  for k in range(100):
    print("------ iterations", k+1)
    for i,j in zip(x,y):
      if j[0] - pred(i,w) > 0:
        w = w + 0.1*i
      elif j[0] - pred(i,w) < 0:
        w = w- 0.1*i
  return w

x = np.array([[0,0], [0,1], [1,0], [1,1]])
y = np.array([[0],[0],[0],[1]])
w = np.random.randn((len(x[0])))
w = learning(x,y,w)
# print(w)
# pred(x[1],w)

#test
print("test ---- ")
for i in x:
  print(pred(i,w))

print("updated wight: ", w)
print(y)
```

```
Initial weight:  [0.19983571 0.95959967]
------ iterations 1
------ iterations 2
------ iterations 3
------ iterations 4
------ iterations 5
------ iterations 6
------ iterations 7
------ iterations 8
------ iterations 9
------ iterations 10
------ iterations 11
------ iterations 12
------ iterations 13
------ iterations 14
------ iterations 15
------ iterations 16
------ iterations 17
------ iterations 18
------ iterations 19
------ iterations 20
------ iterations 21
------ iterations 22
------ iterations 23
------ iterations 24
------ iterations 25
------ iterations 26
------ iterations 27
------ iterations 28
------ iterations 29
------ iterations 30
------ iterations 31
------ iterations 32
------ iterations 33
------ iterations 34
------ iterations 35
------ iterations 36
------ iterations 37
------ iterations 38
------ iterations 39
------ iterations 40
------ iterations 41
------ iterations 42
------ iterations 43
------ iterations 44
------ iterations 45
------ iterations 46
------ iterations 47
------ iterations 48
------ iterations 49
------ iterations 50
------ iterations 51
------ iterations 52
------ iterations 53
------ iterations 54
------ iterations 55
------ iterations 56
------ iterations 57
```