Perceptrons - Making Predictions

Creating a gate with Perceptron

```
import numpy as np
```

AND Gate

```
def AND(x1, x2):
  # 교수님이 w가 다 더해서 1이 되어야 한다는 의미는. 다 더해서 1을 넘어도 상관이 없으나, 나중에 추
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5]) # w는 가중치를 의미합니다. 기울기라고 봐도 괜찮습니다.
    b = -0.7 # bias == 절편
    # activation 계산
    tmp = w[0]*x[0] + w[1]*x[1] + b # tmp = np.sum(w*x) + b
    print("AND({0},{1}) # activation : {2}".format(x1,x2,tmp))
    if tmp <= 0:
      print("result = 0")
      return 0
    else:
      print("result = 1")
      return 1
```

```
AND(0,0)
AND(0,1)
AND(1,0)
AND(1,1)
```

```
AND(0,0) # activation : -0.7
result = 0
AND(0,1) # activation : -0.2
result = 0
AND(1,0) # activation : -0.2
result = 0
AND(1,1) # activation : 0.3
result = 1
1
```

NAND Gate

```
def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.5, -0.5])
    b = 0.7
    tmp = w[0]*x[0] + w[1]*x[1] + b
    print("NAND({0},{1}) # activation : {2}".format(x1,x2,tmp))
    if tmp <= 0:
      print("result = 0")
      return 0
    else:
      print("result = 1")
      return 1
```

```
NAND(0,0)
NAND(0,1)
NAND(1,0)
```

```
NAND(1,1)
```


OR Gate

```
def OR(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.2
    tmp = np.sum(w*x) + b
    print("OR({0},{1}) # activation : {2}".format(x1,x2,tmp))
    if tmp <= 0:
      print("result = 0")
      return 0
    else:
      print("result = 1")
      return 1


OR(0,0)
OR(0,1)
OR(1,0)
OR(1,1)
```


XOR Gate

```
def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    if y <= 0:
      print("########## ############")
      print("XOR({0},{1}) # result = 0)".format(x1,x2))
      print("########## ############")
      return y
    else:
      print("########## ############")
      print("XOR({0},{1}) # result = 1)".format(x1,x2))
      print("########## ############")
      return y
```

```
XOR(0,0)
XOR(0,1)
XOR(1,0)
XOR(1,1)
```


XOR cannot be expressed as a single layer Perceptron.