Lab Seminar: 2022. 09. 16.

Decision Tree & Intro of Neural Network

Data Science from Scratch 2nd – Chapter 17, 18

IDEALAB
Improving lives through learning

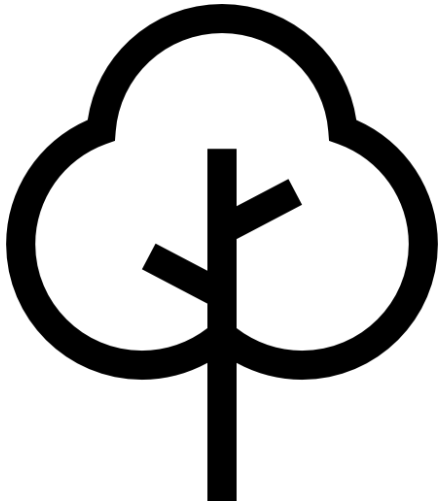
ChanKi Kim
School of Computer Science/Department of AI Convergence Engineering
Gyeongsang National University (GNU)

GNU 경상국립대학교 Gyeongsang National University

Contents

Decision Tree

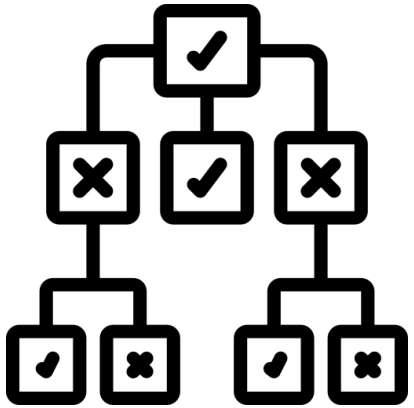
- Decision Tree
  - 계속해서 질문을 던져 나가면서 범주를 좁혀 나가 특정한 값이 최종적으로 도달한 값으로 예측하는 것
    - 마치 “스무고개 놀이”와 유사
      - ex) 다리가 네 개 이하인가요?, 아가미가 있나요?



Decision Tree

- Mechanism of Decision Tree
  - Dataset의 Feature들을 X로 두고, 예측하고자 하는 Feature를 Y로 설정
  - 각 층마다 Condition을 주어 각 노드에 속한 값의 개수 감소시키며, Correction 향상

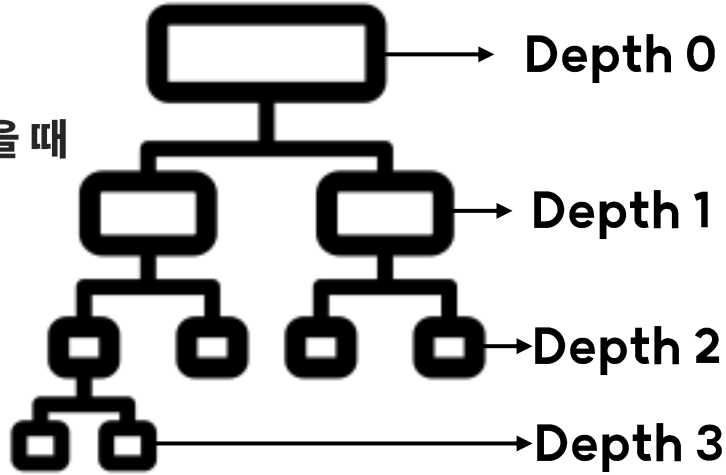
	Ear Shape	Face Shape	Whiskers	Cat?
Cat	Pointy	Round	Present	1
Dog	Floppy	Not Round	Absent	0
…	…	…	…	…
Cat	Pointy	Round	Present	1



Decision Tree

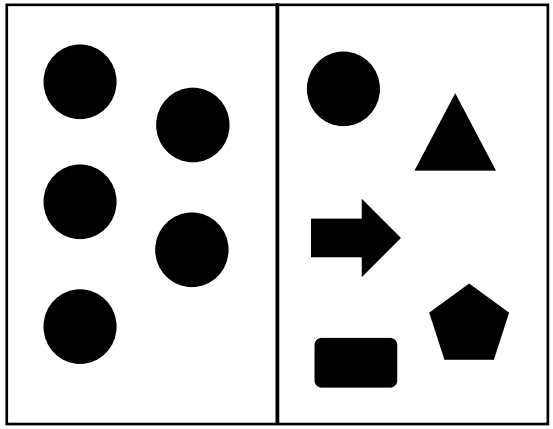
- Decision 1
  - "언제까지 쪼개 나갈 것인가?"
    - 1. 노드가 100% 하나의 클래스로만 이루어져 있을 때
    - 2. 설정한 최대 깊이(Maximum Depth)에 도달하였을 때

      - Maximum Depth = Hyperparameter



Decision Tree

- Decision 2
  - "각 노드로 쪼갤 feature를 어떻게 선택할 것인가?"
    - Maximize Purity(=Minimize Impurity)
      - 위와 같은 실행은 높은 정확도 도출



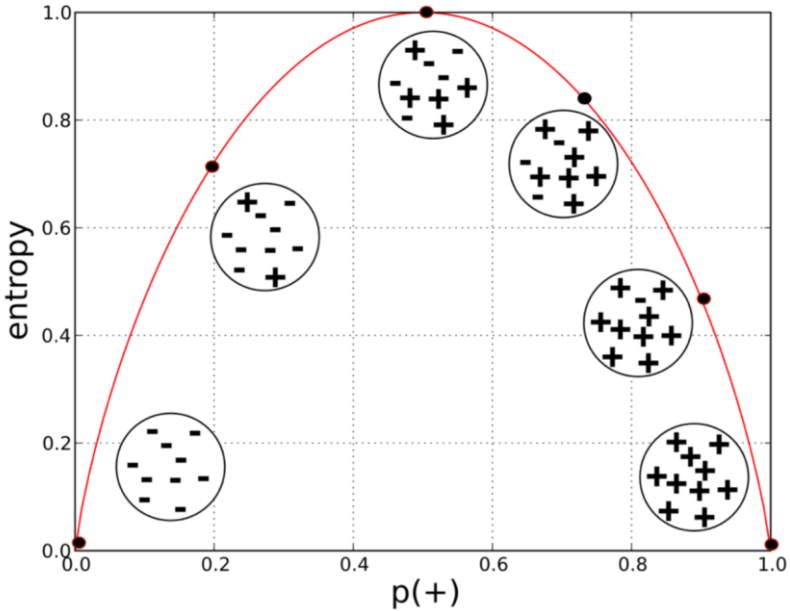
Decision Tree

- Entropy
  - Impurity를 나타내는 척도
    - p(+) = 예시의 일부가 +일 확률 (‘+’ = 1, ‘–’ = 0)

$$- H\big(p(+)\big) = -p(+)\log_2\big(p(+)\big) - p(-)\log_2\big(p(-)\big)$$

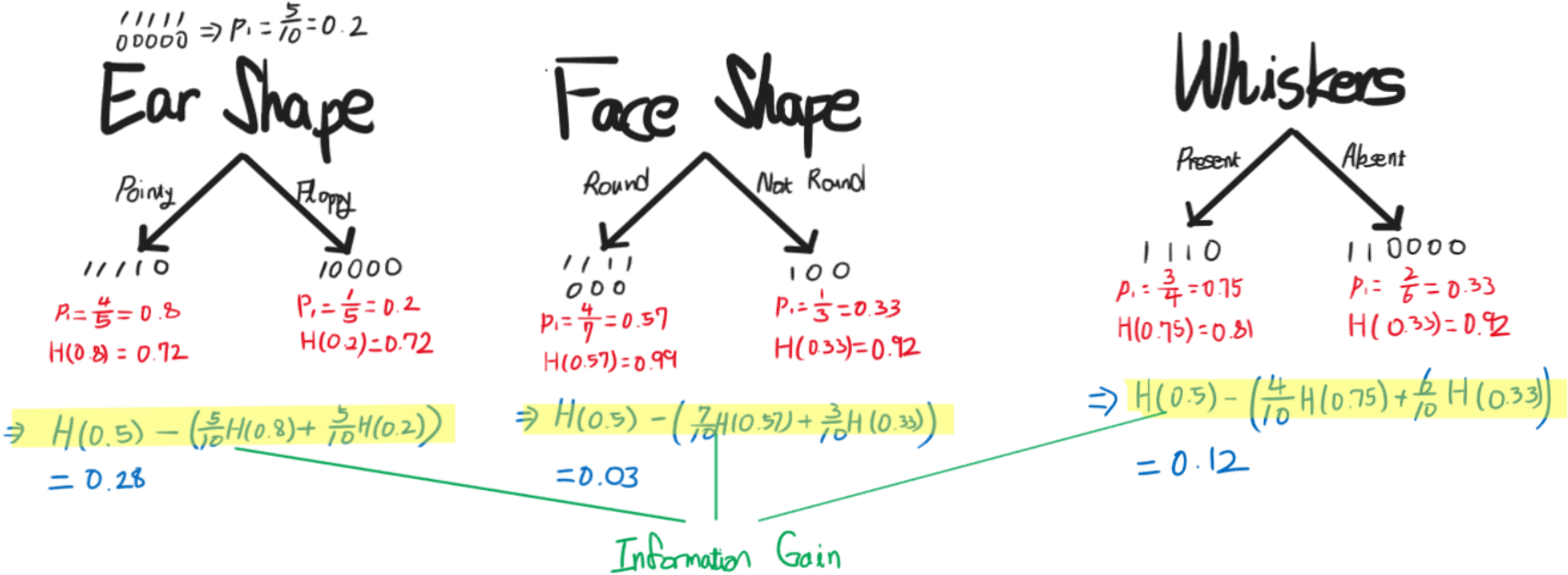
Entropy		
– – – – – –	p(+) =0	H(p(+))=0
+ + – – – –	p(+) =2/6	H(p(+))=0.92
+ + + – – –	p(+) =3/6	H(p(+))=1
+ + + + + –	p(+) =5/6	H(p(+))=0.65
+ + + + + +	p(+) =6/6	H(p(+))=0

H(p(+))



Decision Tree

- **Choosing a Split**
  - **Goal : Reduce Entropy**



Decision Tree

- Information Gain
  - Split을 통해 Entropy가 얼마나 줄었는지 나타내는 값
    - 상위 Node의 Entropy에서 Split을 통해 Entropy가 얼마나 개선됐는지 Focus
    - 수식 : $H(p_i^{root}) - (w^{left} H(p_i^{left}) + w^{right} H(p_i^{right}))$
      - 높을수록 pure!

$$\Rightarrow H(0.5) - \left(\frac{5}{10} H(0.8) + \frac{5}{10} H(0.2)\right)$$

$$= 0.28$$

Decision Tree

- One-Hot Encoding
  - 범주형 자료를 수치형 자료로 변환하는 전 처리 기법
    - 장점 : 범주형 자료는 값에 순위를 매길 수 없기에, 예측 수행의 결과가 좋게 나오지 않는 결과를 원-핫 인코딩을 통해 순위를 매길 수 있게 하여 보다 더 나은 결과 도출 가능
    - 단점 : Category의 개수가 많아질수록 표현하기 위한 벡터의 개수의 증가로 인한 요구하는 저장 공간의 증가

      - Deep Learning 학문에서 전반적으로 많이 사용되지만, 특히 NLP에서 많이 사용됨

Decision Tree

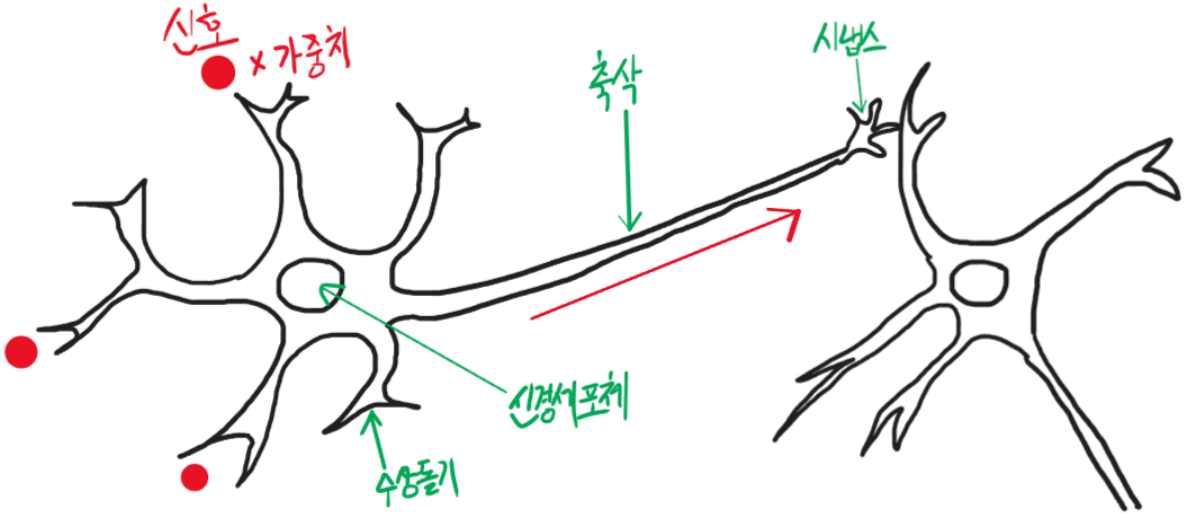
- Example of One-Hot Encoding

Cat? Dog?	Whiskers
Cat	Present
Dog	Absent
…	…
Cat	Absent

Cat? Dog?	Present	Absent
Cat	1	0
Dog	0	1
…	…	…
Cat	0	1

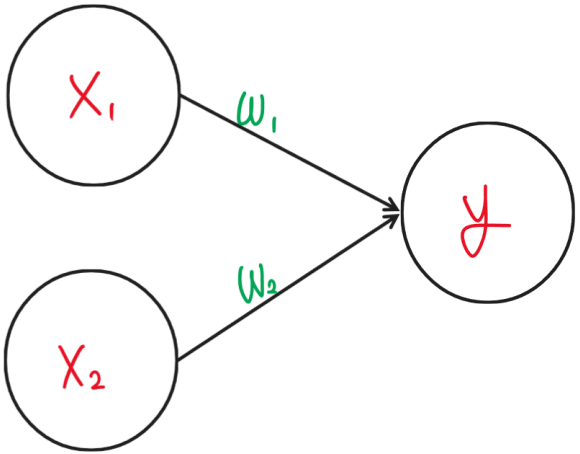
Neural Network

- Neuron Signal Transduction Process
  - 뉴런이 여러 뉴런으로부터 받은 **신호**와 **가중치**를 곱한 값의 합이 **임계값**보다 크면 다른 뉴런에게 신호를 전달하는 방식



Neural Network

- Perceptron
  - 뉴런의 신호 전달 과정과 동일한 방법으로 신호 전달
    - 뉴런과 동일하게 입력 받은 신호와 가중치를 곱하여 합한 값과 임계값을 비교하여 신호 전달 여부 선택
    - y <= θ인 경우 신호 전달 X, y > θ인 경우 신호 전달 O



$$y = \begin{cases} 0 & (w_1x_1 + w_2x_2 \le \theta) \\ 1 & (w_1x_1 + w_2x_2 > \theta) \end{cases}$$

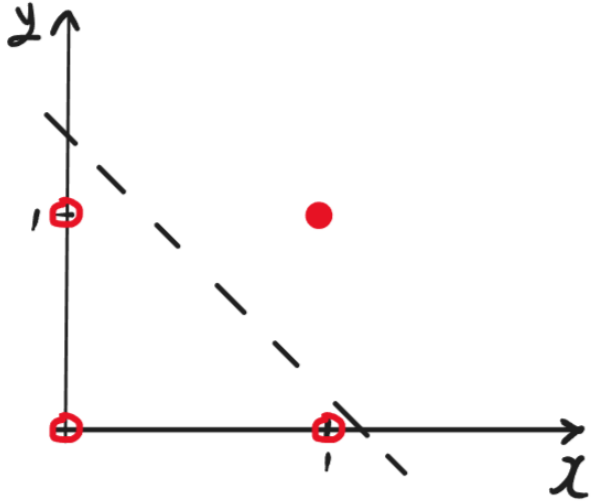
Neural Network

- Simple Logic Gate
  - Simple Logic Gate, 단순한 논리 회로에는 AND Gate, OR Gate, NAND Gate 존재
    - 위에서 언급된 세 가지의 논리 회로를 Perceptron으로 표현 가능하며 Perceptron의 한계를 파악하고 Perceptron을 쌓아 더욱 발전된 신경망 도출

Neural Network

- **AND Gate**
  - 두 입력이 모두 1일 때만 1 출력, 그 이외에는 0 출력하는 논리 회로
  - 같은 출력 값끼리 하나의 직선만으로 분리 가능 -> 단층 퍼셉트론으로 표현 가능

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1



Neural Network

- **AND Gate**

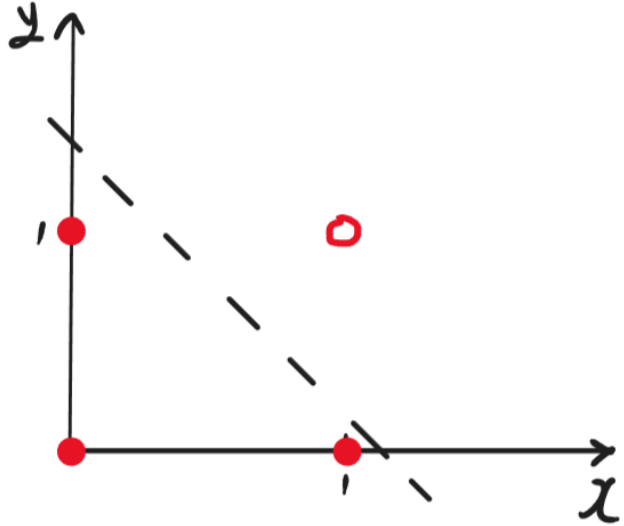
```python
import numpy as np

def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    print("AND Gate")
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = AND(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
AND Gate
(0, 0) -> 0
(1, 0) -> 0
(0, 1) -> 0
(1, 1) -> 1
```

Neural Network

- **NAND Gate**
  - Not AND로 AND Gate의 출력을 뒤집은 논리 회로
  - 같은 출력 값끼리 하나의 직선만으로 분리 가능 -> 단층 퍼셉트론으로 표현 가능

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0



Neural Network

- **NAND Gate**

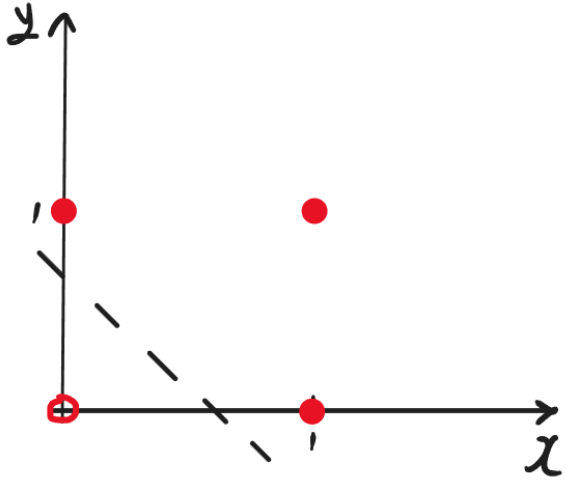
```python
import numpy as np

def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.5, -0.5])
    b = 0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    print("NAND Gate")
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = NAND(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
NAND Gate
(0, 0) -> 1
(1, 0) -> 1
(0, 1) -> 1
(1, 1) -> 0
```

Neural Network

- OR Gate
  - 입력 신호 중 하나 이상이 1 이상이면 출력 1, 그 이외에는 0을 출력하는 논리 회로
  - 같은 출력 값끼리 하나의 직선만으로 분리 가능 -> 단층 퍼셉트론으로 표현 가능

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1



Neural Network

- OR Gate

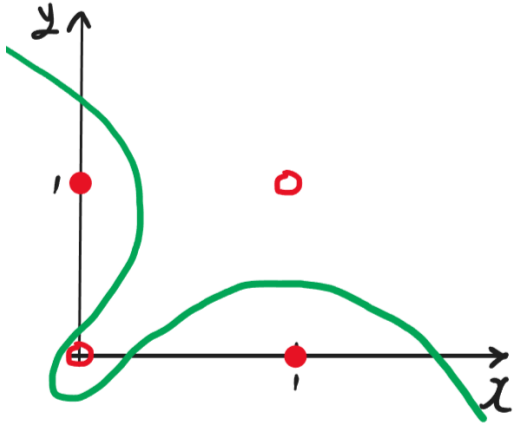
```python
import numpy as np

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
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    print("OR Gate")
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = OR(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
OR Gate
(0, 0) -> 0
(1, 0) -> 1
(0, 1) -> 1
(1, 1) -> 1
```

Neural Network

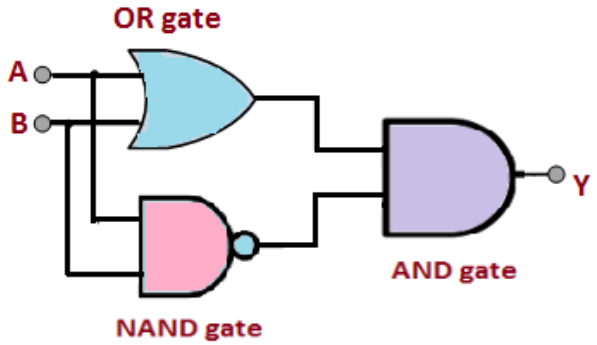
- Limitation of Perceptron
  - XOR Gate
    - X_1 혹은 X_2 중 하나의 입력만이 1일 때만 1을 출력하는 논리 회로
    - 같은 값끼리 하나의 직선만으로 분리 불가 -> 단층 퍼셉트론으로는 표현 불가

X_1	X_2	Output
0	0	0
0	1	1
1	0	1
1	1	0



Neural Network

- Limitation of Perceptron
  - XOR Gate
    - 하지만 OR Gate, NAND Gate, AND Gate를 조합하면 XOR Gate를 표현 가능
    - 즉, 조합에 사용된 세 가지 회로는 단층 퍼셉트론으로 표현 가능, XOR Gate는 다층 퍼셉트론으로 표현 가능



Neural Network

- Limitation of Perceptron
  - XOR Gate

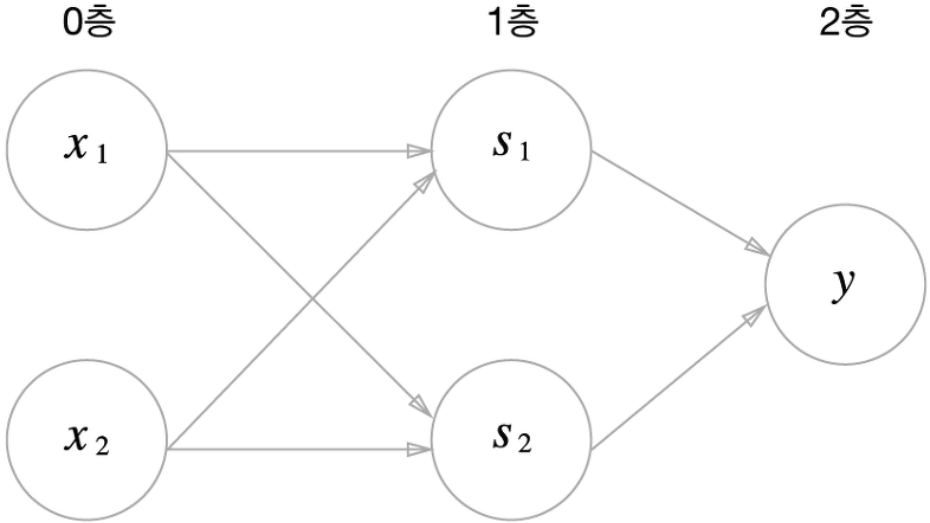
```python
import numpy as np

def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.5, -0.5])
    b = 0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1
```

```python
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
    if tmp <= 0:
        return 0
    else:
        return 1

def XOR(x1, x2):
    s1 = OR(x1, x2)
    s2 = NAND(x1, x2)
    y = AND(s1, s2)
    return y

if __name__ == '__main__':
    print("XOR Gate")
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = XOR(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
XOR Gate
(0, 0) -> 0
(1, 0) -> 1
(0, 1) -> 1
(1, 1) -> 0
```

Neural Network

- Limitation of Perceptron
  - XOR Gate
    - XOR의 퍼셉트론의 구조를 통해 단층 퍼셉트론인 AND Gate, OR Gate, NAND Gate와 달리 2층 퍼셉트론(다층 퍼셉트론)



Realization & Conclusion

- 이번 챕터 관련 세미나 관련 챕터에서는 알고 넘어가야 할 개념적인 부분들이 많았기에 구현보다는 개념적인 부분에 좀 더 Focus
- 다음 챕터는 Deep Learning과 Clustering에 관련된 부분이고 이번 세미나와 유기적으로 연결하여 발표할 예정
- Decision Tree을 베이스로 한 모델들을 차후에 구현해볼 예정

경상국립대학교

Gyeongsang National University

Improving lives through learning

IDEALAB