로지스틱 회귀분석(Logistic Regression) - 분류(범주예측)

```
import warnings
warnings.filterwarnings('ignore')
```

I. Sigmoid Activation

1) sigmoid( ) 정의

```
import numpy as np

def sigmoid(x):
    y = 1 / (1 + np.exp(-x))
    return y
```

2) sigmoid( ) 실행

```
sigmoid(0)
```

```
0.5
```

```
sigmoid(100000000)
```

```
1.0
```

```
sigmoid(-100000000)
```

```
0.0
```

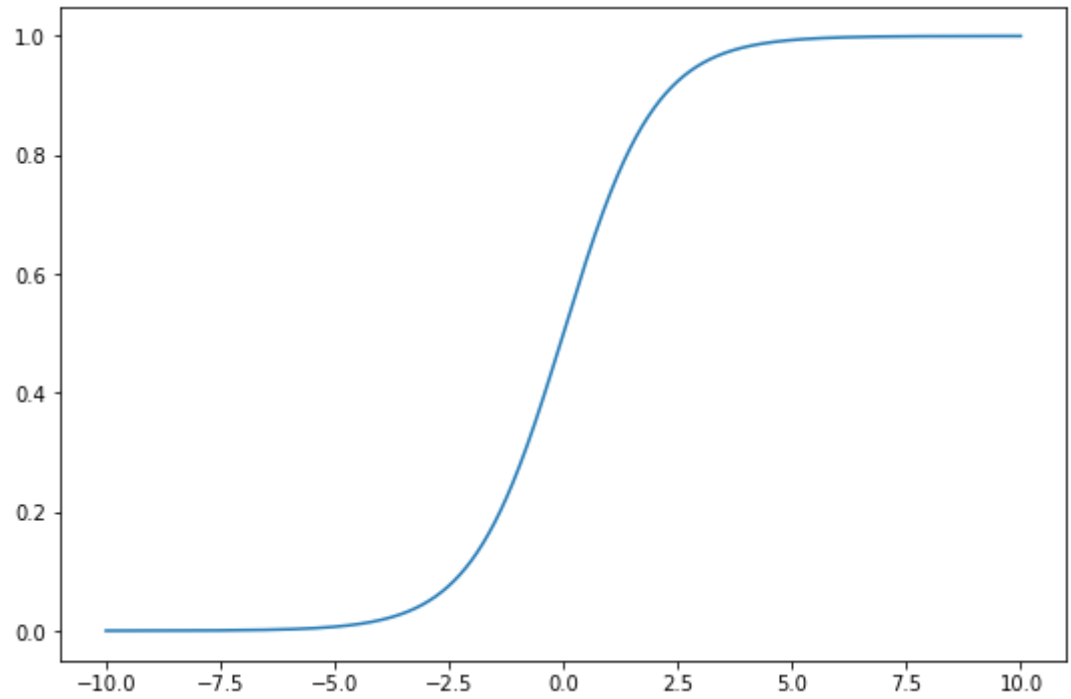
3) sigmoid( ) 시각화

```
import matplotlib.pyplot as plt

n = np.linspace(-10.0, 10.0, 2000)

plt.figure(figsize = (9, 6))
```

```
plt.plot(n, sigmoid(n))
plt.show()
```



II. Cross Entropy Error

1) Cross Entropy

- 서로 다른 사건의 확률을 곱하여 Entropy를 계산
  - y : 실제값, y_hat : 예측값(can be incorrect)
- y를 Cross-Entropy의 가중치로 적용
  - Binary Cross-Entropy Error = −y * log(y_hat) − (1 - y) * log(1 - y_hat)
  - Categorical Cross-Entropy Error = −y * log(y_hat)

(1) y = 1 vs. y_hat = 1

```
y = 1
y_hat = 1

-y * np.log(y_hat)
```

```
-0.0
```

(2) y = 1 vs. y_hat = 0.0001

```
y = 1
y_hat = 0.0001

-y * np.log(y_hat)
```

```
9.210340371976182
```

> (3) y = 0 vs. y_hat = 0

```
y = 0
y_hat = 0

-(1 - y) * np.log(1 - y_hat)
```

```
-0.0
```

> (3) y = 0 vs. y_hat = 0.9999

```
y = 0
y_hat = 0.9999

-(1 - y) * np.log(1 - y_hat)
```

```
9.210340371976294
```

2) Information Theory

> (1) 발생 확률이 서로 다른 사건 A, B, C - Information Gain

- Information Gain(정보 이득량)
  - 자주 발생하지 않는 사건은 자주 발생하는 사건보다 전달하는 정보량이 많음
  - Information Gain(정보 이득량)은 정보의 희귀성(발생가능성)에 반비례
  - I(x) = −log(P(x))

```
A = 0.9
B = 0.5
C = 0.1

print('%.3f' % -np.log(A), '%.3f' % -np.log(B), '%.3f' % -np.log(C))
```

```
0.105 0.693 2.303
```

> (2) AlphaGo와 Apes의 바둑대결 승리 확률 - Degree of Surprise

- Degree of Surprise(놀람의 정도)
  - 예상하기 어려운 정보에 더 높은 가치를 매기는 것

```
Alphago = 0.999
Apes = 0.001

print('%.3f' % -np.log(Alphago), '%.3f' % -np.log(Apes))
```

```
0.001 6.908
```

3) Entropy

- 불확실성의 정도
  - Entropy = E(−log(P(x)))
- 확률변수의 평균 정보량(기댓값)
  - −sum(p(x) * log(p(x)))
- 불확실성(Entropy)이 낮으면 분류정확도가 높아짐

> (1) 승률이 비슷한 두팀의 Entropy

```
P1 = 0.5
P2 = 0.5

-P1 * np.log(P1) - P2 * np.log(P2)
```

```
0.6931471805599453
```

> (2) 승률 차이가 큰 두팀의 Entropy

```
P1 = 0.999
P2 = 0.001

-P1 * np.log(P1) - P2 * np.log(P2)
```

```
0.007907255112232087
```

#

#

#

The End

#

#

#