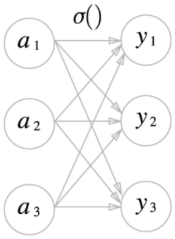
3장. 신경망 (계속)

출력층 설계

출력층은 분류(classification)과 회귀(regression)으로 나뉜다.

소프트맥스

분류를 위한 출력층에는 소프트맥스를 사용한다.



$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)}$$

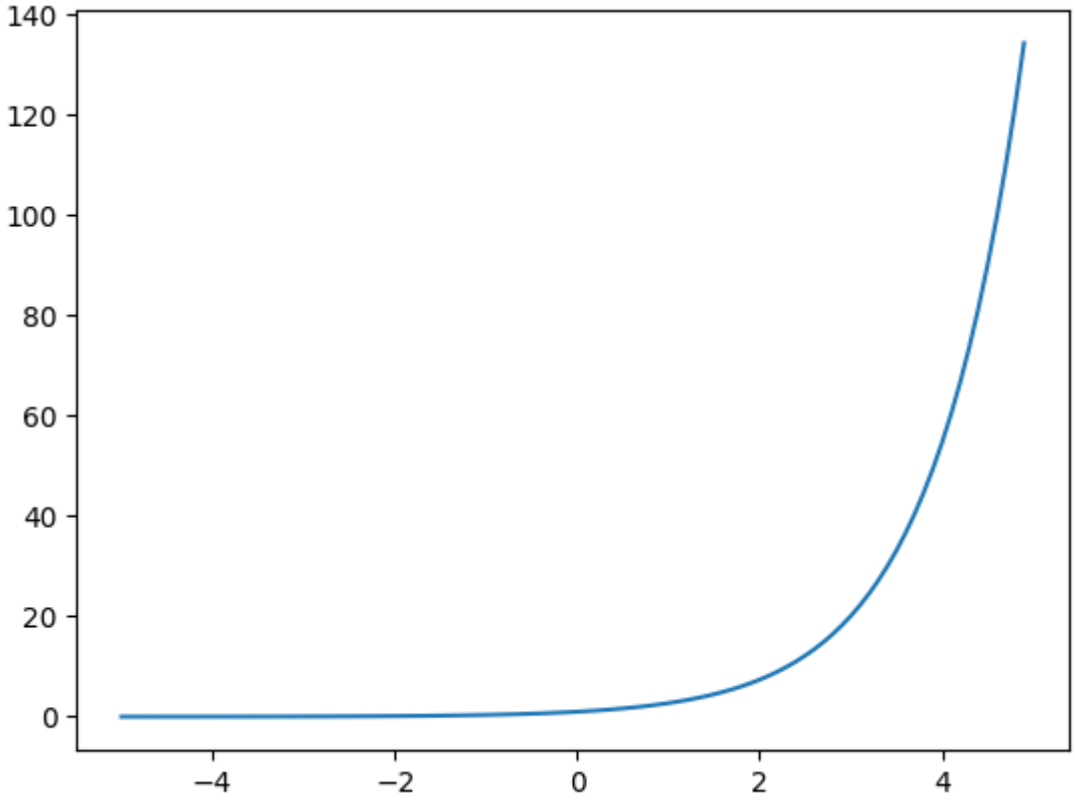
`exp(x)` 는 e^x 를 의미하는 지수함수(exponential function)이다. y_k는 y의 k번째 항이다. n은 출력층의 뉴런의 수.

분자에 모든 항들의 지수함수(= e^{a_i})의 합이 들어간다. 분자는 입력신호 중 한개의 지수함수의 값이 된다.

단, e^{1000} 정도만 되어도 크기가 너무 커져서 나눗셈이 불안정해진다. 따라서 다음 수식처럼 exp()안쪽을 적당히 작은 수로 만들어준다. (a_i)중 가장 큰수를 빼준다.

$$\begin{aligned} y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)} &= \frac{C\exp(a_k)}{C\sum_{i=1}^{n} \exp(a_i)} \\ &= \frac{\exp(a_k + \log C)}{\sum_{i=1}^{n} \exp(a_i + \log C)} \\ &= \frac{\exp(a_k + C')}{\sum_{i=1}^{n} \exp(a_i + C')} \end{aligned}$$

In [3]:
```python
# 지수함수의 그래프를 먼저 보자
import numpy as np
import matplotlib.pylab as plt

x = np.arange(-5.0, 5.0, 0.1)
y = np.exp(x)
plt.plot(x, y)
plt.show()
```



In [5]:
```
# 소프트맥스 함수
def softmax(a):
    c = np.max(a)
    exp_a = np.exp(a - c)
    sum_exp_a = np.sum(exp_a)
    y = exp_a / sum_exp_a
    return y
```

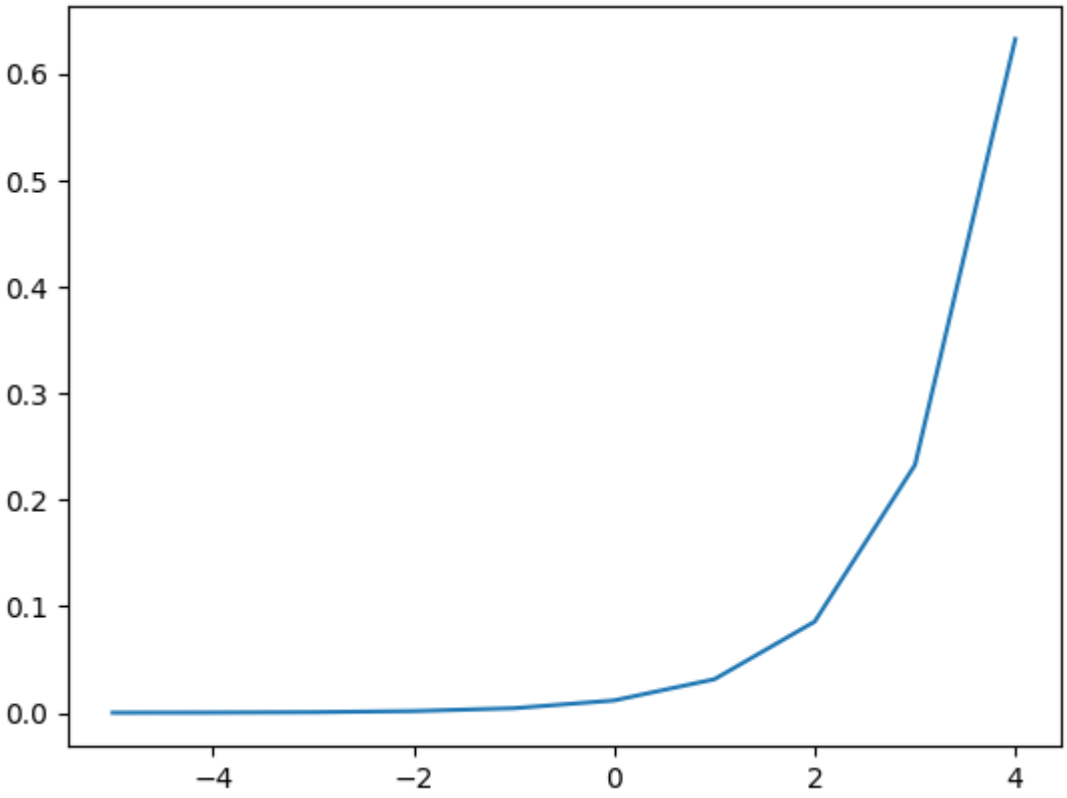
In [6]:
```
a = np.array([0.3, 2.9, 4.0])
y= softmax(a)
print(y)

# 합쳐서 1이 된다는 것은 중요한 성질 -> 확률로 해석할 수 있다.
print(np.sum(y))
```

```
[0.01821127 0.24519181 0.73659691]
1.0
```

In [14]:
```
# softmax 의 특징중 하나. 대소관계가 유지된다. (실무에서는 계산 부담 줄이려고 빼버리기도 한다.)
a = np.array([4,6])
print(softmax(a))
print(sum(softmax(a)))
```

```
[0.11920292 0.88079708]
0.9999999999999999
```

In [15]:
```
# 소프트맥스 그래프도 보자
x = np.arange(-5.0, 5.0, 1)
y = softmax(x)
plt.plot(x, y)
plt.show()
```



In [10]:
```
# 단순히 exp(x) 를 납작하게 눌러서 모든 값의 합이 1이 되게 한것 뿐이구나 !!!
```