公式4.1-4.2

$$
\begin{aligned}
z &= \sum_{i=1}^{d} w_i x_i + b \\
&= \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b
\end{aligned}
$$

实例4.1-4.2

```
import numpy as np

z = 0  # 净输入
w = np.asarray([1, 2, 3])  # 3维权重向量
x = np.asarray([1, 2, 3])  # 输入向量
b = 1  # 偏置

z = np.vdot(w, x) + b  # 公式4.1，4.2

print(z)

'''
15
'''
```

公式4.3

$$
a = f(z)
$$

实例4.3

```
import numpy as np


def f(x):  # 非线性函数f，假设为正弦函数
    return np.sin(x)


a = 0  # 活性值
z = 15  # 净输入

a = f(15)  # 公式4.3

print(a)

'''
0.6502878401571168
'''
```

公式4.4

$$
\sigma(x) = \frac{1}{1 + \exp(-x)}
$$

实例4.4

```
import numpy as np
import math


def sigma(x):  # Logistic函数
    return 1 / (1 + math.exp(-x))


sigma_x = 0  # 输出
x = 1  # 输入

sigma_x = sigma(x)  # 公式4.4

print(sigma_x)

'''
0.7310585786300049
'''
```

公式4.5

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

实例4.5

```
import numpy as np
import math


def tanh(x):  # Tanh函数
    return (math.exp(x) - math.exp(-x)) / (math.exp(x) + math.exp(-x))


tanh_x = 0  # 输出
x = 1  # 输入

tanh_x = tanh(x)  # 公式4.5

print(tanh_x)

'''
0.7615941559557649
'''
```

公式4.6

$$\tanh(x) = 2\sigma(2x) - 1$$

实例4.6

```
import numpy as np
import math


def sigma(x):  # Logistic函数
    return 1 / (1 + math.exp(-x))
```

```
def tanh(x):  # Tanh函数
    return 2 * sigma(2 * x) - 1


tanh_x = 0  # 输出
x = 1  # 输入

tanh_x = tanh(x)  # 公式4.6

print(tanh_x)

'''
0.7615941559557646
'''
```

公式4.7-4.8

$$
\begin{aligned}
g_l(x) &\approx \sigma(0) + x \times \sigma'(0) \\
&= 0.25x + 0.5
\end{aligned}
$$

实例4.7-4.9

```
import numpy as np
import math


def sigma(x):  # Logistic函数
    return 1 / (1 + math.exp(-x))


def sigma_fd(x):  # Logistic函数的一阶导函数
    return sigma(x)*(1-sigma(x))


def g_l(x):  # Logistic函数的一阶泰勒展开函数
    return sigma(0) + x * sigma_fd(0)


g_l_x = 0  # 输出
x = 1  # 输入

g_l_x = g_l(x)  # 公式4.7-4.8

print(g_l_x)

'''
0.75
'''
```

公式4.9-4.11

$$
\begin{aligned}
\text{hard-logistic}\,(x) &= \begin{cases} 1 & g_l(x) \geq 1 \\ g_l & 0 < g_l(x) < 1 \\ 0 & g_l(x) \leq 0 \end{cases} \\
&= \max\left(\min\left(g_l(x), 1\right), 0\right) \\
&= \max(\min(0.25x + 0.5, 1), 0)
\end{aligned}
$$

实例4.9-4.11

```
import numpy as np
import math


def sigma(x):  # Logistic函数
    return 1 / (1 + math.exp(-x))


def sigma_fd(x):  # Logistic函数的一阶导函数
    return sigma(x)*(1-sigma(x))


def g_l(x):  # Logistic函数的一阶泰勒展开函数
    return sigma(0) + x * sigma_fd(0)


def hard_logistic(x):  # 分段函数hard_logistic近似logistic函数
    if g_l(x) >= 1:
        return 1
    elif g_l(x) <= 0:
        return 0
    else:
        return g_l(x)


hard_logistic_x = 0  # 输出
x = 1  # 输入

hard_logistic_x = hard_logistic(x)  # 公式4.9-4.11

print(hard_logistic_x)

'''
0.75
'''
```

公式4.12-4.13

$$
\begin{aligned}
g_t(x) &\approx \tanh(0) + x \times \tanh'(0) \\
&= x
\end{aligned}
$$

实例4.12-4.13

```
import numpy as np
import math


def tanh(x):  # Tanh函数
    return (math.exp(x) - math.exp(-x)) / (math.exp(x) + math.exp(-x))
```

```
def tanh_fd(x):  # Tanh函数的一阶导函数
    return 1 - tanh(x) ** 2


def g_t(x):  # Tanh函数在0附近的一阶泰勒展开函数
    return tanh(0) + x * tanh_fd(0)


g_t_x = 0  # 输出
x = 1  # 输入

g_t_x = g_t(x)  # 公式4.12-4.13

print(g_t_x)

'''
1.0
'''
```

公式4.14-4.15

$$\begin{aligned}\text{hard-tanh}\left(x\right) &= \begin{cases} 1 & g_t(x) \geq 1 \\ g_t & 0 < g_t(x) < 1 \\ -1 & g_t(x) \leq -1 \end{cases} \\ &= \max\left(\min\left(g_t(x), 1\right), -1\right) \\ &= \max(\min(x, 1), -1)\end{aligned}$$

实例4.14-4.15

```
import numpy as np
import math


def tanh(x):  # Tanh函数
    return (math.exp(x) - math.exp(-x)) / (math.exp(x) + math.exp(-x))


def tanh_fd(x):  # Tanh函数的一阶导函数
    return 1 - tanh(x) ** 2


def g_t(x):  # Tanh函数在0附近的一阶泰勒展开函数
    return tanh(0) + x * tanh_fd(0)


def hard_tanh(x):  # 分段函数hard_tanh近似tanh函数
    if g_t(x) >= 1:
        return 1
    elif g_t(x) <= -1:
        return - 1
    else:
        return g_t(x)


hard_tanh_x = 0  # 输出
```

```
x = 1  # 输入

hard_tanh_x = hard_tanh(x)  # 公式4.14-4.15

print(hard_tanh_x)

'''
1
'''
```

公式4.16-4.17

$$\begin{aligned}\mathrm{ReLU}(x) &= \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases} \\ &= \max(0, x)\end{aligned}$$

实例4.16-4.17

```
import numpy as np
import math


def relu(x):  # ReLU函数
    if x >= 0:
        return x
    else:
        return 0


relu_x = 0  # 输出
x = 1  # 输入

relu_x = relu(x)  # 公式4.16-4.17

print(relu_x)

'''
1
'''
```

公式4.18-4.19

$$\begin{aligned}\mathrm{LeakyReLU}\ (x) &= \begin{cases} x & \text{if } x > 0 \\ \gamma x & \text{if } x \leq 0 \end{cases} \\ &= \max(0, x) + \gamma \min(0, x)\end{aligned}$$

实例4.18-4.19

```
import numpy as np
import math


def leaky_relu(x):  # 带泄露的Relu函数
    gamma = 0.001
    if x > 0:
        return x
    else:
```

```
        return gamma * x


leaky_relu_x = 0  # 输出
x = 1  # 输入

leaky_relu_x = leaky_relu(x)  # 公式4.18-4.19

print(leaky_relu_x)

'''
1
'''
```

公式4.20

$$\text{LeakyReLU}\ (x) = \max(x, \gamma x)$$

实例4.20

```
import numpy as np
import math


def leaky_relu(x):  # 带泄露的Relu函数
    gamma = 0.001
    return max(x, gamma * x)

leaky_relu_x = 0  # 输出
x = 1  # 输入

leaky_relu_x = leaky_relu(x)  # 公式4.20

print(leaky_relu_x)

'''
1
'''
```