Python小练习：激活函数

作者：凯鲁嘎吉 - 博客园 http://www.cnblogs.com/kailugaji/

本文介绍几种常见的激活函数，并用Python来实现，包括：Sigmoid、tanh、ReLU、LeakyReLU、ELU、Swish、softmax、softplus。

1．常见激活函数定义

- Sigmoid

$$\text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$$

- Swish

$$\text{Swish}(x) = x \cdot \text{Sigmoid}(x)$$

- tanh

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

- softmax

$$\text{softmax}(x) = \frac{e^{x}}{\sum_{d=1}^{D} e^{x_d}}$$

- ReLU

$$\text{ReLU}(x) = \begin{cases} x, & x > 0 \\ 0, & x \le 0 \end{cases}$$

- LeakyReLU

$$\text{LeakyReLU}(x) = \begin{cases} x, & x > 0 \\ \gamma x, & x \le 0 \end{cases}$$

- ELU

$$\text{ELU}(x) = \begin{cases} x, & x > 0 \\ \gamma(e^{x} - 1), & x \le 0 \end{cases}$$

- softplus

$$\text{softplus}(x) = \log(1 + e^{x})$$

2. activation_function_test1.py

```python
# -*- coding: utf-8 -*-
# Author：凯鲁嘎吉 Coral Gajic
# https://www.cnblogs.com/kailugaji/
# Python小练习：激活函数
# 用Python实现常见的激活函数：'sigmoid', 'tanh', 'relu', 'leakyrelu', 'elu', 'swish', 'softmax','softplus'
'''
部分参考：
    https://zhuanlan.zhihu.com/p/397494815
'''
import numpy as np
import torch
import torch.nn.functional as F
import matplotlib.pyplot as plt
plt.rc('font',family='Times New Roman')

# 激活函数
def activation_function(index, x, gamma = None, dim = -1):
    y = torch.empty([]) # 自己手动写的激活函数
    z = torch.empty([]) # 调用Pytorch内置的激活函数
    if index == 'sigmoid':
        y = 1 / (1 + torch.exp(-x))
        z = torch.sigmoid(x)
    elif index == 'tanh':
        y = (torch.exp(x) - torch.exp(-x)) / (torch.exp(x) + torch.exp(-x))
        z = torch.tanh(x)
    elif index == 'relu':
        y = np.where(x >= 0, x, 0)
        y = torch.tensor(y)
        z = F.relu(x)
    elif index == 'leakyrelu':
        y = np.where(x > 0, x, x * gamma)
        y = torch.tensor(y)
        z = F.leaky_relu(x, gamma)
    elif index == 'elu':
        y = np.where(x > 0, x, gamma * (np.exp(x) - 1))
        y = torch.tensor(y)
        z = F.elu(x,gamma)
    elif index == 'swish':
        y = x * (1 / (1 + torch.exp(-x)))
        z = x * torch.sigmoid(x)
    elif index == 'softmax':
        y = torch.exp(x) / torch.exp(x).sum(dim = dim, keepdim = True)
        z = F.softmax(x, dim = dim)
    elif index == 'softplus':
        y = torch.log(1 + torch.exp(x))
        z = F.softplus(x)
    return y, z
```

```
48
49 torch.manual_seed(1)
50 x = torch.randn(2, 3) # 原始数据
51 print('原始数据：\n', x)
52 # activation_function()参数设置
53 index = ['sigmoid', 'tanh', 'relu', 'leakyrelu', 'elu', 'swish', 'softmax', 'softplus']
54 gamma = 0.1 # 超参数
55 num = 4 # 小数点后保留几位
56 for idx in index:
57     y, z = activation_function(idx, x, gamma)
58     print('-----------------------------------------')
59     print('激活函数为：', idx)
60     print('自己写的函数：\n', np.around(y, num))
61     print('调用内置函数：\n', np.around(z, num))
62 # --------------------画图-----------------------
63 # 手动设置横纵坐标范围
64 plt.xlim([-4, 4])
65 plt.ylim([-1, 4])
66 x = np.linspace(-4, 4, 100, endpoint=True)
67 color = ['green', 'red', 'yellow', 'cyan', 'orangered', 'dodgerblue', 'black', 'pink']
68 ls = ['-', '-', ':', ':', ':', '-', '-', '-']
69 for i in range(len(index)):
70     _, z = activation_function(index[i], torch.tensor(x), gamma)
71     if color[i] == 'yellow':
72         plt.plot(x, z.numpy(), color=color[i], ls=ls[i], lw = 3, label=index[i])
73     else:
74         plt.plot(x, z.numpy(), color=color[i], ls=ls[i], label=index[i])
75 # 添加 y = 1, x = 0 基准线
76 plt.plot([x.min(), x.max()], [1, 1], color = 'gray', ls = '--', alpha = 0.3)
77 plt.plot([0, 0], [-1, 4], color = 'gray', ls = '--', alpha = 0.3)
78 # 添加x轴和y轴标签
79 plt.xlabel('x')
80 plt.ylabel('f(x)')
81 plt.legend(ncol = 1, fontsize='small', facecolor='lavenderblush', edgecolor='black')
82 plt.tight_layout()
83 plt.savefig('Activation Functions.png', bbox_inches='tight', dpi=500)
84 plt.show()
```

3. 结果

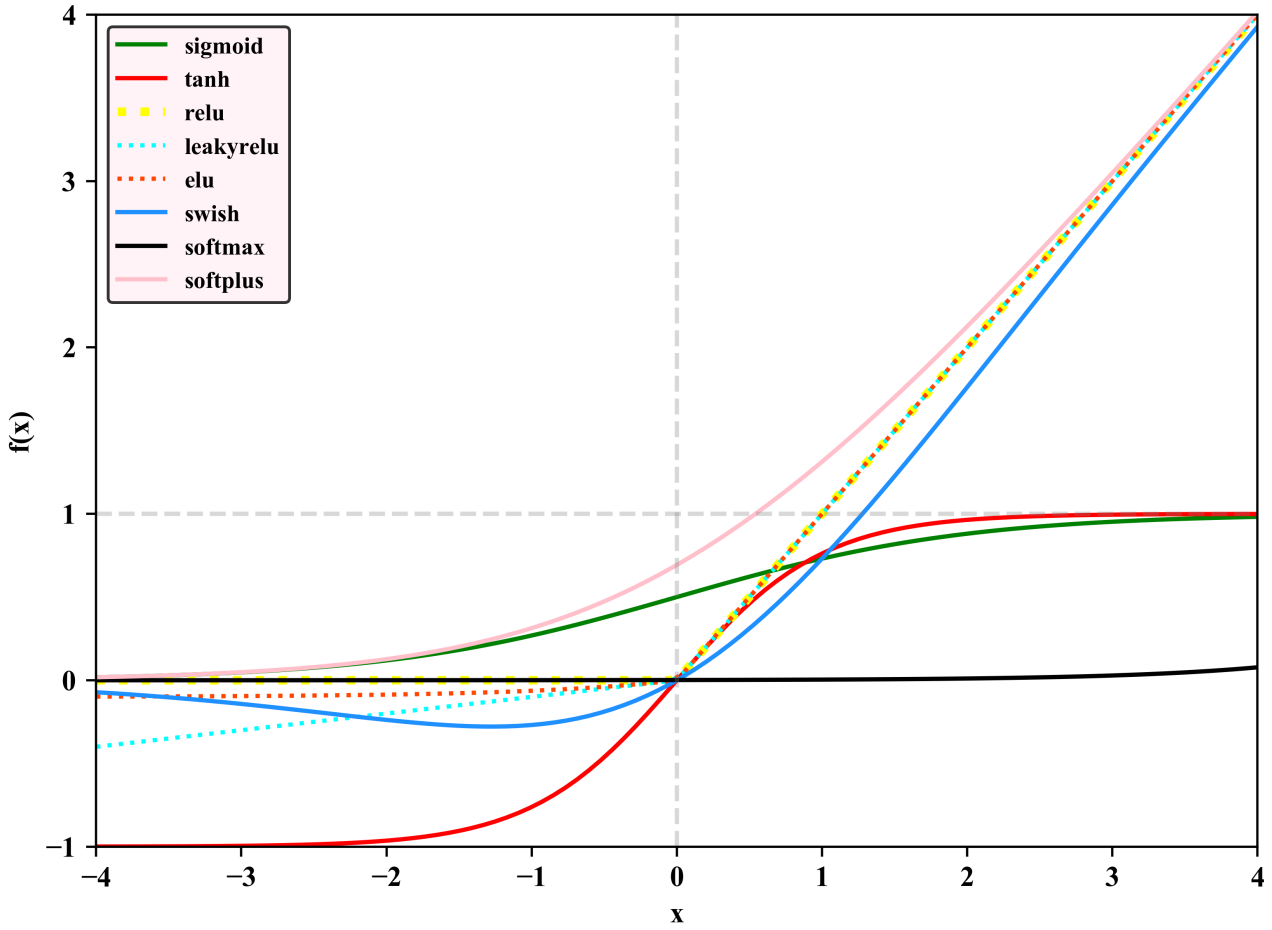
```
D:\ProgramData\Anaconda3\python.exe "D:/Python code/2023.3 exercise/Activation Functions/activation_function_test1.py"
原始数据：
 tensor([[ 0.6614,  0.2669,  0.0617],
        [ 0.6213, -0.4519, -0.1661]])
-----------------------------------------
```

```
激活函数为： sigmoid
自己写的函数：
 tensor([[0.6596, 0.5663, 0.5154],
        [0.6505, 0.3889, 0.4586]])
调用内置函数：
 tensor([[0.6596, 0.5663, 0.5154],
        [0.6505, 0.3889, 0.4586]])
----------------------------------------
激活函数为： tanh
自己写的函数：
 tensor([[ 0.5793,  0.2608,  0.0616],
        [ 0.5520, -0.4235, -0.1646]])
调用内置函数：
 tensor([[ 0.5793,  0.2608,  0.0616],
        [ 0.5520, -0.4235, -0.1646]])
----------------------------------------
激活函数为： relu
自己写的函数：
 tensor([[0.6614, 0.2669, 0.0617],
        [0.6213, 0.0000, 0.0000]])
调用内置函数：
 tensor([[0.6614, 0.2669, 0.0617],
        [0.6213, 0.0000, 0.0000]])
----------------------------------------
激活函数为： leakyrelu
自己写的函数：
 tensor([[ 0.6614,  0.2669,  0.0617],
        [ 0.6213, -0.0452, -0.0166]])
调用内置函数：
 tensor([[ 0.6614,  0.2669,  0.0617],
        [ 0.6213, -0.0452, -0.0166]])
----------------------------------------
激活函数为： elu
自己写的函数：
 tensor([[ 0.6614,  0.2669,  0.0617],
        [ 0.6213, -0.0364, -0.0153]])
调用内置函数：
 tensor([[ 0.6614,  0.2669,  0.0617],
        [ 0.6213, -0.0364, -0.0153]])
----------------------------------------
激活函数为： swish
自己写的函数：
 tensor([[ 0.4362,  0.1512,  0.0318],
        [ 0.4042, -0.1757, -0.0762]])
调用内置函数：
 tensor([[ 0.4362,  0.1512,  0.0318],
        [ 0.4042, -0.1757, -0.0762]])
```

```
----------------------------------------
激活函数为： softmax
自己写的函数：
 tensor([[0.4498, 0.3032, 0.2470],
        [0.5565, 0.1903, 0.2532]])
调用内置函数：
 tensor([[0.4498, 0.3032, 0.2470],
        [0.5565, 0.1903, 0.2532]])
----------------------------------------
激活函数为： softplus
自己写的函数：
 tensor([[1.0775, 0.8355, 0.7245],
        [1.0513, 0.4925, 0.6135]])
调用内置函数：
 tensor([[1.0775, 0.8355, 0.7245],
        [1.0513, 0.4925, 0.6135]])

Process finished with exit code 0
```

可以看到，自己写的激活函数与内置的结果一致。


sigmoid
tanh
relu
leakyrelu
elu
swish
softmax
softplus
f(x)
x
4
3
2
1
0
−1
−4
−3
−2
−1
0
1
2
3
4