In [283…

```
#设计一个单特征值输入的五层的全连接神经网络，激活函数使用ReLU函数，达到对于y=x^2+2x-3的拟合
import numpy as np
neuronNum=[0,3,3,3,3,1]
wAll = {}
wAll[0] = None
bAll = {}
bAll[0] = None
input = {}
input[0] = np.random.randn(1, 1)
input[1] = input[0]
for i in range(2, 6):
    input[i] = np.ones((3, 1))
outputAll = {}
outputAll[0] = input[0]
linearAll = {}
linearAll[0] = None
dwAll = {}
dbAll = {}
dlinearAll = {}
dLineardwAll = {}
dLineardbAll = {}
dback = {}
```

In [239…

```
# 初始化每层的参数
def initialize(layerNum, neuronNum, shape):
    # 因为使用了ReLU函数，所以每一层的参数只有线性函数计算时使用到的系数w和偏置b
    # 因为这里的输入层输入只有x一个特征，所以每个隐层神经元和输出层神经元处理后也只能得到一个输出值
    # 所以输入层、输出层和隐层所接收到的都是一个一维向量，只不过对于输入层来说，这个一维向量只有一个分量
    # 所以初始化参数时根据每一层的神经元数量与上一层输出的一维向量的维数就可以确定这一层需要初始化的参数的维数，使用numpy.random.ra
    w = np.random.randn(neuronNum, shape)
    wAll[layerNum] = w
    b = np.random.randn(neuronNum, 1)
    bAll[layerNum] = b
    return w, b


# 计算线性函数
def linear(w, b, input):
    linear = np.dot(w, input)+b
    return linear


# 计算ReLU函数
def ReLU(intermedium):
    ReLU = np.maximum(intermedium, 0)
    return ReLU
```

In [240…

```
# 计算损失函数-->目标为最小化这个函数
def origin():
    output = np.dot(input[1], input[1])+2*input[1]-3
    np.dot(input[1], input[1])+2*input[1]-3
    return output


def cost(outputNN):
    ori=origin()
    cost = np.sqrt(outputNN-origin())
    print(outputNN)
    print(ori)
    if outputNN>ori:
        dcost=2*(outputNN-ori)
    else:
        dcost=2*(ori-outputNN)*(-1)
    return cost,dcost
```

In [241…

```
# 计算每一层的正向计算-->求线性函数与ReLU函数处理后得到的值
def forward(neuronNum):
    # 第一层
    # 初始化第一层的参数
    for i in range(1,5):
        w, b = initialize(i,neuronNum[i], input[i].shape[0])
        wAll[i]=w
        bAll[i]=b
        liout = linear(w, b, input[i])
        linearAll[i] = liout
        output = ReLU(liout)
        print(f"第{i+1}层的输出为{output}")
        outputAll[i] = output
        input[i+1]=output
    # 此时i为4
    w, b = initialize(i+1, neuronNum[i+1], input[i+1].shape[0])
    wAll[i+1]=w
    bAll[i+1]=b
```

```python
    liout = linear(w, b, input[i+1])
    linearAll[i+1] = liout
    output = ReLU(liout)
    print(f"第{i+1}层的输出为{output}")
    outputAll[i+1] = output
    costNN,dcost=cost(output)
    print(f"此轮迭代的损失为{costNN}")
    return output,costNN,dcost
```

In [242…

```python
# 计算每一层的反向计算-->求梯度
def backward(dcost):
    # 第五层
    # 先对ReLU求梯度

    i=5
    dlinear = outputAll[i]
    dlinear[dlinear < 0] = 0
    dlinear[dlinear > 0] = 1
    dlinearAll[i] = dlinear
    dlinear=dlinear*dcost
    # 然后对线性求梯度
    dLineardw = input[i]
    dLineardwAll[i] = dLineardw
    dLineardb = np.ones((input[i].shape[0], 1))
    dLineardbAll[i] = dLineardb
    # 然后利用链式法则求梯度
    dw = dlinear*dLineardw
    db = dlinear*dLineardb
    dwAll[i] = dw
    dbAll[i] = db
    dx=wAll[i]
    dback[i]=dlinear*dx
    i-=1
    for j in range(i,0,-1):
        # 下一层反向到这一层的时候需要乘一个dfundamental，这是整个模型的最终输出对上一层的输入求得的梯度，维度是3*1
        dfundamention=dback[j+1]
        dlinear = outputAll[j]
        dlinear[dlinear<0]=0
        dlinear[dlinear>0]=1
        # dlinear是该层的输出对ReLU进行求梯度，维度是3*1，和dfundamental作逐元素相乘
        dlinearAll[j]=dlinear*dfundamention
        # 然后对线性求梯度
        # dlineardw 是该层的输出对该层的w进行求梯度
        dLineardw=input[j]
        dLineardwAll[j]=dLineardw
        # dlineardb 是该层的输出对该层的b进行求梯度
        dLineardb=np.ones((input[j].shape[0],1))
        dLineardbAll[j]=dLineardb
        # 然后利用链式法则求梯度
        # dlineardw 是模型的最终输出对该层的b进行求梯度
        dw=dlinear*dLineardw
        # dlineardb 是模型的最终输出对该层的b进行求梯度
        db=dlinear*dLineardb
        # 将该层求得的梯度放入散列表中
        dwAll[j]=dw
        dbAll[j]=db
        # 算出最终模型输出对该层的x的梯度，作为下一层的dfundamental
        dback[j]=dlinear*wAll[j]
    return dwAll,dbAll
```

In [280…

```python
# 定义一个手写神经网络类
def NnInHand(input,epoches):
    for i in range(epoches):
        print(f"第{i+1}次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
        # 正向传播
        outputNN,costNN,dcost = forward(neuronNum)
        print(f"给模型最终输出为{outputNN}")
        # 逆向传播
        dwAll,dbAll=backward(dcost)
        print("各层参数的梯度为")
        print(dwAll)
        print(dbAll)
        # 以最小化损失为目的，迭代更新参数
        for j in range(1,6):
            print(f"第{j}层参数更新")
            wAll[j]=wAll[j]-a*dwAll[j]
            bAll[j]=bAll[j]-a*dbAll[j]
    # 计算模型的准确率
    print(f"acc:{(1-abs(costNN/origin()))*100}%")
    # 计算模型的损失
    print(f"loss:{abs((costNN/origin())*100)}%")
    return None
```

In [284…

```python
if __name__ == "__main__":
```

```
    a=0.1
    epochs=50
    NnInHand(input, epochs)
```

```
第1次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.65265707]
 [1.27717213]
 [0.        ]]
第3层的输出为[[0.52718176]
 [2.14713144]
 [0.        ]]
第4层的输出为[[0.        ]
 [0.        ]
 [2.42751647]]
第5层的输出为[[0.        ]
 [4.17754081]
 [0.44236134]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [0.        ],
       [2.42751647]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.65265707],
       [1.27717213],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[1.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第2次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.        ]
 [0.54998947]]
第3层的输出为[[0.        ]
 [0.        ]
 [0.14842399]]
第4层的输出为[[0.      ]
 [1.533327]
 [0.      ]]
第5层的输出为[[0.        ]
 [0.87947544]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.      ],
       [1.533327],
       [0.      ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [0.        ],
       [0.54998947]]), 1: array([[-0.        ],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[0.],
       [0.],
```

```
       [1.]]), 1: array([[0.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第3次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.02826995]
 [0.74756215]
 [0.        ]]
第3层的输出为[[0.       ]
 [0.9044665]
 [0.       ]]
第4层的输出为[[0.        ]
 [1.60684353]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.89807078]
 [1.51298252]]
第5层的输出为[[0.58891842]]
[[0.58891842]]
[[-3.96337637]]
此轮迭代的损失为[[2.13361074]]
给模型最终输出为[[0.58891842]]
各层参数的梯度为
{5: array([[ 0.        ],
       [ 8.17656584],
       [13.77508484]]), 4: array([[0.        ],
       [1.60684353],
       [0.        ]]), 3: array([[0.       ],
       [0.9044665],
       [0.       ]]), 2: array([[0.        ],
       [0.74756215],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[9.10458957],
       [9.10458957],
       [9.10458957]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第4次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.04270058]
 [0.36982724]
 [0.        ]]
第3层的输出为[[0.14061922]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.        ]
 [0.        ]
 [0.23459006]]
第5层的输出为[[1.02338486]
 [0.77839353]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.04270058],
       [0.        ],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [1.],
       [0.]]), 3: array([[0.],
```

```
       [0.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第5次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[2.12425946]
 [1.21362938]
 [1.70024782]]
第3层的输出为[[2.17785642]
 [0.        ]
 [0.39115245]]
第4层的输出为[[3.88879871]
 [4.60587049]
 [1.37666712]]
第5层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.86138393]]
[[0.86138393]]
[[-3.96337637]]
此轮迭代的损失为[[2.1965337]]
给模型最终输出为[[0.86138393]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[2.17785642],
       [0.        ],
       [0.39115245]]), 2: array([[2.12425946],
       [0.        ],
       [1.70024782]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[9.64952059],
       [9.64952059],
       [9.64952059]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第6次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[3.5486165]
 [0.       ]
 [0.1044351]]
第3层的输出为[[2.31518275]
 [1.52006575]
 [0.        ]]
第4层的输出为[[0.        ]
 [0.        ]
 [0.59091005]]
第5层的输出为[[0.        ]
 [0.08365987]
 [1.48600369]]
第5层的输出为[[0.88050189]]
[[0.88050189]]
[[-3.96337637]]
此轮迭代的损失为[[2.20088125]]
给模型最终输出为[[0.88050189]]
各层参数的梯度为
{5: array([[ 0.        ],
       [ 0.81047648],
       [14.39604189]]), 4: array([[0.        ],
       [0.        ],
       [0.59091005]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[3.5486165],
       [0.       ],
       [0.       ]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[9.68775651],
       [9.68775651],
```

```
       [9.68775651]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[1.],
       [1.],
       [0.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第7次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.24461639]
 [0.        ]
 [0.46994919]]
第3层的输出为[[0.55405445]
 [1.79526227]
 [1.6434993 ]]
第4层的输出为[[0.       ]
 [3.3073281]
 [0.       ]]
第5层的输出为[[0.       ]
 [0.       ]
 [0.4581452]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.        ],
       [1.79526227],
       [0.        ]]), 2: array([[0.24461639],
       [0.        ],
       [0.46994919]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第8次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.13422819]
 [2.37231005]
 [0.86634597]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.        ]
 [0.40074023]
 [0.        ]]
第5层的输出为[[0.       ]
 [1.7006244]
 [0.       ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [0.40074023],
       [0.        ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.80862698],
```

```
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第9次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.00850372]
 [0.        ]
 [0.30757644]]
第3层的输出为[[0.        ]
 [1.19522537]
 [0.        ]]
第4层的输出为[[0.43722953]
 [0.        ]
 [1.16738857]]
第5层的输出为[[1.72947944]
 [0.        ]
 [2.80707395]]
第5层的输出为[[4.22439445]]
[[4.22439445]]
[[-3.96337637]]
此轮迭代的损失为[[2.86142811]]
给模型最终输出为[[4.22439445]]
各层参数的梯度为
{5: array([[28.32116264],
       [ 0.        ],
       [45.96735627]]), 4: array([[0.43722953],
       [0.        ],
       [1.16738857]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[16.37554164],
       [16.37554164],
       [16.37554164]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第10次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.20877394]
 [0.        ]]
第3层的输出为[[3.12980187]
 [0.10954587]
 [0.00758449]]
第4层的输出为[[0.        ]
 [0.        ]
 [4.25708438]]
第5层的输出为[[2.05499823]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.        ],
       [0.        ],
       [0.00758449]]), 2: array([[0.        ],
```

```
       [0.20877394],
       [0.        ]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第11次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [4.31038673]
 [0.        ]]
第3层的输出为[[0.        ]
 [0.        ]
 [6.51178405]]
第4层的输出为[[0.        ]
 [0.        ]
 [1.90580425]]
第5层的输出为[[0.        ]
 [0.        ]
 [0.98435252]]
第5层的输出为[[1.219999]]
[[1.219999]]
[[-3.96337637]]
此轮迭代的损失为[[2.27670274]]
给模型最终输出为[[1.219999]]
各层参数的梯度为
{5: array([[ 0.        ],
       [ 0.        ],
       [10.20453725]]), 4: array([[0.        ],
       [0.        ],
       [1.90580425]]), 3: array([[0.        ],
       [0.        ],
       [6.51178405]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[10.36675073],
       [10.36675073],
       [10.36675073]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[0.],
       [0.],
       [1.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第12次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[2.22301743]
 [0.07957082]
 [1.20071245]]
第3层的输出为[[1.53632428]
 [1.65475139]
 [0.        ]]
第4层的输出为[[0.        ]
 [3.01607774]
 [1.76687707]]
第5层的输出为[[1.23438629]
 [0.        ]
 [0.        ]]
第5层的输出为[[1.01651027]]
[[1.01651027]]
[[-3.96337637]]
此轮迭代的损失为[[2.23156596]]
给模型最终输出为[[1.01651027]]
各层参数的梯度为
{5: array([[12.29420757],
       [ 0.        ],
       [ 0.        ]]), 4: array([[0.],
       [0.],
```

```
       [0.]]), 3: array([[0.        ],
       [1.65475139],
       [0.        ]]), 2: array([[2.22301743],
       [0.07957082],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[9.95977327],
       [9.95977327],
       [9.95977327]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[0.],
       [1.],
       [1.]]), 2: array([[1.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第13次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.66332184]
 [0.        ]
 [0.96841264]]
第3层的输出为[[0.20811301]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.25758464]
 [0.77417598]
 [1.6023468 ]]
第5层的输出为[[1.19809457]
 [1.5964179 ]
 [5.14738338]]
第5层的输出为[[0.44361312]]
[[0.44361312]]
[[-3.96337637]]
此轮迭代的损失为[[2.09928309]]
给模型最终输出为[[0.44361312]]
各层参数的梯度为
{5: array([[10.55998038],
       [14.07079379],
       [45.36892884]]), 4: array([[0.25758464],
       [0.77417598],
       [1.6023468 ]]), 3: array([[0.20811301],
       [0.        ],
       [0.        ]]), 2: array([[1.66332184],
       [0.        ],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[8.81397897],
       [8.81397897],
       [8.81397897]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第14次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.65500481]
 [3.38367982]
 [2.59043317]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.12471537]
 [0.        ]
 [2.66073189]]
第5层的输出为[[3.44307782]]
[[3.44307782]]
[[-3.96337637]]
此轮迭代的损失为[[2.72148015]]
给模型最终输出为[[3.44307782]]
各层参数的梯度为
{5: array([[ 1.84739736],
```

```
       [ 0.        ],
       [39.41317777]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[14.81290838],
       [14.81290838],
       [14.81290838]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第15次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [1.12084551]
 [0.        ]]
第3层的输出为[[2.83432274]
 [1.23293096]
 [1.44842388]]
第4层的输出为[[0.        ]
 [4.90860052]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.520663  ]
 [1.07083456]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [4.90860052],
       [0.        ]]), 3: array([[0.        ],
       [1.23293096],
       [0.        ]]), 2: array([[0.        ],
       [1.12084551],
       [0.        ]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第16次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.82507937]
 [0.        ]
 [1.1069503 ]]
第3层的输出为[[0.        ]
 [2.24699892]
 [0.        ]]
第4层的输出为[[2.79160321]
 [0.        ]
 [1.44030984]]
第5层的输出为[[4.20355544]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
```

```
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[2.79160321],
       [0.        ],
       [0.        ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第17次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.59732984]
 [1.33656896]
 [0.40079913]]
第3层的输出为[[3.52955235]
 [0.34741012]
 [0.        ]]
第4层的输出为[[2.06636216]
 [0.        ]
 [1.66487733]]
第5层的输出为[[4.26747149]
 [2.49561226]
 [1.10720597]]
第5层的输出为[[6.96943272]]
[[6.96943272]]
[[-3.96337637]]
此轮迭代的损失为[[3.30647986]]
给模型最终输出为[[6.96943272]]
各层参数的梯度为
{5: array([[93.31090214],
       [54.56810469],
       [24.209743  ]]), 4: array([[2.06636216],
       [0.        ],
       [1.66487733]]), 3: array([[3.52955235],
       [0.        ],
       [0.        ]]), 2: array([[0.59732984],
       [1.33656896],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[21.86561817],
       [21.86561817],
       [21.86561817]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[1.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第18次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.88294673]
 [0.        ]
 [0.        ]]
第3层的输出为[[0.64309816]
 [2.56880092]
 [2.63418214]]
第4层的输出为[[3.78636758]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.17542636]
 [0.        ]
 [0.        ]]
第5层的输出为[[1.63113814]]
```

```
[[1.63113814]]
[[-3.96337637]]
此轮迭代的损失为[[2.36527261]]
给模型最终输出为[[1.63113814]]
各层参数的梯度为
{5: array([[1.96285059],
       [0.        ],
       [0.        ]]), 4: array([[3.78636758],
       [0.        ],
       [0.        ]]), 3: array([[0.64309816],
       [0.        ],
       [0.        ]]), 2: array([[0.88294673],
       [0.        ],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.        ]])}
{5: array([[11.18902902],
       [11.18902902],
       [11.18902902]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[1.],
       [0.],
       [0.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[1.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第19次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.12819962]
 [1.80658922]]
第3层的输出为[[0.        ]
 [2.8536567 ]
 [0.03621139]]
第4层的输出为[[0.       ]
 [0.2124579]
 [0.       ]]
第5层的输出为[[0.39502499]
 [0.21807491]
 [1.7990837 ]]
第5层的输出为[[1.25118422]]
[[1.25118422]]
[[-3.96337637]]
此轮迭代的损失为[[2.28354124]]
给模型最终输出为[[1.25118422]]
各层参数的梯度为
{5: array([[ 4.11976344],
       [ 2.27432962],
       [18.76286193]]), 4: array([[0.       ],
       [0.2124579],
       [0.       ]]), 3: array([[0.       ],
       [2.8536567],
       [0.       ]]), 2: array([[0.        ],
       [0.12819962],
       [1.80658922]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[10.42912117],
       [10.42912117],
       [10.42912117]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[0.],
       [1.],
       [1.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第20次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.90650628]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.        ]
 [0.67935438]
 [1.1739777 ]]
第5层的输出为[[0.        ]
```

```
 [0.        ]
 [0.05407307]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.       ],
       [0.       ],
       [1.1739777]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[0.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[0.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第21次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [1.14334116]
 [0.        ]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.20503765]
 [0.        ]
 [0.05386754]]
第5层的输出为[[1.68164733]
 [0.86261261]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.20503765],
       [0.        ],
       [0.        ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [1.],
       [0.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第22次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.4977868 ]
 [3.41127744]]
第3层的输出为[[1.10233381]
 [0.        ]
 [0.        ]]
第4层的输出为[[1.8752031 ]
```

```
 [0.89390551]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.66705449]
 [2.47565234]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [0.89390551],
       [0.        ]]), 3: array([[1.10233381],
       [0.        ],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[1.],
       [1.],
       [0.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第23次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.31177837]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.11510404]
 [1.0949718 ]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.        ]
 [0.45166859]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.31177837],
       [0.        ],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[1.],
       [1.],
       [0.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[0.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第24次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.        ]
```

```
 [1.30244042]
 [1.09649149]]
第4层的输出为[[0.55678508]
 [0.        ]
 [0.2204622 ]]
第5层的输出为[[0.11829046]
 [0.        ]
 [0.00454529]]
第5层的输出为[[0.87193949]]
[[0.87193949]]
[[-3.96337637]]
此轮迭代的损失为[[2.19893516]]
给模型最终输出为[[0.87193949]]
各层参数的梯度为
{5: array([[1.14394343],
       [0.        ],
       [0.04395587]]), 4: array([[0.55678508],
       [0.        ],
       [0.2204622 ]]), 3: array([[0.        ],
       [0.        ],
       [1.09649149]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[9.67063172],
       [9.67063172],
       [9.67063172]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [1.],
       [1.]]), 1: array([[0.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第25次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.        ]
 [0.24605854]]
第3层的输出为[[0.        ]
 [0.        ]
 [1.70872329]]
第4层的输出为[[3.29422232]
 [3.78000403]
 [0.        ]]
第5层的输出为[[0.        ]
 [2.3121805 ]
 [5.79571435]]
第5层的输出为[[2.8838275]]
[[2.8838275]]
[[-3.96337637]]
此轮迭代的损失为[[2.61671624]]
给模型最终输出为[[2.8838275]]
各层参数的梯度为
{5: array([[ 0.        ],
       [31.66394257],
       [79.36887547]]), 4: array([[0.        ],
       [3.78000403],
       [0.        ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [0.        ],
       [0.24605854]]), 1: array([[-0.        ],
       [-0.        ],
       [-0.80862698]])}
{5: array([[13.69440773],
       [13.69440773],
       [13.69440773]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[1.],
       [1.],
       [0.]]), 2: array([[0.],
       [0.],
       [1.]]), 1: array([[0.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第26次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.05121923]
```

```
 [0.        ]
 [2.17539496]]
第3层的输出为[[0.16807679]
 [5.10207906]
 [2.10818046]]
第4层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.        ]
 [0.        ]
 [0.10616571]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[1.05121923],
       [0.        ],
       [2.17539496]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第27次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.99842359]
 [0.        ]]
第3层的输出为[[1.4912695 ]
 [0.71768394]
 [3.31898385]]
第4层的输出为[[0.        ]
 [2.87360674]
 [0.33891109]]
第5层的输出为[[3.03012076]
 [0.        ]
 [5.29198615]]
第5层的输出为[[6.4621258]]
[[6.4621258]]
[[-3.96337637]]
此轮迭代的损失为[[3.22885462]]
给模型最终输出为[[6.4621258]]
各层参数的梯度为
{5: array([[ 63.18106108],
       [  0.        ],
       [110.34322609]]), 4: array([[0.        ],
       [0.        ],
       [0.33891109]]), 3: array([[0.        ],
       [0.71768394],
       [3.31898385]]), 2: array([[0.        ],
       [0.99842359],
       [0.        ]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[20.85100434],
       [20.85100434],
       [20.85100434]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[0.],
       [1.],
       [1.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
```

```
第5层参数更新
第28次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.33461766]
 [0.        ]
 [0.        ]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.        ]
 [0.        ]
 [2.74293532]]
第5层的输出为[[0.        ]
 [3.55912972]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[1.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第29次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.66795551]
 [1.12962712]
 [0.        ]]
第3层的输出为[[1.97088625]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.04554158]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.54150765]
 [2.48132058]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.97088625],
       [0.        ],
       [0.        ]]), 2: array([[1.66795551],
       [0.        ],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[1.],
       [0.],
       [0.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
```

```
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第30次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.6060975 ]
 [0.82286578]
 [0.        ]]
第4层的输出为[[1.07907744]
 [0.80395075]
 [0.6573588 ]]
第5层的输出为[[3.33957027]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.07907744],
       [0.        ],
       [0.        ]]), 3: array([[0.6060975 ],
       [0.82286578],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[1.],
       [1.],
       [0.]]), 1: array([[0.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第31次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.        ]
 [0.        ]
 [1.22930935]]
第5层的输出为[[0.        ]
 [1.20621987]
 [0.        ]]
第5层的输出为[[0.31475925]]
[[0.31475925]]
[[-3.96337637]]
此轮迭代的损失为[[2.06836545]]
给模型最终输出为[[0.31475925]]
各层参数的梯度为
{5: array([[ 0.        ],
       [10.32074437],
       [ 0.        ]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[8.55627124],
       [8.55627124],
       [8.55627124]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[0.],
```

```
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第32次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.69486919]
 [1.58974397]]
第3层的输出为[[0.44536272]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.5507804 ]
 [0.16217225]
 [0.59293108]]
第5层的输出为[[1.21536077]
 [1.15795389]
 [0.99470316]]
第5层的输出为[[0.46969886]]
[[0.46969886]]
[[-3.96337637]]
此轮迭代的损失为[[2.10548693]]
给模型最终输出为[[0.46969886]]
各层参数的梯度为
{5: array([[10.77557142],
       [10.26659338],
       [ 8.81918786]]), 4: array([[0.5507804 ],
       [0.16217225],
       [0.59293108]]), 3: array([[0.44536272],
       [0.        ],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[8.86615045],
       [8.86615045],
       [8.86615045]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第33次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.       ]
 [0.       ]
 [1.2158617]]
第3层的输出为[[0.        ]
 [0.        ]
 [0.70011771]]
第4层的输出为[[0.97926711]
 [0.65394918]
 [2.0606312 ]]
第5层的输出为[[2.0797557]
 [0.       ]
 [0.       ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.97926711],
       [0.        ],
       [0.        ]]), 3: array([[0.        ],
       [0.        ],
       [0.70011771]]), 2: array([[0.       ],
       [0.       ],
       [1.2158617]]), 1: array([[-0.        ],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[1.],
       [1.],
```

```
       [1.]]), 2: array([[0.],
       [0.],
       [1.]]), 1: array([[0.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第34次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.23867456]
 [0.84264748]
 [0.        ]]
第3层的输出为[[0.        ]
 [0.85133926]
 [1.07069962]]
第4层的输出为[[0.01616933]
 [0.        ]
 [0.4947304 ]]
第5层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.        ],
       [0.        ],
       [1.07069962]]), 2: array([[0.        ],
       [0.84264748],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [1.],
       [1.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第35次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.65226836]
 [1.34899492]]
第3层的输出为[[0.        ]
 [0.53678577]
 [0.        ]]
第4层的输出为[[0.13866188]
 [0.05018642]
 [0.26648347]]
第5层的输出为[[0.36678147]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.50245177]]
[[0.50245177]]
[[-3.96337637]]
此轮迭代的损失为[[2.11325061]]
给模型最终输出为[[0.50245177]]
各层参数的梯度为
{5: array([[3.27596604],
       [0.        ],
       [0.        ]]), 4: array([[0.13866188],
       [0.        ],
       [0.        ]]), 3: array([[0.        ],
       [0.53678577],
       [0.        ]]), 2: array([[0.        ],
       [0.65226836],
       [0.        ]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[8.93165628],
       [8.93165628],
       [8.93165628]]), 4: array([[1.],
```

```
       [0.],
       [0.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第36次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.02318986]
 [0.        ]]
第3层的输出为[[1.91200665]
 [0.        ]
 [0.        ]]
第4层的输出为[[1.60598945]
 [2.85761927]
 [1.40170481]]
第5层的输出为[[0.        ]
 [1.64907486]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [2.85761927],
       [0.        ]]), 3: array([[1.91200665],
       [0.        ],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第37次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.12024781]
 [0.11919019]]
第3层的输出为[[1.25193571]
 [0.        ]
 [0.55900323]]
第4层的输出为[[1.04418416]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.34990675]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.04418416],
       [0.        ],
       [0.        ]]), 3: array([[1.25193571],
       [0.        ],
       [0.        ]]), 2: array([[0.        ],
       [0.        ],
       [0.11919019]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
```

```
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[1.],
       [0.],
       [0.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第38次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.63395856]
 [0.        ]
 [0.        ]]
第3层的输出为[[0.80770525]
 [0.        ]
 [0.        ]]
第4层的输出为[[0.        ]
 [1.21608906]
 [0.59088575]]
第5层的输出为[[0.0404896 ]
 [1.4017177 ]
 [2.85454612]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [1.21608906],
       [0.59088575]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[1.63395856],
       [0.        ],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[0.],
       [1.],
       [1.]]), 2: array([[1.],
       [0.],
       [0.]]), 1: array([[1.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第39次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.56061104]
 [0.        ]
 [2.93699227]]
第3层的输出为[[0.        ]
 [0.        ]
 [0.41675341]]
第4层的输出为[[0.05789153]
 [0.35612935]
 [0.42226636]]
第5层的输出为[[0.24563996]
 [0.        ]
 [0.97957294]]
第5层的输出为[[3.49669646]]
[[3.49669646]]
[[-3.96337637]]
此轮迭代的损失为[[2.73131339]]
给模型最终输出为[[3.49669646]]
各层参数的梯度为
{5: array([[ 3.66498391],
       [ 0.        ],
       [14.61537095]]), 4: array([[0.05789153],
       [0.        ],
       [0.42226636]]), 3: array([[0.        ],
       [0.        ],
       [0.41675341]]), 2: array([[0.        ],
       [0.        ],
```

```
       [2.93699227]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[14.92014565],
       [14.92014565],
       [14.92014565]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[1.],
       [1.],
       [1.]]), 2: array([[0.],
       [0.],
       [1.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第40次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.        ]
 [0.50445988]]
第3层的输出为[[3.09305925]
 [0.        ]
 [0.09894463]]
第4层的输出为[[0.        ]
 [5.89021224]
 [0.        ]]
第5层的输出为[[7.85851229]
 [0.        ]
 [0.28883932]]
第5层的输出为[[10.26656039]]
[[10.26656039]]
[[-3.96337637]]
此轮迭代的损失为[[3.77225884]]
给模型最终输出为[[10.26656039]]
各层参数的梯度为
{5: array([[223.65226572],
       [  0.        ],
       [  8.22033043]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [0.        ],
       [0.50445988]]), 1: array([[-0.        ],
       [-0.        ],
       [-0.80862698]])}
{5: array([[28.45987351],
       [28.45987351],
       [28.45987351]]), 4: array([[1.],
       [0.],
       [1.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[0.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第41次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.09008884]
 [0.        ]]
第3层的输出为[[1.20070279]
 [0.        ]
 [0.74333436]]
第4层的输出为[[1.14647587]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.        ]
 [0.4363217 ]
 [1.27897393]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.20070279],
```

```
       [0.        ],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [1.]]), 3: array([[1.],
       [0.],
       [0.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第42次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.27224164]
 [0.52432733]
 [0.        ]]
第3层的输出为[[0.        ]
 [2.30671354]
 [0.        ]]
第4层的输出为[[1.11550214]
 [0.        ]
 [0.        ]]
第5层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.51358137]]
[[0.51358137]]
[[-3.96337637]]
此轮迭代的损失为[[2.11588226]]
给模型最终输出为[[0.51358137]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [0.52432733],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[8.95391547],
       [8.95391547],
       [8.95391547]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.],
       [0.],
       [0.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第43次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.07257493]
 [0.26650392]]
第3层的输出为[[0.24307457]
 [0.        ]
 [1.75951949]]
第4层的输出为[[0.        ]
 [0.        ]
 [0.14604007]]
第5层的输出为[[0.01994264]
 [0.85754564]
 [0.11354152]]
第5层的输出为[[0.26701141]]
[[0.26701141]]
[[-3.96337637]]
此轮迭代的损失为[[2.05679065]]
给模型最终输出为[[0.26701141]]
各层参数的梯度为
{5: array([[0.16873024],
       [7.25550121],
```

```
       [0.96064931]]), 4: array([[0.        ],
       [0.        ],
       [0.14604007]]), 3: array([[0.        ],
       [0.        ],
       [1.75951949]]), 2: array([[0.        ],
       [0.        ],
       [0.26650392]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[8.46077556],
       [8.46077556],
       [8.46077556]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[0.],
       [0.],
       [1.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[0.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第44次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.08484275]
 [0.        ]
 [0.68276756]]
第3层的输出为[[0.        ]
 [0.        ]
 [0.36121774]]
第4层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.53253974]
 [0.61157711]
 [0.99459633]]
第5层的输出为[[0.31519834]]
[[0.31519834]]
[[-3.96337637]]
此轮迭代的损失为[[2.06847159]]
给模型最终输出为[[0.31519834]]
各层参数的梯度为
{5: array([[4.5570221 ],
       [5.23335668],
       [8.51090942]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [0.        ],
       [0.68276756]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[8.55714942],
       [8.55714942],
       [8.55714942]]), 4: array([[1.],
       [1.],
       [1.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [1.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第45次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.78877549]
 [0.20172409]
 [0.23782857]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[1.34670015]
 [0.        ]
 [0.01436491]]
第5层的输出为[[0.        ]
 [0.        ]
 [0.26251563]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
```

```
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [0.        ],
       [0.01436491]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [1.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[1.],
       [1.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第46次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [0.        ]
 [1.01704235]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.        ]
 [2.02219293]
 [1.7645464 ]]
第5层的输出为[[4.50336766]
 [0.        ]
 [0.        ]]
第5层的输出为[[13.2555245]]
[[13.2555245]]
[[-3.96337637]]
此轮迭代的损失为[[4.14956635]]
给模型最终输出为[[13.2555245]]
各层参数的梯度为
{5: array([[155.08608281],
       [  0.        ],
       [  0.        ]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.        ],
       [-0.80862698]])}
{5: array([[34.43780174],
       [34.43780174],
       [34.43780174]]), 4: array([[1.],
       [0.],
       [0.]]), 3: array([[0.],
       [1.],
       [1.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[0.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第47次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.]
 [0.]
 [0.]]
第3层的输出为[[0.7065216 ]
 [0.46275826]
 [0.3002268 ]]
第4层的输出为[[0.70196322]
 [0.33559419]
 [0.        ]]
第5层的输出为[[0.]
 [0.]
 [0.]]
第5层的输出为[[0.]]
[[0.]]
```

```
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.7065216 ],
       [0.46275826],
       [0.        ]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.],
       [-0.],
       [-0.]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[1.],
       [1.],
       [0.]]), 2: array([[1.],
       [1.],
       [1.]]), 1: array([[0.],
       [0.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第48次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.09023462]
 [0.        ]
 [1.89005833]]
第3层的输出为[[2.08038112]
 [0.        ]
 [1.3898352 ]]
第4层的输出为[[0.       ]
 [3.3200196]
 [0.       ]]
第5层的输出为[[1.39190028]
 [2.13612863]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.       ],
       [3.3200196],
       [0.       ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.09023462],
       [0.        ],
       [1.89005833]]), 1: array([[-0.80862698],
       [-0.        ],
       [-0.80862698]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[1.],
       [1.],
       [0.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[1.],
       [0.],
       [1.]]), 1: array([[1.],
       [0.],
       [1.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第49次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[0.        ]
 [1.17208355]
 [0.        ]]
第3层的输出为[[0.]
 [0.]
 [0.]]
第4层的输出为[[0.        ]
 [0.15477894]
 [0.        ]]
第5层的输出为[[0.        ]
 [1.18602979]
```

```
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.        ],
       [0.15477894],
       [0.        ]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[-0.        ],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[0.],
       [1.],
       [0.]]), 2: array([[0.],
       [0.],
       [0.]]), 1: array([[0.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
第50次迭代>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
第2层的输出为[[1.1431002 ]
 [1.70977084]
 [0.        ]]
第3层的输出为[[0.        ]
 [2.13225533]
 [0.        ]]
第4层的输出为[[1.41456763]
 [0.        ]
 [1.83010239]]
第5层的输出为[[0.        ]
 [5.83105154]
 [0.        ]]
第5层的输出为[[0.]]
[[0.]]
[[-3.96337637]]
此轮迭代的损失为[[1.99082304]]
给模型最终输出为[[0.]]
各层参数的梯度为
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [0.],
       [0.]]), 3: array([[0.],
       [0.],
       [0.]]), 2: array([[0.        ],
       [1.70977084],
       [0.        ]]), 1: array([[-0.80862698],
       [-0.80862698],
       [-0.        ]])}
{5: array([[0.],
       [0.],
       [0.]]), 4: array([[0.],
       [1.],
       [0.]]), 3: array([[1.],
       [0.],
       [1.]]), 2: array([[0.],
       [1.],
       [0.]]), 1: array([[1.],
       [1.],
       [0.]])}
第1层参数更新
第2层参数更新
第3层参数更新
第4层参数更新
第5层参数更新
acc:[[49.76951838]]%
loss:[[50.23048162]]%
```