



深蓝学院  
shenlanxueyuan.com

## 第二章作业思路提示



主讲人 陈彭鑫



- 第一部分：感知机
- 第二部分：神经网络
- 第三部分：手写数字分类器

## 作业 I

现有如下训练数据集，正样本点 $M_1(3,3)$ 和 $M_2(4,3)$ ，负样本点 $M_3(1,1)$ ，请利用感知机算法求感知机模型 $f(x) = \text{sign}(w \cdot x + b)$ 。下方表格中已给出了前4步迭代的步骤，第5步迭代中选择 $M_2$ 为误分类点，请补全后续详细的迭代过程。

迭代次数	误分类点	$w$	$b$	超平面
0		(0, 0)	0	0
1	$M_1$	(3, 3)	1	$3x_1 + 3x_2 + 1$
2	$M_3$	(2, 2)	0	$2 + 2x_2$
3	$M_3$	(1, 1)	-1	$x_1 + x_2 - 1$
4	$M_3$	(0, 0)	-2	-2
5	$M_2$			
...				
N	无			

设直线  $L$  的方程为  $Ax+By+C=0$ ，点  $P$  的坐标为  $(x_0, y_0)$ ，则点  $P$  到直线  $L$  的距离为  $\frac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$

空间中任意一点  $x_i$  到超平面  $S$  的距离：

$$\frac{1}{\|w\|} |w \cdot x_i + b|$$

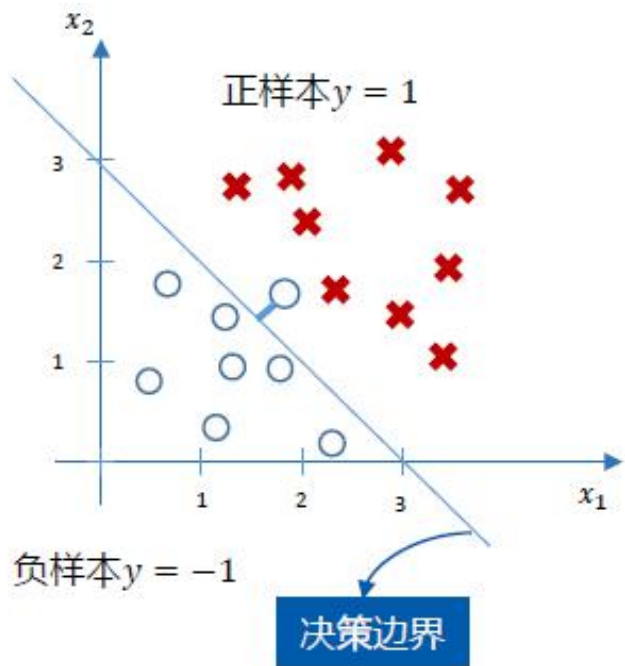
如果一个点是误分类的，

当  $w \cdot x_i + b > 0$  时，  $y_i = -1$ ，

当  $w \cdot x_i + b < 0$  时，  $y_i = +1$ ，

因此，距离又可以表示成：

$$-\frac{1}{\|w\|} y_i (w \cdot x_i + b)$$



# 感知机(算法流程)

## 感知机算法

**输入:** 训练数据集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ ; 学习率  $\eta$ 。

**输出:** 感知机模型参数  $w, b$ , 以及感知机模型  $f(x) = \text{sign}(w \cdot x + b)$

1. 选取初值  $w_0, b_0$ ;
2. 在训练集中选取数据  $(x_i, y_i)$ ;
3. 如果  $y_i(w \cdot x_i + b) \leq 0$ , 更新参数  $w \leftarrow w + \eta y_i x_i$   $b \leftarrow b + \eta y_i$
4. 跳转到第2步, 继续学习直到训练集中没有误分类的点。

```
class Perceptron:
```

```
    def __init__(self, X, y):  
        self.X = X  
        self.y = y
```

```
        self.w = np.array([0, 0]).reshape(2, 1)  
        self.b = 0  
        self.lr = 1          # learning rate  
        self.max_iter = 100  # maximum iteration
```

```
        print('迭代次数 \t 误分类点 \t w \t b \n')  
        print('0 \t N/A \t (%.4f, %.4f) \t %.4f \n' % (self.w[0], self.w[1], self.b))
```

```
    def hypothesis(self, X):  
        return (X @ self.w + self.b).ravel()
```

```
    def train(self):
```

```
        for iter in range(self.max_iter):  
            # search misclassified point  
            H = self.hypothesis(X)  
            misclassified_indices = np.where(self.y * H <= 0)  
            if(misclassified_indices[0].size == 0):  # no misclassification  
                print('没有误分类点, 感知结束\n')  
                break
```

```
            # stochastic gradient descent  
            i = np.random.choice(misclassified_indices[0], 1)  
            xi = X[i, :].reshape(2, 1)
```

```
            # update  
            self.w = self.w + self.lr * self.y[i] * xi  
            self.b = self.b + self.lr * self.y[i]
```

```
        print('%d \t (%.4f, %.4f) \t (%.4f, %.4f) \t %.4f \n' % (iter, xi[0], xi[1], self.w[0], self.w[1], self.b))
```

- 第一部分：感知机
- 第二部分：神经网络
- 第三部分：手写数字分类器

# 手写数字分类器

---

- 第一部分：感知机
- 第二部分：神经网络
- 第三部分：手写数字分类器

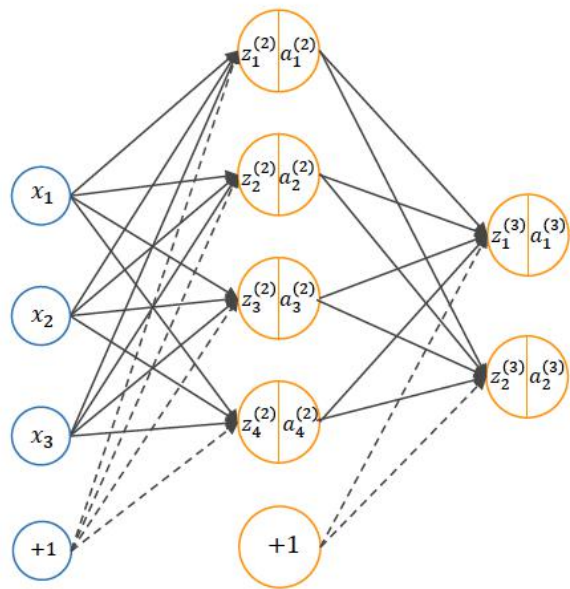


## 作业 III

我们的手写数字识别案例中，多层神经网络模型的参数总共有多少个？如果图片变成了 $256 \times 256$ 大小，那么模型的参数将会发生如何变化？当模型参数过大的时候，有什么措施可以降低计算的复杂度或是提升计算的速度？

# 手写数字分类器

## 3. 神经网络—多层神经网络



输入层

隐藏层

输出层

- $z^{(l)}$  第 $l$ 层神经元的权重和  $\mathbb{R}^{n_l}$   
 $a^{(l)}$  第 $l$ 层神经元输出（经过激励函数）  $\mathbb{R}^{n_l}$   
 $\theta^{(l)}$  从 $l-1$ 层到 $l$ 层的权重参数组  $\mathbb{R}^{n_l \times n_{l-1}}$

总结：第 $l$  ( $2 \leq l \leq L$ ) 层神经元的状态及激活值的向量表示形式为：

$$z^{(l)} = \theta^{(l)} a^{(l-1)} + b^{(l)}$$

$$a^{(l)} = f(z^{(l)})$$

$$a^{(1)} = x$$

前向传播过程：

$$x = a^{(1)} \rightarrow z^{(2)} \rightarrow a^{(2)} \rightarrow \dots \rightarrow a^{(L-1)} \rightarrow z^{(L)} \rightarrow a^{(L)}$$

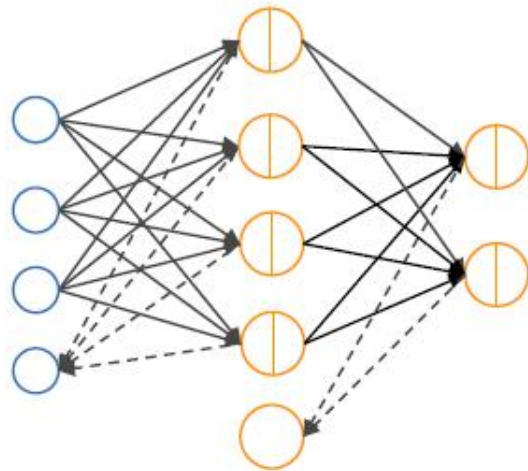


## 4. 反向传播

### BP算法的核心

**输入：** 训练数据集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ ; 学习率  $\mu$ 。

**输出：** 模型权重，偏置



1. 随机初始化输出层和隐藏层的权重、偏置

2. 前向传播:

$$z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)}$$

$$a^{(l)} = f(z^{(l)})$$

3. 计算每一层的误差:

$$\delta_i^{(L)} = -(y_i - a_i^{(L)}) f'(z_i^{(L)}), \quad (1 \leq i \leq n_L)$$

$$\delta_i^{(l)} = \left( \sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \Theta_{ji}^{(l+1)} \right) f'(z_i^{(l)}), \quad (1 \leq i \leq n_l)$$

4. 求各层权重，偏置的偏导:

$$\frac{\partial E}{\partial \Theta_{ij}^{(l)}} = \delta_i^{(l)} a_j^{(l-1)}$$

$$\frac{\partial E}{\partial b_i^{(l)}} = \delta_i^{(l)}$$

5. 更新参数:

$$W^{(l)} = W^{(l)} - \frac{\mu}{N} \sum_{i=1}^N \frac{\partial E_{(i)}}{\partial \Theta^{(l)}}$$

$$b^{(l)} = b^{(l)} - \frac{\mu}{N} \sum_{i=1}^N \frac{\partial E_{(i)}}{\partial b^{(l)}}$$



感谢各位聆听 !

Thanks for Listening

