深蓝学院
shenlanxueyuan.com

深度学习理论与实践

第二课：神经网络

主讲人 郑元春

中科院大数据挖掘与知识管理重点实验室
中科院虚拟经济与数据科学研究中心

知识引入

感知机

神经网络

反向传播

手写数字识别

1. 知识引入-极大似然估计

什么是极大似然估计?

利用已知的样本结果信息（观测值，输入特征），反推最具有可能（概率最大）导致这些样本结果出现的模型参数。

- 假设一：数据独立同分布
- 假设二：分布已知

极大似然估计：模型已经确定，参数未知；对模型的求解就是利用观测值来求解模型的参数。

$$f(x)=\frac{1}{\sqrt{2\pi\sigma}}\exp\left(-\frac{(x-\boxed{\mu})^2}{2\boxed{\sigma}^2}\right)$$

预估参数

当参数求解出来，则整个模型则是确定的

1. 知识引入-极大似然估计

什么是极大似然估计?

对于一个函数 $p(\boxed{x}|\boxed{\theta})$

观测数据 模型参数

- 如果 θ 是已知的，则该函数叫做概率函数(Probability Function)，描述的是不同样本点 x 出现的概率；
- 如果 x 是已知的，则该函数叫做似然函数(Likelihood Function)，描述的是对于不同的模型参数，出现样本点 x 的概率。

1. 知识引入-极大似然估计

案例：估计白球比例

有一个罐子，里面有黑球和白球，其各自数目与比例不知。想知道罐子中黑球和白球的比例，怎么做?

方案：从罐子中随机拿出一个球，记录其颜色并放回，这个抽样过程可以重复N次，通过N次实验记录的小球颜色来估计罐子中黑白球的比例。假如$N = 100$，而且其中有70次是白色球，则罐子中白球所占比例最有可能是多少?

直觉上罐子中白球比例最有可能是0.7
那么怎么证明呢?

1. 知识引入-**极大似然估计**

案例：估计白球比例

假设罐子中白球比例是p，那么黑球的比例就是$1-p$,由于每次是有放回的随机取出一个球，则每个拿出球的颜色这一事件$\{x_1, x_2, \cdots, x_{100}\}$服从**独立同分布假设。**将100次抽样中有70次白球的概率记做$p(O|M)$，则有:

$$
\begin{aligned}
p(O|M) &= p(x_1, x_2, \cdots, x_{100}|M) \\
&= p(x_1|M)p(x_2|M)\cdots p(x_{100}|M) \\
&= \boxed{p}^{70}\,(1-p)^{30}
\end{aligned}
$$

模型参数

不同的p对应不同的分布情况，那么该按照什么方法去选择这个分布呢?

1. 知识引入-极大似然估计

案例：估计白球比例

$p(O|M) = p^{70}\,(1-p)^{30}$

$p(O|M)$是p的函数，使得函数取**极大值**，则样本结果O出现的可能性最大

对$p(O|M)$中的p进行求导，令其导数为0，则可求出$p(O|M)$的极大值。

$$\frac{\partial p(O|M)}{\partial p} = 70p^{69}(1-p)^{30} - 30p^{70}(1-p)^{29} = 0$$

即 $70p^{69}(1-p)^{30} - 30p^{70}(1-p)^{29}$

$= p^{69}(1-p)^{29}[70(1-p) - 30p]$

$= 0$

得 $p = 0.7$

$p = 0$和$p = 1$也是极值点，为什么不使用这两个点

1. 知识引入-逻辑回归

逻辑回归

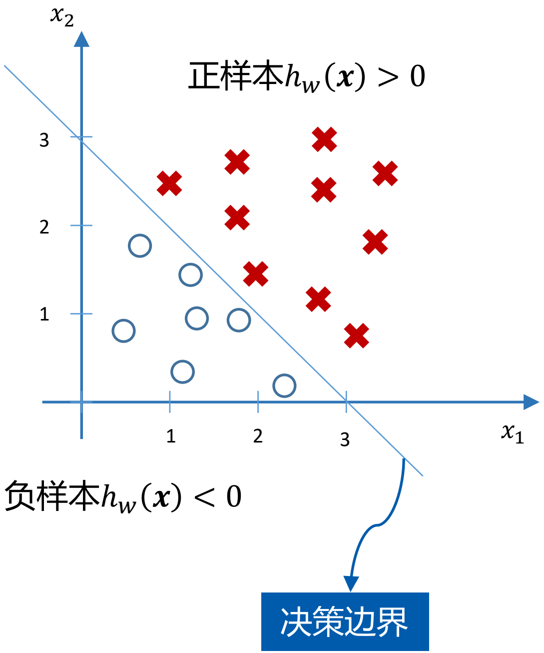
概述：一个分类算法，它可以处理二元分类以及多元分类。

本质：假设数据服从一个分布，然后使用**极大似然估计**的方法估计该分布中的参数。

对于二分类问题，令分类界面（决策边界）为：

$$w_1x_1 + w_2x_2 + b = 0$$

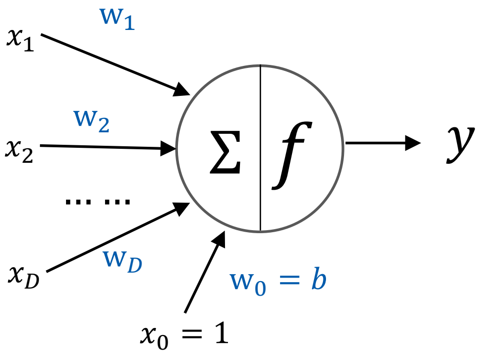
若某样本点满足 $h_w(x) = w_1x_1 + w_2x_2 + b > 0$ ，则该样本为正样本（类别为1）；相反，则亦然。

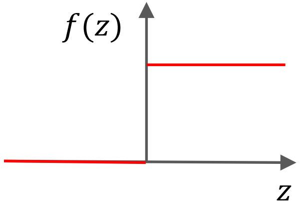


参考资料：https://blog.csdn.net/weixin_39445556/article/details/83930186

1. 知识引入-逻辑回归

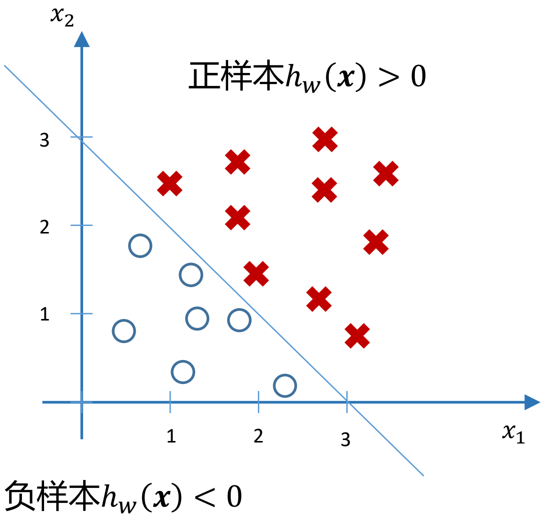
逻辑回归





$$y = f\left(\sum_{i=0}^{D} w_i x_i\right)$$

感知机M-P模型



线性分类器

线性分类器执行过程其实就是感知机

1. 知识引入

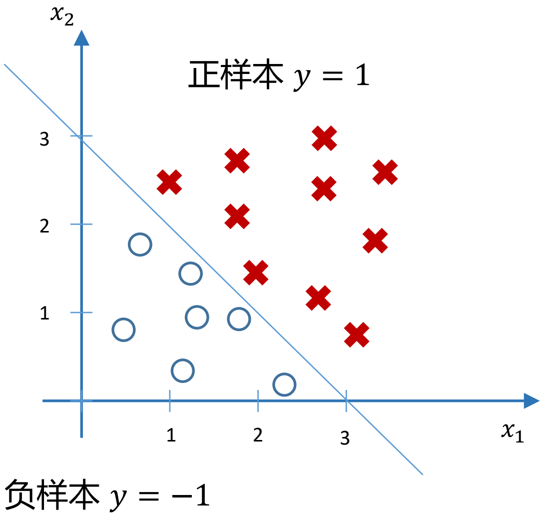
逻辑回归

目标：分类概率$P(y=1)$与输入向量x之间的直接关系，然后通过比较概率值来判别类别。

以二分类问题为例，给定数据集：

$D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_N,y_N)\}$,

其中，$\boldsymbol{x}_i\in\mathcal{R}^n, y_i\in\{0,1\}$



如何将决策边界$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$与分类概率建立联系?

$$P(y=1|\boldsymbol{x})=\frac{1}{1+e^{-(\boldsymbol{w}^T\boldsymbol{x})}}$$

Logistic函数可以将实数空间压缩到[0,1]范围内

1. 知识引入

逻辑回归

为了方便，将样本$\boldsymbol{x}$是正/负样本概率分别表示为：

$$P(y=1|\boldsymbol{x})=p(\boldsymbol{x})$$

$$P(y=0|\boldsymbol{x})=1-p(\boldsymbol{x})$$

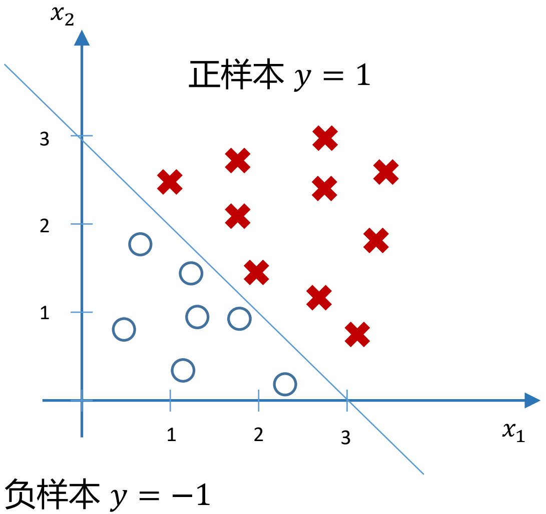
则对于数据集D中，所有样本的似然函数为：

$$\prod[p(x_i)]^{y_i}[1-p(x_i)]^{1-y_i}$$

求其极大值

为了求导方便，将连乘转化为累加的形式（取对数）

$$L(w)=\sum[y_i lnp(x_i)+(1-y_i)ln(1-p(x_i))]$$



1. 知识引入

逻辑回归

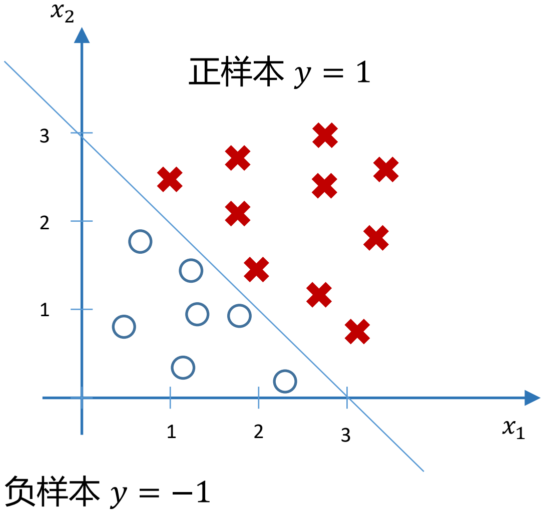
对于数据集D中所有样本的对数似然函数为：

$$L(\boldsymbol{w}) = \sum y_i lnp(\boldsymbol{x}_i) + (1 - y_i)\ln(1 - p(\boldsymbol{x}_i))$$

$$= \sum y_i ln \frac{p(\boldsymbol{x}_i)}{1 - p(\boldsymbol{x}_i)} + \ln(1 - p(\boldsymbol{x}_i))$$

$$= \sum (y_i - 1)\boldsymbol{w}^T\boldsymbol{x} - \ln(1 + e^{-\boldsymbol{w}^T\boldsymbol{x}})$$

求其极大值



$L(w)$是关于w的高阶连续可导函数。根据凸优化理论，可采用梯度下降法，牛顿法等优化方法求解。

1. 知识引入

逻辑回归

损失函数：交叉熵，对数似然函数取负号（求极小值）。

$$J(\boldsymbol{w}) = -L(\boldsymbol{w})$$

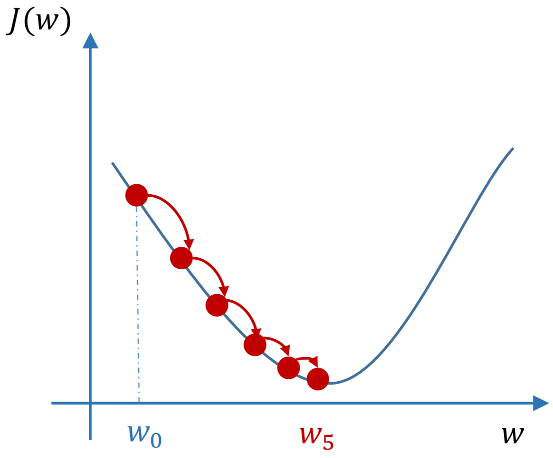
$$= -\sum y_i lnp(\boldsymbol{x}_i) + (1 - y_i)\ln(1 - p(\boldsymbol{x}_i))$$

随机梯度法：通过一阶导数来寻找下降方向，迭代更新参数。

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = (p(\boldsymbol{x}_i) - y_i)\boldsymbol{x}_i$$

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \alpha \frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}}$$

其中k为算法的迭代次数，α是更新步长（缩放系数）。



梯度下降动画演示

知识引入

感知机

神经网络

反向传播

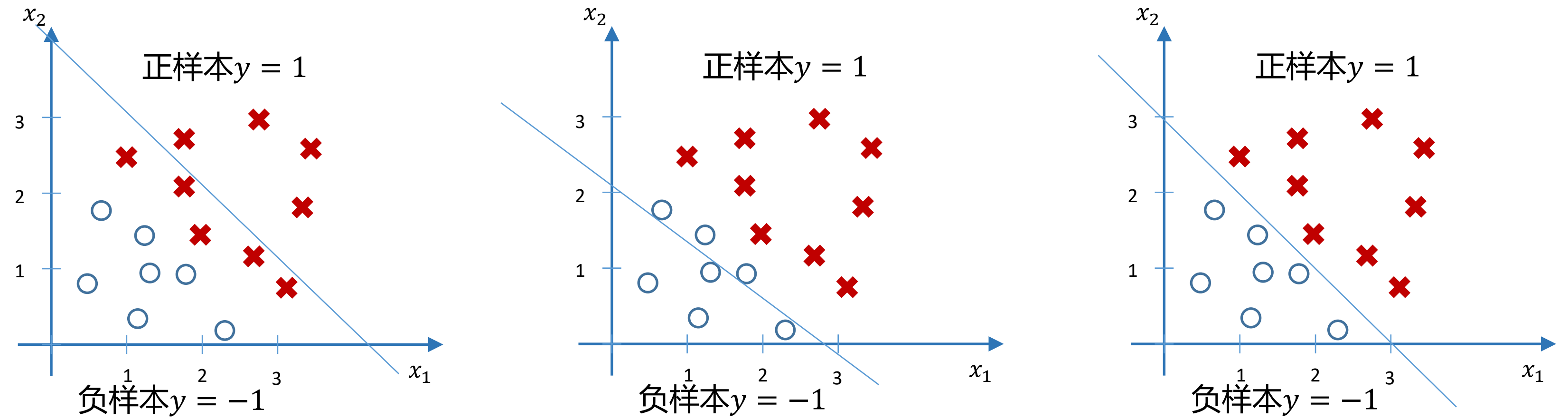
手写数字识别

2. 感知机

感知机学习策略

假设数据集是线形可分的，感知机的学习目标是求得一个能够将训练集正样本和负样本完全分开的分离超平面（决策边界）。为了找出这样的超平面（确定感知机模型参数$\boldsymbol{w}, b$），需要确定一个学习策略，即定义损失函数并将损失函数最小化。

思路1：误分类样本点的总数（如右图）定义为损失函数。 有何缺点?



2. 感知机

感知机学习策略

思路2：选择误分类样本点到超平面S的总距离作为损失函数。

空间中任意一点x_i到超平面S的距离：

$$\frac{1}{||\boldsymbol{w}||}|\boldsymbol{w}\cdot\boldsymbol{x}_i+b|$$

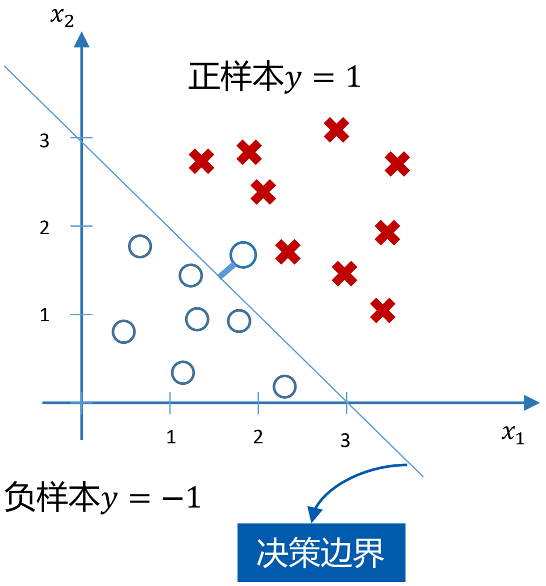
如果一个点是误分类的，

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：

$$-\frac{1}{||\boldsymbol{w}||}y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)$$



2. 感知机

感知机学习策略

假设所有误分类样本点的集合是M，则总距离表示为：

$$-\frac{1}{||\boldsymbol{w}||}\sum_{\boldsymbol{x}_i\in M} y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)$$

因此，给定训练集合$\mathrm{D}=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_N,y_N)\}$，感知机$\mathrm{sign}(\boldsymbol{w}\cdot\boldsymbol{x}+b)$学习的损失函数定义为：

$$\mathrm{L}(w,b)=-\sum_{\boldsymbol{x}_i\in M} y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)$$

感知机学习策略

优化目标：求一组参数w, b，使损失函数最小化。即

$$\min_{w,b} \mathrm{L}(w, b) = -\sum_{\boldsymbol{x}_i \in M} y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)$$

感知机学习算法是误分类驱动的（也是数据驱动的一种），

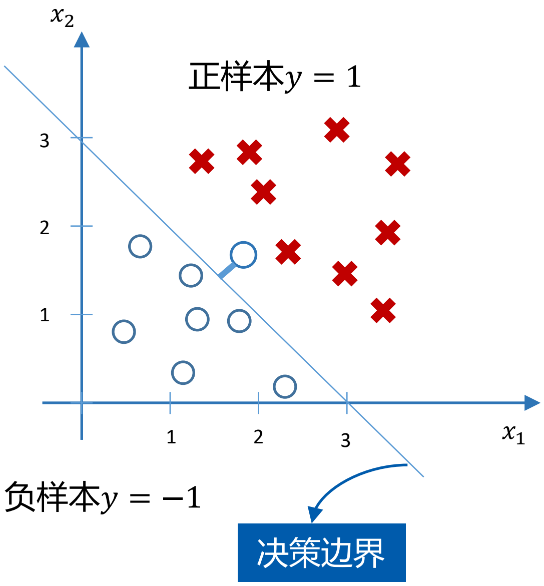
具体采用的是随机梯度下降算法：

$$\frac{\partial \mathrm{L}(w, b)}{w} = -\sum_{\boldsymbol{x}_i \in M} y_i \boldsymbol{x}_i \qquad w \leftarrow w + \eta y_i \boldsymbol{x}_i$$

$$\frac{\partial \mathrm{L}(w, b)}{b} = -\sum_{\boldsymbol{x}_i \in M} y_i \qquad b \leftarrow b + \eta y_i$$

参数的梯度

参数更新策略



2. 感知机

感知机算法

输入： 训练数据集合$D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_N,y_N)\}$；学习率$\eta$。

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

1. 选取初值$\boldsymbol{w}_0,b_0$；
2. 在训练集中选取数据$(\boldsymbol{x}_i,y_i)$；
3. 如果$y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)\leq 0$，更新参数 $w\leftarrow w+\eta y_i\boldsymbol{x}_i \quad b\leftarrow b+\eta y_i$
4. 跳转到第2步，继续学习直到训练集中没有误分类的点。

思考：对于线性不可分的情形，感知机是否还有用？

2. 感知机

感知机实例

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$f(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{w} \cdot \boldsymbol{x} + b)$。

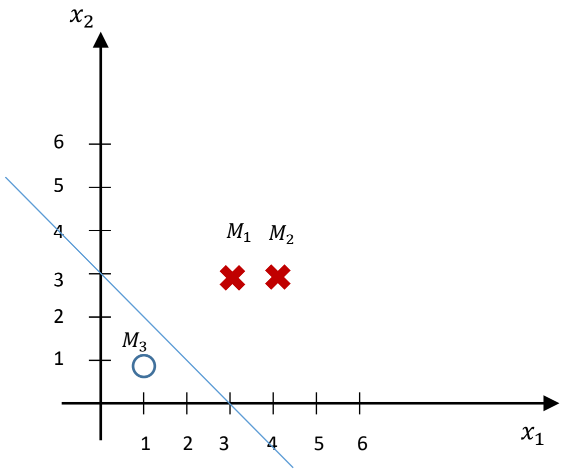
1. 取初值$\boldsymbol{w}_0 = \boldsymbol{0}, b_0 = 0$
2. M_1对应的感知机模型输出为0，未被正确分类，更新：

$$\boldsymbol{w}_1 = \boldsymbol{w}_0 + y_1\boldsymbol{x}_{M1} = (3,3) \quad b_1 = b_0 + y_1 = 1$$

3. M_1和M_2已经满足，M_3对应感知机输出为1，未被正确分类，更新：

$$\boldsymbol{w}_2 = \boldsymbol{w}_1 + y_3\boldsymbol{x}_{M3} = (2,2) \quad b_2 = b_1 + y_3 = 0$$

4. 重复更新步骤，直至没有误分类点。



2. 感知机

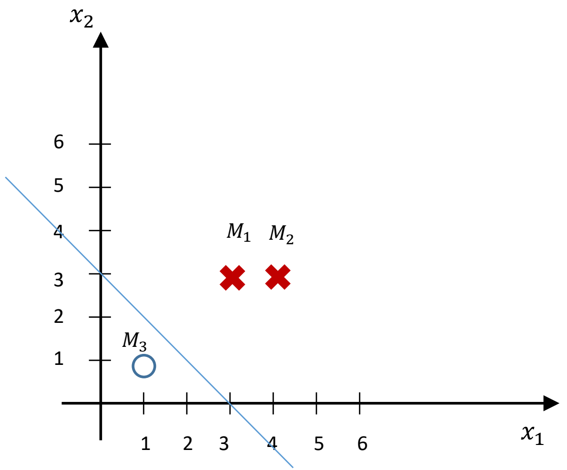
感知机实例

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_1	(3，3)	-1	$3x_1+3x_2-1$
6	M_3	(2，2)	-2	$2x_1+2x_2-2$
7	M_3	(1，1)	-3	x_1+x_2-3
8	无	(1，1)	-3	x_1+x_2-3

假如在第5次迭代时候，选择的是M_2作为误分类点，

那么结果又是什么（见作业）

感知机模型： $f(\boldsymbol{x})=\text{sign}(x_1+x_2-3)$



知识引入

感知机

神经网络

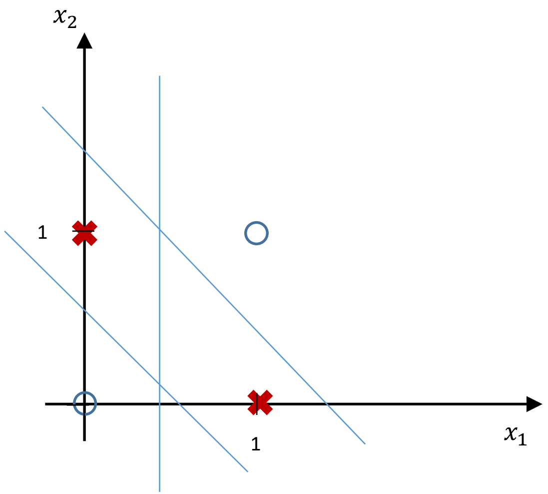
反向传播

手写数字识别

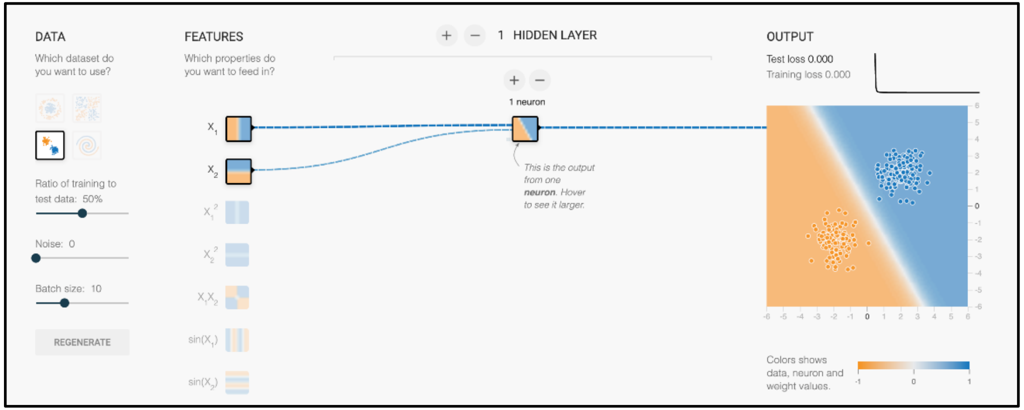
3. 神经网络

感知机问题?

非线性问题：在“异或”问题上找不到一条直线能把**O**和**X**分开，即这是一个不能用直线分类的问题，该类问题叫做非线性问题。



异或问题在二维平面上的分布



网页演示

3. 神经网络

非线性分类问题

对于非线性分类问题，可以通过增加特征的方式来完成非线性的表达。

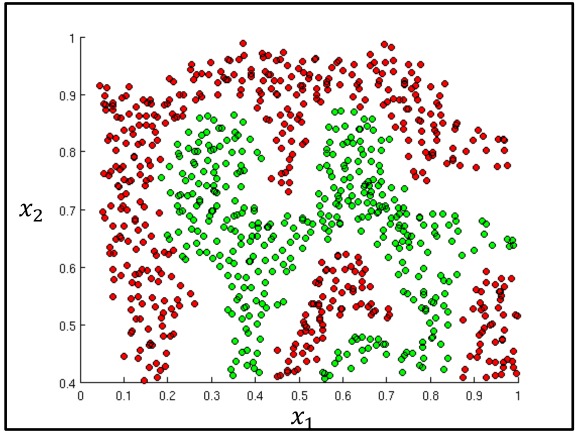
线性分类：$f(\boldsymbol{x}) = g(w_0 + w_1 x_1 + w_2 x_2)$

非线性分类：

$$f(\boldsymbol{x}) = g(w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2 + w_4 x_1^2 x_2 + w_5 x_1 x_2^2 + \cdots)$$

假设有100个特征，那么使用“增加特征”的方式构造的特征，例如$x_1 x_2, x_1 x_3, \dots\dots, x_1 x_{100}$ 和 $x_1^2 x_2, x_1^2 x_3, \dots\dots x_1^2 x_{100}$等将会很快的达到 $\boldsymbol{n^2}$ 甚至是 $\boldsymbol{n^3}$ 个。

优化和计算增加极大的困难。



非线性分类问题

如何解决这个问题？

【网页演示】

3. 神经网络

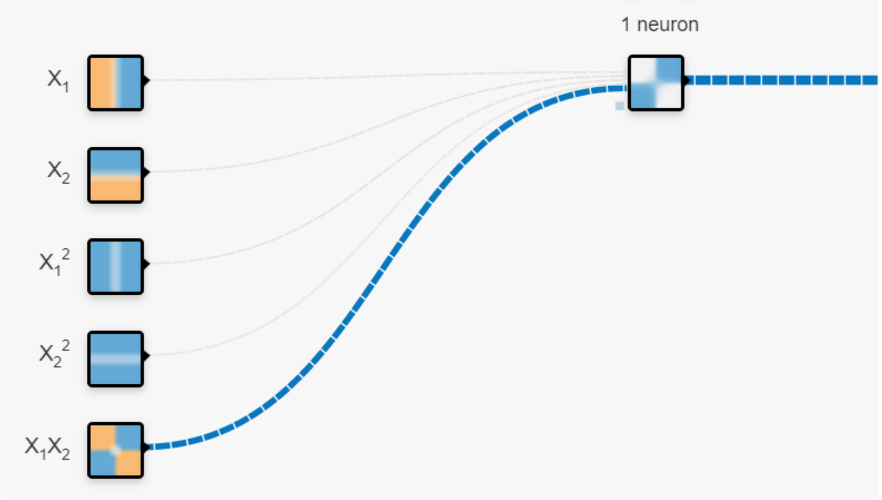
神经元的解释

神经元和线性分类器

神经元未激活之前的数学表达与线形回归用的数学公式一模一样；

假设现在将激活函数选定为sigmoid函数，那么这个神经元本身的作用和线性分类器就一致了。

本质上，一个单层的神经网络就是一个线形分类器。



3. 神经网络

神经元的解释

神经元和非线性模型

模型	实现非线性转换的方式	理论基础
Kernel SVM	利用Kernel (简单说就是一个泛函数的线性空间)	有一定的数学理论基础
多层神经网络	通过多个隐藏层	理论支持并不完备 (带隐藏层的神经网络可以拟合任何函数)

3. 神经网络

神经元的解释

神经元的偏置： 若现在的神经元没有偏置，那么输入和权重的乘积总和就是$z=\sum_i w_i x_i$，这时候想要神经元产生正/负的输出，就需要人工选择一个阈值T。

$$output=\begin{cases}0 & if \sum_i w_i x_i < T\\ 1 & if \sum_i w_i x_i \geq T\end{cases}$$

而神经元当中的偏置度量了神经元产生正（负）激励的难易程度。也就是说，人工选择的阈值越大，则产生正激励的难度也就越大。在神经元模型中，将形式统一起来：

$$output=\begin{cases}0 & if \sum_i w_i x_i + b < 0\\ 1 & if \sum_i w_i x_i + b \geq 0\end{cases}$$

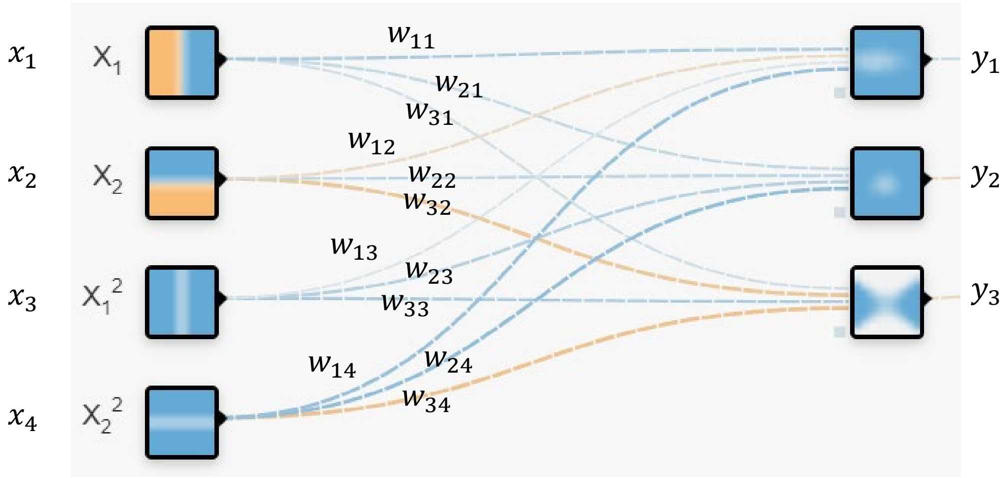
偏置的大小度量了神经元产生正 (负) 激励的难易程度

3. 神经网络

神经元的解释

神经网络是由一层一层神经元构建的，那么每一层究竟在干什么，或者是完成什么事情呢?

每层神经元理解： $\boldsymbol{y}=f(\boldsymbol{W}\cdot\boldsymbol{x}+\boldsymbol{b})$,其中$\boldsymbol{x}$是输入向量，$\boldsymbol{y}$是输出向量，$\boldsymbol{b}$是偏移向量，$\boldsymbol{W}$是权重矩阵。每一层就相当于把输入$\boldsymbol{x}$经过简单的数学操作得到$\boldsymbol{y}$。

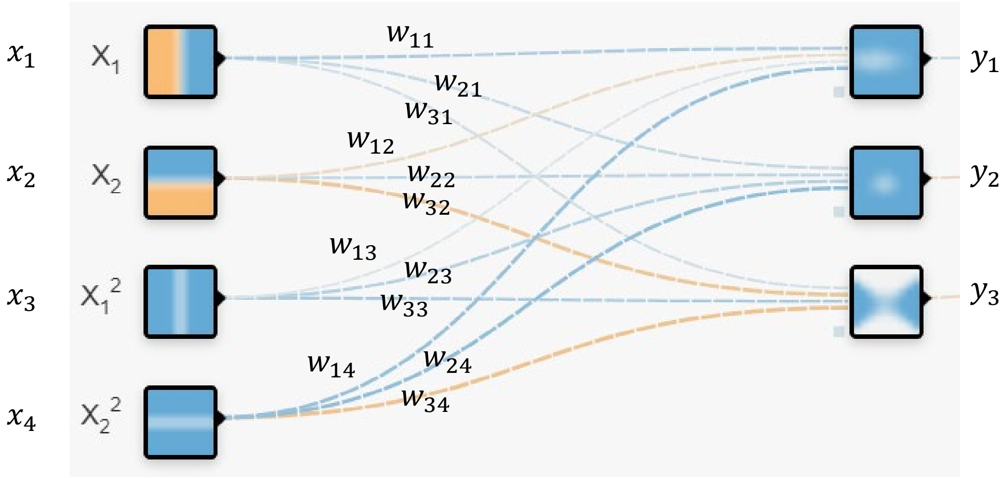


3. 神经网络

神经元的解释

神经元数学理解： 通过如下5种变换，对输入空间（输入向量的集合）进行操作，完成输入空间->输出空间的变换。

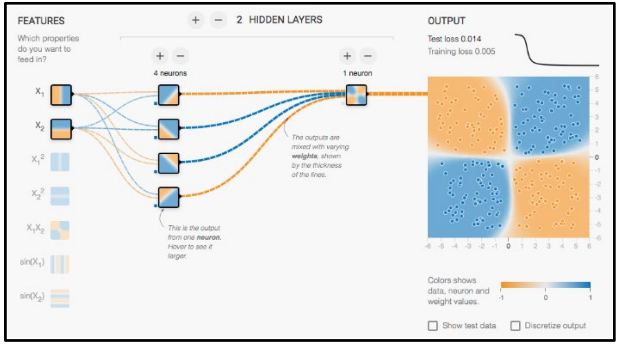
- 降维/升维
- 放大/缩小
- 旋转
- 平移
- 弯曲



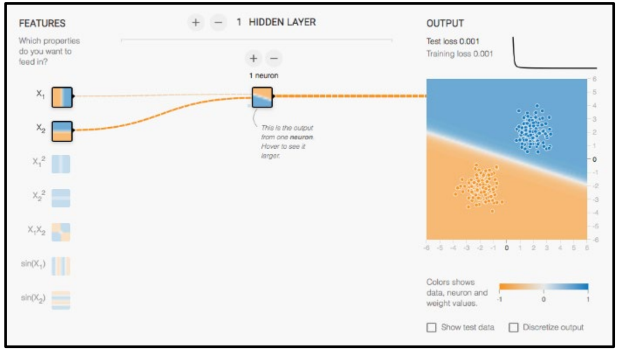
3. 神经网络

神经元的解释

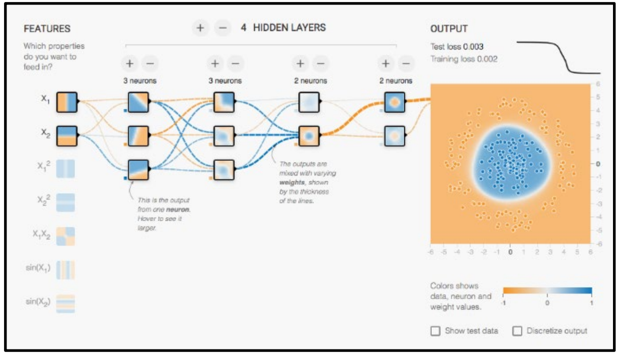
- 单个的神经元，只能做二分类问题；
- 增加神经元个数，可以做更加复杂的分类；
- 增加神经元个数，同时增加神经元的层数，完成更加复杂的非线性分类问题。



(b)多层多个神经元

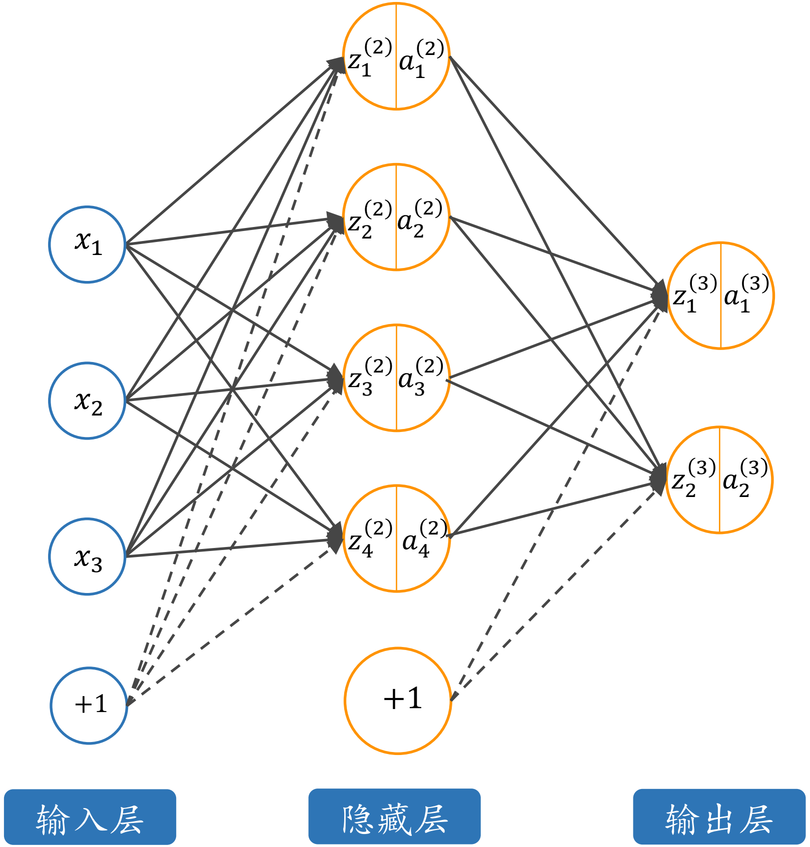


(a)单层单神经元



(c)更多层多个神经元

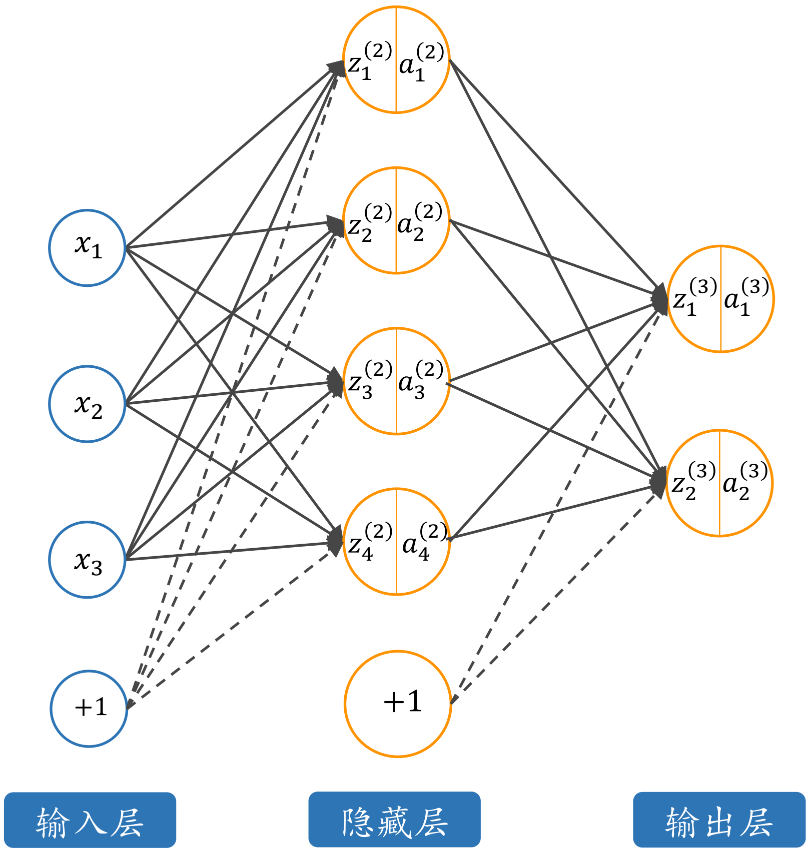
3. 神经网络—多层神经网络



定义符号：

n_l	第 l层的神经元个数
$f(\cdot)$	神经元的激活函数
$z_i^{(l)}$	第l层的第i个神经元的加权和
$a_i^{(l)}$	第l层的第i个神经元的输出（经过激励函数）
$\Theta^{(l)}$	从l-1层到l层的权重参数组
$\Theta_{ij}^{(l)}$	从l-1层第j个到l层第i个神经元连接的权重

3. 神经网络—多层神经网络



$z_i^{(l)}$ 第l层的第i个神经元的加权和

$a_i^{(l)}$ 第l层的第i个神经元的输出（经过激励函数）

$\Theta_{ij}^{(l)}$ 从l-1层第j个到l层第i个神经元连接的权重

$$a_1^{(2)} = f\left(\Theta_{11}^{(2)} x_1 + \Theta_{12}^{(2)} x_2 + \Theta_{13}^{(2)} x_3 + b_1^{(2)}\right)$$

$$a_2^{(2)} = f\left(\Theta_{21}^{(2)} x_1 + \Theta_{22}^{(2)} x_2 + \Theta_{23}^{(2)} x_3 + b_2^{(2)}\right)$$

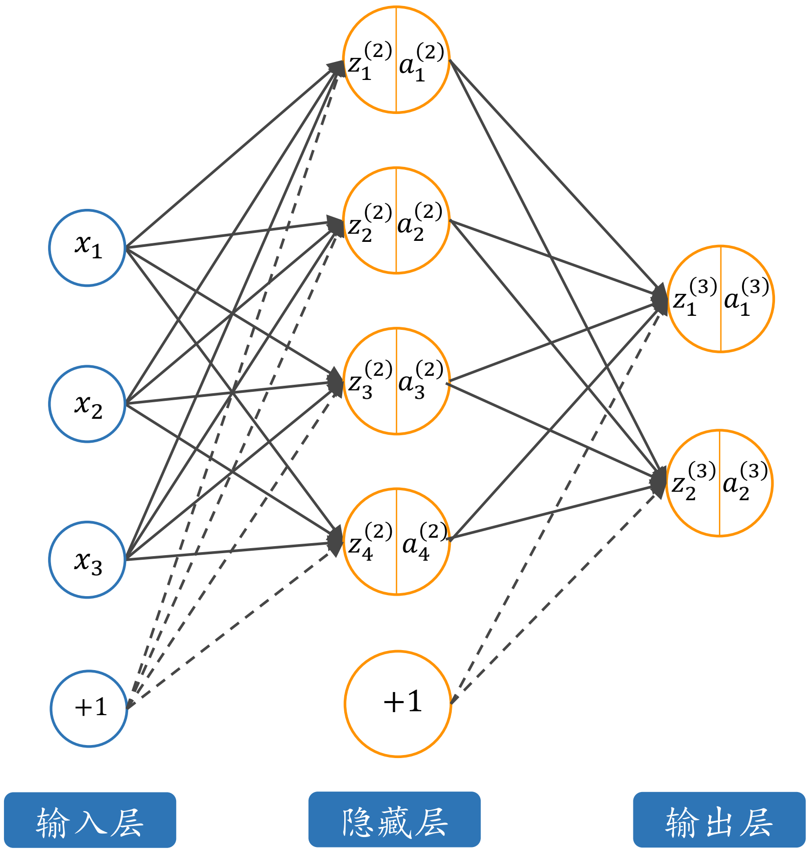
$$a_3^{(2)} = f\left(\Theta_{31}^{(2)} x_1 + \Theta_{32}^{(2)} x_2 + \Theta_{33}^{(2)} x_3 + b_3^{(2)}\right)$$

$$a_4^{(2)} = f\left(\Theta_{41}^{(2)} x_1 + \Theta_{42}^{(2)} x_2 + \Theta_{43}^{(2)} x_3 + b_4^{(2)}\right)$$

$$a_1^{(3)} = f\left(\Theta_{11}^{(3)} a_1^{(2)} + \Theta_{12}^{(3)} a_2^{(2)} + \Theta_{13}^{(3)} a_3^{(2)} + \Theta_{14}^{(3)} a_4^{(2)} + b_1^{(3)}\right)$$

$$a_2^{(3)} = f\left(\Theta_{21}^{(3)} a_1^{(2)} + \Theta_{22}^{(3)} a_2^{(2)} + \Theta_{23}^{(3)} a_3^{(2)} + \Theta_{24}^{(3)} a_4^{(2)} + b_2^{(3)}\right)$$

3. 神经网络—多层神经网络



$\boldsymbol{z}^{(l)}$ 第l层神经元的权重和 $\mathbb{R}^{n_l}$

$\boldsymbol{a}^{(l)}$ 第l层神经元输出（经过激励函数） $\mathbb{R}^{n_l}$

$\boldsymbol{\Theta}^{(l)}$ 从l-1层到l层的权重参数组 $\mathbb{R}^{n_l \times n_{l-1}}$

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

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

$$\boldsymbol{a}^{(l)} = f\left(\boldsymbol{z}^{(l)}\right)$$

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

前向传播过程：

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

3. 神经网络—多层神经网络

损失函数

逻辑回归的损失函数

$\boxed{P(y=1|\boldsymbol{x})=p(\boldsymbol{x})}$ $\boxed{\|\boldsymbol{\theta}\|_F^2}$

$$J(\theta)=-\frac{1}{m}\underbrace{\left[\sum_{i=1}^{m}y^{(i)}logh_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)+\left(1-y^{(i)}\right)log\left(1-h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right]}_{\text{交叉熵损失函数}}+\underbrace{\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2}_{\text{正则项}}$$

神经网络的损失函数

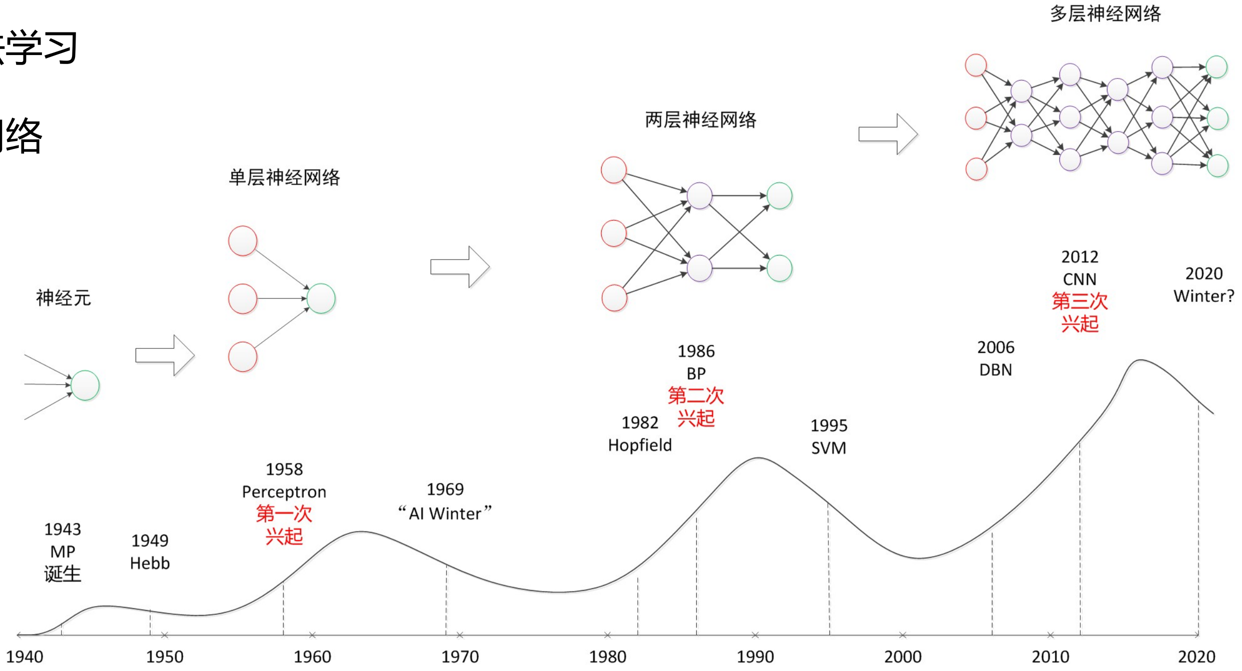
$$J(\theta)=-\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K}y_k^{(i)}log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{i}\right)\right)_k+\left(1-y_k^{(i)}\right)log\left(1-\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{i}\right)\right)_k\right)\right]+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{ji}^{(l)}\right)^2$$

其中，$h_{\boldsymbol{\Theta}}(\boldsymbol{x})\in\mathbb{R}^K$，$\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x})\right)_k$表示$k^{th}$输出层的输出，$K=n_L$

3. 神经网络

神经网络的发展与比较

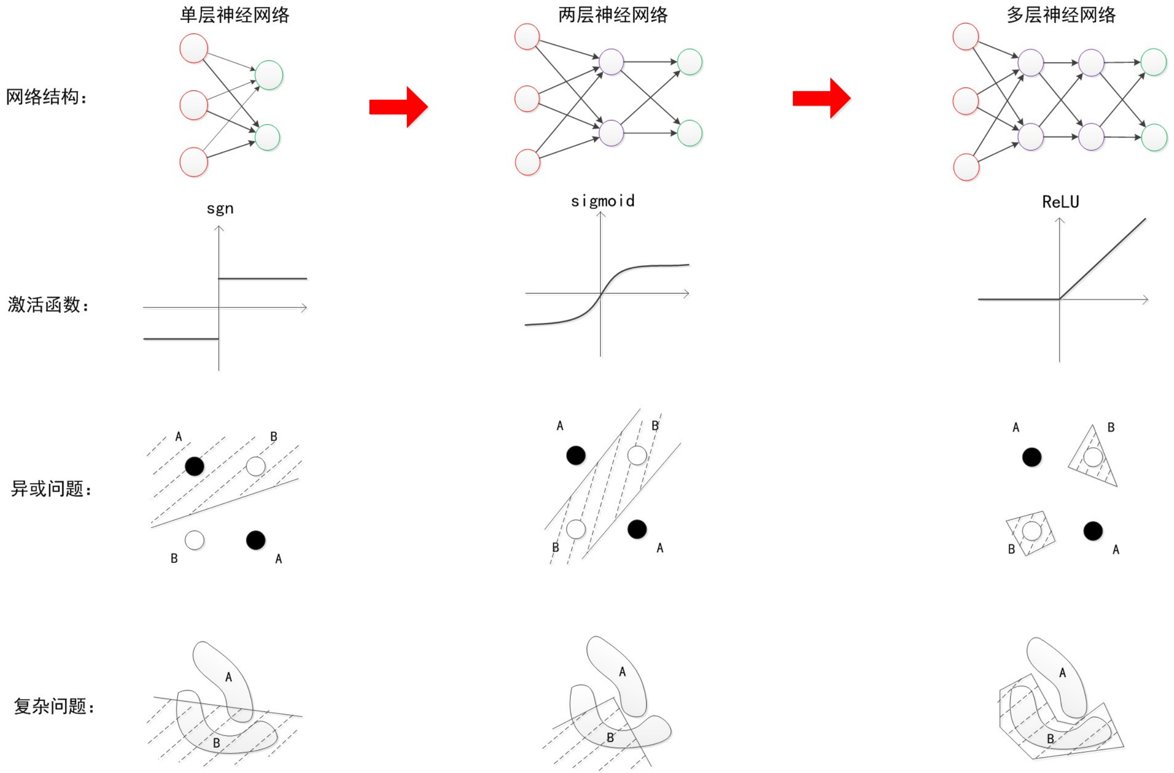
- 从人工指定权值到算法学习
- 从感知机到多层神经网络
- 神经网络的起起落落



3. 神经网络

神经网络的发展与比较

- 激活函数的改变
- 决策边界的改变
- 解决问题能力改变



知识引入

感知机

神经网络

反向传播

手写数字识别

4. 反向传播

梯度下降

反向传播：前向传播算法对于未训练完全的网络，网络的输出值与真实值之间存在差异，把这个差异叫做网络的**误差**，误差的传播与前向传播方向正好相反，最后对每一个参数利用梯度下降的方法来更新。

训练数据

$$\left\{\left(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}\right),\left(\boldsymbol{x}^{(2)}, \boldsymbol{y}^{(2)}\right), \cdots,\left(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}\right), \cdots,\left(\boldsymbol{x}^{(N)}, \boldsymbol{y}^{(N)}\right)\right\}$$

输出数据

第i个训练样本的输出：$\boldsymbol{y}^{(i)}=\left(y_1^{(i)}, \cdots, y_{n_L}^{(i)}\right)^{\top}$

$n_L=K$：输出层神经元（节点）个数

对于第i个训练数据来说，其代价函数为：

$$E^{(i)}=\frac{1}{2}\left\|\boldsymbol{y}^{(i)}-\boldsymbol{a}^{(i)}\right\|^2=\frac{1}{2}\sum_{k=1}^{n_L}\left(y_k^i-a_k^i\right)^2$$

对于所有训练数据，其平均代价函数为：

$$E_{total}=\frac{1}{N}\sum_{i=1}^{N}E^{(i)}$$

4. 反向传播

梯度下降

梯度下降法（又称为“批量梯度下降法”），$\Theta^{(l)}$与$b^{(l)}$的参数更新方式为:

$$\begin{aligned}\Theta^{(l)} &= \Theta^{(l)} - \mu\frac{\partial E_{total}}{\partial \Theta^{(l)}} \\ &= \Theta^{(l)} - \frac{\mu}{N}\sum_{i=1}^{N}\frac{\partial E^{(i)}}{\partial \Theta^{(l)}}\end{aligned} \qquad \begin{aligned}b^{(l)} &= b^{(l)} - \mu\frac{\partial E_{total}}{\partial b^{(l)}} \\ &= b^{(l)} - \frac{\mu}{N}\sum_{i=1}^{N}\frac{\partial E^{(i)}}{\partial b^{(l)}}\end{aligned}$$

只需求得每个训练数据的代价函数对参数的偏导数 $\frac{\partial E^{(i)}}{\partial \Theta^{(l)}}, \frac{\partial E^{(i)}}{\partial b^{(l)}}$，即可得到参数的迭代更新表示。

为了表示简洁，之后的推导中，我们将$E^{(i)}$直接记作E。

4. 反向传播

梯度下降

对于前述的神经网络（下图），任意一个训练样本，将误差表示（代价函数）在输出层展开，则能得到：

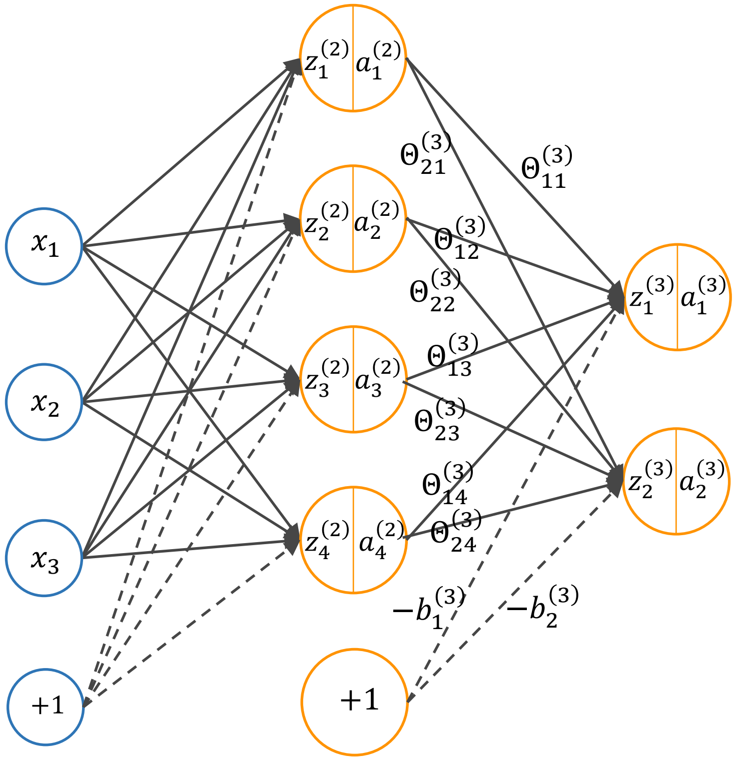
$$E=\frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{a}^{(3)}\right\|^2=\frac{1}{2}\sum_{k=1}^{2}\left(y_k-a_k^{(3)}\right)^2$$

$$=\frac{1}{2}\left((y_1-a_1^{(3)})^2+(y_2-a_2^{(3)})^2\right)$$

$$=\frac{1}{2}\left((y_1-f(z_1^{(3)}))^2+(y_2-f(z_2^{(3)}))^2\right)$$

$$=\frac{1}{2}\Big((y_1-f(\Theta_{11}^{(3)}a_1^{(2)}+\Theta_{12}^{(3)}a_2^{(2)}+\Theta_{13}^{(3)}a_3^{(2)}+\Theta_{14}^{(3)}a_4^{(2)})+b_1^{(3)})^2+$$

$$(y_2-f(\Theta_{21}^{(3)}a_1^{(2)}+\Theta_{22}^{(3)}a_2^{(2)}+\Theta_{23}^{(3)}a_3^{(2)}+\Theta_{24}^{(3)}a_4^{(2)})+b_2^{(3)})^2\Big)$$



4. 反向传播

梯度下降-输出层权重更新

对输出层权重求偏导数，得到：

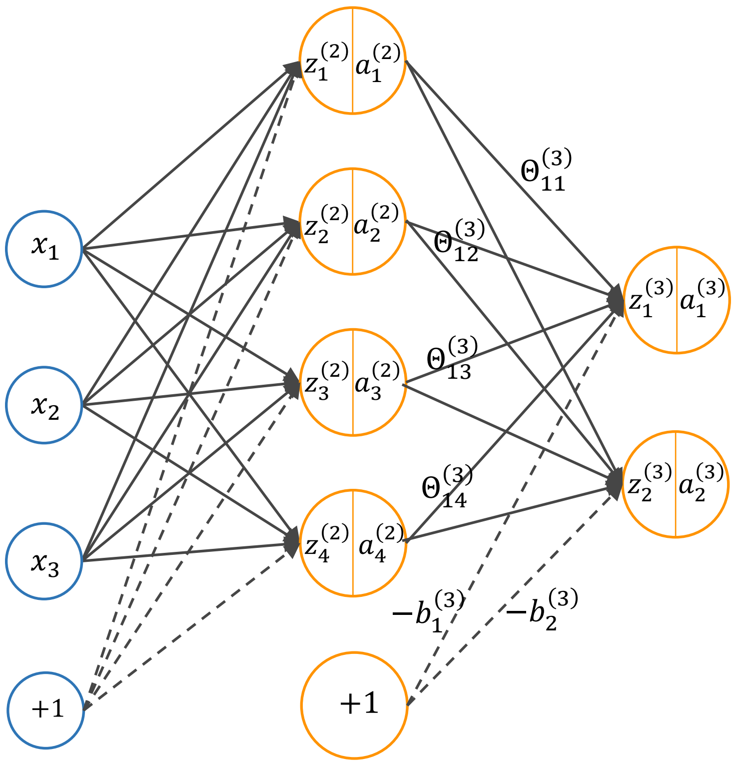
$$\frac{\partial E}{\partial \Theta_{11}^{(3)}} = \frac{\partial E}{\partial a_1^{(3)}} \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \frac{\partial z_1^{(3)}}{\partial \Theta_{11}^{(3)}}$$

$$= \frac{1}{2} \cdot 2 \left(y_1 - a_1^{(3)}\right) \left(-\frac{\partial a_1^{(3)}}{\partial \Theta_{11}^{(3)}}\right)$$

$$= -\left(y_1 - a_1^{(3)}\right) f'\left(z_1^{(3)}\right) \frac{\partial z_1^{(3)}}{\partial \Theta_{11}^{(3)}}$$

$$= \boxed{-\left(y_1 - a_1^{(3)}\right) f'\left(z_1^{(3)}\right)} a_1^{(2)}$$

令：$\delta_i^{(3)} = \frac{\partial E}{\partial a_i^{(3)}} \frac{\partial a_i^{(3)}}{\partial z_i^{(3)}}$　　则：$\frac{\partial E}{\partial \Theta_{11}^{(3)}} = \boxed{\delta_1^{(3)}} a_1^{(2)}$



4. 反向传播

梯度下降-输出层权重更新

对于输出层其他的权重参数，同样可得：

$$\delta_1^{(3)} = -\left(y_1 - a_1^{(3)}\right) f'\left(z_1^{(3)}\right)$$

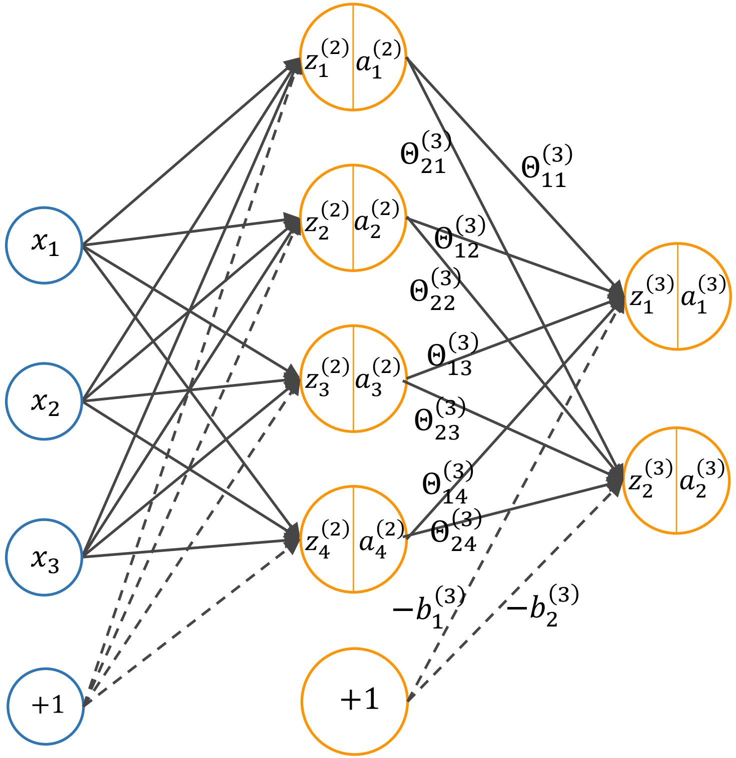
$$\delta_2^{(3)} = -\left(y_2 - a_2^{(3)}\right) f'\left(z_2^{(3)}\right)$$

$$\frac{\partial E}{\partial \Theta_{11}^{(3)}} = \delta_1^{(3)} a_1^{(2)} \qquad \frac{\partial E}{\partial \Theta_{21}^{(3)}} = \delta_2^{(3)} a_1^{(2)}$$

$$\frac{\partial E}{\partial \Theta_{12}^{(3)}} = \delta_1^{(3)} a_2^{(2)} \qquad \frac{\partial E}{\partial \Theta_{22}^{(3)}} = \delta_2^{(3)} a_2^{(2)}$$

$$\frac{\partial E}{\partial \Theta_{13}^{(3)}} = \delta_1^{(3)} a_3^{(2)} \qquad \frac{\partial E}{\partial \Theta_{23}^{(3)}} = \delta_2^{(3)} a_3^{(2)}$$

$$\frac{\partial E}{\partial \Theta_{14}^{(3)}} = \delta_1^{(3)} a_4^{(2)} \qquad \frac{\partial E}{\partial \Theta_{24}^{(3)}} = \delta_2^{(3)} a_4^{(2)}$$



4. 反向传播

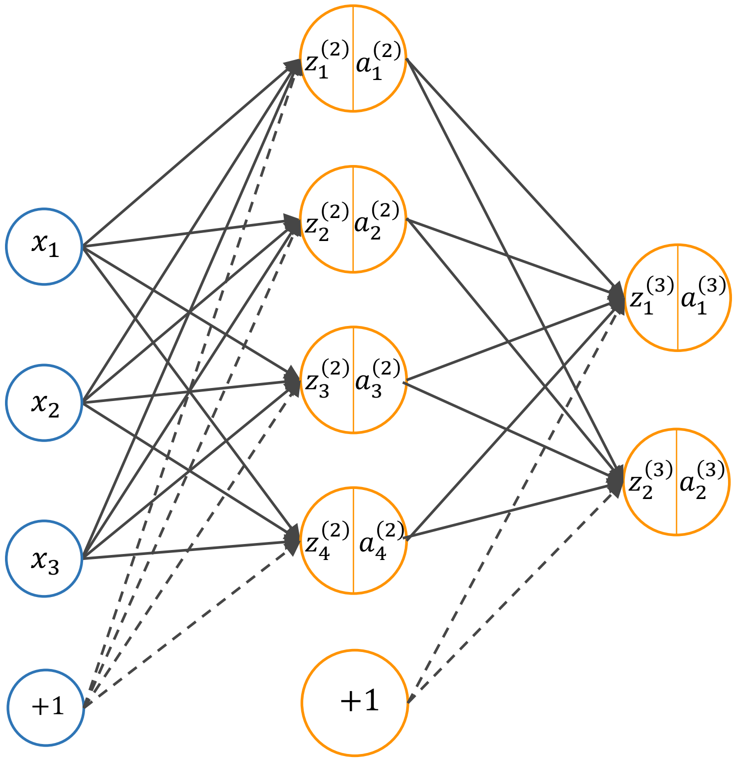
梯度下降-输出层权重更新

$$\delta_i^{(3)} = \frac{\partial E}{\partial z_i^{(3)}} = \frac{\partial E}{\partial a_i^{(3)}} \frac{\partial a_i^{(3)}}{\partial z_i^{(3)}}$$

$$\delta_i^{(2)} = \frac{\partial E}{\partial z_i^{(2)}} = \frac{\partial E}{\partial a_i^{(2)}} \frac{\partial a_i^{(2)}}{\partial z_i^{(2)}}$$

为什么引入 $\delta_i^{(l)}$?

- 简化 $\frac{\partial E_{(i)}}{\partial \Theta^{(l)}}$ $\frac{\partial E_{(i)}}{\partial b^{(l)}}$ 的表达形式
- 通过 $\delta_i^{(l+1)}$ 来计算 $\delta_i^{(l)}$（误差的反向传播）



4. 反向传播

梯度下降-输出层权重更新

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

推广到一般情况，假设网络有L层，则：

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

$$\frac{\partial E}{\partial \Theta_{ij}^{(L)}} = \delta_i^{(L)} a_j^{(L-1)} \quad (1 \leq i \leq n_L, 1 \leq j \leq n_{L-1})$$

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

转化为向量形式：

$$\boldsymbol{\delta}^{(L)} = -\left(\boldsymbol{y} - \boldsymbol{a}^{(L)}\right) \odot f'\left(\boldsymbol{z}^{(L)}\right)$$

$$\nabla_{\Theta^{(L)}} E = \boldsymbol{\delta}^{(L)} \left(\boldsymbol{a}^{(L-1)}\right)^T$$

4. 反向传播

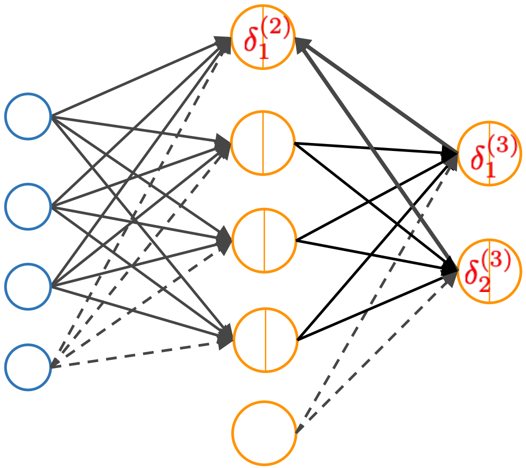
梯度下降-隐藏层权重更新

$$\frac{\partial E}{\partial z_i^{(l)}} = \frac{\partial E}{\partial z_1^{(l+1)}}\frac{\partial z_1^{(l+1)}}{\partial z_i^{(l)}} + \frac{\partial E}{\partial z_2^{(l+1)}}\frac{\partial z_2^{(l+1)}}{\partial z_i^{(l)}} + \cdots + \frac{\partial E}{\partial z_{n_{l+1}}^{(l+1)}}\frac{\partial z_{n_{l+1}}^{(l+1)}}{\partial z_i^{(l)}}$$
$$= \sum_{j=1}^{n_{l+1}} \frac{\partial E}{\partial z_j^{(l+1)}}\frac{\partial z_j^{(l+1)}}{\partial z_i^{(l)}}$$

输出层误差 → 隐藏层误差

误差反向传播

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



$$\delta_i^{(l)} \equiv \frac{\partial E}{\partial z_i^{(l)}} = \sum_{j=1}^{n_{l+1}} \frac{\partial E}{\partial z_j^{(l+1)}}\frac{\partial z_j^{(l+1)}}{\partial z_i^{(l)}} = \sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)}\frac{\partial z_j^{(l+1)}}{\partial z_i^{(l)}}$$

4. 反向传播

梯度下降-隐藏层权重更新

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

$$= \sum_{j=1}^{n_{l+1}} \frac{\partial E}{\partial z_j^{(l+1)}} \frac{\partial z_j^{(l+1)}}{\partial z_i^{(l)}}$$

$$= \sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \frac{\partial z_j^{(l+1)}}{\partial z_i^{(l)}}$$

$$= \sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \frac{\partial z_j^{(l+1)}}{\partial a_i^{(l)}} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}}$$

$$= \sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \Theta_{ji}^{(l+1)} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}}$$

$$= \left(\sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \Theta_{ji}^{(l+1)} \right) f'\left(z_i^{(l)}\right)$$

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

这就是BP传播算法的核心公式，利用$l+1$层的$\delta_i^{(l+1)}$来计算l层的$\delta_i^{(l)}$，这就是“误差反向传播”名字的由来。

$$\boldsymbol{\delta}^{(l)} = \left(\left(\boldsymbol{\Theta}^{(l+1)}\right)^T \boldsymbol{\delta}^{(l+1)} \right) \odot f'\left(\boldsymbol{z}^{(l)}\right)$$

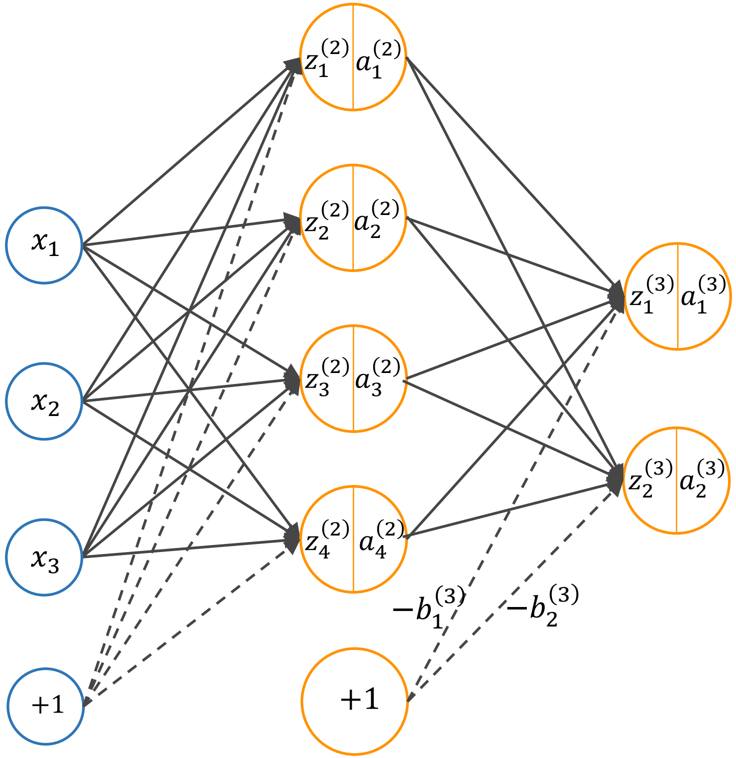
4. 反向传播

梯度下降-偏置更新

对输出层偏置求偏导数，得到：

$$\frac{\partial E}{\partial b_1^{(3)}} = \frac{\partial E}{\partial a_1^{(3)}} \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \frac{\partial z_1^{(3)}}{\partial b_1^{(3)}} = \delta_1^{(3)}$$

$$\frac{\partial E}{\partial b_2^{(3)}} = \frac{\partial E}{\partial a_2^{(3)}} \frac{\partial a_2^{(3)}}{\partial z_2^{(3)}} \frac{\partial z_1^{(3)}}{\partial b_2^{(3)}} = \delta_2^{(3)}$$



4. 反向传播

梯度下降-偏置更新

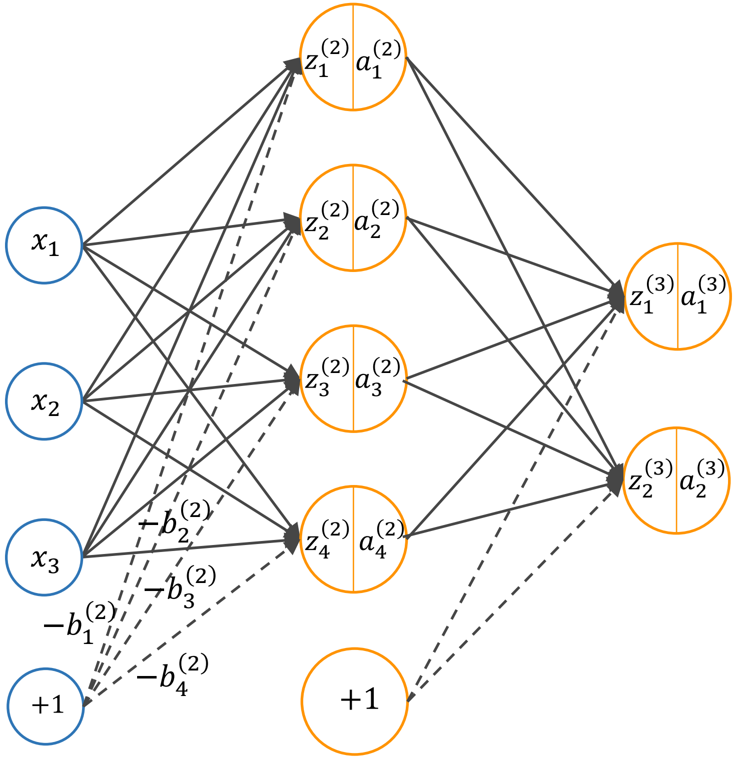
对隐藏层偏置求偏导数，得到：

$$\frac{\partial E}{\partial b_1^{(2)}} = \frac{\partial E}{\partial a_1^{(3)}} \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \frac{\partial z_1^{(3)}}{\partial z_1^{(2)}} \frac{z_1^{(2)}}{b_1^{(2)}} = \delta_1^{(2)}$$

$$\frac{\partial E}{\partial b_2^{(2)}} = \frac{\partial E}{\partial a_2^{(3)}} \frac{\partial a_2^{(3)}}{\partial z_2^{(3)}} \frac{\partial z_2^{(3)}}{\partial z_2^{(2)}} \frac{z_2^{(2)}}{b_2^{(2)}} = \delta_2^{(2)}$$

$$\frac{\partial E}{\partial b_3^{(2)}} = \frac{\partial E}{\partial a_3^{(3)}} \frac{\partial a_3^{(3)}}{\partial z_3^{(3)}} \frac{\partial z_3^{(3)}}{\partial z_3^{(2)}} \frac{z_3^{(2)}}{b_3^{(2)}} = \delta_3^{(2)}$$

$$\frac{\partial E}{\partial b_4^{(2)}} = \frac{\partial E}{\partial a_4^{(3)}} \frac{\partial a_4^{(3)}}{\partial z_4^{(3)}} \frac{\partial z_4^{(3)}}{\partial z_4^{(2)}} \frac{z_4^{(2)}}{b_4^{(2)}} = \delta_4^{(2)}$$



4. 反向传播

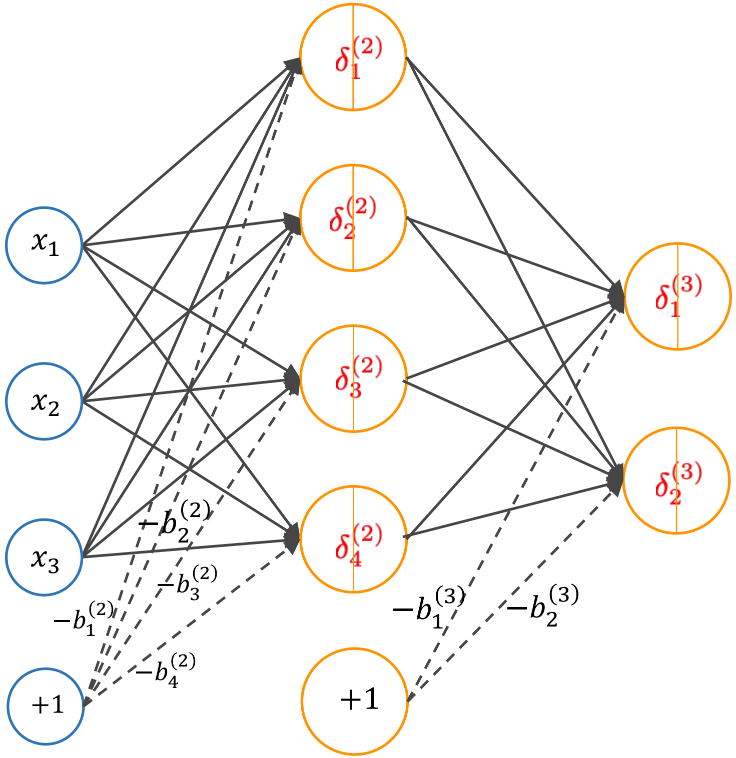
梯度下降-偏置更新

因此，输出层和隐藏层的偏置为：

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

向量形式为：

$$\nabla_{b^{(l)}} E = \boldsymbol{\delta}^l$$



4. 反向传播

BP算法的核心

总结，反向传播算法的核心公式为：

$$\delta_i^{(L)} = -\left(y_i - a_i^{(L)}\right) f'\left(z_i^{(L)}\right) \qquad \delta_i^{(l)} = \left(\sum_{j=1}^{n_{l+1}} \delta_j^{(l+1)} \Theta_{ji}^{(l+1)}\right) f'\left(z_i^{(l)}\right)$$

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

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

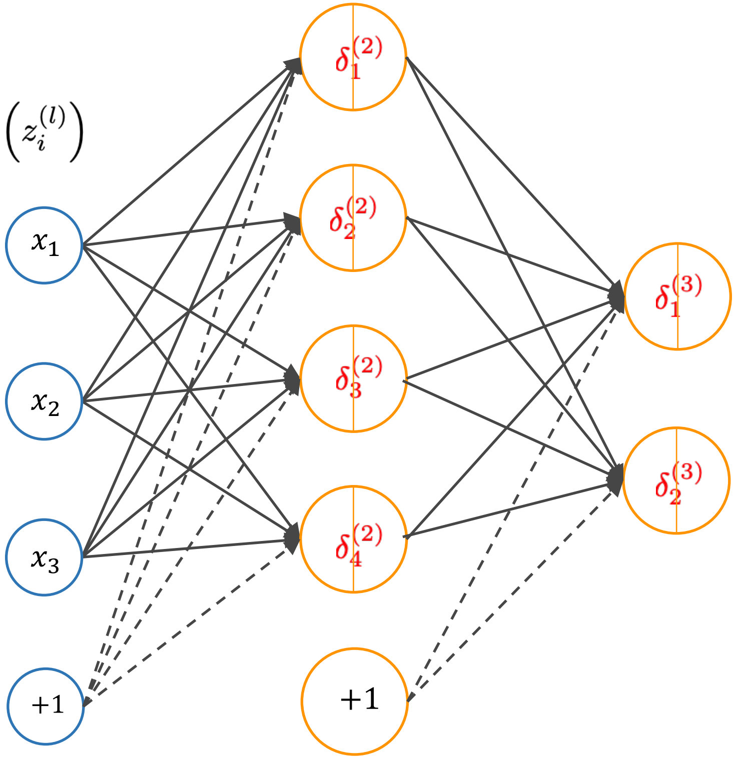
向量形式为：

$$\boldsymbol{\delta}^{(L)} = -\left(\boldsymbol{y} - \boldsymbol{a}^{(L)}\right) \odot f'\left(\boldsymbol{z}^{(L)}\right)$$

$$\boldsymbol{\delta}^{(l)} = \left(\left(\Theta^{(l+1)}\right)^{\top} \boldsymbol{\delta}^{(l+1)}\right) \odot f'\left(\boldsymbol{z}^{(l)}\right)$$

$$\nabla_{\Theta^{(l)}} E = \boldsymbol{\delta}^{(l)} \left(\boldsymbol{a}^{(l-1)}\right)^{\top}$$

$$\nabla_{b^{(l)}} E = \delta^{l}$$

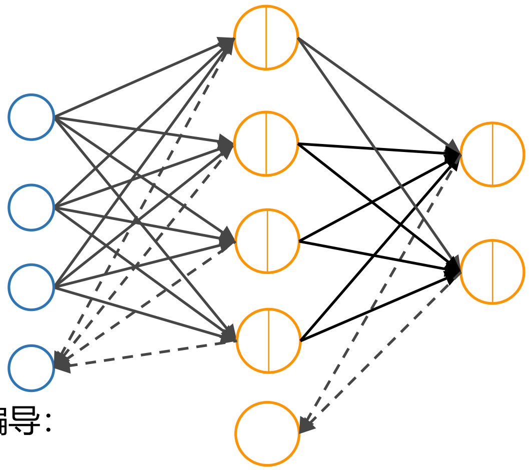


4. 反向传播

BP算法的核心

输入： 训练数据集合D $= \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$；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：

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

$$a^{(l)} = f\left(\boldsymbol{z}^{(l)}\right)$$

3. 计算每一层的误差：

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

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

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：

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

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

知识引入

感知机

神经网络

反向传播

手写数字识别

课后作业

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$f(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{w} \cdot \boldsymbol{x} + b)$。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	$\boldsymbol{w}$	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			

课后作业

作业 II

为什么多层神经网络可以拟合任意函数？其与浅层宽网络的区别是什么？

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*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？