学习小结

留出验证法（hold out）：留出一定比例的数据作为测试集。留出验证法是一种简单易用的评估方法。缺点：在原始数据较少的情况下，验证集和测试集包含的样本很少，无法代表数据，如果在数据划分为三个集合之前，将数据进行随机打乱，存在着最终得到的模型性能差别很大的问题。

交叉验证法（cross validation）：交叉验证法又称 K 折验证，顾名思义将数据划分为大小相同的 K 个分区，在 K 次运算中选择不同的分区作为测试集。交叉验证法的缺点：在原始数据比较大时，训练计算开销较大。

自助验证法（bootstrapping）：从给定训练集中有放回的随机抽样，在数据集较小、难以有效划分训练集与测试集时经常使用自助验证法，自助验证法可以减少训练样本规模不同造成的影响，同时还能比较高效地进行实验估计。缺点：由于自助验证法产生的数据集改变了原始数据集的分布，这会产生估计偏差，不适合原始数据比较大的场合。在原始数据足够时，留出验证法和交叉验证法更常用。

评估模型的注意事项：

（1）训练集和测试集都能够代表当前数据。

（2）评估模型与预测时间有关时，应该确保测试集中所有数据的时间都晚于训练集数据，且划分数据前不应该随机打乱数据。

（3）降低数据冗余，尽可能降低训练集和验证集之间的交集。

2.3　深度学习基础知识

深度学习是机器学习中一种基于对数据进行表征学习的方法，它的概念源于人工神经网络的研究，所以涉及的专业名词有神经元、多层感知机等。对于初次接触深度学习的人而言相关的计算公式及术语比较抽象，下面将用通俗易懂的方式推导相关计算公式并介绍深度学习相关基础知识。

2.3.1　线性回归

在学习深度学习基础知识之前，有必要了解线性回归模型，以便于后面理解神经网络中的激活函数、损失函数、反向传播等相关术语。

在统计学中，只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示，这种回归分析称为一元线性回归；如果回归分析中包括两个或两个以上的自变量，且自变量之间存在线性相关，则称为多重线性回归。

本节中只讨论一元线性回归，一元线性回归是分析只有一个自变量 X 和一个因变量 Y 的线性相关关系的方法。也就是说，对于自变量 X 的某个值 x_i，因变量 Y 对应的取值 y_i 不是唯一确定的，而是有很多的可能取值，它们分布在一条直线的上下，这是因为 Y 还受除自变量以外的其他因素的影响。这些因素的影响大小和方向都是不确定的，通常用一个随机变量（记

为 e）来表示，又称随机扰动项。e 是无法直接观测的随机变量，通常随机误差 e 的期望值为 0。在一元线性回归模型中还有 a 和 b 两个常数。其中，a 表示为总体回归直线在 Y 轴上的截距；b 表示为总体回归直线的斜率。根据理论回归模型式（2.1）将 Y 和 X 之间的依存关系可表示为一元线性回归方程式（2.2）。式（2.2）中 $\hat{y}$ 为 y 的估计值，也称 $\hat{y}$ 为 y 的拟合值或回归值。

$$y_i = a + bx_i + e_i \tag{2.1}$$

$$\hat{y}_i = a + bx_i \tag{2.2}$$

在统计学中根据散布点去拟合回归直线时采用误差（残差）平方最小化进行模型求解的方法称为最小二乘估计法，使得拟合回归直线上的 $\hat{y}$ 对应的实际观测值 y 之间的差为最小，而最优拟合线应该使各点到直线的距离的平方和（残差平方和 SSE）最小，即

$$Q = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - a - bx_i)^2 \tag{2.3}$$

利用数学偏导数求极值的方法，通过对 Q 分别求 a 和 b 的偏导数

$$\begin{aligned} \frac{\partial Q}{\partial a} &= -2\sum(y_i - a - bx_i) = 0 \\ \frac{\partial Q}{\partial b} &= -2\sum(y_i - a - bx_i)x_i = 0 \end{aligned} \tag{2.4}$$

可得标准方程组，即式（2.5）

$$\begin{aligned} \sum y_i &= na + b\sum x_i \\ \sum x_i y_i &= a\sum x_i + b\sum x_i^2 \end{aligned} \tag{2.5}$$

根据公式进行求解，得到 a、b 的值，即

$$\begin{aligned} b &= \frac{\sum x_i y_i - \frac{1}{n}\left(\sum x_i\right)\left(\sum y_i\right)}{\sum x_i^2 - \frac{1}{n}\left(\sum x_i\right)^2} \\ a &= \frac{\sum y_i}{n} - b\frac{\sum x_i}{n} \end{aligned} \tag{2.6}$$

根据式（2.6）推导得到的 a、b 值，代入式（2.2）计算出最佳情况的 $\hat{y}$ 值，分别求出 X、Y 的平均值：$\overline{X} = \left(\sum x_i\right)/n$、$\overline{Y} = \left(\sum y_i\right)/n$，可计算出回归平方和（sum of squares for regression，SSR）计算公式为

$$\mathrm{SSR} = \sum_{i=1}^{n}(\hat{y}_i - \overline{y}_i)^2 \tag{2.7}$$

残差平方和（sum of squares for error，SSE）计算公式为

$$\mathrm{SSE} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.8}$$

根据回归平方和与残差平方和计算出总偏差平方和（sum of squares for total，SST）和均方误差（mean-square error，MSE）。

$$\mathrm{SST} = \mathrm{SSR} + \mathrm{SSE} \tag{2.9}$$

$$\mathrm{MSE} = \frac{\mathrm{SSE}}{n} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.10}$$

均方根误差（root mean square error，RMSE）又称回归系统的拟合标准误差，是 MSE 的平方根，计算公式如下：

$$\mathrm{RMSE}=\sqrt{\mathrm{MSE}}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2} \tag{2.11}$$

线性回归决定系数又称拟合优度 R^2，拟合优度表达式如下：

$$R^2=\frac{\mathrm{SSR}}{\mathrm{SST}} \tag{2.12}$$

一般来说，拟合优度 $R^2\geqslant 0.9$ 评估模型为优，$0.9>R^2\geqslant 0.8$ 评估模型为良好，$0.8>R^2\geqslant 0.6$ 评估模型为一般，$R^2\leqslant 0.6$ 评估模型为差，拟合优度越大，自变量对因变量的解释程度越高。

下面通过蛋糕销售的例子对一元线性回归进一步加深理解。通常蛋糕中的材料投入越多，蛋糕就越好吃，销售额就会越高。所以，蛋糕的成本与销售额之间的关系是直接相关的关系。通过表 2.3 中生产总成本以及销售总金额等部分数据，创建一个线性回归模型，通过分析蛋糕的成本与价格数据的线性关系，来预测任意尺寸蛋糕的利润。

表 2.3　蛋糕销售毛利表

样本	尺寸/英寸	销售总数	生产总成本/万元	销售总金额/万元
1	4	1 100	1.5	3.3
2	5	652	2	4.5
3	6	562	2.5	5
4	7	918	4	9
5	8	921	5	11.8
6	9	561	6	10
7	10	1011	6	18
8	11	780	7	17
9	12	630	8	15
10	13	765	8	20.5
11	14	729	9	21
12	15	692	9	22
13	16	668	9.9	22.6
14	17	698	10	25
15	18	590	11	22.9
16	19	550	12	23
17	20	510	13	24
18	21	500	14	26
19	22	498	15	30.8
20	23	480	16	32

成本支出显然是影响销售额的一个重要因素，应该以表 2.3 中的生产总成本为自变量 X，以销售总金额为因变量 Y，表示在二维坐标内就能够得到一个散点图。图 2.6 所示为销售毛利散点图。当然蛋糕的销售额不仅受成本费用影响，同时还受许多其他因素影响，这些影响因素

存在不确定性，比如蛋糕的尺寸、口味等。

如果想进一步分析生产成本和销售金额的关系，就可以利用一元线性回归方程式（2.2）做出图 2.7 所示的拟合直线。

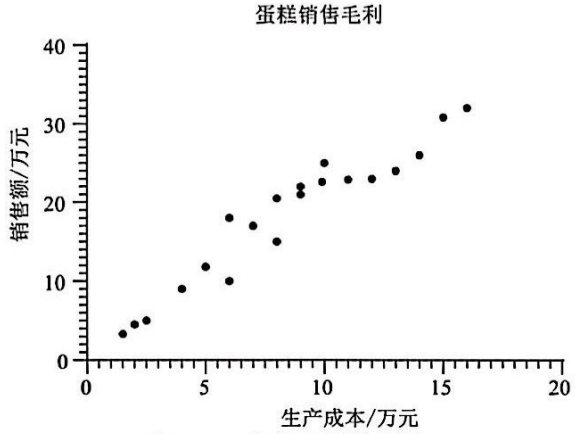

图 2.6 销售毛利散点图

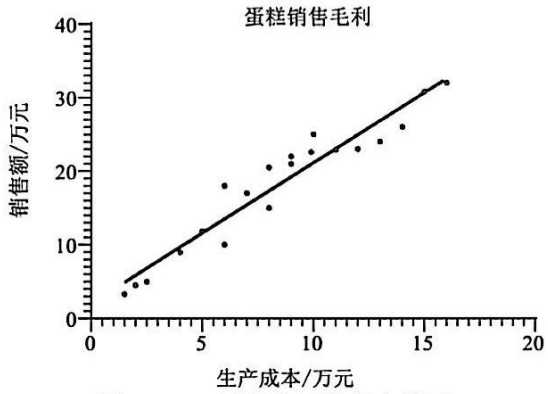

图 2.7 一元线性回归拟合模型

图 2.7 中的这条拟合直线代表的并不是真实数据，是通过表 2.3 中的样本数据$(x, y)=\{(1.5,3.3),(2,4.5),(2.5,5),\cdots,(16,32)\}$推算出来的，能粗略地表示毛利关系的一个线性方程。

将样本数据(x,y)代入式（2.3），采用最小二乘估计法的手段找出这个函数的最小值：$Q=(3.3-a-1.5b)^2+(3.3-a-1.5b)^2+(3.3-a-1.5b)^2+\cdots+(32-a-16b)^2$，通过标准方程组式（2.5）求解得到 $a=2.061$，$b=1.907$。根据拟合优度 R^2 公式计算得到：$R^2=SSR/SST=0.9216$，因为拟合优度大于 0.9，证明该评估模型为优。

由于拟合直线只是一个近似表达值，因此有很多点都没有落在直线上，那么这条直线拟合程度到底怎么样呢？最理想的回归直线应该尽可能从整体来看最接近各个实际观察点，如图 2.7 所示，分别从散点图的各个数据标记点，做一条平行于 y 轴的平行线，相交于图中直线，即散点图中各点到回归直线的垂直距离，平行线的长度在统计学中称为“偏差”（bias）或在预测中称为“残差”。误差是指分析结果的运算值和实际值之间的差，残差大小可以衡量预测的准确性。平行线的距离越大，“偏差”值就越大，通俗地理解偏差越小拟合度就越高。

通过提升算法模型的复杂度，使用 Sigmoidal 算法拟合，图 2.8 可以明显看出点到模型的沿 Y 轴的垂直距离更小了，即拟合度更高了，可见当模型复杂度上升时，“偏差”减小了。图 2.8 所示为用 95%的置信区间画出的偏差线。对样本的总体参数在一定概率下真值的取值范围（可靠范围）称为置信区间，其概率称为置信概率或置信度。简单地说，就是以测量值为中心，在一定范围内，真值出现在该范围内的概率。置信区间则是在某一置信度下，以测量值为中心，真值出现的范围。

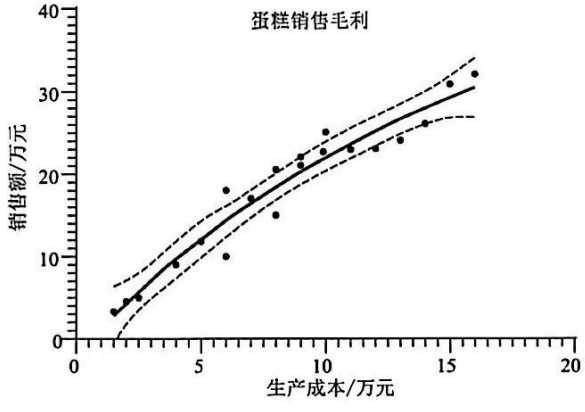

图 2.8 非线性回归

2.3.2　神经元

人类神经系统最基本的结构和功能单位是神经细胞，即神经元，根据神经元的机能，可分为感知（传入）神经元、联络（中间）神经元、运动（传出）神经元。神经元接受来自体内外的刺激时，将神经冲动传输到其他神经元。如图 2.9 所示，当人们收到外部的感官刺激时，兴奋在相邻神经元之间传递，“树突”作为输入端接收传入的兴奋信息，“细胞体”处理收到的信息并将其转换成输出激活，从“轴突”将输出激活传送给“突触”，突触小体释放“神经递质”，神经递质经自由扩散到达突触后膜，作用于下一个神经元，引起神经元的兴奋或抑制。由于神经递质经过突触间隙时是一个扩散的过程，所以兴奋经过突触时的速度较慢。如果神经递质多，扩散的神经递质的量增加，释放的神经递质总体速度会加快，则会提高兴奋程度、传递兴奋的速度也会更快。从上述例子神经元处理流程归纳为：树突（输入）、轴突（传输）、突触（输出）。

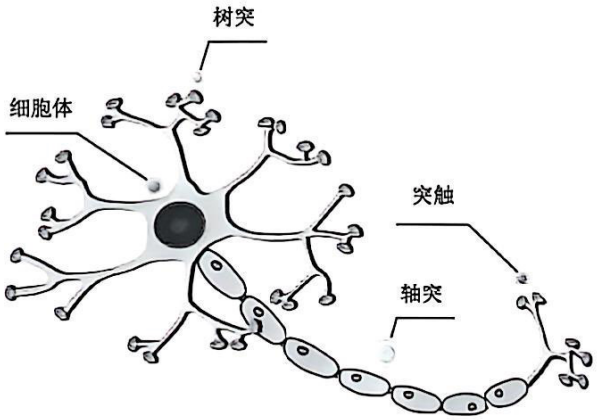

图 2.9　神经元

人工神经网络（artificial neural network，ANN）是一种模仿生物神经系统功能和结构的运算模型。从信息处理角度对神经网络进行抽象，建立某种模型。它由大量的神经元（节点）之间相互连接构成，每个神经元（节点）都有输入连接和输出连接，在神经网络中这些节点连接模拟了神经元“树突”与“突触”的行为，信息从一个神经元传递到另一个神经元，在传递过程中模拟了生物神经元之间传递的神经递质的量称为“权重”（weight），也就是训练出来的特征值，每一个连接都有“权重”，发送到每个连接的值要乘以这个权重并加上“偏置”，并在输出时作用了一个实现非线性结果的函数，称为激活函数（activation function）。

如图 2.10 所示，当 X 输入到神经元时，会乘以一个“权重”（weight），在训练过程中随机初始化权重，并通过反向求导更新这些权重。“偏置”主要是为了改变权重的范围，在添加偏置后，允许将激活函数向左或向右移位，偏置的存在是为了更好地拟合数据。权重的值是如何获得的呢？首先将权重初始化，然后通过神经元经过多轮迭代修改 W，直至训练完成训练出神经元的权重。

“激活函数”就是在人工神经网络的神经元上运行的函数，它通过一定的公式将输入的线性值逼近任何非线性值的函数，让神经网络可以应用到其他非线性模型中，如图 2.11 所示。

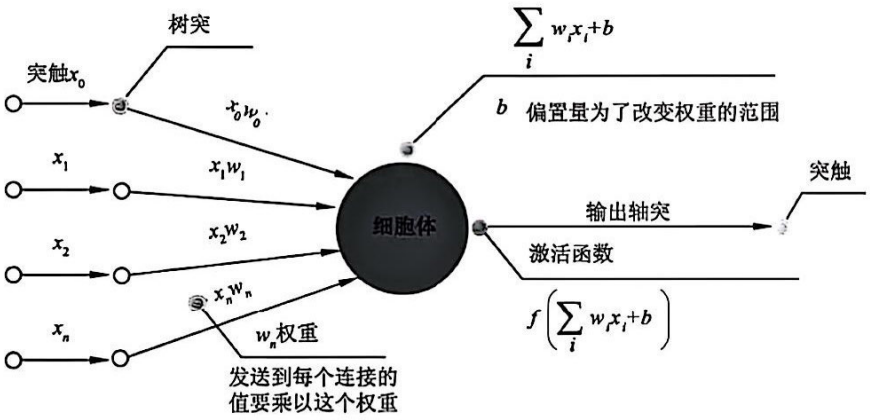

图 2.10 人工神经网络

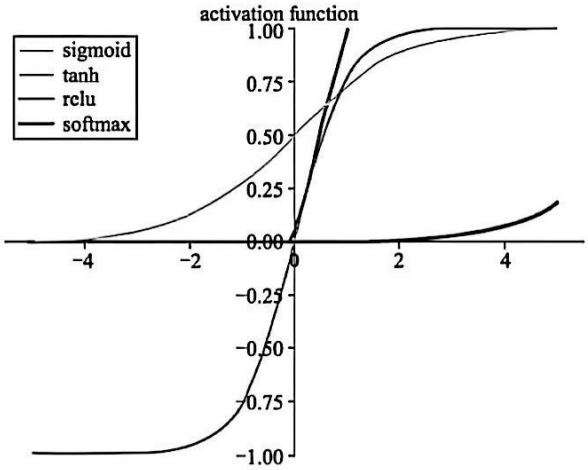

图 2.11 激活函数

常用的激活函数有 sigmoid、tanh、relu、softmax 和 relu 等，如图 2.12 所示，下面简单的介绍这些激活函数的区别。

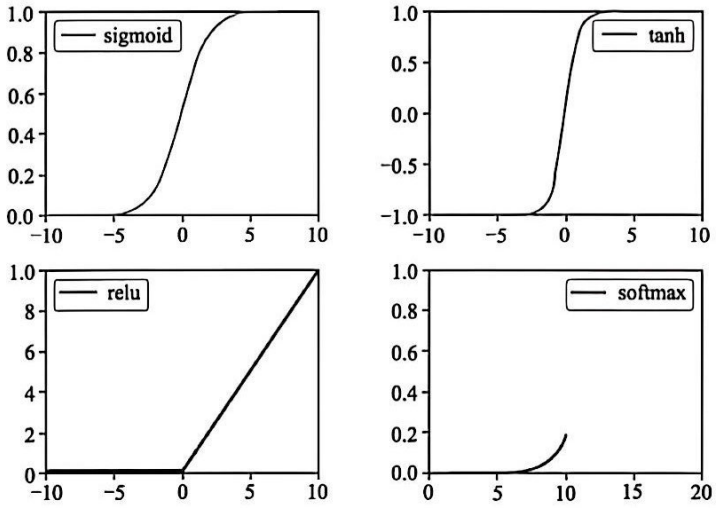

图 2.12 四种激活函数

（1）sigmoid 函数，取值范围为（0,1），在二分类的概率中常常用这个函数。在特征相差不大时效果较好，具有较好的对称性。sigmoid 最大的缺点是激活函数计算量大而且耗时，反向传播时，很容易就会出现梯度消失的情况，从而无法完成深层网络的训练。

（2）tanh 函数又称双曲正切函数，取值范围为（-1,1），它不会像 sigmoid 函数在导数从 0 开始很快就又趋近于 0，而造成梯度消失现象，虽然解决了原点对称问题，但是并没有彻底解决梯度消失现象。tanh 函数相较于 sigmoid 函数收敛速度更快，在实际应用中更好，在特征相差明显时的效果会很好，在循环过程中会不断扩大特征效果。

（3）relu 函数又称线性整流函数，用于隐藏层神经元输出，弥补了 sigmoid 函数以及 tanh 函数的部分梯度消失问题，而且收敛速度更快、计算速度更快。如图 2.7 所示，当输入值为负时，输出始终为 0。这个缺点导致神经元无法激活，也就是神经元不学习了，这种现象称为死神经元。为了解决 relu 函数的这个缺点，在 relu 函数的负半区间引入一个泄露（leaky）值，所以称为 leaky relu 函数。

（4）softmax 函数又称归一化指数函数，取值范围为[0,1]，用于多分类神经网络输出，是二分类函数 sigmoid 在多分类上的推广，在二分类问题时与 sigmoid 函数是一样的，softmax 函数的目的是将多分类的结果以概率的形式展现出来且概率和为 1。

2.3.3　人工神经网络

通过上面的介绍我们对神经元有了进一步的了解，每个神经元的权重和偏置，在神经网络训练期间根据错误进行更新。激活函数对线性组合进行非线性转换后生成输出。

当然，单个神经元无法执行高度复杂的任务，因此，使用多个神经元构成一个层，在这样的层中每个神经元都连接到第二层的所有神经元，即输入层与输出层，称其为“感知机”（perception）。如果在输入层与输出层之间加入新的层，则称为隐藏层，具有多个隐藏层的神经网络就构成了所谓的“多层感知机”（multi-layer perception，MLP）。

当输入层数据通过隐藏层到输出层的传播，信息沿着一个方向前进，没有反向传播过程称为“前向传播”算法（forward propagation)。

假设存在一个网络结构如图 2.13 所示，图中 X=0.2 为输入值，Y=0.6 为实际值。

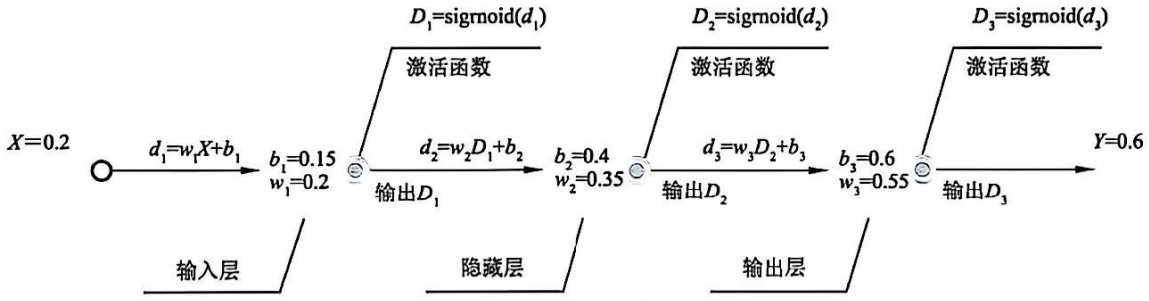

图 2.13　前向传播

初始权重：w_1=0.2、w_2=0.35、w_3=0.55；偏置设置为：b_1=0.15、b_2=0.4、b_3=0.6。图 2.13 中 d_i 为输入值乘以权重 w_i 并加上一个偏置量 b_i，D_i 为激活函数作用后的输出结果。根据计算公式（2.13）

$$\hat{y}=f\left(\sum_{i}^{n}w_i x_i+b\right) \tag{2.13}$$

进行前向传播计算

$$f\left(w_3 \cdot f\left(w_2 \cdot f\left(w_1 \cdot X+b_1\right)+b_2\right)+b_3\right)$$

分别求出输入层、隐藏层、输出层的结果：

$$d_1=w_1\times X+b_1=0.2\times 0.2+0.15=0.19$$

$$D_1=\text{sigmoid}(d_1)=\frac{1}{1+e^{-d_1}}=0.547\ 357\ 6$$

$$d_2=w_2\times D_1+b_2=0.35\times 0.547\ 357\ 6+0.4=0.591\ 575$$

$$D_2=\text{sigmoid}(d_2)=\frac{1}{1+e^{-d_2}}=0.643\ 726$$

$$d_3=w_3\times D_2+b_3=0.55\times 0.643\ 726+0.6=0.954$$

$$D_3=\text{sigmoid}(d_3)=\frac{1}{1+e^{-d_3}}=0.721\ 9$$

至此前向传播过程结束，得到输出值为 D_3=0.721 9，与实际值 Y=0.6 有一定的误差，由式（2.10）计算出均方误差 MSE=$E(Y-D_3)^2$=(0.6−0.721 9)^2=0.014 859 61。

神经网络中一旦获取了单次迭代的输出值，就可以计算网络的错误 MSE。把这个错误反馈给网络，以及损失函数的梯度来更新网络的权重。权重更新后可以减少后续迭代中的错误。使用损失函数梯度进行权重的更新称为反向传播(back propagation)。在反向传播中，网络的运动是反向的，误差随梯度从外层流入，穿过隐含层，权重被更新。

如图 2.14 所示，对误差进行反向传播求导，根据求导结果将初始化分配给每个节点的权重进行更新。

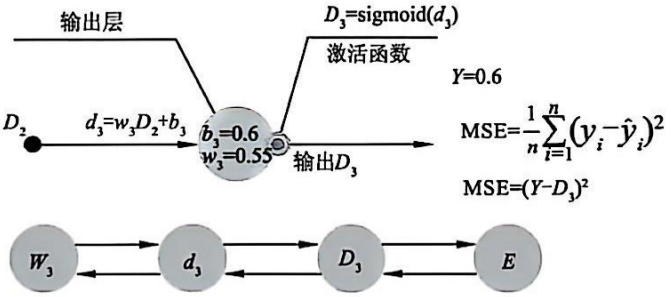

图 2.14 反向传播

输出层到隐含层的权重更新，根据链式法对 w_3 求导：

$$\frac{\partial E}{\partial w_3}=\frac{\partial E}{\partial D_3}\cdot\frac{\partial D_3}{\partial d_3}\cdot\frac{\partial d_3}{\partial w_3} \tag{2.14}$$

$$\frac{\partial E}{\partial D_3}=\left|2\left(Y-D_3\right)\right|=0.243\ 8$$

$$\frac{\partial D_3}{\partial d_3}=\frac{-1\bullet(-e^{-d_3})}{(1+e^{-d_3})^2}=0.200\ 76$$

$$\frac{\partial d_3}{\partial w_3}=D_2=0.643\ 726$$

$$\frac{\partial E}{\partial w_3}=\frac{\partial E}{\partial D_3}\bullet\frac{\partial D_3}{\partial d_3}\bullet\frac{\partial d_3}{\partial w_3}=0.031\ 51$$

通过整体误差 MSE 对 w_3 的偏导值，下面进行对 w_3 权重进行更新，设更新后的权重为 w_3^{new}，学习速率选择 η=0.5，由式（2.15）计算出新的权重 w_3^{new}。

$$w_3^{\text{new}}=w_3-\eta\bullet\frac{\partial E}{\partial w_3} \tag{2.15}$$

$$w_3^{\text{new}}=0.55-0.5\times0.031\ 51=0.534\ 245$$

学习速率（learning rate，η）决定着目标函数能否收敛到局部最小值以及何时收敛到最小值。学习速率选择必须合理，不能过高或过低。如何确定学习速率在很多文献中都有详细描述，本章不做介绍。

根据输出层到隐含层的权重更新方法，根据链式法则分别对隐含层与输入层的权重求导：

$$\frac{\partial E}{\partial w_2}=\frac{\partial E}{\partial D_3}\bullet\frac{\partial D_3}{\partial d_3}\bullet\frac{\partial d_3}{\partial D_2}\bullet\frac{\partial D_2}{\partial d_2}\bullet\frac{\partial d_2}{\partial w_2}$$

$$\frac{\partial E}{\partial w_2}=2(Y-D_3)\bullet\frac{e^{-d_3}}{(1+e^{-d_3})^2}\bullet w_3\bullet\frac{e^{-d_2}}{(1+e^{-d_2})^2}\bullet D_1$$

$$\frac{\partial E}{\partial w_2}=0.003\ 379\ 32$$

$$\frac{\partial E}{\partial w_1}=\frac{\partial E}{\partial D_3}\bullet\frac{\partial D_3}{\partial d_3}\bullet\frac{\partial d_3}{\partial D_2}\bullet\frac{\partial D_2}{\partial d_2}\bullet\frac{\partial d_2}{\partial D_1}\bullet\frac{\partial D_1}{\partial d_1}\bullet\frac{\partial d_1}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1}=2(Y-D_3)\bullet\frac{e^{-d_3}}{(1+e^{-d_3})^2}\bullet w_3\bullet\frac{e^{-d_2}}{(1+e^{-d_2})^2}\bullet w_2\bullet\frac{e^{-d_1}}{(1+e^{-d_1})^2}\bullet X$$

$$\frac{\partial E}{\partial w_1}=0.000\ 107\ 07$$

由式（2.15）分别求出 w_1、w_2 的新权重：w_2^{new}=0.348 31、w_1^{new}=0.199 946 46。至此对误差进行反向传播求导完成，根据更新后的权重，重新计算输出 D_3^{new}=0.719 865，第一次迭代之后，原来的均方误差 MSE = 0.014 859 61 下降到 MSE_{new} = 0.014 367 62，通过反向传播求导多次迭代后求出最终模型的权重。

在前向传播与反向传播的例子中所使用的激活函数都是 sigmoid 函数，如图 2.15 所示，从求导结果可以看出，sigmoid 导数的取值范围为 0 ~ 0.25。

在多个隐藏层的神经网络中，如果初始化节点的权重较小，那么各个层次的相乘都是 0 ~ 1 的小数，而激活函数 sigmoid 的导数也是 0 ~ 1，函数在反向传播过程中连乘，随着隐藏层数目的增加，结果变得越来越小，这种现象称为“梯度消失”（vanishing gradient）。

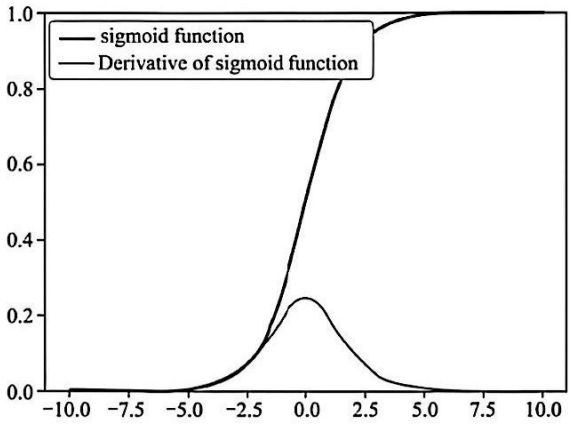

图 2.15 sigmoid 函数求导

“梯度爆炸”（exploding gradient）与梯度消失正好相反，在反向传播过程中，初始化节点的权重非常高，权重大到乘以激活函数的导数都大于 1，那么连乘后，可能会导致求导的结果很大，而其他节点的权重显得微不足道。

2.3.3 卷积神经网络

卷积神经网络（convolutional neural networks，CNN）大部分用于处理图像数据。图 2.16（a）所示图片大小为 200×200 像素的黑白图片，每个像素点只有一个值，总的数值个数为 40 000 个特征。

图 2.16（b）所示为彩色图片，彩色图片由 RGB 三通道组成，意味着总的数值有 200×200×3=120 000 个特征的输入。如果将这张彩色图片作为神经网络输入，即神经网络当中与若干个神经元连接，假设第一个隐层是 10 个神经元，就会有 120 000×10 个权重参数。随着图像的大小增加，参数的数量变得非常大。这样的神经网络参数更新不仅需要大量的计算，而且很难达到理想效果，于是在 1974 年，Paul Werbos 提出了误差反向传导来训练人工神经网络，使得训练多层神经网络成为可能。1979 年，Kunihiko Fukushima（福岛邦彦）提出了 Neocognitron，卷积、池化的概念基本形成。

（a） （b）

图 2.16 对比图片

卷积神经网络由一个或多个卷积层、池化层以及全连接层等组成。与其他深度学习结构相比，卷积神经网络在图像等方面能够给出更好的结果。这一模型也可以使用反向传播算法进行

训练。相比较其他浅层或深度神经网络，卷积神经网络需要考量的参数更少，使之成为一种颇具吸引力的深度学习结构。通过图 2.17 可以更清晰地了解卷积神经网络的整个工作流程。卷积层的目的是提取输入的不同特征，某些卷积层可能只能提取一些低级的特征，如边缘、线条和角等层级，更多层的网络能从低级特征中迭代提取更复杂的特征。

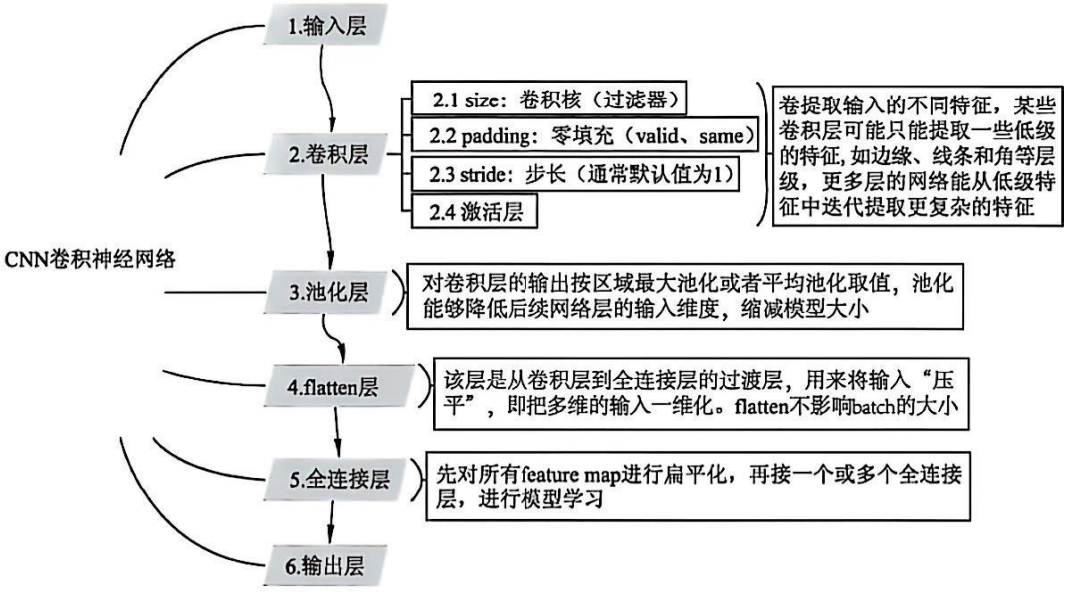

图 2.17　卷积神经网络

为了更详细地理解卷积层运算过程，下面用图 2.18 来表示更好理解些。假设图片长宽相等，N=5，F=3 × 3 为卷积核/过滤器大小。卷积层中重要的参数分别为 size、padding、stride。

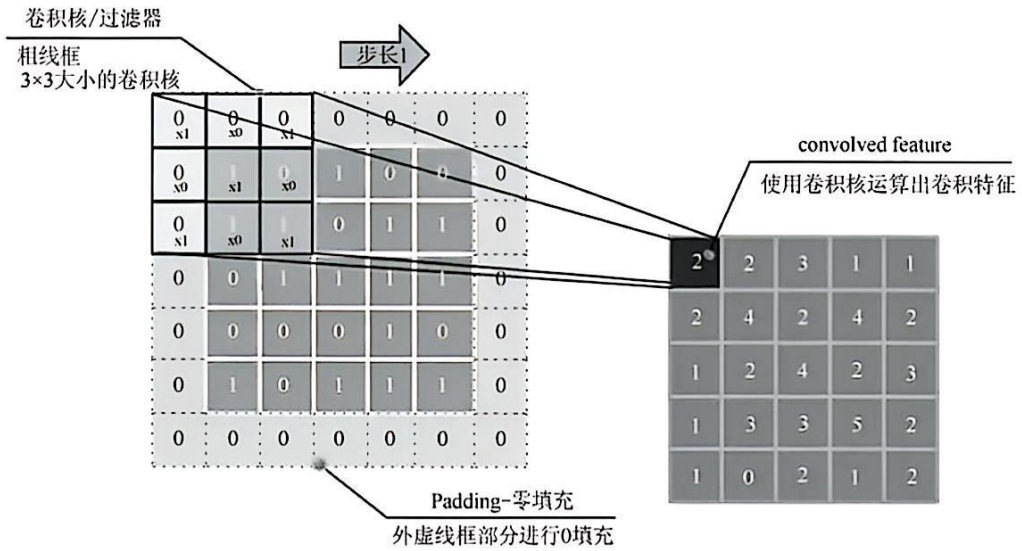

图 2.18　卷积运算过程

（1）size：卷积核又称过滤器，选择有 1 × 1、3 × 3、5 × 5。过滤器就像一个加权矩阵，将输入图像的一部分相乘，生成一个复杂的输出。一般情况下卷积核为奇数维度的过滤器，如果卷

积核不是奇数而是偶数个，就会造成最终计算结果非整数情况，这样的填充不均匀，所以卷积核默认都使用奇数维度大小。

（2）padding：又称零填充，如图 2.18 所示，原来的图片大小 N=5，卷积核 F=3 时，得到的卷积特征为 3×3。进行卷积之后的图片变小了，如果换一个更大的卷积核或者加入很多层卷积之后，得到的卷积特征可能越变越小，对于原始图片当中的边缘像素来说，只计算了一遍，对于中间的像素会有很多次过滤器与之计算，这样导致对边缘信息的丢失。padding 有两种形式填充：valid 不填充、same 输出大小与原图大小一致填充。所以为了避免上述情况，默认选择 same 填充卷积计算方式，图 2.18 中外框虚线部分进行了零填充，零在权重乘积和运算中对最终结果不造成影响，能够在避免图片增加额外干扰信息的情况下有效地解决边缘信息的丢失问题。

（3）stride：卷积核移动的步长，通常默认为 1。图 2.18 中看到的都是每次移动一个步长计算后的结果，如果将这个步长修改为 2 或者更大的数值，也会出现卷积运算后的卷积特征变小、边缘信息丢失的问题。

池化层，在卷积层之间引入池化层是很常见的，池化层主要对卷积层学习到的特征图进行亚采样（subsampling）处理。池化层有最大池化（max pooling）和平均池化（avg pooling）两种处理方式。最大池化是取窗口内的最大值作为输出，平均池化是取窗口内所有值的均值作为输出。池化层的主要意义是降低后续网络层的输入维度、缩减模型大小、提高计算速度、提高了图像的特征（feature map）的健壮性、防止过拟合。

全连接层与进行扁平化层（flatten），flatten 层主要是对卷积层、池化层所学习到的特征（feature map）进行扁平化，全连接层在整个卷积神经网络中起到“分类器”的作用。

小　结

线性回归（linear regression）：在统计学中，线性回归是利用称为线性回归方程的最小平方函数对一个或多个自变量和因变量之间关系进行建模的一种回归分析，在机器学习中是入门的学习模型，线性模型形式简单、易于建模，但蕴涵着机器学习中一些重要的基本思想。

测试数据（testing data）/验证集（validation set）：测试数据用于判断学得模型是否足够有效。