# 斯坦福大学公开 Machine Learning 学习笔记

## 第四章 神经网络(Neural Networks)

之前学习了线性回归和逻辑回归，这两个分别用于线性和非线性的分类问题。但是线性回归无法解决非线性的分类问题，因为即使是对复杂的多维分类面，它还是线性的，无法分类需要曲线（面）分类器的问题。逻辑（非线性）回归不适用于存在大量特征值的情况，因为计算量太大（$h_\theta(x) = g(\Theta^T x)$每次都要进行大量的矩阵运算）。
神经网络在解决复杂的非线性分类问题上，被证明是一种好得多的算法，即使特征空间很大。

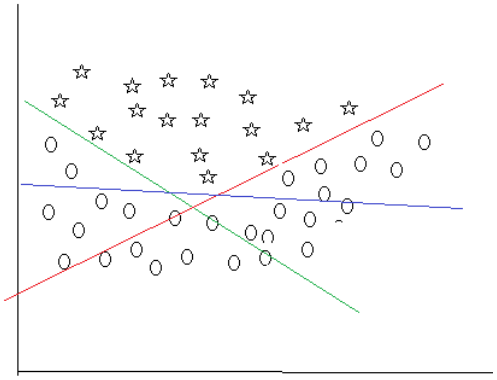


图 4-1 线性分类器无法解决非线性分类问题

## 神经网络模型基本概念介绍

神经网络算法模仿人类的大脑，由很多的“神经元”连接而成，每个“神经元”就是一个逻辑单元，每个神经单元有输入和输出，输入一般为数据集的特征值或者为上一个神经单元的输出，输出为神经单元处理（计算）后的输出或者为最终的计算结果。基本的网络结构如下图所示。

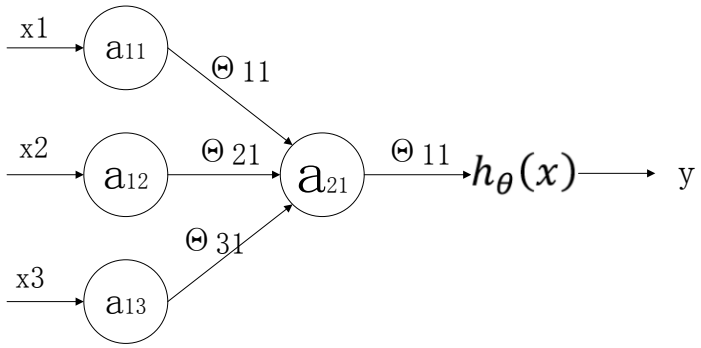



图 4-2 简单的神经网络算法网络结构

上图中的（x1, x2, x3）为一个数据单元的特征值，y 为经过计算特征值最终得出的分类结果，$a_{11}$等神经单元为根据前面单元计算得出的激励，计算激励的算法（激励函数）一般有感知器（perception）、线性单元（Linear unit）和 sigmod 单元（sigmod unit）。课堂中使用的是

sigmod 单元。$\Theta_{11}$等是前一个单元的输出对下一个单元激励的权重，即前一个单元对下一个单元的贡献权重。如果计算激励的算法为 sigmoid 函数的话，那么上面的神经网络就叫做一个由 sigmod 函数作为激励函数的神经网络算法。后面我们学习的神经网络都是使用 sigmod 函数作为单元的神经网络。
下面使用另外一张神经网络图介绍神经网络中的其他概念，并定义本章后续使用的符号。

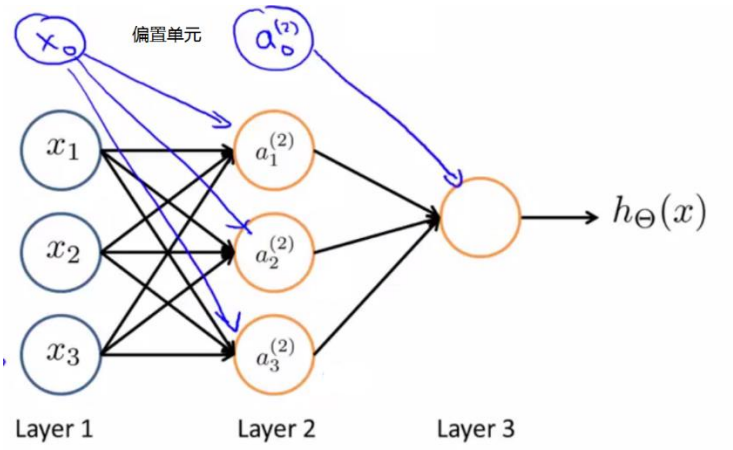



图 4-3 三层神经网络结构

图 4-3 是具有三层的神经网络，其中，Layer1 是接受原始数据的单元层，称为输入层，Layer3 是输出最终结果的单元层，称为输出层，位于输入层和输出层的单元层，我们一般是看不见的，称之为隐藏层(图中的 Layer2)，一个神经网络可以有多个隐藏层，图 4-3 是只有一个隐藏的神经网络。事实上，神经网络不一定是像 4-2 那样严格分层的，只是课堂中我们学习的是严格分层的神经网络，所以下面的符号定义都是针对这样的神经网络算法，更一般的定义可以参考 Mitchell 的《机器学习》中人工神经网络章节。下面给出部分符号定义：

$a_i^{(j)}$ :表示第 j 层的第 i 个激励值；

$\theta_{ij}^{(l)}$ :表示第 l 层的激励$a_j^{(l)}$对下一层(l+1 层)的激励$a_i^{(l+1)}$计算作用的权重；

$\theta^{(l)}$ :表示第 l 层对第 l+1 层所有权重的矩阵；

$a^{(l)}$ :表示第 l 层的所有激励的矩阵；

特别地，上图中的$x_0$，$a_0^{(2)}$,以及更多层的$a_0^{(3)}$, $a_0^{(4)}$…都是每一层的偏置单元，其中第一层的（$x_1$，$x_2$，$x_3$）其实就是（$a_1^{(1)}$,，$a_2^{(1)}$,，$a_3^{(1)}$,）。只不过作为输入层，其值就是原始数据的特征值，故如此表达。
如果在第 j 层有$s_j$个单元，第 j+1 层有$s_{j+1}$个单元，那么$\theta^{(j)}$是$s_{j+1} * (s_j + 1)$维的矩阵，其中考虑了 j 层的偏置单元。

# 激励计算-前向传播（Forward Propagation）

上一节我们说过本笔记中的神经网络算法都是使用的 sigmoid 函数作为激励函数的，那么图 4-3 中的激励计算公式如下。
第二层单元的计算:

$$a_1^{(2)} = g(\theta_{10}^{(1)}x_0 + \theta_{11}^{(1)}x_1 + \theta_{12}^{(1)}x_2 + \theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\theta_{20}^{(1)}x_0 + \theta_{21}^{(1)}x_1 + \theta_{22}^{(1)}x_2 + \theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\theta_{30}^{(1)}x_0 + \theta_{31}^{(1)}x_1 + \theta_{32}^{(1)}x_2 + \theta_{33}^{(1)}x_3)$$

第三层的计算：

$$h_\theta(x) = a_1^{(3)} = \mathrm{g}(\theta_{10}^{(2)}a_0^{(2)} + \theta_{11}^{(2)}a_1^{(2)} + \theta_{12}^{(2)}a_2^{(2)} + \theta_{13}^{(2)}a_3^{(2)})$$

假设：

$$z_1^{(2)} = \theta_{10}^{(1)}x_0 + \theta_{11}^{(1)}x_1 + \theta_{12}^{(1)}x_2 + \theta_{13}^{(1)}x_3)$$

其中 2 表示与第 2 层有关，1 表示与第 2 层的第一个单元$a_1^{(2)}$有关，同理可推，上面计算单元的公式可表示如下：

$$a_1^{(2)} = g(z_1^{(2)})$$
$$a_2^{(2)} = g(z_2^{(2)})$$
$$a_3^{(2)} = g(z_3^{(2)})$$

综上，我们可以得出在具有特征值 X=($\mathrm{x}_0$，$\mathrm{x}_1$，$\mathrm{x}_2$，$\mathrm{x}_3$)$^T$的情况下如何计算出预测值 y 的方法：

计算第 2 层：$Z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix} = \Theta^{(1)}X$，则$a^{(2)} = \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix} = g(Z^{(2)})$

然后将偏置单元$a_0^{(2)}$加入到$a^{(2)}$中（$a^{(2)} \in R^4$），计算第三层输出层单元：$Z^{(3)} = \Theta^{(2)}a^{(2)}$，

最终$h_\theta(X) = a^{(3)} = g\left(Z^{(3)}\right) = \mathrm{y}$。

以上依据上一层的激励和权值计算下一层激励，依次向前传播计算值，最终计算出预测结果的算法称为前向传播：从输入层的激励开始，然后向前传播给隐藏层计算，隐藏层计算后向前传播，最终计算出输出层的激励。

# 前向传播应用与体会

下面通过使用神经网络算法模拟同或运算公式来形象地理解前馈算法，以及神经网络在非线性分类问题中的应用。

下图是同或运算的结果分布图，圆圈代表真（1），三角代表假（0）。从图中我们可以看出，同或运算的真值分布是非线性的，一般的线性分类器是无法计算同或运算的结果。

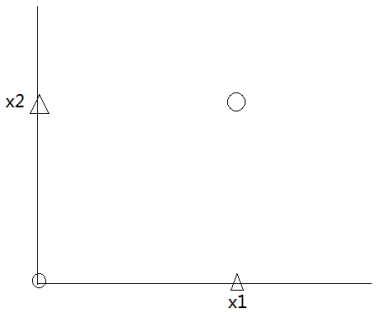



图 4-4 同或运算真值分布

直观地，我们可以直接通过简单的神经网络算法计算出逻辑与 AND、或 OR 和(NOT x1) AND

(NOT x2)的结果，其中简单的神经网络指只有输入层和输出层，没有隐藏层，这样的算法其实相当于逻辑分类算法，大家可以通过第二节的公式计算图 4-2 的假设函数体会出来。上面说到的三种运算的神经网络算法如下图 4-5。

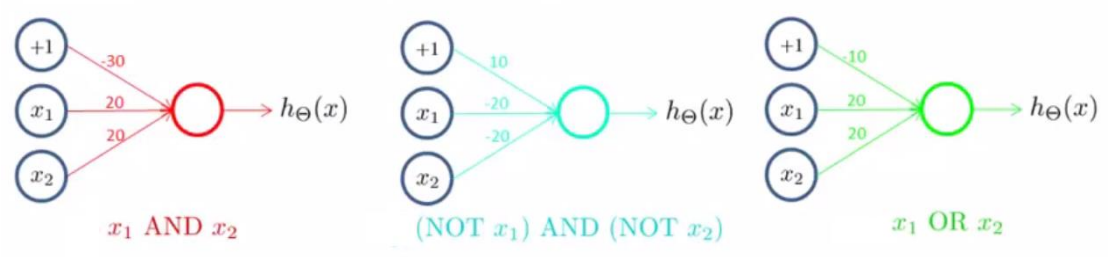



图 4-5 计算逻辑与、非和(NOT x1) AND (NOT x2)的神经网络

下面只给出神经网络是如何计算出逻辑与运算的，其他两个计算方式相似。

使用两层神经网络的假设函数为：

$$h_\theta(x) = \mathrm{g}(z^{(2)}) = \mathrm{g}(\theta_{10}^{(1)}x_0 + \theta_{11}^{(1)}x_1 + \theta_{12}^{(1)}x_2)$$

从逻辑分类算法我们了解到 sigmod 函数的图形如下：

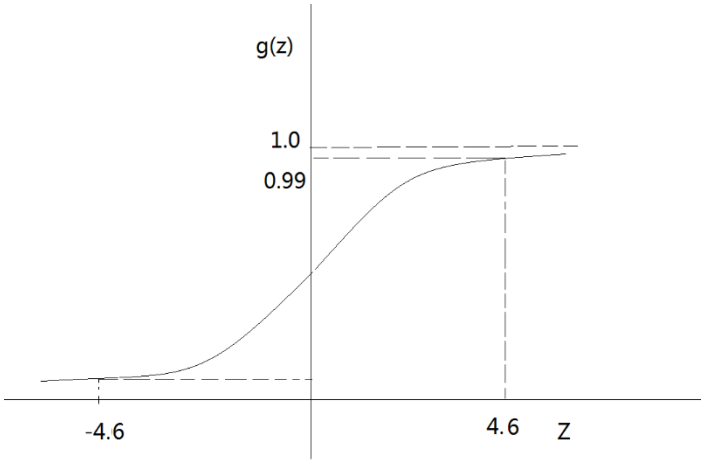



图 4-6 sigmoid 函数的图形

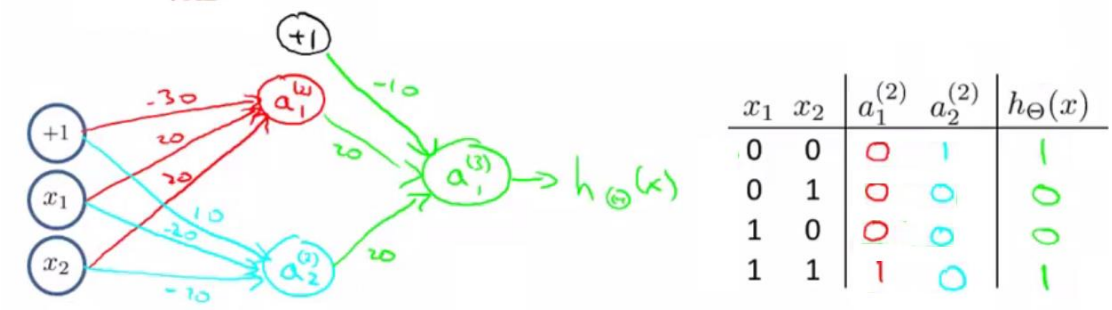
当 z=4.6 时 g(z)=0.99≈1，当 z=-4.6 时 g(z)=0.01≈0。那么对于逻辑与运算，只要我们指定的权值（$\theta_{10}^1$，$\theta_{11}^1$，$\theta_{12}^1$）使得 Z 满足以下情况时，就能很好的模拟逻辑与运算。

| $x_1$ | $x_2$ | z | g(z) |
|---|---|---|---|
| 0 | 0 | <=-4.6 | 0.01≈0 |
| 0 | 1 | <=-4.6 | 0.01≈0 |
| 1 | 0 | <=-4.6 | 0.01≈0 |
| 1 | 1 | >=4.6 | 0.99≈1 |

当我们对（$\theta_{10}^1$，$\theta_{11}^1$，$\theta_{12}^1$）取值（−30，20, 20）时，z 满足以上条件，得到的假设函数

$h_\theta(x) = \mathrm{g}(z^2) = \mathrm{g}(-30 + 20x_1 + 20x_2)$ 能够很好地模拟(预测)逻辑与运算。

下面我们在上面三个逻辑运算的基础上计算同或逻辑运算，体会下前向反馈在多层（含有隐藏层）神经网络中的作用，运算过程如图 4-7。




| $x_1$ | $x_2$ | $a_1^{(2)}$ | $a_2^{(2)}$ | $h_\Theta(x)$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 |

图 4-7 三层神经网络计算逻辑同或运算

由于同或运算比之前的三个逻辑运算要复杂，为了处理同或的复杂性，我们需要添加神经网络的层数，从而有更多的单元数。这就好比人类的大脑，处理越复杂的问题，就得调动更多的神经元，更多的神经元之间传递并交流信息，才能将事问题的所有复杂条件考虑清楚，从而给出解决方案。当然，更多的单元数，也意味着更多的计算量。图中我们添加了一个隐藏层，引入了两个神经元。将图 4-7 与图 4-5 对比，我们可以看出，其实计算$a_1^{(2)}$的过程是在计算逻辑与，计算$a_2^{(2)}$就是在计算(NOT x1) AND (NOT x2)，最后计算的$a_1^{(3)}$即计算或运算。通过图右侧的计算真值表可以看出该神经网络计算出了正确的结果。整个过程就是先在第一层计算出一些特征，然后下一层再计算出一些复杂的特征，然后是更复杂的特征，最终这些特征别传递给最后一层，计算出最终的结果。从图中，我们可以深刻的体会到前向传播时将分类器的复杂特征分散到不同的层进行计算，这样便减少了一次计算的复杂性，却又保证了计算的正确性，而不像逻辑分类器，需要一次计算所有的特征，从而一次需要大量的计算。至于图中为什么这样计算便会产生正确的结果，这不是神经网络算法关心的问题，对于一些复杂的学习，神经网络最终的学习结果为什么有效目前是无法理解的。最后，对于我们如何取得网络中的权值的问题，这是下篇神经网络笔记的重点，方法是反向传播算法。

# 多类别分类（Multiclass Classification）

与逻辑回归算法处理多类别分类问题不同，神经网络可以通过一次计算，得出分类结果。为了解决多类别问题，我们可以让神经网络的输出层最终输出一个值向量。假如我们需要分类的数据可能是四种类别中的一种，我们可以让输出层的单元数为 4，这样输出的值为一个四维的向量。神经网络的计算过程就如下图这样：

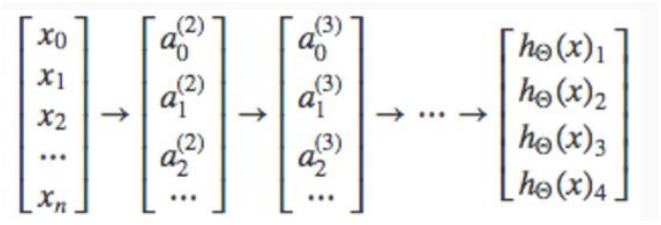



图 4-8 神经网络计算四元分类过程

如果数据属于第一种类别，我们的输出为$h_\theta(X) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$，如果为第二种，输出为$h_\theta(X) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$，

第三种第四种也分别为$\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$，$\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$。根据同样的效果，我们可以将真实的分类结果转换为$\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$，$\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$，$\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$，$\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$。哪一位结果是 1，就表示预测的类别为那一种。