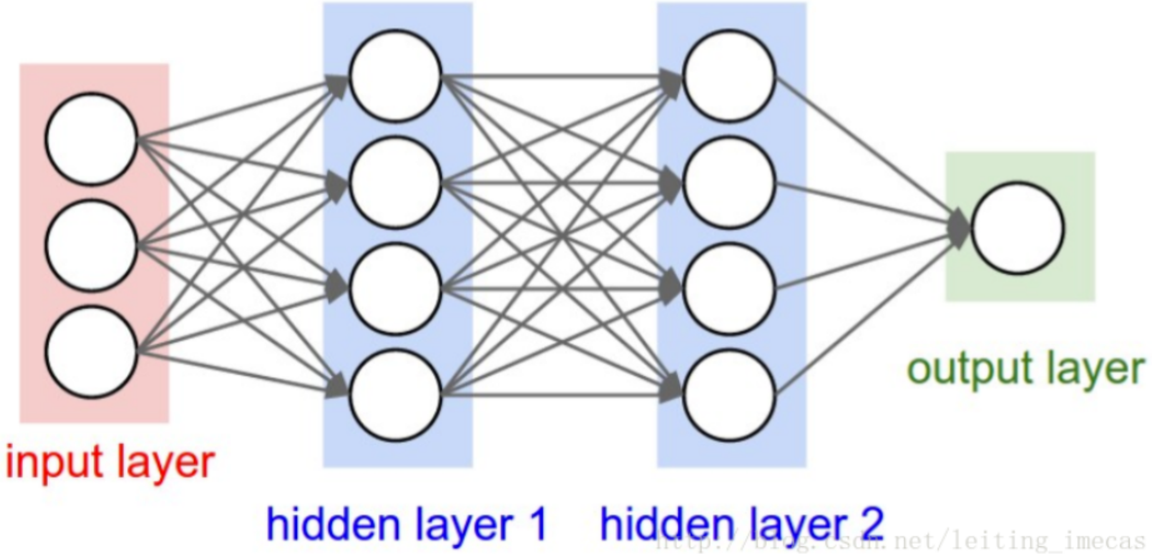
神经网络

- 首先介绍一些常用的神经网络
  - 人工神经网络(ANN)
  - 卷积神经网络(CNN)
  - 循环神经网络(RNN)
  - 生成对抗网络(GAN)

基本结构

- 神经网络的结构如下



- 通常一个神经网络由一个输入层(input layer)，多个隐藏层(hidden layer)和一个输出层(output layer)构成。
- 每个圆圈看做是一个神经元，神经元之间有权值。

从逻辑回归到神经元

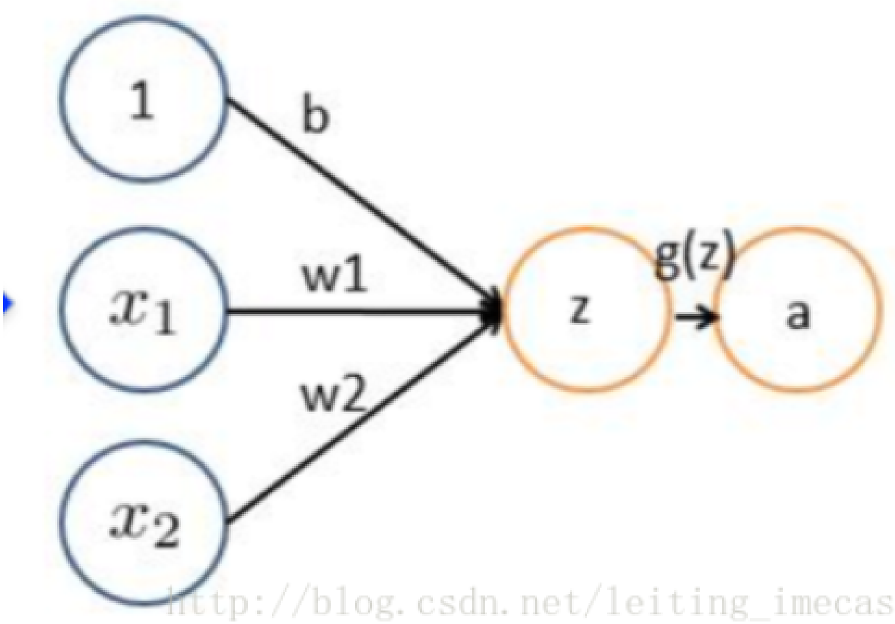
- LinearRegression模型：

$$y = b + w_1 x_1 + w_2 x_2$$

- sigmoid函数：

$$g(z) = \frac{1}{1 + e^{-z}}$$

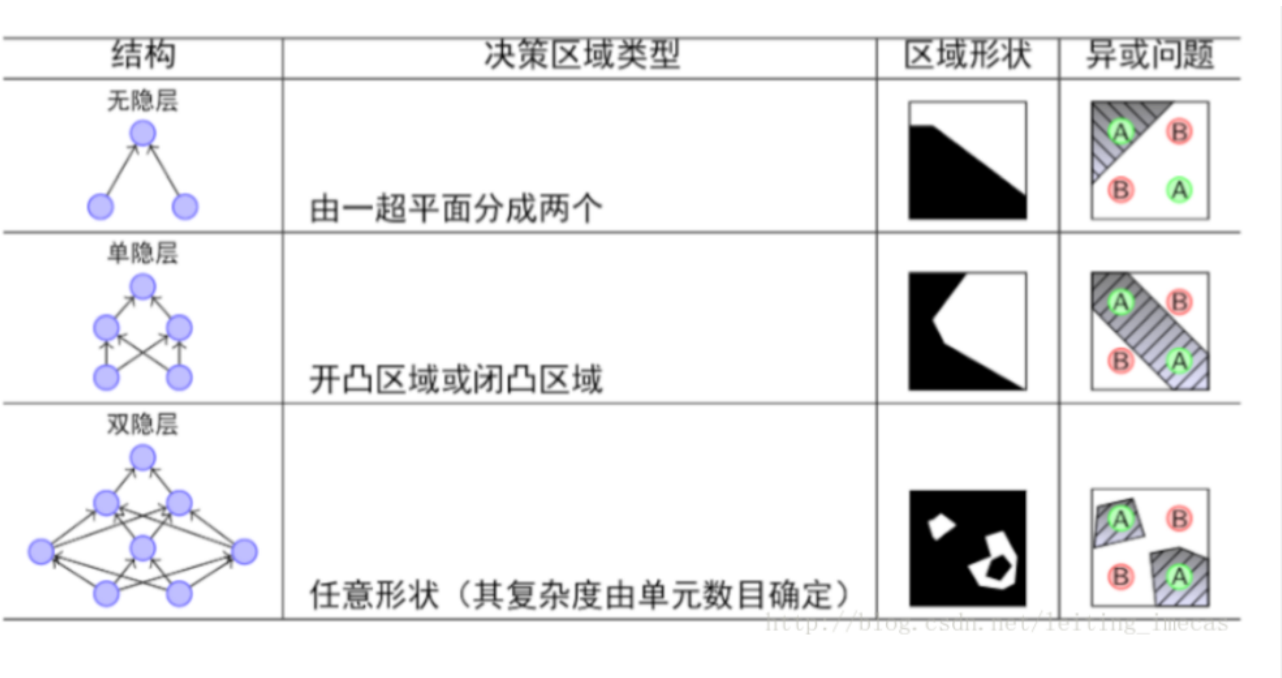
- 线性回归可以理解为：



是一个没有隐藏的单层神经网络，将输入进行线性组合，通过sigmoid激活函数，输出到输出层。

神经网络的重要性

- 神经网络在分类问题中的效果很好，LR或者线性SVM适合线性分割。如果数据非线性可分，LR需要做特征映射，增加高斯项或组合项；SVM需要选择核函数。
- 如果数据非线性可分，需要使用多分类解决。使用多个二分类器进行分类。
- 隐藏层的多少决定分类效果。



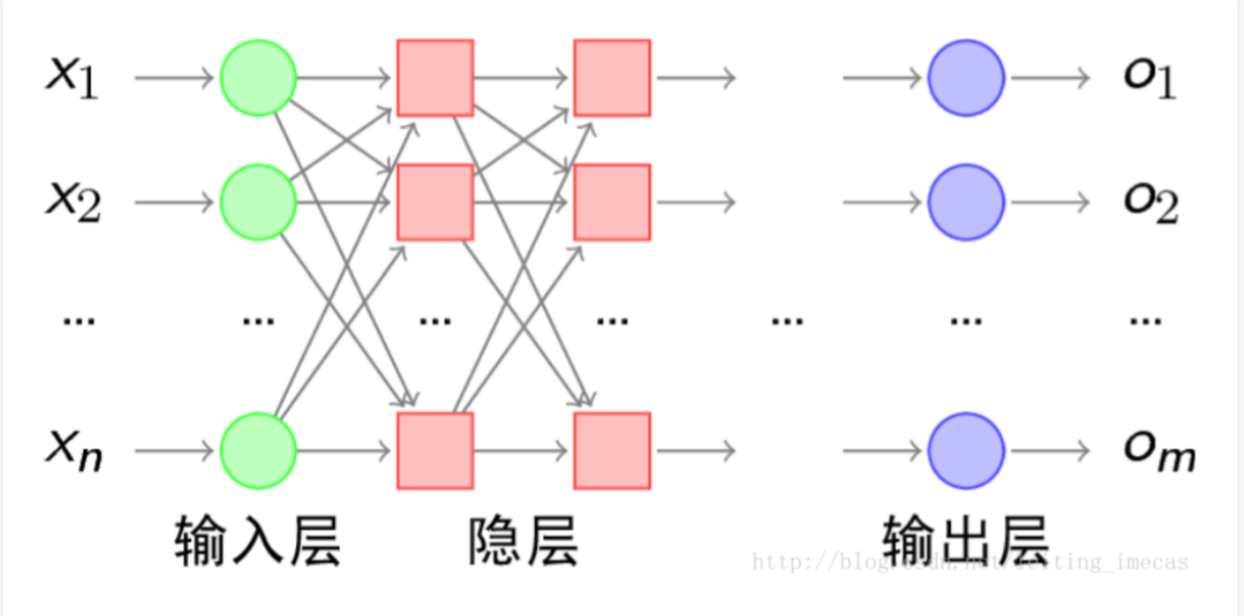
结构	决策区域类型	区域形状	异或问题
无隐层	由一超平面分成两个		
单隐层	开凸区域或闭凸区域		
双隐层	任意形状（其复杂度由单元数目确定）		

神经网络的过拟合

- 隐藏层的数量过多或者是神经元的数量过多会导致过拟合问题。
- 解决过拟合问题的方法是正则化或者dropout(暂时以某种概率去掉某些神经元)。

神经网络结构

- 基本结构



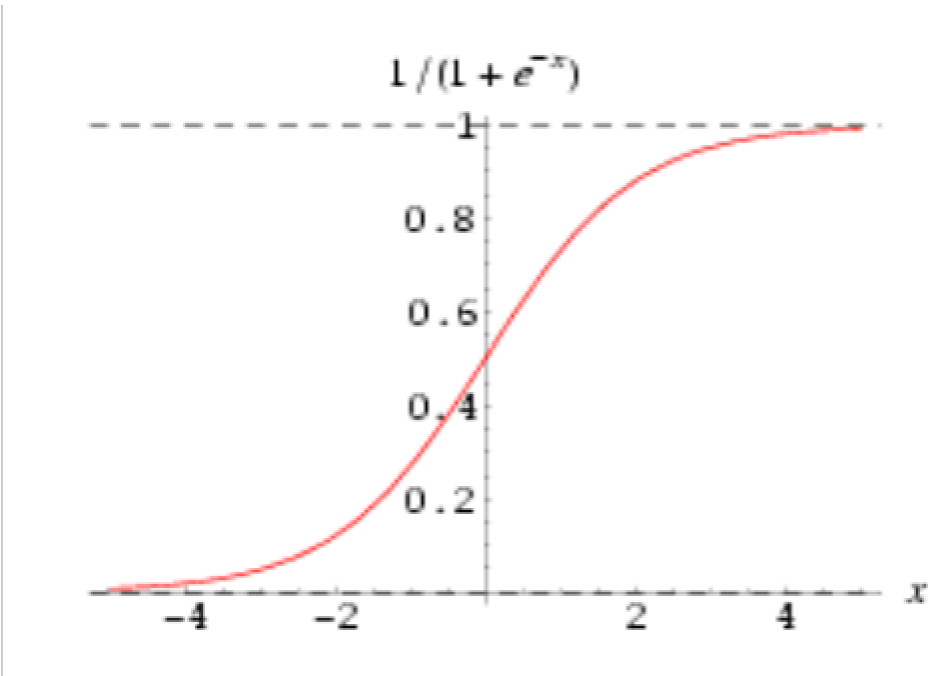
  n个输入，m个输出。

- 激活函数
  - 激活函数的作用就是将线性的输出转化为非线性，使得神经网络课可以任意逼近任何非线性函数，可以应用到众多的非线性模型中。

  常见的激活函数有：

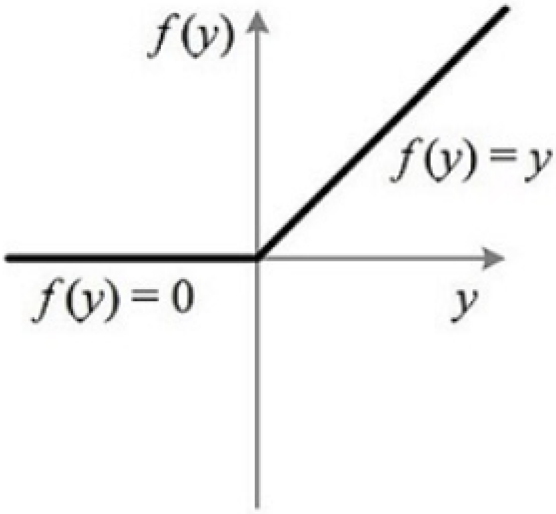
  - sigmoid函数

$$g(z) = \frac{1}{1+e^{-z}}$$



  - ReLu函数（用于隐藏层神经元输出）

$$ReLu(x) = x, x > 0$$
$$ReLu(x) = 0, x \leq 0$$

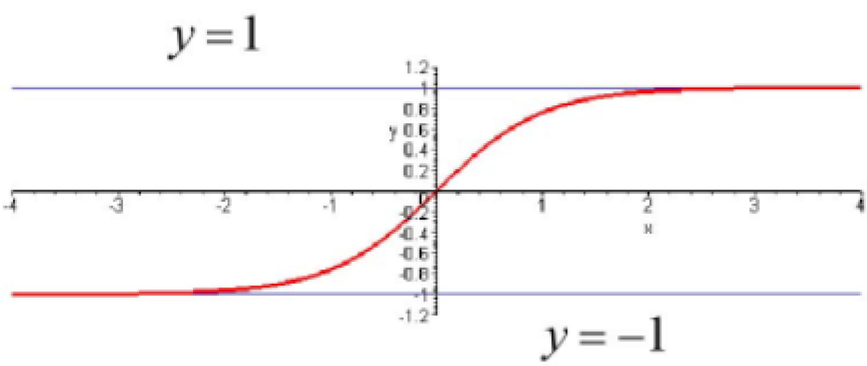


- Softmax函数 （多分类神经网络输出）

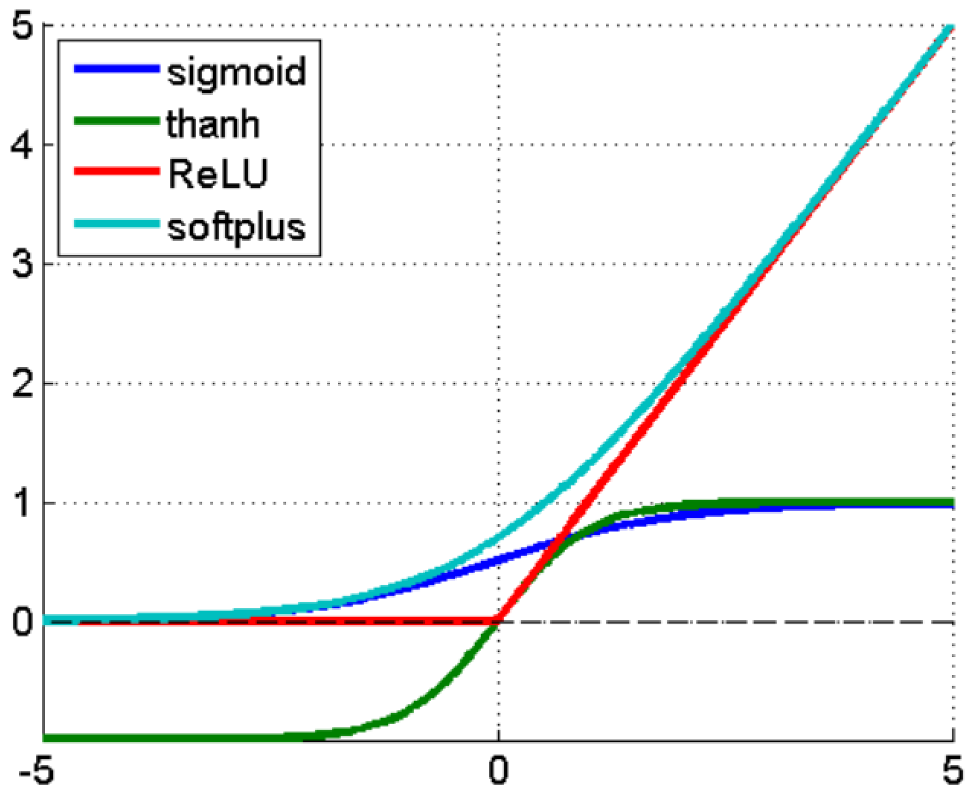
$$f(x) = \frac{e_j^i}{\sum_{i=1}^{m} e_j^i}$$

  - tanh函数

$$\begin{aligned} tanh(x) &= 2g(2x) - 1 \\ &= \frac{1 - e^{-2x}}{1 + e^{-2x}} \end{aligned}$$



- 几种激活函数的比较

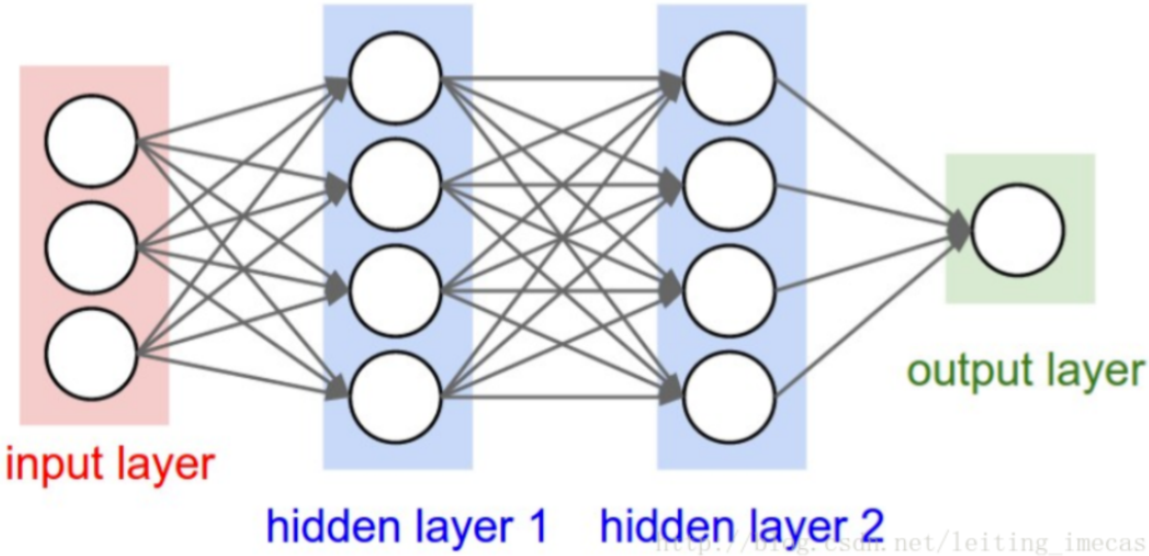


- ReLu函数的优点：对于随机梯度下降的收敛有加速作用，可以 通过对一个矩阵进行阈值计算得到。
- 缺点：ReLu单元在训练的时候有可能死掉。一个非常大的梯度流过一个ReLu神经元，更新参数后，这个神经元再也不会对任何数据有激活现象了，这个神经元的梯度永远是0。如果学习率过大，很有可能网络的40%的神经元都死掉了。
- Leaky Relu
  - Leaky ReLu是为了解决ReLu死亡的问题的，在ReLu中，x<0，函数值为0，而在Leaky ReLU中则是一个很小的梯度值，比如0.01。
- 除过Leaky ReLu之外，还有PReLu，Maxout。

BP算法(Backpropagation反向传播)

- 主要思想是：
  - 将训练集数据输入到ANN的输入层，经过隐藏层，最后达到输出层并输出结果，这是ANN的前向传播过程；
  - 由于ANN的输出结果与实际结果有误差，则计算估计值与实际值之间的误差，并将该误差从输出层向隐藏层反向传播，直至传播到输入层；
  - 在反向传播的过程中，根据误差调整各种参数的值；不断迭代上述过程，直至收敛。

    通过一个例子我们先看看前向传播：



- 首先通过线性组合，得到

  $$a = W^T X + b$$

  a是每一层的输出，然后通过激活函数，比如sigmoid函数，得到

  $$g(a) = \frac{1}{1+e^{-a}}$$

  代入进去，会得到第一层的输出，也就是第二层的输入，对第二层继续使用线性组合，通过激活函数，依次类推，直到到达输出层。

  这就是前向传播。

- 下面介绍一下反向传播：
  - 回归问题经常使用MSE作为损失函数，分类问题使用交叉熵作为损失函数。
  - 我们通过前向传播得到的预测值与真实值的差别较大，需要通过梯度下降算法更新参数，从而减小损失函数的值，使预测值接近真实值。
  - 反向传播算法是通过代价函数对参数θ求导，从而更新参数。
  - 首先介绍一下损失函数：
    - 输出层采用Logistic Regression

      $$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1-(h_\Theta(x^{(i)}))_k)\right]$$

    - 输出层采用softmax Regression

      $$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} 1\left\{y^{(i)}=k\right\} \log \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K}\exp(\theta^{(j)\top} x^{(i)})}\right]$$

    - 加入正则项之后的损失函数
      - Logistic Regression loss function regularization

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

- Softmax Regression loss function regularization

$$J(\theta) = -\left[\sum_{i=1}^{m}\sum_{k=1}^{K} 1\left\{y^{(i)} = k\right\} \log \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K}\exp(\theta^{(j)\top} x^{(i)})}\right] + \frac{\lambda}{2}\sum_{k=1}^{K}\sum_{j=1}^{n}\theta_{kj}^2$$

- 先定义一些标记
  - L：神经网络总层数
  - S：第`l`层神经网络单元的个数，不包括`bias`
  - k：第`K`个输出单元
  - θ：第`l`层到第`l+1`层的权值矩阵的`i`行`j`列
  - Z：第j层第`i`个神经元的输入值
  - ai：第`j`层第`i`个神经元的输出值
  - `a(j) = g(Z(j))`
- 输出层

  这是输出层的函数值

$$h_\Theta(x) = a^{(L)} = g(z^{(L)})$$

最后一层的输入值

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

输出层的代价函数对θ求导

$$\frac{\partial}{\partial \Theta_{i,j}^{(L-1)}} J(\Theta) = \frac{\partial J(\Theta)}{\partial h_\theta(x)_i}\frac{\partial h_\theta(x)_i}{\partial z_i^{(L)}}\frac{\partial z_i^{(L)}}{\partial \Theta_{i,j}^{(L-1)}} = \frac{\partial J(\Theta)}{\partial a_i^{(L)}}\frac{\partial a_i^{(L)}}{\partial z_i^{(L)}}\frac{\partial z_i^{(L)}}{\partial \Theta_{i,j}^{(L-1)}}$$

代价函数

$$loss(\Theta) = -y^{(i)}\log(h_\Theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\Theta(x^{(i)}))$$

第一项求导：

$$\frac{\partial J(\Theta)}{\partial a_i^{(L)}} = \frac{a_i^{(L)} - y_i}{(1-a_i^{(L)})a_i^{(L)}}$$

第二项求导：

首先由下式得

$$
\begin{aligned}
\frac{\partial g(z)}{\partial z} &= -\left(\frac{1}{1+e^{-z}}\right)^2 \frac{\partial}{\partial z}(1+e^{-z}) \\
&= -\left(\frac{1}{1+e^{-z}}\right)^2 e^{-z}\,(-1) \\
&= \left(\frac{1}{1+e^{-z}}\right)\left(\frac{1}{1+e^{-z}}\right)(e^{-z}) \\
&= \left(\frac{1}{1+e^{-z}}\right)\left(\frac{e^{-z}}{1+e^{-z}}\right) \\
&= \left(\frac{1}{1+e^{-z}}\right)\left(\frac{1+e^{-z}}{1+e^{-z}} - \frac{1}{1+e^{-z}}\right) \\
&= g(z)\,(1-g(z))
\end{aligned}
$$

$$
\frac{\partial a_i^{(L)}}{\partial z_i^{(L)}} = \frac{\partial g(z_i^{(L)})}{\partial z_i^{(L)}} = g(z_i^{(L)})(1-g(z_i^{(L)})) = a_i^{(L)}(1-a_i^{(L)})
$$

第三项求导:

$$
\frac{\partial z_i^{(L)}}{\partial \Theta_{i,j}^{(L-1)}} = a_j^{(L-1)}
$$

综合上面三项式子可得

$$
\begin{aligned}
\frac{\partial}{\partial \Theta_{i,j}^{(L-1)}} J(\Theta) &= \frac{\partial J(\Theta)}{\partial a_i^{(L)}} \frac{\partial a_i^{(L)}}{\partial z_i^{(L)}} \frac{\partial z_i^{(L)}}{\partial \Theta_{i,j}^{(L-1)}} \\
&= \frac{a_i^{(L)} - y_i}{(1-a_i^{(L)})a_i^{(L)}} a_i^{(L)}(1-a_i^{(L)}) a_j^{(L-1)} \\
&= (a_i^{(L)} - y_i) a_j^{(L-1)}
\end{aligned}
$$

上面是对输出层的反向传播，下面接着对隐层进行反向传播

$$
\frac{\partial}{\partial \Theta_{i,j}^{(l-1)}} J(\Theta) = \frac{\partial J(\Theta)}{\partial a_i^{(l)}} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} \frac{\partial z_i^{(l)}}{\partial \Theta_{i,j}^{(l-1)}} \ (l = 2, 3, \ldots, L-1)
$$

$$
\frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} = \frac{\partial g(z_i^{(l)})}{\partial z_i^{(l)}} = g(z_i^{(l)})(1-g(z_i^{(l)})) = a_i^{(l)}(1-a_i^{(l)})
$$

$$\frac{\partial z_i^{(l)}}{\partial \Theta_{i,j}^{(l-1)}} = a_j^{(l-1)}$$

展开

$$\frac{\partial J(\Theta)}{\partial a_i^{(l)}} = \sum_{k=1}^{s_{l+1}} \left[\frac{\partial J(\Theta)}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial a_i^{(l)}} \right]$$

$$\frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} = a_k^{(l+1)}(1 - a_k^{(l+1)})$$

$$\frac{\partial z_k^{(l+1)}}{\partial a_i^{(l)}} = \Theta_{k,i}^{(l)}$$

求得递推式

$$\begin{aligned} \frac{\partial J(\Theta)}{\partial a_i^{(l)}} &= \sum_{k=1}^{s_{l+1}} \left[\frac{\partial J(\Theta)}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial a_i^{(l)}} \right] \\ &= \sum_{k=1}^{s_{l+1}} \left[\frac{\partial J(\Theta)}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} \Theta_{k,i}^{(l)} \right] \\ &= \sum_{k=1}^{s_{l+1}} \left[\frac{\partial J(\Theta)}{\partial a_k^{(l+1)}} a_k^{(l+1)}(1 - a_k^{(l+1)}) \Theta_{k,i}^{(l)} \right] \end{aligned}$$

定义第`l`层第`i`个节点的误差为：

$$\begin{aligned} \delta_i^{(l)} &= \frac{\partial}{\partial z_i^{(l)}} J(\Theta) \\ &= \frac{\partial J(\Theta)}{\partial a_i^{(l)}} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} \\ &= \frac{\partial J(\Theta)}{\partial a_i^{(l)}} a_i^{(l)}(1 - a_i^{(l)}) \\ &= \sum_{k=1}^{s_{l+1}} \left[\frac{\partial J(\Theta)}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial z_k^{(l+1)}} \Theta_{k,i}^{(l)} \right] a_i^{(l)}(1 - a_i^{(l)}) \\ &= \sum_{k=1}^{s_{l+1}} \left[\delta_k^{(l+1)} \Theta_{k,i}^{(l)} \right] a_i^{(l)}(1 - a_i^{(l)}) \end{aligned}$$

输出层的误差

$$
\begin{aligned}
\delta_i^{(L)} &= \frac{\partial J(\Theta)}{\partial z_i^{(L)}} \\
&= \frac{\partial J(\Theta)}{\partial a_i^{(L)}} \frac{\partial a_i^{(L)}}{\partial z_i^{(L)}} \\
&= \frac{a_i^{(L)} - y_i}{(1-a_i^{(L)})a_i^{(L)}} a_i^{(L)}(1-a_i^{(L)}) \\
&= a_i^{(L)} - y_i
\end{aligned}
$$

最终代价函数的偏导数为

$$
\begin{aligned}
\frac{\partial}{\partial \Theta_{i,j}^{(l-1)}} J(\Theta) &= \frac{\partial J(\Theta)}{\partial a_i^{(l)}} \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} \frac{\partial z_i^{(l)}}{\partial \Theta_{i,j}^{(l-1)}} \\
&= \frac{\partial J(\Theta)}{\partial z_i^{(l)}} \frac{\partial z_i^{(l)}}{\partial \Theta_{i,j}^{(l-1)}} \\
&= \delta_i^{(l)} \frac{\partial z_i^{(l)}}{\partial \Theta_{i,j}^{(l-1)}} \\
&= \delta_i^{(l)} a_j^{(l-1)} \\
&= \delta_i^{(l+1)} a_j^{(l)}
\end{aligned}
$$

总结

- 输出层误差

$$\delta_i^{(L)} = a_i^{(L)} - y_i$$

- 隐藏层误差

$$\delta_i^{(l)} = \sum_{k=1}^{s_{l+1}} \left[\delta_k^{(l+1)} \Theta_{k,i}^{(l)} \right] a_i^{(l)}(1-a_i^{(l)})$$

- 代价函数偏导项

$$\frac{\partial}{\partial \Theta_{i,j}^{(l-1)}} J(\Theta) = \delta_i^{(l)} a_j^{(l-1)}$$

即

$$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) = \delta_i^{(l+1)} a_j^{(l)}$$