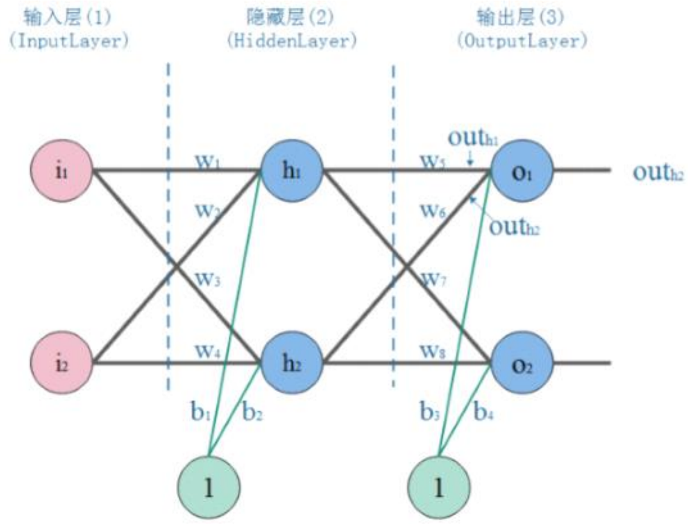
Class 5 人工神经网络

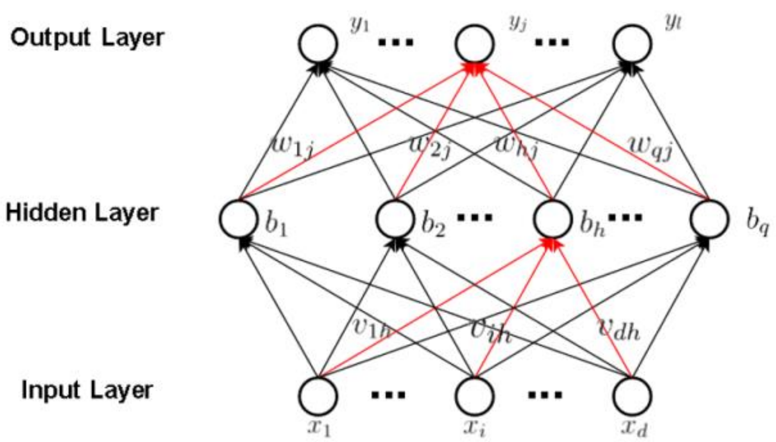
2018年11月27日　　9:03

- ## 神经网络的结构：



  - **输入层：**
    输入层对应的是输入实例的特征向量；
  - **隐藏层：**
    输出层是经过多层神经网络计算之后得出的计算结果向量
  - **输出层:**
    在整个神经网络结构的所有层次中，除了输入层和输出层以外的全都叫做隐藏层。

- ## 前向传播算法：



$$b_h = \delta\left(\sum_{i=1}^{d} \nu_{ih}x_i + \gamma_h\right)$$

$$\hat{y}_j = \delta\left(\sum_{h=1}^{q} w_{hj} b_h + \theta_j\right)$$

其中，

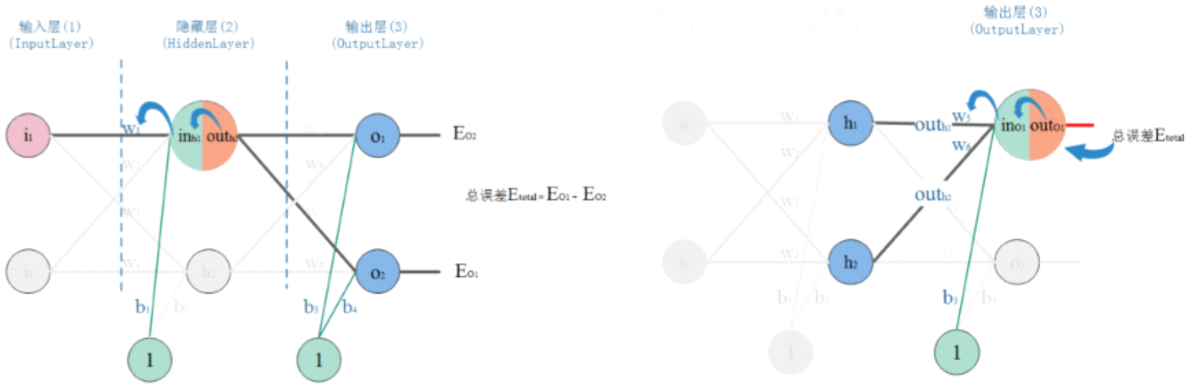
b_h 为隐藏层的输出，$\hat{y}_j$ 为输出层的输出；

v_{ih} 和w_{hj} 分别表示输入层到隐藏层的权重、隐藏层到输出层的权重；

γ_h 和θ_j 分别表示隐藏层和输出层的偏置。

- **当输入一个样例后，获得该样例的特征向量，再根据权向量得到后一层神经元的输入值，然后使用 Sigmoid 函数计算出每个神经元的输出，再将此输出作为下一层神经元的输入，依次类推，直到输出层。**这样的计算过程就叫做**前向传播算法。**

• 反向传播算法：



- **损失函数：**

$$E^{(k)} = \frac{1}{2}\sum_{j=1}^{l}\left(\hat{y}_j^{(k)} - y_j^{(k)}\right)^2$$

- **参数：**

$$v \in R^{d*q}, \gamma \in R^{q}, \omega \in R^{q*l}, \theta \in R^{l}$$

- **梯度计算：**

$$\frac{\partial E^{(k)}}{\partial v_{ih}}, \frac{\partial E^{(k)}}{\partial \gamma_h}, \frac{\partial E^{(k)}}{\partial \omega_{hj}}, \frac{\partial E^{(k)}}{\partial \theta_j}$$

1. 求$E^{(k)}$ 对于 w_{hj} 的梯度：

$$\frac{\partial E^{(k)}}{\partial \omega_{hj}} = \frac{\partial E^{(k)}}{\partial \hat{y}_j^{(k)}} \cdot \frac{\partial \hat{y}_j^{(k)}}{\partial (\beta_j + \theta_j)} \cdot \frac{\partial (\beta_j + \theta_j)}{\partial \omega_{hj}}$$

其中

$$\frac{\partial E^{(k)}}{\partial \hat{y}_j^{(k)}} = \left(\hat{y}_j^{(k)} - y_j^{(k)}\right)$$

,

$$\frac{\partial \hat{y}_j^{(k)}}{\partial(\beta_j + \theta_j)} = \delta'\left(\beta_j + \theta_j\right) = \delta\left(\beta_j + \theta_j\right) \cdot \left(1 - \delta\left(\beta_j + \theta_j\right)\right) = \hat{y}_j^{(k)} \cdot \left(1 - \hat{y}_j^{(k)}\right)$$

$$\frac{\partial(\beta_j + \theta_j)}{\partial \omega_{hj}} = b_h$$

定义:

$$error_j^{OutputLayer} = \frac{\partial E^{(k)}}{\partial(\beta_j + \theta_j)} = \frac{\partial E^{(k)}}{\partial \hat{y}_j^{(k)}} \cdot \frac{\partial \hat{y}_j^{(k)}}{\partial(\beta_j + \theta_j} = \left(\hat{y}_j^{(k)} - y_j^{(k)}\right) \cdot \hat{y}_j^{(k)} \cdot \left(1 - \hat{y}_j^{(k)}\right)$$

所以:

$$\frac{\partial E^{(k)}}{\partial \omega_{hj}} = error_j^{OutputLayer} \cdot b_h$$

1．求$\boldsymbol{E^{(k)}}$ 对于 $\boldsymbol{\theta_j}$ 的梯度:

$$\frac{\partial E^{(k)}}{\partial \theta_j} = \frac{\partial E^{(k)}}{\partial \hat{y}_j^{(k)}} \cdot \frac{\partial \hat{y}_j^{(k)}}{\partial(\beta_j + \theta_j)} \cdot \frac{\partial(\beta_j + \theta_j)}{\partial \theta_j} = error_j^{OutputLayer} \cdot 1$$

2．求$\boldsymbol{E^{(k)}}$ 对于 $\boldsymbol{v_{ih}}$ 的梯度:

$$\frac{\partial E^{(k)}}{\partial v_{ih}} = \sum_{j=1}^{l} \frac{\partial E^{(k)}}{\partial(\beta_j + \theta_j)} \cdot \frac{\partial(\beta_j + \theta_j)}{\partial b_h} \cdot \frac{\partial b_h}{\partial(\alpha_h + \gamma_h)} \cdot \frac{\partial(\alpha_h + \gamma_h)}{\partial v_{ih}}$$

其中

$$\frac{\partial E^{(k)}}{\partial(\beta_j + \theta_j)} = error_j^{OutputLayer} \qquad \frac{\partial(\beta_j + \theta_j)}{\partial b_h} = \omega_{hj}$$

$$\frac{\partial b_h}{\partial(\alpha_h + \gamma_h)} = \delta'(\alpha_h + \gamma_h) = \delta(\alpha_h + \gamma_h) \cdot \left(1 - \delta(\alpha_h + \gamma_h)\right) = b_h \cdot (1 - b_h)$$

$$\frac{\partial(\alpha_h + \gamma_h)}{\partial v_{ih}} = x_i^{(k)}$$

定义:

$$
\begin{aligned}
error_h^{HiddenLayer} &= \frac{\partial E^{(k)}}{\partial(\alpha_h+\gamma_h)} \\
&= \sum_{j=1}^{l} \frac{\partial E^{(k)}}{\partial(\beta_j+\theta_j)} \cdot \frac{\partial(\beta_j+\theta_j)}{\partial b_h} \cdot \frac{\partial b_h}{\partial(\alpha_h+\gamma_h)} \\
&= \sum_{j=1}^{l} error_j^{OutputLayer} \cdot \omega_{hj} \cdot \delta'(\alpha_h+\gamma_h) \\
&= \sum_{j=1}^{l} error_j^{OutputLayer} \cdot \omega_{hj} \cdot b_h \cdot (1-b_h)
\end{aligned}
$$

所以：

$$\frac{\partial E^{(k)}}{\partial v_{ih}} = error_h^{HiddenLayer} \cdot x_i^{(k)}$$

3．求$\boldsymbol{E}^{(\boldsymbol{k})}$ 对于 $\boldsymbol{\gamma_h}$ 的梯度：

$$\frac{\partial E^{(k)}}{\partial \gamma_h} = \sum_{j=1}^{l} \frac{\partial E^{(k)}}{\partial(\beta_j+\theta_j)} \cdot \frac{\partial(\beta_j+\theta_j)}{\partial b_h} \cdot \frac{\partial b_h}{\partial(\alpha_h+\gamma_h)} \cdot \frac{\partial(\alpha_h+\gamma_h)}{\partial \gamma_h} = error_h^{HiddenLayer} \cdot 1$$

- ## 算法流程：

Input: training set: $\mathcal{D} = \{(x^{(k)}, y^{(k)})\}_{k=1}^{m}$
            learning rate η
Steps：
1: initialize all parameters within (0,1)
2: repeat:
3:    for all $(x^{(k)}, y^{(k)}) \in \mathcal{D}$ do:
4:       calculate $y^{(k)}$
5:       calculate $error^{OutputLayer}$:
6:       calculate $error^{HiddenLayer}$:
7:       update v , θ , v and γ
8:    end for
9: until reach stop condition
Output: trained ANN

- **梯度更新：**

$$\omega_{hj} \coloneqq \omega_{hj} - \eta \cdot \frac{\partial E^{(k)}}{\partial \omega_{hj}}$$
$$\theta_j \coloneqq \theta_j - \eta \cdot \frac{\partial E^{(k)}}{\partial \theta_j}$$
$$v_{ih} \coloneqq v_{ih} - \eta \cdot \frac{\partial E^{(k)}}{\partial v_{ih}}$$
$$\gamma_h \coloneqq \gamma_h - \eta \cdot \frac{\partial E^{(k)}}{\partial \gamma_h}$$

其中 $\boldsymbol{\eta}$ 是学习率