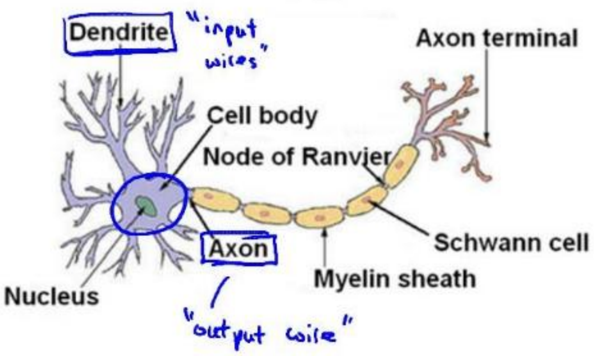
8.3 模型表示 1

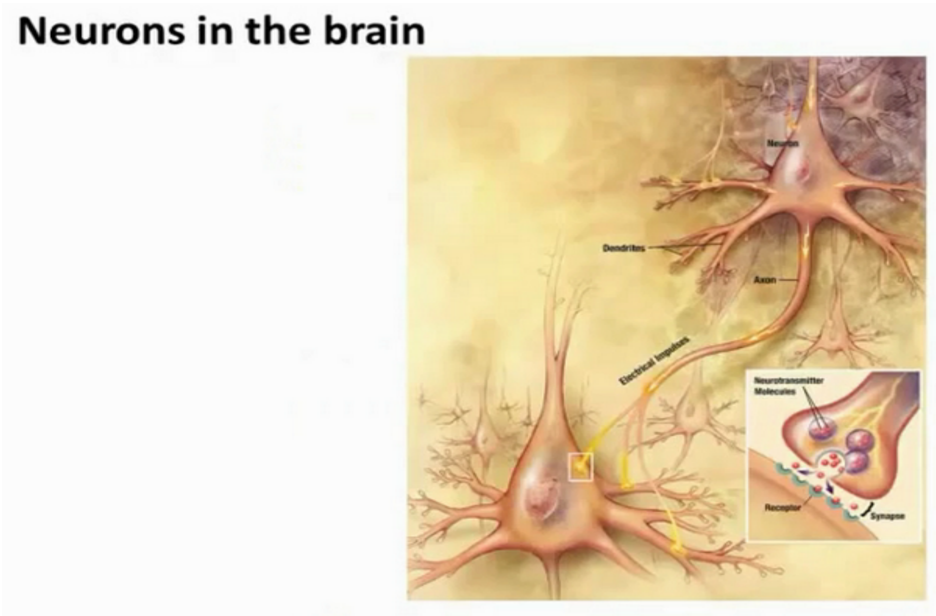
参考视频: 8 - 3 - Model Representation I (12 min).mkv

为了构建神经网络模型，我们需要首先思考大脑中的神经网络是怎样的？每一个神经元都可以被认为是一个**处理单元/神经核（processing unit/Nucleus）**，它含有许多**输入/树突（input/Dendrite）**，并且有一个**输出/轴突（output/Axon）**。神经网络是大量神经元相互链接并通过电脉冲来交流的一个网络。

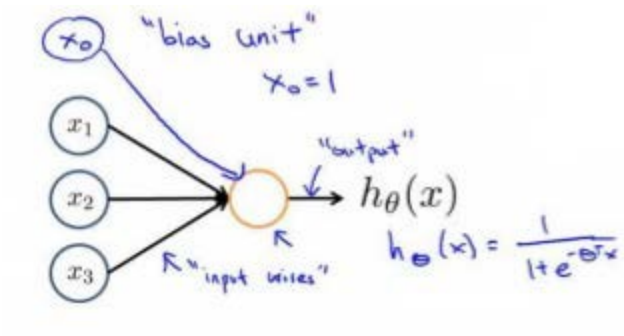


下面是一组神经元的示意图，神经元利用微弱的电流进行沟通。这些弱电流也称作动作电位，其实就是一些微弱的电流。所以如果神经元想要传递一个消息，它就会就通过它的轴突，发送一段微弱电流给其他神经元，这就是轴突。

这里是一条连接到输入神经，或者连接另一个神经元树突的神经，接下来这个神经元接收这条消息，做一些计算，它有可能会反过来将在轴突上的自己的消息传给其他神经元。这就是所有人类思考的模型：我们的神经元把自己的收到的消息进行计算，并向其他神经元传递消息。这也是我们的感觉和肌肉运转的原理。如果你想活动一块肌肉，就会触发一个神经元给你的肌肉发送脉冲，并引起你的肌肉收缩。如果一些感官：比如说眼睛想要给大脑传递一个消息，那么它就像这样发送电脉冲给大脑的。

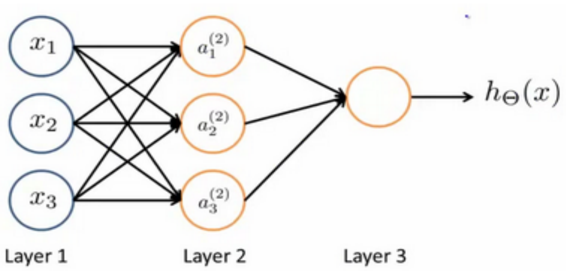


神经网络模型建立在很多神经元之上，每一个神经元又是一个个学习模型。这些神经元（也叫激活单元，**activation unit**）采纳一些特征作为输出，并且根据本身的模型提供一个输出。下图是一个以逻辑回归模型作为自身学习模型的神经元示例，在神经网络中，参数又可被成为权重（**weight**）。



Sigmoid (logistic) activation function.

我们设计出了类似于神经元的神经网络，效果如下：

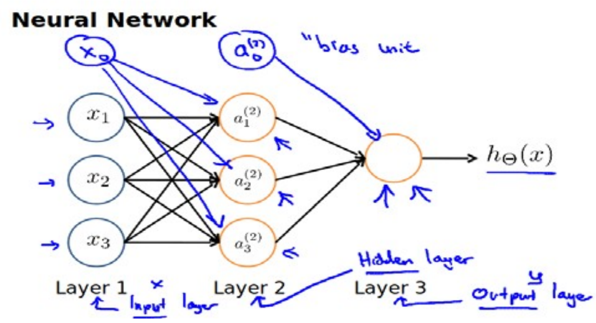


其中x_1，x_2，x_3是输入单元（**input units**），我们将原始数据输入给它们。

a_1，a_2，a_3是中间单元，它们负责将数据进行处理，然后呈递到下一层。

最后是输出单元，它负责计算$h_\theta(x)$。

神经网络模型是许多逻辑单元按照不同层级组织起来的网络，每一层的输出变量都是下一层的输入变量。下图为一个 3 层的神经网络，第一层成为输入层（**Input Layer**），最后一层称为输出层（**Output Layer**），中间一层成为隐藏层（**Hidden Layers**）。我们为每一层都增加一个偏差单位（**bias unit**）：



下面引入一些标记法来帮助描述模型：

$a_i^{(j)}$ 代表第j 层的第 i 个激活单元。$\theta^{(j)}$代表从第 j 层映射到第$j+1$ 层时的权重的矩阵，例如$\theta^{(1)}$代表从第一层映射到第二层的权重的矩阵。其尺寸为：以第 $j+1$层的激活单元数量为行数，以第 j 层的激活单元数加一为列数的矩阵。例如：上图所示的神经网络中$\theta^{(1)}$的尺寸为 3*4。

对于上图所示的模型，激活单元和输出分别表达为：

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

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

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

$$h_\Theta(x) = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

上面进行的讨论中只是将特征矩阵中的一行（一个训练实例）喂给了神经网络，我们需要将整个训练集都喂给我们的神经网络算法来学习模型。

我们可以知道：每一个a都是由上一层所有的x和每一个x所对应的决定的。

（我们把这样从左到右的算法称为前向传播算法(**FORWARD PROPAGATION**)）

把x，θ，a 分别用矩阵表示，我们可以得到$\theta \cdot X = a$ ：

$$X = \begin{matrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{matrix} \quad , \quad \theta = \begin{matrix} \theta_{10} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \theta_{33} \end{matrix} \quad , \quad a = \begin{matrix} a_1 \\ a_2 \\ a_3 \end{matrix}$$

8.4 模型表示 2

参考视频: 8 - 4 - Model Representation II (12 min).mkv

（**FORWARD PROPAGATION**）相对于使用循环来编码，利用向量化的方法会使得计算更为简便。以上面的神经网络为例，试着计算第二层的值：

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}$$

$$z^{(2)} = \Theta^{(1)} x$$
$$a^{(2)} = g(z^{(2)})$$

$$g\left(\begin{bmatrix} \theta_{10}^{(1)} & \theta_{11}^{(1)} & \theta_{12}^{(1)} & \theta_{13}^{(1)} \\ \theta_{20}^{(1)} & \theta_{21}^{(1)} & \theta_{22}^{(1)} & \theta_{23}^{(1)} \\ \theta_{30}^{(1)} & \theta_{31}^{(1)} & \theta_{32}^{(1)} & \theta_{33}^{(1)} \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) = g\left(\begin{bmatrix} \theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3 \\ \theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3 \\ \theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3 \end{bmatrix}\right) = \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix}$$

我们令 $z^{(2)} = \theta^{(1)} x$，则 $a^{(2)} = g(z^{(2)})$，计算后添加 $a_0^{(2)} = 1$。 计算输出的值为：

$$g\left(\begin{bmatrix} \theta_{10}^{(2)} & \theta_{11}^{(2)} & \theta_{12}^{(2)} & \theta_{13}^{(2)} \end{bmatrix} \times \begin{bmatrix} a_0^{(2)} \\ a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix}\right) = g\left(\theta_{10}^{(2)} a_0^{(2)} + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)}\right) = h_\theta(x)$$

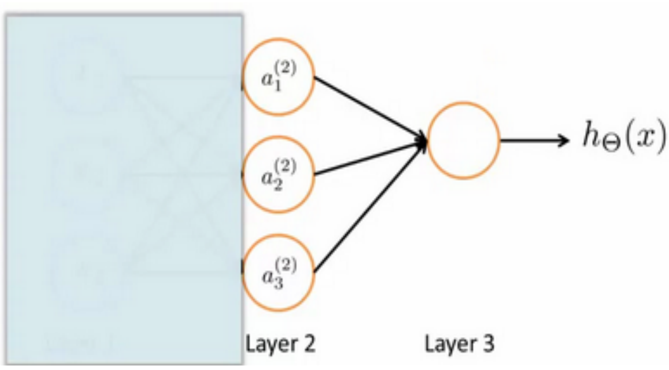
我们令 $z^{(3)} = \theta^{(2)} a^{(2)}$，则 $h_\theta(x) = a^{(3)} = g(z^{(3)})$。

这只是针对训练集中一个训练实例所进行的计算。如果我们要对整个训练集进行计算，我们需要将训练集特征矩阵进行转置，使得同一个实例的特征都在同一列里。即：

$z^{(2)} = \Theta^{(1)} \times X^T$

$a^{(2)} = g(z^{(2)})$

为了更好了了解 **Neuron Networks** 的工作原理，我们先把左半部分遮住：



右半部分其实就是以a_0, a_1, a_2, a_3，按照 **Logistic Regression** 的方式输出$h_\theta(x)$：

$$h_\Theta(x) = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

其实神经网络就像是 **logistic regression**，只不过我们把 **logistic regression** 中的输入向量$[x_1 \sim x_3]$ 变成了中间层的$\left[a_1^{(2)} \sim a_3^{(2)}\right]$，即:

$$h_\theta(x) = g\left(\theta_0^{(2)} a_0^{(2)} + \theta_1^{(2)} a_1^{(2)} + \theta_2^{(2)} a_2^{(2)} + \theta_3^{(2)} a_3^{(2)}\right)$$

我们可以把a_0, a_1, a_2, a_3看成更为高级的特征值，也就是x_0, x_1, x_2, x_3的进化体，并且它们是由 x与决定的，因为是梯度下降的，所以a是变化的，并且变得越来越厉害，所以这些更高级的特征值远比仅仅将 x次方厉害，也能更好的预测新数据。

这就是神经网络相比于逻辑回归和线性回归的优势。