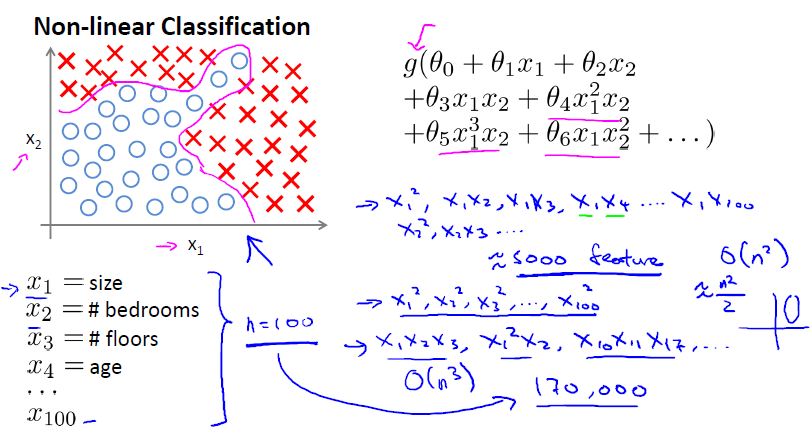
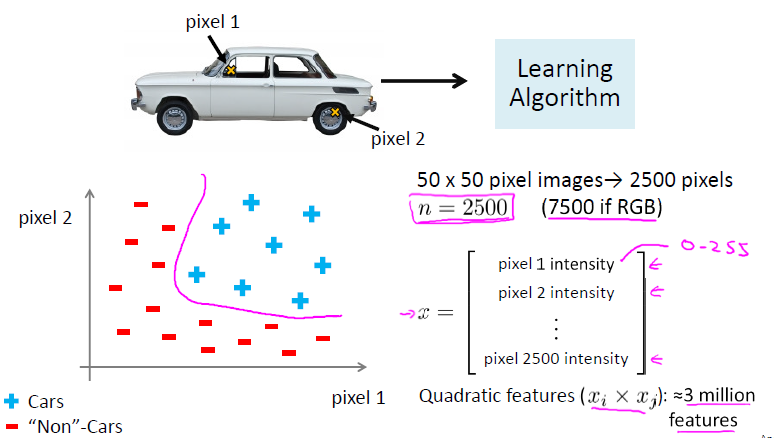
**视频一：**

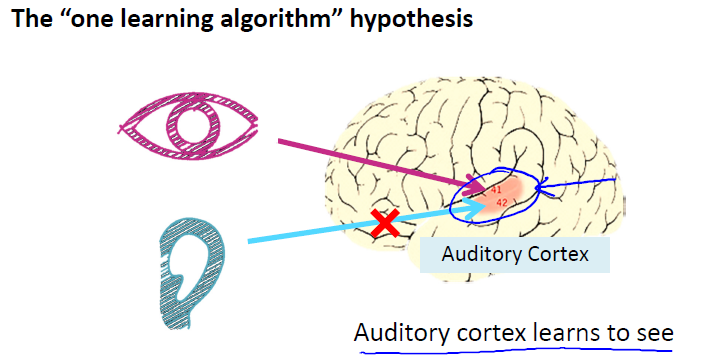
首先讲了在已经有了线性回归后为什么要引入神经网络：线性回归要自己配对变量，单单是计算二次项就已经很多了，更不用说更高；而如果只是单变量指数，又会损失很多特征的相关性，而且即使会出现椭圆什么的图形，也不会出现紫色线那么拟合的线；



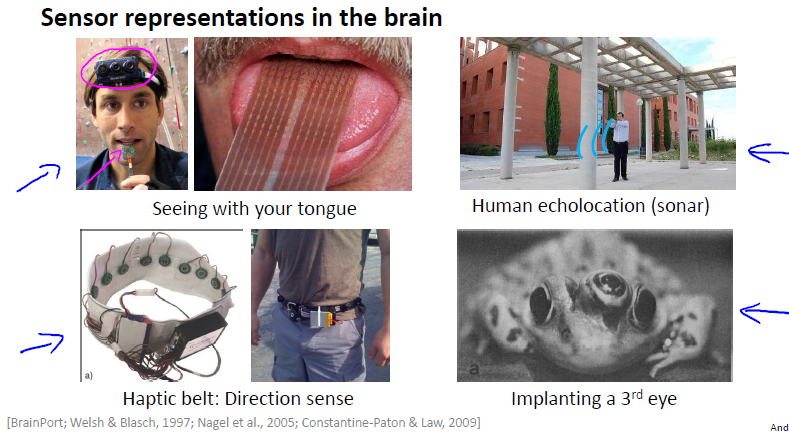
对于车辆识别，人眼看到的是图像，但是计算机看到的是一个矩阵，灰度图的话就是灰度级别，仅仅提取车身的两个像素进行分类问题的建模还可以接受，类似于回归分析里的x1和x2，但是对于任一幅图，非常小的50\*50的图就有2500个点，用这所有的点用逻辑回归就不现实。



在说明了之前方法的局限性后，就开始讲新方法的起由：本来大脑皮层中用于分辨听觉信息的区域，在切断了和耳朵的连接，将其连接到眼睛，发现，之后这个部位也可以利用眼睛的视觉来完成眼睛的功能，等等；

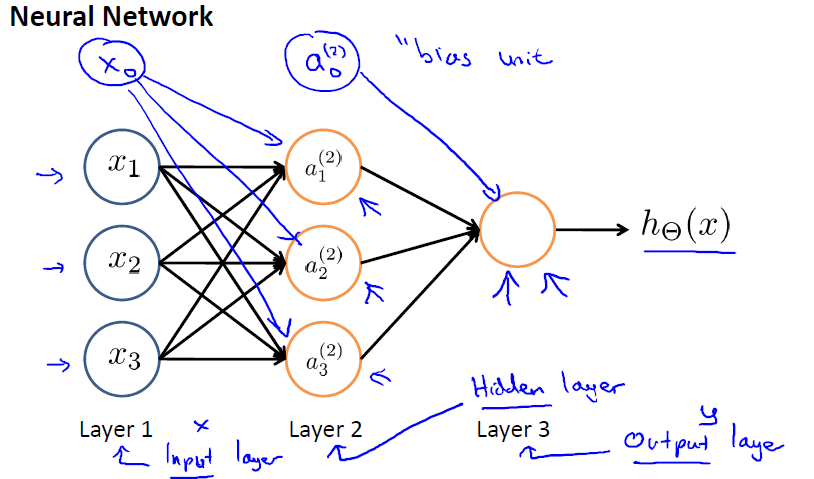


下面的例子也展现了大脑学习功能的奥秘：第一个是盲人，头戴图像拍摄传感器，将其连接到舌头组织上，利用原本处理舌头信息的大脑来处理图片信息，奏效，这都说明并不是大脑的某一部分只能做一件事，而是可以通过学习来重新训练获取新功能，这个是将坏的部分接到好的部分上；第二个是盲人，人类回声定位，通过咂舌的声音等，将声音传导接到了视觉皮层上，让人类也有了类似回声定位的功能，这里举例了一个美国盲人篮球女孩的故事，都是具有回声定位能力；第三个是定位皮带，某个位置必须要朝向北（举例来说的），当不朝向北时就发出警报，让人类也有鸟类一样的方向能力；最后是给青蛙装上了第3只眼，发现它也可以学会使用这只眼睛，大脑皮层的功能并不是固定的。

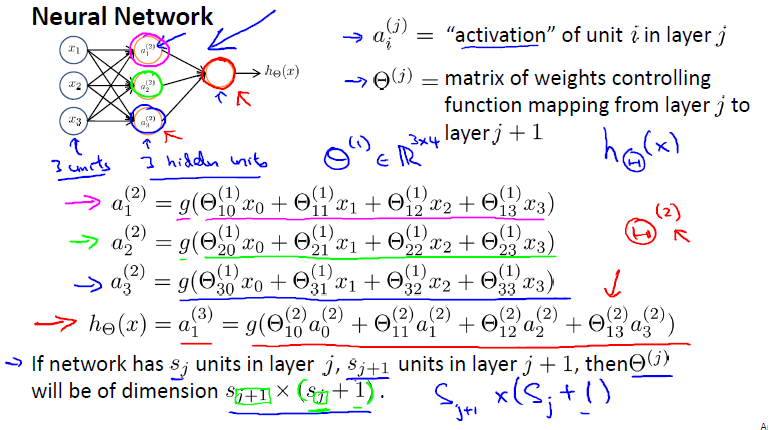


**视频二：**

由神经细胞进行引入，介绍神经网络单元的构成模仿的是神经细胞：每一层都有一个偏置常量设为1，因为要与theta0相乘，所以总是0，之后就不会画出来了；



然后就解释了每一层的参数，而且给出了theta维数的计算公式。

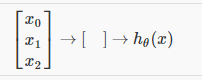


阅读材料给的总结相当的到位：

Model Representation I

Let's examine how we will represent a hypothesis function using neural networks. At a very simple level, neurons are basically computational units that take inputs (**dendrites**) as electrical inputs (called "spikes") that are channeled to outputs (**axons**). In our model, our dendrites are like the input features *x*1​⋯*xn*​, and the output is the result of our hypothesis function. In this model our *x*0​ input node is sometimes called the "bias unit." It is always equal to 1. In neural networks, we use the same logistic function as in classification,  ​, yet we sometimes call it a sigmoid (logistic) **activation** function. In this situation, our "theta" parameters are sometimes called "weights".

Visually, a simplistic representation looks like:



Our input nodes (layer 1), also known as the "input layer", go into another node (layer 2), which finally outputs the hypothesis function, known as the "output layer".

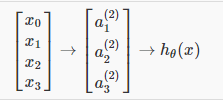
We can have intermediate layers of nodes between the input and output layers called the "hidden layers."

In this example, we label these intermediate or "hidden" layer nodes  *a*02​⋯*an*2​ and call them "activation units."

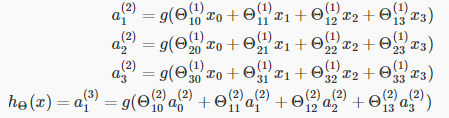
*a*(*j*)*i*="activation" of unit *i* in layer *j*

Θ(*j*)=matrix of weights controlling function mapping from layer *j* to layer *j*+1

If we had one hidden layer, it would look like:



The values for each of the "activation" nodes is obtained as follows:



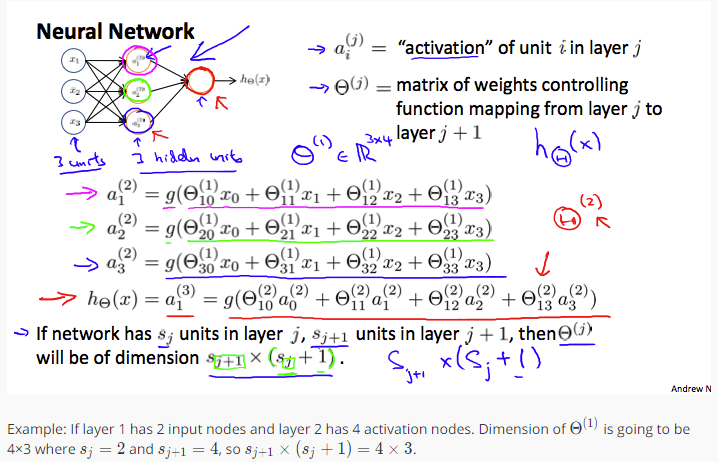
This is saying that we compute our activation nodes by using a 3×4 matrix of parameters. We apply each row of the parameters to our inputs to obtain the value for one activation node. Our hypothesis output is the logistic function applied to the sum of the values of our activation nodes, which have been multiplied by yet another parameter matrixΘ(2) containing the weights for our second layer of nodes.

Each layer gets its own matrix of weights, Θ(*j*).

The dimensions of these matrices of weights is determined as follows:

If network has s\_*j* units in layer j and s\_*j+1*units in layer j+1, then Θ(*j*) will be of dimension s\_*j+1* times s\_*j + 1*.If network has *sj*​ units in layer *j* and *s\_j*+1​ units in layer *j+1*, then Θ(*j*) will be of dimension *s\_j*+1​×(*s\_j​+1*).

The +1 comes from the addition in Θ(*j*) of the "bias nodes," *x*0​ and Θ0(*j*)​. In other words the output nodes will not include the bias nodes while the inputs will. The following image summarizes our model representation:



**视频三和四：**

讲了前向传播：theta和X的相乘，一步布向激活函数的思想靠近，原来逻辑回归的时候只是利用了输入变量X，相当于只有一个输入层和一个输出层，现在有了中间的隐藏层，就可以利用中间变量学习到更加灵活和复杂的特征，毕竟原来只有一个输入层特征可以利用，现在可以通过调节每一层的theta得到更灵活和复杂的特征，所以加了隐藏层就可以比单层学习更加灵活；

