输出层用 sigmoid，因为它的变化范围是（0,1），但是当 z 的绝对值较大时，导数趋于 0，会减慢神经网络的迭代。所以适合在输出层做二分类。
隐层用 tanh 或者 ReLU 或者 Leaky ReLU
tanh：a=（e^{z}-e^{-z}）/（e^{z}+e^{-z}）。它的变化范围是（-1,1），当在隐层中使用时，它的均值是 0，数据更有平均性。但是，和 sigmoid 一样，当 z 的绝对值较大时，导数趋于 0，会减慢神经网络的迭代。
ReLU：a=max(0,z) 当 z=0 时，a 对 z 的偏导直接取 0，虽然 z=0 时导数不连续。
Leaky ReLU：a=max(0.01z, z) 将在 z=0 处产生拐点

为什么要用非线性激活函数？
因为如果用线性激活函数，神经网络所做的工作只是在对输入进行线性组合。不论神经网络有多少层，都是在计算输入的线性组合，因为线性组合的组合还是线性组合，那还不如直接把所有隐层删掉，这样会更快完成线性组合。非线性激活函数的存在是增加神经网络的复杂度，增大可以由输入产生的空间，从而给机器在大空间中的参数寻找提供空间基础。

只有一个地方可以使用线性激活函数，就是在让机器去学习回归问题的时候。而且还只能在输出层用。不可以隐层中使用线性激活函数，否则又成线性组合了。比如预测房价，房价的变化区间是非负实数，可以在网络的输出层，使用线性激活函数得到从 0 到正无穷的实数。

随机初始化权重 w，是为了防止神经网络中出现破坏性的对称性分布，因为对称性会使若干条线路在计算重复的内容，而使用神经网络的目的恰恰是尽可能多地计算不同的内容。初始化的时候，w 一般会取很小的值。举例来说，对于 tanh 和 sigmoid 函数，当 z 越小时函数梯度越大，当在越大时函数梯度越小。如果一开始 w 就取太大的数值，很容易在函数中梯度较小的部分进行迭代，而因为这里梯度几乎趋近于 0，所以迭代的速度很慢。

超参数：神经网络中，影响着 w 和 b 数值的所有参数都是超参数。几乎除了 w 和 b 之外的所有参数都是超参数，比如学习率α（就是梯度下降的步长）、梯度下降的循环步数、隐层的层数、节点的总数、激活函数的形式，等等。
会有系统的调节超参数的方法。