1. Thinking
   1. **请列举几种常见的激活函数，激活函数有什么作用**
      1. 定义：

激活函数是在标准范围或对称范围之间规范化数据的方式。有不同函数。根据在训练算法中决定权重，需要不要激活函数。

* + 1. 最大优点：

作为缓存每层输入值的一种方式。神经网络可以借此寻找模式或忽略噪音。

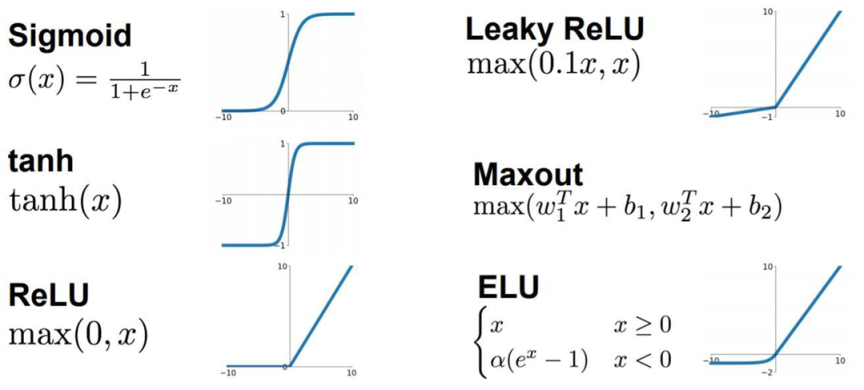
* + 1. 作用：

如果不用激活函数，就相当于激励函数f(x) =x， 此时每一层节点的输入都是上层输出的线性函数， 那么无论神经网络有多少层，输出都是输入的线性组合 => 与没有隐藏层效果相当

引入非线性函数作为激励函数，这样神经网络表达能力会更加强大 => 不再是输入的线性组合，而是几乎可以逼近任意函数

* + 1. 激活函数用来对数据作标准化，分为对称型或标准型

倾斜函数



处理概率值

Sigmoid和tanh函数输出值在(0,1)和(-1,1)之间

深层网络训练

relu的有效导数是常数1，解决了深层网络中出现的梯度消失问题

周期函数

用于给具有大量噪音的数据建模

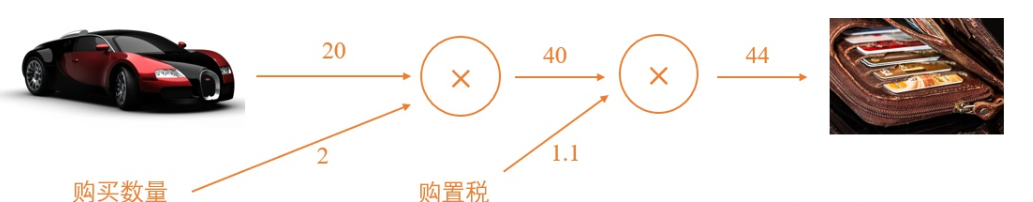
Sine

Cosine

* 1. **什么是反向传播中的链式法则**
     1. 链式法则

如果某个函数由复合函数表示，则该复合函数的导数可以用构成复合函数的各个函数的导数的乘积表示。我们只需要知道每个节点导数值，然后求乘积就可以了。

正向传播



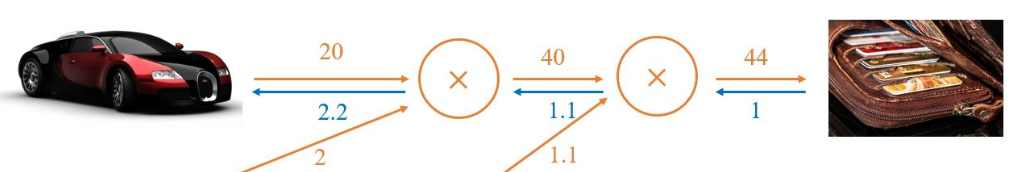
汽车单价20万，最终需要支付44万，我现在想知道汽车单价每波动1万，对最终支付价格的影响是多少。参看下图：我们从右向左依次求导，得到的值分别为

①44/44=1

②44/40=1.1

③40/20=2

那么最终价格相对于汽车单价的导数就是①×②×③=2.2



* 1. **利用梯度下降法训练神经网络，发现模型loss不变，可能有哪些问题？怎么解决？**
     1. 问题描述：

模型训练通常会遇到瓶颈， 一般包括：梯度消失、大量神经元失活、梯度爆炸和弥散、学习率过大或过小等。

* 梯度消失：
* 模型的loss难以下降，可以通过梯度的检验来验证模型当前所处的状态。有时梯度的更新和反向传播代码存在bug时，也会有这样的问题。
* 使用sigmoid作为激活函数，如果梯度小于1，会发生梯度消失。可以考虑使用Relu作为激活函数，以稳定模型的收纳速度。
* *大量神经元失活：在使用Relu激活函数的时候，当每一个神经元的输入X为负时，会使得该神经元输出恒为0，导致失活，由于此时梯度为0，无法恢复。有一种解决方案是使用LeakyRelu，这时，Y轴的左边图线会有一个很小的正梯度，使得神经网络在一定时间后可以得到恢复。不过LeakyRelu并不常用，因为部分神经元失活并不影响结果，相反，这种输出为0还有很多积极的作用。因为Relu方程输入为负时，输出值为0，利用此特性可以很好地忽略掉卷积核输出负相关信息，同时保留相关信息。*
* *梯度爆炸和梯度弥散: 问题产生的根本原因是，根据链式法则，深度学习中的梯度在逐层累积。如1.1的n次方无穷大，0.9的n次方无穷小。网络中某些层过大的输出也会造成梯度爆炸，此时应该为该输出取一个上界，可用最大范数约束。*

*loss不下降一般分为三种：*

*1. 训练集上loss不下降*

* *模型结构和特征工程存在问题*
* *权重初始化方案有问题*
* *正则化过度*
* *选择合适的激活函数、损失函数*
* *选择合适的优化器和学习速率*
* *训练时间不足*
* *模型训练遇到瓶颈*
* *batch size过大*
* *数据集未打乱*
* *数据集有问题*
* *未进行归一化*

*验证集上loss不下降*

一种是训练集上的loss也不下降，这时问题主要在训练集的loss上，应当先参考上述方法解决。

另一种是训练集上的loss可以下降，但验证集上的loss已经不降了

由于验证集是从同一批训练数据中划分出来的， 所以一般不存在数据集的问题，所以主要是过拟合

* 适当的正则化和降维
* 适当降低模型的规模
* 获取更多的数据集

1. 测试集上loss不下降

训练集和验证集的loss不下降时， 参见以上两种情况

* 默认训练集和验证集的loss情况是正常的。所以，如果测试集的loss很高，或者正确率很低. 训练数据的分布和场景与测试数据的分布和应用场景不一致
* 测试集一般为模型之前训练时从未见过的新数据，或者目标应用场景下的真实数据应用场景不一致

噪声问题

* + 1. 梯度下降需要每一层都有明确的误差才能更新参数，所以接下来的重点是如何将输出层的误差反向传播给隐藏层

梯度消失时，模型的loss难以下降.可以通过梯度的检验来验证模型当前所处的状态。有时梯度的更新和反向传播代码存在bug时，也会有这样的问题。