深度学习实用层面

http://blog.csdn.net/red\_stone1/article/details/78208851

训练集、交叉验证集和测试集的比例

什么是欠拟合与过拟合。欠拟合过拟合与偏差方差有什么关系

Regularization和dropout方法防止过拟合

Early stop方法防止过拟合

输入归一化及其好处

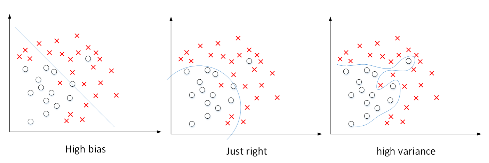
梯度消失与爆炸，以及如何减缓

1w以上样本，98%训练集，1%交叉验证集（dev），1%测试集。三者务必处于同一分布

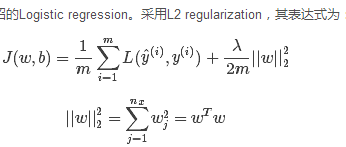
深度学习中的偏差（Bias）和方差（Variance）是抽象意义上的，并不实际指某一数据集分布状态。

偏差指训练集误差与贝叶斯误差（人类最好表现）的差值。偏差大表明欠拟合。解决方法是增大训练次数，增加神经网络隐藏层数、神经元数，或者选择其它更复杂的NN模型

方差指交叉验证集（或测试集）与偏差的差值。方差大表明过拟合。解决方法是增大样本量，使用正则化（Regularization）方法或者dropout方法

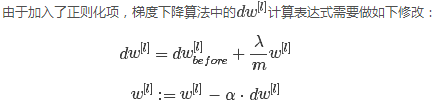


传统机器学习算法中，减小Bias会增加Variance，减小Variance会增加Bias。而现在的深度学习中，通过使用更复杂的神经网络和海量的训练样本，一般能够同时有效减小Bias和Variance。



b值对整体模型影响很小，所以忽略对其正则化

L2 regularization比L1 regularization更加常用，因为其得到的w更加稀疏（很多w为零值），可以节约存储空间；L2在微分求导方面更简单。而且二者解决过拟合问题能力接近。



L2 regularization也被称做weight decay。因为在更新w[l]的时候，会多减去增加的正则项，，使得w[l]比没有正则项的值要小一些



< keep\_prob使得产生的随机矩阵中，每个元素为1的概率是keep\_prob，为0的概率是1-keep\_prob。所以d1矩阵元素，不是1就是0。

Numpy中multiply是矩阵对应位置相乘，element-wise

对于m个样本，单次迭代训练时，随机删除掉隐藏层一定数量的神经元；然后，在删除后的剩下的神经元上正向和反向更新权重w和常数项b？？？怎么实现；接着，下一次迭代中，再恢复之前删除的神经元，重新随机删除一定数量的神经元，进行正向和反向更新w和b。不断重复上述过程，直至迭代训练完成。使用dropout训练结束后，在测试和实际应用时，不需要进行dropout和随机删减神经元，所有的神经元都在工作

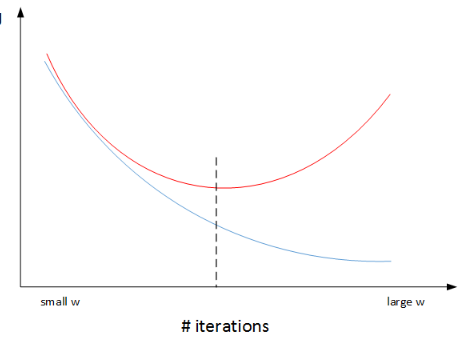
Dropout使得神经元不会受某个输入非常大的影响。因为输入是不确定的，如果在多次训练后能够达到稳定性能，那么神经元受各个输入的影响必然是均衡的，否则就会产生剧烈波动。输入影响均衡，表示为权重参数都不是很大。这从从效果上来说，dropout与L2 regularization是类似的

影响被均匀化了。也就是说，对应的权重w不会很大。这从从效果上来说，与L2 regularization是类似的

Regulation和dropout方法的本质，都是使训练出的权重参数更小。原因可以从极端情况来思考（这似乎也是神经网络问题常见的思考模式）。当某些权重参数很小，近似为0时，意味着该神经网络中某些神经元的实际作用很小，该神经元可以忽略。从而使得原本过于复杂的神经网络变得简单。

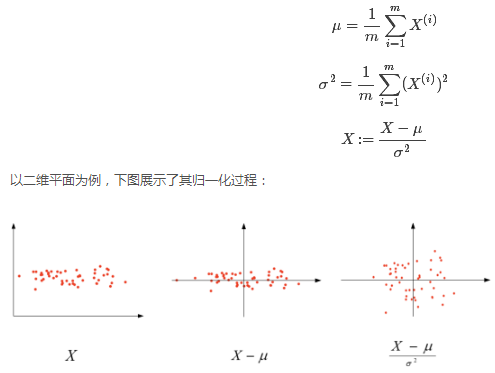
欠拟合与过拟合的本质。借助空间几何理解，欠拟合是采用了过于低维的曲面（比如平面，折面）来划分样本空间中复杂分布的情况。而过拟合则是采用了过于高维的曲面来划分样本空间中不那么复杂的分布情况。维度高低对应着神经网络的复杂程度，直观上就是神经元个数。而神经元个数又是通过权重指数表征的。因此解决过拟合问题，着眼于减小权重参数大小。

使用early stop防止过拟合。一个神经网络模型随着迭代训练次数增加，train set error一般是单调减小的，而dev set error 先减小，之后又增大。在dev seterror开始增大时停止训练，即为early stop

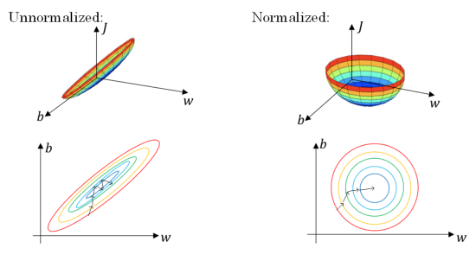


Early stop不符合正交化原则，其将减小代价函数和防止过拟合混合在一起。不像regularizatoin和dropout，通过足够的迭代（迭代多通常意味着代价函数减小）来防止过拟合。因此后二者更常用

机器学习训练模型有两个目标：一是优化cost function，尽量减小J；二是防止过拟合。这两个目标彼此对立的，即减小J的同时可能会造成过拟合，反之亦然



训练集进行了标准化处理，那么对于测试集或在实际应用时，应该使用同样的μ和σ对其进行标准化处理



输入归一化可以提高训练速度。输入归一化使得代价函数在不同参数上的梯度变得更均匀，从而可以使用更大的学习率而不至于引起震荡。

如何减缓梯度消失与梯度爆炸

