**过拟合**

# 定义

给定一个假设空间，一个假设属于，如果存在其他的假设属于,使得在训练样本集上的错误率比小，但在整个实例分布上比的错误率小，那么就说假设过度拟合训练数据。 ——《Machine Learning》Tom M.Mitchell

# 出现过拟合的原因

1. 训练集的数量级和模型的复杂度不匹配。训练集的数量级要小于模型的复杂度；
2. 训练集和测试集的特征分布不一致；
3. 样本里的噪音数据干扰过大，大到模型过分记住了噪音特征，反而忽略了真实的输入输出间的关系；
4. 权值学习迭代次数足够多(Overtraining)，拟合了训练数据中的噪声和训练样例中没有代表性的特征。

# 解决方案

1. Simpler Model Structure

调小模型复杂度，使其适合自己训练集的数量级（缩小宽度和减小深度）

1. Data Augmentation

训练集越多，过拟合的概率越小。在计算机视觉领域中，增广的方式是对图像旋转，缩放，剪切，添加噪声等。

1. Regularization

参数太多，会导致我们的模型复杂度上升，容易过拟合，也就是我们的训练误差会很小。 正则化是指通过引入额外新信息来解决机器学习中过拟合问题的一种方法。这种额外信息通常的形式是模型复杂性带来的惩罚度。 正则化可以保持模型简单，另外，规则项的使用还可以约束我们的模型的特性。

A）L0范数与L1范数

L0范数是指向量中非0的元素的个数。如果我们用L0范数来规则化一个参数矩阵W的话，就是希望W的大部分元素都是0即让参数W是稀疏的。

L1范数是指向量中各个元素绝对值之和，也叫“稀疏规则算子”（Lasso Regularization）。为什么L1范数会使权值稀疏？有人可能会这样给你回答“它是L0范数的最优凸近似”。实际上，还存在一个更美的回答：任何的规则化算子，如果他在Wi=0的地方不可微，并且可以分解为一个“求和”的形式，那么这个规则化算子就可以实现稀疏。W的L1范数是绝对值，|W|在W=0处是不可微，

为什么L0和L1都可以实现稀疏，但常用的为L1？因为L0范数很难优化求解（NP难问题），二是L1范数是L0范数的最优凸近似，而且它比L0范数要容易优化求解。所以大家才把目光和万千宠爱转于L1范数。

综上，L1范数和L0范数可以实现稀疏，L1因具有比L0更好的优化求解特性而被广泛应用。

那么参数稀疏有什么好处呢？这里扯两点：

（1）特征选择(Feature Selection)：

大家对稀疏规则化趋之若鹜的一个关键原因在于它能实现特征的自动选择。一般来说，Xi的大部分元素（也就是特征）都是和最终的输出Yi没有关系或者不提供任何信息的，在最小化目标函数的时候考虑Xi这些额外的特征，虽然可以获得更小的训练误差，但在预测新的样本时，这些没用的信息反而会被考虑，从而干扰了对正确Yi的预测。稀疏规则化算子的引入就是为了完成特征自动选择的光荣使命，它会学习地去掉这些没有信息的特征，也就是把这些特征对应的权重置为0。

（2）可解释性(Interpretability)：

另一个青睐于稀疏的理由是，模型更容易解释。例如患某种病的概率是Y，然后我们收集到的数据X是1000维的，也就是我们需要寻找这1000种因素到底是怎么影响患上这种病的概率的。假设我们这个是个回归模型：Y=W1\*X1+W2\*X2+…+W1000\*X1000+B（当然了，为了让Y限定在[0,1]的范围，一般还得加个Logistic函数）。通过学习，如果最后学习到的W\*就只有很少的非零元素，例如只有5个非零的Wi，那么我们就有理由相信，这些对应的特征在患病分析上面提供的信息是巨大的，决策性的。也就是说，患不患这种病只和这5个因素有关，那医生就好分析多了。但如果1000个Wi都非0，医生面对这1000种因素，累觉不爱。

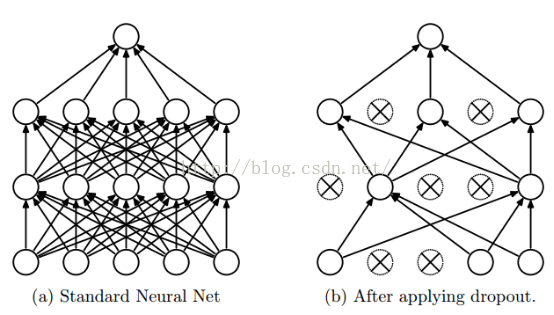
B）L2范数

除了L1范数，还有一种更受宠幸的规则化范数是L2范数: ||W||2。它也不逊于L1范数，它有两个美称，在回归里面，有人把有它的回归叫“岭回归”（Ridge Regression），有人也叫它“权值衰减”(Weight Decay)。 Weight Decay还有一个好处，它使得目标函数变为凸函数，梯度下降法和L-BFGS都能收敛到全局最优解。

L2范数是指向量各元素的平方和然后求平方根。我们让L2范数的规则项||W||2最小，可以使得W的每个元素都很小，都接近于0，但与L1范数不同，它不会让它等于0，而是接近于0。因为一般认为参数值小的模型比较简单，能适应不同的数据集，也在一定程度上避免了过拟合现象。可以设想一下对于一个线性回归方程，若参数很大，那么只要数据偏移一点点，就会对结果造成很大的影响；但如果参数足够小，数据偏移得多一点也不会对结果造成什么影响，专业一点的说法是『抗扰动能力强』。

1. Dropout

这个方法在神经网络里面很常用。Dropout方法是Imagenet中提出的一种方法，通俗一点讲就是Dropout方法在训练的时候让神经元以一定的概率不工作。具体看下图：



1. Early Stopping

     对模型进行训练的过程即是对模型的参数进行学习更新的过程，这个参数学习的过程往往会用到一些迭代方法，如梯度下降（Gradient Descent）学习算法。Early Stopping便是一种迭代次数截断的方法来防止过拟合的方法，即在模型对训练数据集迭代收敛之前停止迭代来防止过拟合。

  Early Stopping方法的具体做法是，在每一个Epoch结束时（一个Epoch集为对所有的训练数据的一轮遍历）计算Validation Data的Accuracy，当Accuracy不再提高时，就停止训练。这种做法很符合直观感受，因为Accurary都不再提高了，在继续训练也是无益的，只会提高训练的时间。那么该做法的一个重点便是怎样才认为Validation Accurary不再提高了呢？并不是说Validation Accuracy一降下来便认为不再提高了，因为可能经过这个Epoch后，Accuracy降低了，但是随后的Epoch又让Accuracy又上去了，所以不能根据一两次的连续降低就判断不再提高。一般的做法是，在训练的过程中，记录到目前为止最好的Validation Accuracy，当连续10次Epoch（或者更多次）没达到最佳Accuracy时，则可以认为Accuracy不再提高了。此时便可以停止迭代了（Early Stopping）。这种策略也称为“No-Improvement-In-N”，N即Epoch的次数，可以根据实际情况取，如10、20、30……

6、 Ensemble

集成学习算法也可以有效的减轻过拟合。Bagging通过平均多个模型的结果，来降低模型的方差。Boosting不仅能够减小偏差，还能减小方差。

7、重新清洗数据

数据清洗从名字上也看的出就是把“脏”的“洗掉”，指发现并纠正数据文件中可识别的错误的最后一道程序，包括检查数据一致性，处理无效值和缺失值等。导致过拟合的一个原因也有可能是数据不纯导致的，如果出现了过拟合就需要我们重新清洗数据。

SupportVectorMachine <IKernel> svm;

IKernel kernel = new Accord.Statistics.Kernels.Gaussian(Convert.ToDouble(numSigma.Text.Trim()));

var teacher = new SequentialMinimalOptimizationRegression()

{

// Set learning parameters

Complexity = 1,

Tolerance = 0.001,

Epsilon =0.001,

Kernel = kernel

};

try

{

// Use the teacher to create a machine

svm = teacher.Learn(TrainInput, TrainOutput);

}