正则化防止过拟合

正则化可以防止模型过拟合，提高模型的泛化能力，如下图是模型欠拟合、正确的模型和过拟合的例子。

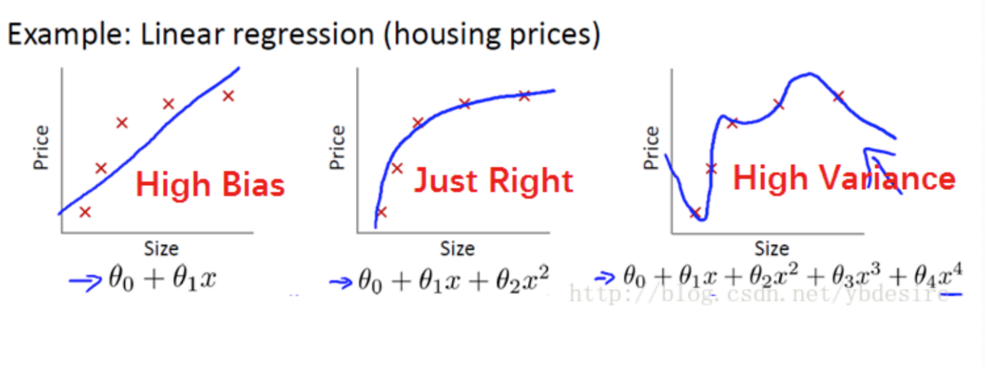

图 1 欠拟合、正确的、过拟合模型

如图 1 中所示，一般欠拟合的模型具有较大的偏差[正确值与预测值之间的差值]，过拟合的模型具有较大的方差[数据分布太散]。

造成过拟合的本质原因是模型学习的太过精密，导致连训练集中的样本噪声也一丝不差的训练进入了模型。

造成欠拟合的原因则是与过拟合相反，模型学习的太过粗糙，连训练集中的样本数据的特征关系都没能学出来。

解决过拟合的方法主要有以下几种：

1、 数据层面：
- 数据集扩增，获取更多的数据
- 特征工程，筛选组合得到更高质量的特征。

2、 模型层面：
- 选择较为简单的模型
- 集成学习，Bagging 策略组合模型，降低模型的方差
- 加入正则项，如 L1、L2 正则项，以及树模型的剪枝策略，XGBoost 中的正则惩罚项（叶子节点值+叶子节点的个数）。

3、 更多方法
- 早停（Early stopping），在模型的训练精度已经达到一定的需求时停止训练，以防止模型学习过多的样本噪声。
- 加入噪声，给定训练样本集更多的样本噪声，使得模型不易完全拟合这些噪声，从而只训练学习我们

想要的数据特征关系（这个的方法现实使用时效果并不好）。

- dropout，在深度学习中，我们经常使用 dropout 方法来防止过拟合，dropout 实际借鉴来自 bagging 的思想。
- 正则化，常用的正则化方法就是加入 L1、L2 正则项。
- BN(Batch Normalization)，BN 每一次训练中所组成的 Mini-Batch 类似于 Bagging 策略，不同的 Mini-Batch 训练出来的 BN 参数也不同。
- 权重衰减，有时我们也会称 L2 正则为权重衰减，因为 L2 正则化会使得权重偏向于 0. 权重衰减实际上是使得模型在训练后期，权重的变化边的很慢很慢，从而使得模型不至于在迭代后期转而去学习更多的样本噪声

范数

在讲 L1 正则和 L2 正则之前，我们先来了解下什么是范数(norm)，有时为了便于理解，我们把范数当做距离来理解，比如 L-P 范数，它不单指一个范数，而是指一群范数，其定义如下：

$$\mathrm{LP} = \sqrt[p]{\sum_{1}^{n} x_i^p}, \quad \mathrm{x} = (x_1, \ldots x_n)$$

- L0 范数

L0 范数就是 p 取 0 时，显然只要x_i ! = 0，那么它的 0 次方就是 1，所以 L0 范数指的是向量中非 0 的元素的个数，注意这里指的是元素的个数。所以如果我们用 L0 范数取规则化一个参数矩阵的话，就是希望 W 的大部分元素都是 0，话句话说，让参数 W 是稀疏的。

- L1 范数

L1 范数就是 p 取值为 1 时的表达式，它指的是向量中各个元素的绝对值之和，L1 范数也会使权值稀疏，有个规则是这样说的：任何的规则化算子，如果在 W=0 出不可微，且可以分解为求和的形式，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，在 W=0 处不可微（具体规则的来源与证明暂找到。）

既然 L0 范数能实现稀疏，为什么还要用 L1 范数呢，原因有两个：

1. L0 范数很难优化求解，它是一个 NP 难问题，L0 是一个 0-1 跃阶函数，低于 1 范数的都不是凸的。
2. L1 范数是 L0 翻书的最优凸近似，而且它比 L0 范数更容易优化求解。

总而言之就是，L1 范数和 L0 范数都可以实现稀疏，但是 L1 因具有比 L0 更好的优化求解特性而被广泛应用。

这里在介绍下让参数稀疏的好处：

- 特征选择：稀疏参数它能实现对特征的自动选择。一般来说，x_i的大部分元素（也就是特征）都是和最终的输出没有关系或者不提供任何的有用信息的，在最小化目标函数时考虑到了x_i的这些额外的特征，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，从而干扰了对正确的y_i的预测。而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. 总结来说就是，越好的特征包含的数据分布信息越多，差的特征也包含一定的数据分布信息，但同时还会包含大量的噪声，特征选择的宗旨在于选择出好的特征去学习，而不是为了一点点的模型训练提升取引入学习更多的噪声。
- 可解释性：我们最后输出的模型是关于一堆特征的加权组合，如果特征有几千个，解释起来就会很困难。但如果通过特征选择过滤出来 5 个特征，然后经过训练发现效果也不错，那这样的模型解释起来也会容易很多。

- L2 范数

L2 范数指的是 p 取 2，它指的是向量各元素的平方和然后求平方根，L2 范数可以使得 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，这里有很大的区别。同时，更小的权值 W，表示网络的模型复杂度更低，对数据的拟合刚刚好（这个法则也叫做奥卡姆剃刀）。

奥卡姆剃刀规则：当你有两个处于竞争地位的理论能得到相同的结论，那么简单的那个更好。一般的解释就是：如果你有两个原理，它们都能解释观测到的事实，那么你应该选择使用简单的那个；需要假设最少的解释是正确的。奥卡姆剃刀规则从来没有说简单的理论就是正确的理论，它的表述为"当两个假说具有相同的解释力和预测力时，我们以简单的那个假说作为讨论依据"。剃刀规则不是一个理论，而是一个原理，它的目的是为了精简抽象实体，它不能被证明也不能被证伪，它是一个规范性的思考原则。

同时，过拟合的时候，拟合函数的系数往往会很大，为什么呢，是因为过拟合，就是拟合函数往往需要顾及到每一个点，最终形成的拟合函数波动会很大。那么在某些很小的区间里面，函数值变化很剧烈，这就意味着函数在某些小区间里的导数值（绝对值）很大，由于自变量的值可大可小，所以只有系数足够大，才能保证导数值很大。而正则化项是通过约束参数的范数使其不要太大，这适用于 L1 正则和 L2 正则，所以在一定程度上能够减少过拟合情况。

- L1 正则和 L2 正则的直观解释

这部分将解释为什么 L1 正则化可以产生稀疏模型（L1 是怎么让参数为 0 的），以及 L2 正则化是怎么防止过拟合的（L2 正则是怎么让参数趋近于 0 的）。

L1 正则化和 L2 正则化的代价函数，可以写成如下的形式：

$$\text{L1 正则} - \text{Lasso: } \min \frac{1}{n} ||y - Xw||^2, s.t. ||w|| \leq C$$

$$\text{L2 正则} - \text{Ridge：} \min \frac{1}{n} ||y - Xw||^2, \quad s.t. ||w||^2 \leq C$$

也可以把上式写成如下格式：

$$L1: J = J_0 + \alpha \sum_w |w|$$

$$L2: J = J_0 + \alpha \sum_w w^2$$

我们令$L = \alpha \sum_w |w|$ 和$L = \alpha \sum_w w^2$，这样的话，损失函数就分为了两部分，J_0是原始的损失函数，L 是正则化项，那么我们的任务就变成了在 L 约束下求出J_0取最小值的解。同时也就是说，我们把模型空间限制在 w 的一个 L1-ball 中（这句话我没理解，我的理解是在限制条件 L 下，求J_0的最小值）。

考虑二维的情况，即只有两个权值w_1和w_2，求解J_0的过程可以画出等值线，同时 L1 正则化的函数 L 也可以在$w_1 w_2$平面画出来，如下图：

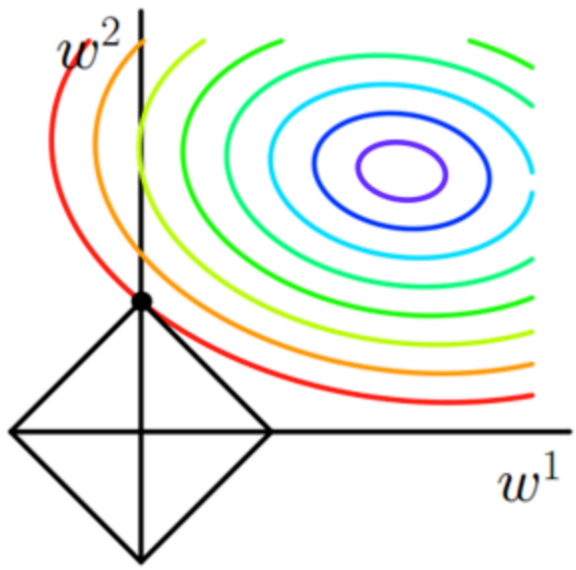

图 2：L1 正则化

图 2 中有颜色的圆是J_0的等值线，黑色方形是 L 函数的图形，在图中，当J_0等值线和 L 图形首次相交的地方才是最优解。可以直观想象，因为 L 函数有很多“突出的角”【二维情况下四个，多位情况下更多】，J_0与这些角接触的几率要远大于与其他 L 部位接触的几率，而在这些角上，会有很多权值为 0，这就是为什么 L1 正则化可以产生稀疏模型，进而可以用于特征选择。同时，正则化前面的系数α，可以控制 L 图形的大小，α越小，L 的图形越大（上图中的黑色方框），α越大，L 的图形就越小。

同样的对于 L2 正则化，也可以画出它们的图像：

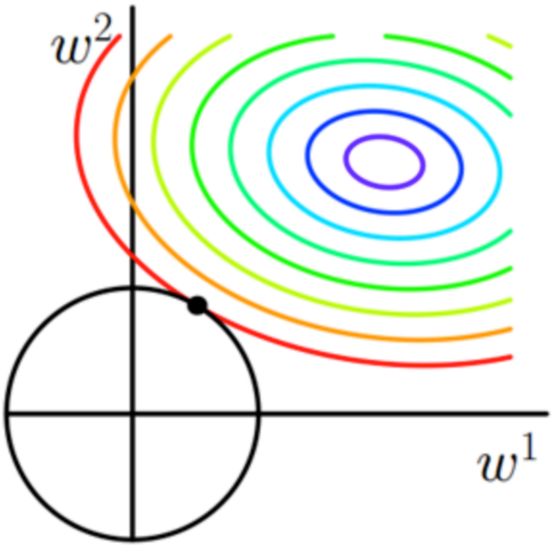

图 3： L2 正则化

二维平面下 L2 正则化的函数图形是个圆，与方形相比，被抹去了棱角。因此J_0与 L 相交时使得w_1和w_2等于零的几率小了很多，这也就是为什么 L2 正则化不具备稀疏性的原因。

■ L2 正则叫做权重衰减的原因

这个我们可以从添加了 L2 正则后的损失函数谈起，正则化项前的损失函数,其中$h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$：

$$\mathrm{J}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^i\right) - y^i\right)^2$$

梯度下降求解后：

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$$

其中

$$\frac{\partial}{\partial\theta_j}J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^i\right) - y^i\right) * x_j^i$$

则带入梯度下降公式得：

$$\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^i\right) - y^i\right) * x_j^i$$

而对损失函数加上 L2 正则后，损失函数为：

$$\mathrm{J}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}[\left(h_\theta\left(x^i\right) - y^i\right)^2 + \lambda\sum_{j=1}^{n}\theta_j]$$

相应的最后的梯度下降更新参数θ_j为：

$$\theta_j := \theta_j(1 - \alpha\frac{\lambda}{m}) - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^i\right) - y^i\right) * x_j^i$$

其中$0 < 1 - \alpha\frac{\lambda}{m} < 1$,所以$\theta_j$在每次更新之前都需要乘以一个小于 1 的因子，这也使得θ_j不断减小，这也是 L2 正则也叫权重衰减的源来。