深入理解GBDT、XGBoost、LightGBM系列(一)

导语：从模型+策略+算法的视角出发理解GBDT中的梯度提升(GB)，解决两个问题：1. 梯度提升(gradient boosting)到底指什么？2. 常说的残差应该如何理解？

在面试中经常会遇到介绍一下GBDT 这个问题，如何回答呢。比较常见说法是：“由多棵CART构成，每棵树学习的是前一棵的残差，在预测时会把每一棵树的结果加在一起作为最终结果...”。好像也没错，但总感觉缺点什么，也没有重点，也就不让人满意。

希望在读完这篇文章之后，可以给出更加系统，条理清晰，重点突出的答案。

怎么让语言组织的更加系统呢，肯定要借助一定的思维模型，回答机器学习和深度学习类问题，最通用的思维模型就是李航博士《统计学习方法》中的统计学习三要素：模型、策略、算法。这里的策略指的是损失函数+正则项最小化，在书中称为经验风险和结构风险最小化。算法也就是常说的最优化方法。

下面把这个思维模型应用到GBDT的理解中。

对于GBDT提到模型首先想到是什么，我首先想到的是树模型，随之而来的就是ID3，C4.5，CART等树的节点分裂策略。而事实上首先应该理解的是boosting，因为在GBDT是Boosting框架的一种，所以首先介绍一下boosting（提升）的概念。

引用参考资料[4]的内容介绍提升方法的概念，下面的内容主旨摘自原书，按我喜欢的方式表达出来。

提升方法的基本思想是：对于一个复杂任务来说，将多个专家的判断进行适当的综合所得出的判断，要比其中任何一个专家单独的判断好。也就是“三个臭皮匠定个诸葛亮”的道理。

在理解提升方法之前，首先要了解两个概念，强可学习(strongly learnable)和弱可学习(weakly learnable)。在概率近似正确(probably approximately correct, PAC) 学习的框架中，一个概念（分类），如果存在一个多项式的学习算法能够学习它，并且正确率很高，那么就称这个概念是强可学习的；一个概念，如果存在一个多项式的学习算法能够学习它，学习的正确率仅比随机猜测略好，那么就称这个概念是弱可学习的。后来Schapire后来证明强可学习与弱可学习是等价的，也就是说，在PAC的学习框架下，一个概念是强可学习的充分必要条件是这个概念是弱可学习的。

那么问题可以转换为，在学习中，如果已经发现了弱学习算法，能否将它提升(boost)为强学习算法。发现弱学习算法要比发现强学习算法容易得多，这里的弱学习算法不用考虑太复杂，最简单的就是卡阈值，比如小于阈值T就分类为0，大于等于T就分类1。那么如何具体实施提升，便成为开发提升方法时所要解决的问题。关于提升的方法的研究很多，可以查阅参考资料[3]，其中最具代表性的是AdaBoost算法。

对于分类问题而言，给定一个训练样本集，求比较粗糙的分类规则（弱分类器）要比求精确的分类规则（强分类器）容易得多。提升方法就是从弱学习算法出发，反复学习，得到一系列弱分类器（又称为基本分类器），然后组合这些弱分类器，构成一个强分类器。大多数的提升方法都是改变训练数据的概率分布（训练数据的权值分布），针对不同的训练数据分布调用弱学习算法学习一系列弱分类器。

这样，对提升方法来说，有两个问题需要回答：一是在每一轮如何改变训练数据的权值或概率分布；二是如何将弱分类器组合成一个强分类器。关于第一个问题，AdBoost的做法是，提高那些被前一轮弱分类器错误分类样本的权值，而降低那些被正确分类样本的权值。这样一来，那些没有得到正确分类的数据，由于其权值的加大而受到后一轮的弱分类器的更大关注。于是，分类问题被一系列的弱分类器“分而治之”。至于第二个问题，即弱分类器的组合，AdaBoost采取加权多数表决的方法。具体地，加大分类误差率小的弱分类器的权值，使其在表决中起较大的作用，减小分类误差率大的弱分类器的权值，使其在表决中起较小的作用。

AdaBoost的具体算法不再这里展开，介绍很多，比如下面这篇。

至此对提升算法有了一个定性的认识，但还是没有看到GBDT和XGBoost中的用到梯度提升（gradient boosting)。在引入梯度提升之前，还需要从把提升(boosting)问题换到另外一个视角看一下，也就是统计学视角，来自参考资料[3]。

在统计学视角下，将提升问题视为加性模型(additive model)，并且用前向分步法优化加性模型，加性模型在统计学中已经有非常悠久的历史。用论文中的原话就是：

We believe that viewing current boosting procedures as stagewise algorithms for fitting additive models goes a long way toward understanding their performance.

加性模型的通用形式如下：

$$F_M(x) = \sum_{m=1}^{M} \beta_m b(x; \gamma_m)$$

其中，γ 为弱分类器的参数，β_m 为弱分类器的权值，即弱分类器为 $f_m(x) = \beta_m b(x; \gamma_m)$

对于回归问题，均方误差作为损失函数，应用前向分步法优化时，迭代过程如下：

$$\{\beta_m, \gamma_m\} \leftarrow \arg\min_{\beta, \gamma} E\left[y - F_{m-1}(x) - \beta b(x; \gamma)\right]^2$$

对于 $m = 1, 2, \ldots, M,$ 参数 $\{\beta_k, \gamma_k\}_1^{m-1}$ 在其对应步骤的迭代中已经确定。

在优化每一个弱分类器时，需要不断修改最初的数据，如下：

$$y_m \leftarrow y - \sum_{k \neq m} f_k(x).$$

对于前向分步法，第m次的迭代的y值 y_m 由m-1步的y值 y_{m-1} 和弱分类器 $f_{m-1}(x)$ 决定，

$$y_m = y_{m-1} - f_{m-1}(x).$$

注意这里经典问题来了，此时当前弱分类器拟合的是数据的残差。

此时刚好和GBDT的原理介绍相合，我在这里犯的错误是把结果当成原因，理解成它就应该拟合残差。而事实上是，为了完成boost问题整体的优化，在任务是拟合，损失函数是均方误差条件下的特定状态。

那么扩展开来，如果不是拟合任务，损失函数也不是均方误差时，怎么办，如何得到每一步的数据变化？

由此，在参考资料[5]中引入了，类似梯度下降法的解决方案，也就是梯度提升(gradient boosting)的方案。其关键是利用损失函数的负梯度在当前模型的值，作为回归问题提升树算法中的残差近似值，拟合一个回归树。

For steepest–descent

$$f_m(\mathbf{x}) = -\rho_m g_m(\mathbf{x}) \tag{6}$$

with

$$g_m(\mathbf{x}) = \left[\frac{\partial \phi(F(\mathbf{x}))}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})} = \left[\frac{\partial E_y[L(y, F(\mathbf{x}))\,|\,\mathbf{x}]}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}$$

and

$$F_{m-1}(\mathbf{x}) = \sum_{i=0}^{m-1} f_i(\mathbf{x}).$$

Assuming sufficient regularity that one can interchange differentiation and integration, this becomes

$$g_m(\mathbf{x}) = E_y\left[\frac{\partial L(y, F(\mathbf{x}))}{\partial F(\mathbf{x})}\,|\,\mathbf{x}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}. \tag{7}$$

The multiplier ρ_m in (6) is given by the line search

$$\rho_m = \arg\min_{\rho} E_{y,\mathbf{x}} L\left(y, F_{m-1}(\mathbf{x}) - \rho g_m(\mathbf{x})\right).$$

至此，也就解决了梯度的问题。

总结一下，结合模型、策略、算法三要素去分析就能更加清晰的认识提升(Boosting)算法。从模型的角度看就是把多个弱分类器加在一起，这里的弱分类器不一定是决策树，也可以是其它模型，比如XGBoost就是支持线性模型。从策略+算法来看，优化方法采用的是前向分步法，这是一种贪婪算法，从前向后的优化各个弱分类器，所以优化每一个弱分类时，需要求当前Boost分类器和标签之间的变化。在解决回归问题时，采用均方误差作为损失时，此时恰好是当前Boost分类器和标签之间的差值，也就是残差这种说法的来源。而当不是回归问题时，损失函数不采用平方或指数形式，则不能用残差来表示这个变化。为了应对这个问题，就仿照梯度下降法，采用负梯度来表示这个变化，这也就是为什么叫做梯度提升(Gradient Boosting)。至此文章开头的两个就得到了比较清晰的解答。

回过头来看，为什么会有文章开头的那两个问题呢，主要是因为看问题的视角有问题，把整体的提升问题割裂开了，目光都集中到了弱分类器树的优化上，有这两个问题也就在所难免了。

既然已经写到这里了，当然不打算放过XGBoost和LightGBM这两个开源神器，这个系列后续会继续深入探索它们，看看大佬们怎么解释这些神器的实现，欢迎关注，共同成长。

参考资料

- [1] 《XGBoost: A Scalable Tree Boosting System》https://arxiv.org/pdf/1603.02754.pdf
- [2] 《对xgboost的理解》 https://zhuanlan.zhihu.com/p/75217528
- [3] 《Additive logistic Regression: a statistical view of boosting》 https://web.stanford.edu/~hastie/Papers/AdditiveLogisticRegression/alr.pdf
- [4] 《统计学习方法》 李航
- [5] 《greedy function approximation: a gradient boosting machine》https://jerryfriedman.su.domains/ftp/trebst.pdf
- [6] 《机器学习算法中 GBDT 和 XGBOOST 的区别有哪些？》 https://www.zhihu.com/question/41354392?sort=created