集成学习

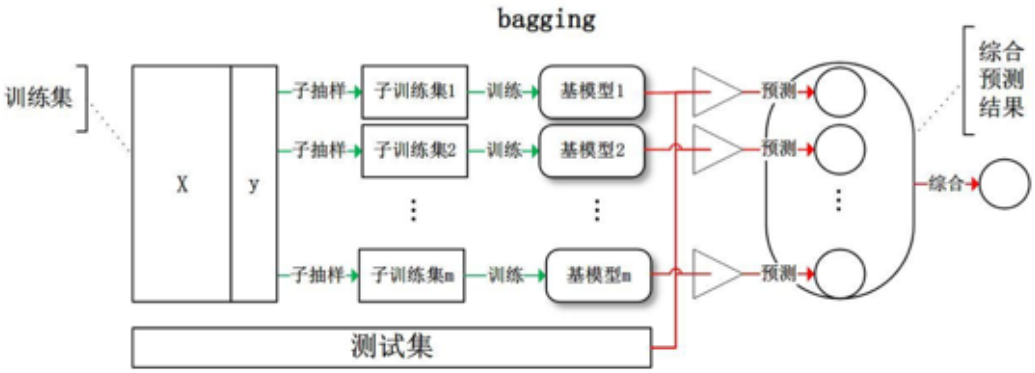
集成学习通过训练多个分类器，然后将其组合起来，从而达到更好的预测性能，提高分类器的泛化能力。

目前集成学习有 3 个主要框架：bagging、boosting、stacking。

bagging套袋法

bagging是并行集成学习方法的最著名代表，其算法过程如下：

1. 从原始样本集中抽取训练集。每轮从原始样本集中使用Bootstraping的方法抽取n个训练样本（在训练集中，有些样本可能被多次抽取到，而有些样本可能一次都没有被抽中）。共进行k轮抽取，得到k个训练集。（k个训练集之间是相互独立的）
2. 每次使用一个训练集得到一个模型，k个训练集共得到k个模型。（注：这里并没有具体的分类算法或回归方法，我们可以根据具体问题采用不同的分类或回归方法，如决策树、感知器等）
3. 对分类问题：将上步得到的k个模型采用投票的方式得到分类结果；对回归问题，计算上述模型的均值作为最后的结果。（所有模型的重要性相同）



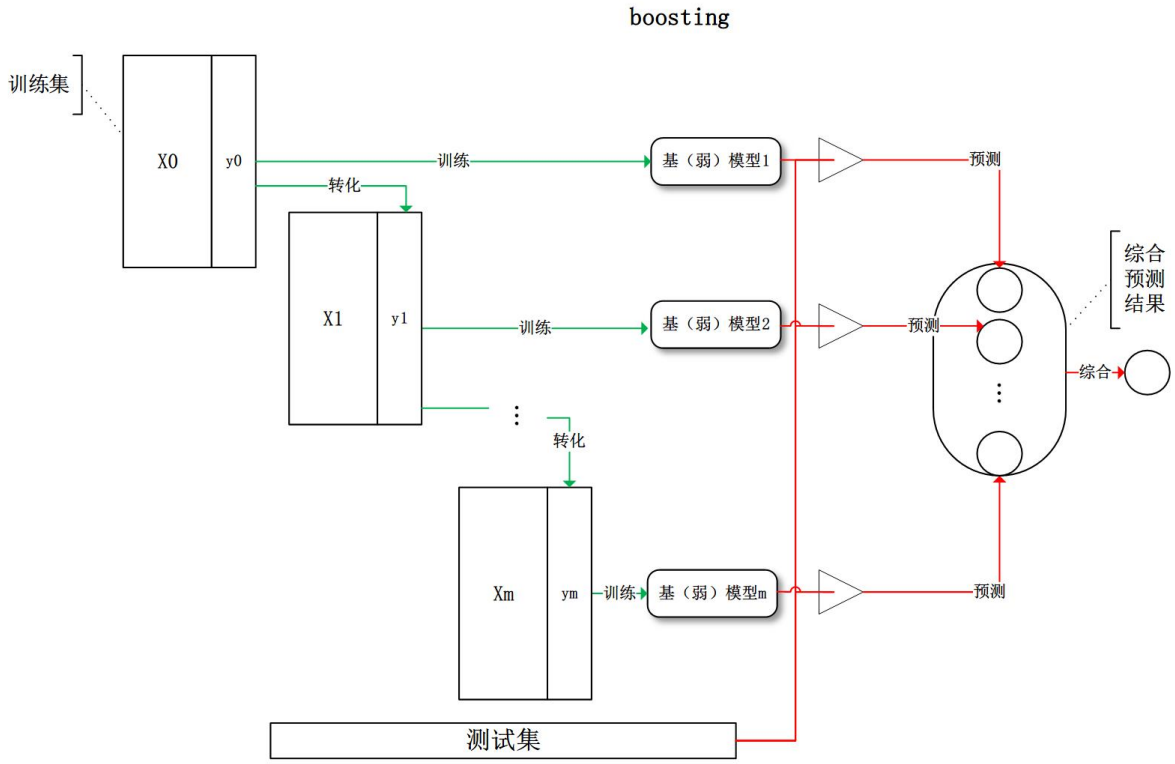
boosting(提升方法)

俗话说“三个臭皮匠顶个诸葛亮”，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。

Leslie Valiant首先提出了“**强可学习**（strongly learnable）”和“**弱可学习**（weakly learnable）”的概念，并且指出：在概率近似正确（probably approximately correct，PAC）学习的框架中，一个概念（一个类），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，如果正确率不高，仅仅比随即猜测略好，那么就称这个概念是弱可学习的。2010年的图灵奖给了L. Valiant，以表彰他的PAC理论。非常有趣的是Schapire后来证明强可学习与弱可学习是等价的，也就是说，在PAC学习的框架下，一个概念是强可学习的充要条件是这个概念是可学习的。

这样一来，问题便成为，在学习中，如果已经发现了“弱学习算法”，那么能否将它提升（boost）为”强学习算法”。大家知道，发现弱学习算法通常比发现强学习算法容易得多。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。

对于分类问题而言，给定一个训练数据，求一个比较粗糙的分类器（即弱分类器）要比求一个精确的分类器（即强分类器）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，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（训练数据中的各个数据点的权值分布），调用弱学习算法得到一个弱分类器，再改变训练数据的概率分布，再调用弱学习算法得到一个弱分类器，如此反复，得到一系列弱分类器**。



这样，对于提升方法来说，有两个问题需要回答：

1. 是在每一轮如何改变训练数据的概率分布
2. 是如何将多个弱分类器组合成一个强分类器。

关于第一个问题，AdaBoost的做法是，提高那些被前几轮弱分类器线性组成的分类器错误分类的的样本的权值。这样一来，那些没有得到正确分类的数据，由于权值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，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。

Bagging和Boosting的区别

1）样本选择上：

Bagging：训练集是在原始集中有放回选取的，从原始集中选出的各轮训练集之间是独立的。

Boosting：每一轮的训练集不变，只是训练集中每个样例在分类器中的权重发生变化。而权值是根据上一轮的分类结果进行调整。

2）样例权重：

Bagging：使用均匀取样，每个样例的权重相等

Boosting：根据错误率不断调整样例的权值，错误率越大则权重越大。

3）预测函数：

Bagging：所有预测函数的权重相等。

Boosting：每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重。

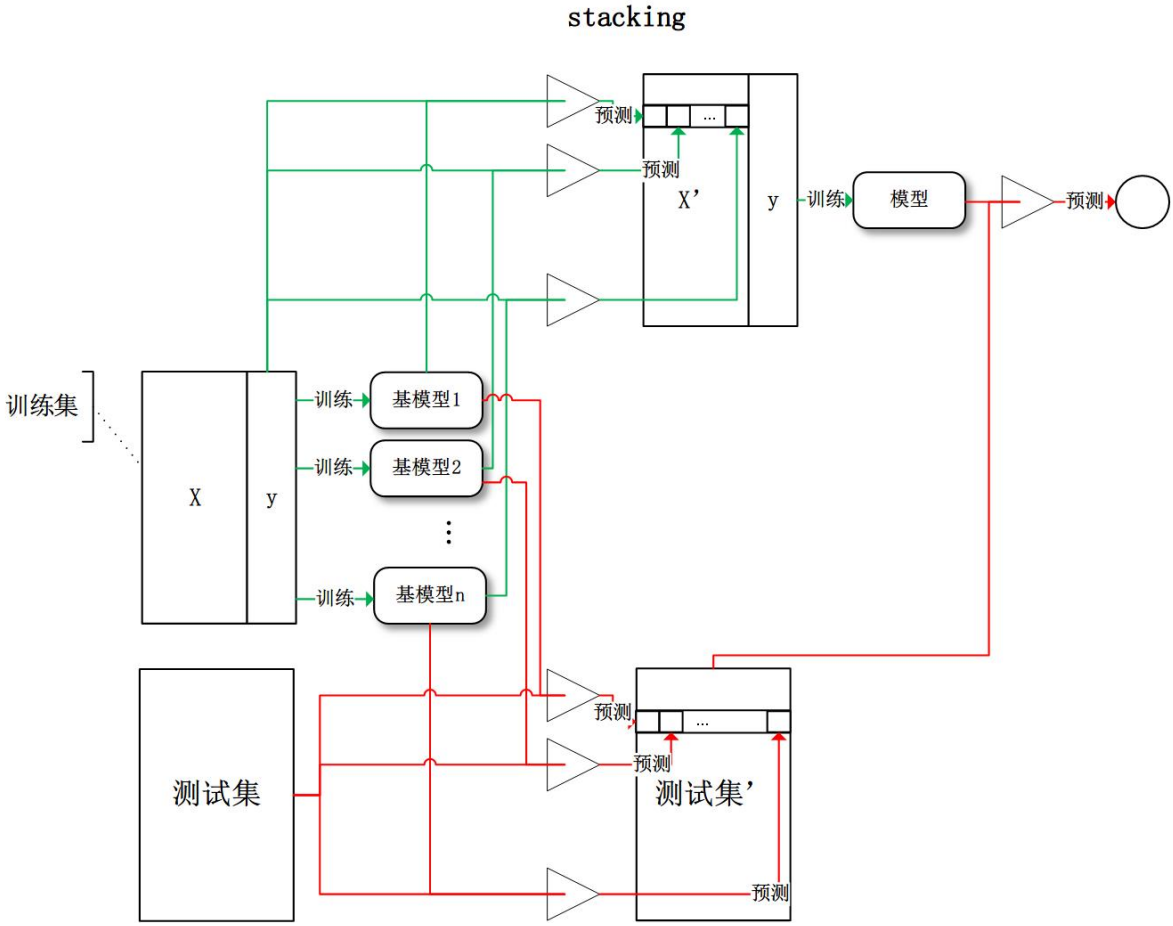
4）并行计算：

Bagging：各个预测函数可以并行生成

Boosting：各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果。

stacking模型融合

stacking 就是当用初始训练数据学习出若干个基学习器后，将这几个学习器的预测结果作为新的训练集，来学习一个新的学习器。stacking在深度学习大数据比赛中经常被使用，大概流程可以看看这篇博客。



参考资料

Boosting原理
提升方法(boosting)详解
Bagging和Boosting的区别