Bagging & Boosting

Bagging 和 Boosting 都是将已有的分类或回归算法通过一定方式组合起来，形成一个性能更加强大的分类器，更准确的说这是一种分类算法的组装方法. 即将弱分类器组装成强分类器的方法.

I. Bagging (套袋法，自主整合法)

1. bootstrapping (自助法)

bootstrapping 是一种有放回的抽样方法，它是非参数统计中一种重要的估计统计量方差进而进行区间估计的统计方法 (如均值、方差等).

其核心思想和基本步骤如下：

(1) 采用重抽样技术从原始样本中抽取一定数量(自己给定)的样本，此过程允许重复抽样.

(2) 根据抽出的样本计算统计量 T.

(3)重复上述 N 次 (一般大于 1000)，得到统计量 T.

(3) 计算上述 N 个统计量 T 的样本方差，得到统计量的方差.

应该说是 bootstrap 是现代统计学较为流行的方法，小样本效果好，通过方差的估计可以构造置信区间等.

2. Bagging (bootstrap aggregating)

Bagging 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 bootstrapping 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)

II. Boosting (提升法)

其主要思想是将弱分类器组装成一个强分类器. 在 PAC (概率近似正确) 学习框架下, 则一定可以将弱分类器组装成一个强分类器. 其中主要的是 AdaBoost (adaptive boosting), 即自适应提升法.

1. 关于 Boosting 的两个核心问题：

（1） 在每一轮如何改变训练数据的权值或概率分布？

通过提高那些在前一轮被弱分类器分错样例的权值, 减小前一轮分对样例的权值, 来使得分类器对误分的数据有较好的效果.

（2） 通过什么方式来组合弱分类器？

通过加法模型将弱分类器进行线性组合, 比如 AdaBoost 通过加权多数表决的方式, 即增大错误率小的分类器的权值, 同时减小错误率较大的分类器的权值. 而提升树通过拟合残差的方式逐步减小残差, 将每一步生成的模型叠加得到最终模型.

2. Gradient Boosting (梯度提升)

Boosting 是一种思想, Gradient Boosting 是一种实现 Boosting 的方法, 它的主要思想是, 每一次建立模型, 是在之前建立模型损失函数的梯度下降方向. 损失函数描述的是模型的不靠谱程度, 损失函数越大, 说明模型越容易出错. 如果我们的模型能够让损失函数持续的下降, 说明我们的模型在不停的改进, 而最好的方式就是让损失函数在其梯度的方向下降.

III. Bagging 和 Boosting 的区别：

在大多数数据集中, Boosting 的准确性要比 Bagging 高.

(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.

IV. 总结

这两种方法都是把若干个分类器整合为一个分类器的方法，只是整合的方式不一样，最终得到不一样的效果，将不同的分类算法套入到此类算法框架中一定程度上会提高了原单一分类器的分类效果，但是也增大了计算量.

下面是将决策树与这些算法框架进行结合所得到的新的算法：

(1) Bagging + 决策树 = 随机森林

(2) AdaBoost + 决策树 = 提升树

(3) Gradient Boosting + 决策树 = GBDT