# 集成算法(Ensembles)

## 摘要：

伴随着人工智能概念的再一次觉醒，机器学习和深度学习的概念也成为了当下研究的热点内容[1]，其中机器学习作为人工智能领域和深度学习领域中一种重要的研究手段，因此学术界和工业界对机器学习中的各种方法进行了大量的研究和应用，并根据机器学习中常见的任务划分出了回归、分类、聚类和降维的问题[2][11]，并相应提出了诸多各式各样的算法，由于算法之间存在应用领域和本身算法适用性问题，因此遇到实际问题时无法立刻找到最优的解决办法，这个问题就是机器学习领域在常见的no free lunch theorem问题[3]。而集成算法(Ensembles)是一种比较独特的机器学习算法，把简单的算法组织起来，这样就能够集合各种算法的优点，来产生一共综合的机器学习策略，这个为实际的应用和研究提供了很好的思路[4]。

本文将对集成学习算法中比较经典和流行的Bagging、Boosting和Stacking算法进行介绍并给出具体的算法设计原理和实现流程，最后简要对其性能上给出一定的比较。同时也会简要介绍一些基于这些算法基础上衍生出来的集成算法，例如Gradient Boosting(GB)、Gradient Boosting Decision Tree(GBDT)以及XGBoost算法的设计原理和实现流程。

## 引言：

集成方法是一种常见的机器学习算法，它构造一组分类器，由一组经过单独训练的分类器(如神经网络或决策树)组成，在对新测试用例进行分类时，这些分类器的预测被组合在一起，然后通过对它们的预测进行加权投票来对新的数据点进行分类。以往的研究表明，集成通常比集成中的任何单个分类器更准确。

最初的集成方法是贝叶斯平均法，随着理论研究的推进，又出现了比较流行的算法包括Bagging，Boosting，Stacking，这三类分别代表了集成算法中并行、串行和树形结合的几种不同方式[4][5]。Bagging是把各个基模型的结果组织起来，取一个折中的结果。Boosting是根据旧模型中的错误来训练新模型，层层改进。Stacking则是把基模型组织起来，基模型本身进行搭配组合，该方法看起来更灵活，也更复杂[6]。

本文将重点回顾Boosting算法和Bagging算法的原理和算法设计思路，给出具体的实现步骤，并解释为什么集合的性能通常优于任何单个分类器，然后分别介绍这两类算法的优点和不足，以及应用当作需要注意的问题。

## 算法介绍

集成学习方法的设计思想是构造一个集合，在集合中的分类器多种多样(多样性)，相互之间独立(独立性)并且都有着较高的准确率(准确性)，通过加法模型将集合中的弱分类器进行线性组合，最后利用平均法、投票法或学习法这三种组合策略来获得高度准确的分类决策。许多作者已经通过集成方法证明了显著的性能改进[6]。

Bagging (Breiman, 1996c)和Boosting (Freund & Shapire, 1996; Shapire, 1990)是两种比较新但却比较流行的集成方法[1]。

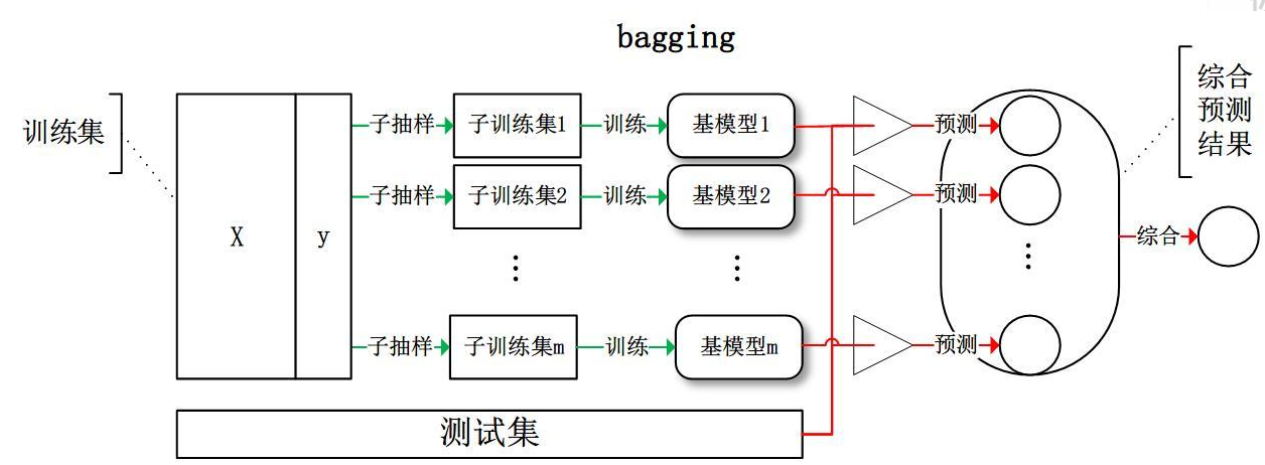
### 3.1 Bagging算法

#### 3.1.1 基本概念

Bagging的全称是bootstrap aggregating，主要借鉴了统计学上的Bootstrap估计方法，Bootstrap是一类非参蒙特卡洛(Monte Carlo)方法[12]，基本原理就是对观测信息进行再抽样，进而对总体的发布特性进行统计推理。Bagging就是按照随机有放回的方式选择训练数据然后构造分类器，最后组合。

下面将以scikit-learn工具包中提供的Bagging集成算法为例来介绍一下Bagging算法需要解决的问题：Bagging Classifier/Regressor是从原始数据集抽选S次(抽取实例，抽取属性)，得到S个新数据集(有的值可能重复，有的值可能不出现)。使用同一模型，训练得到S个分类器，预测时使用投票结果最多的分类。

下面给出的是Bagging算法的结构图。



#### 3.1.2 算法介绍

它是一系列决策树的集成，用随机的方式建立一个决策树的森林。当有一个新的输入样本进入的时候，就让森林中的每一棵决策树分别进行判断，预测时使用投票结果最多的分类，也是少数服从多数的算法。Bagging中各个基算法之间没有依赖，可以并行计算，它的结果参考了各种情况，实现的是在欠拟合和过拟合之间取折中。

Bagging预测器是一种方法，用于生成多个版本的预测器，并使用这些版本获得聚合的预测器。在预测数值结果时，聚合对版本进行平均，在预测类时进行简单投票。通过对学习集进行自举复制并将其作为新的学习集来形成多个版本。利用分类回归树和线性回归中的子集选择对真实数据集和模拟数据集进行了测试，结果表明，套袋法在精度上有较大的提高。关键因素是预测方法的不稳定性。如果对学习集的扰动会导致构建的预测器发生显著变化，那么bagging可以提高预测精度。

Random Forest (RF)算法，是在以决策树为基学习器构建Bagging集成的基础上，进一步在决策树的训练过程中引入了随机属性选择，属于Bagging算法的一种衍生算法。RF算法在模型训练过程中，采用了将属性集合作为模型训练的一个新增超参数的想法，即从节点属性集合M中随机选取k个属性的子集，然后再从这个子集中选择一个最优属性用于决策树构建。RF算法由于输入数据是随机的从整体的训练样本数据中选取的一部分作为决策树的训练，而且是有放回的选取，此外又引入了随机特征作为构建决策树的方法，这两个随机性的引入使得RF能够有效的避免过拟合现象的出现，而且还具有很好的抗噪声能力。同时RF还具备实现简单，处理高维数据时无需做特征选择等优点，但不足的地方时对噪声比较敏感。

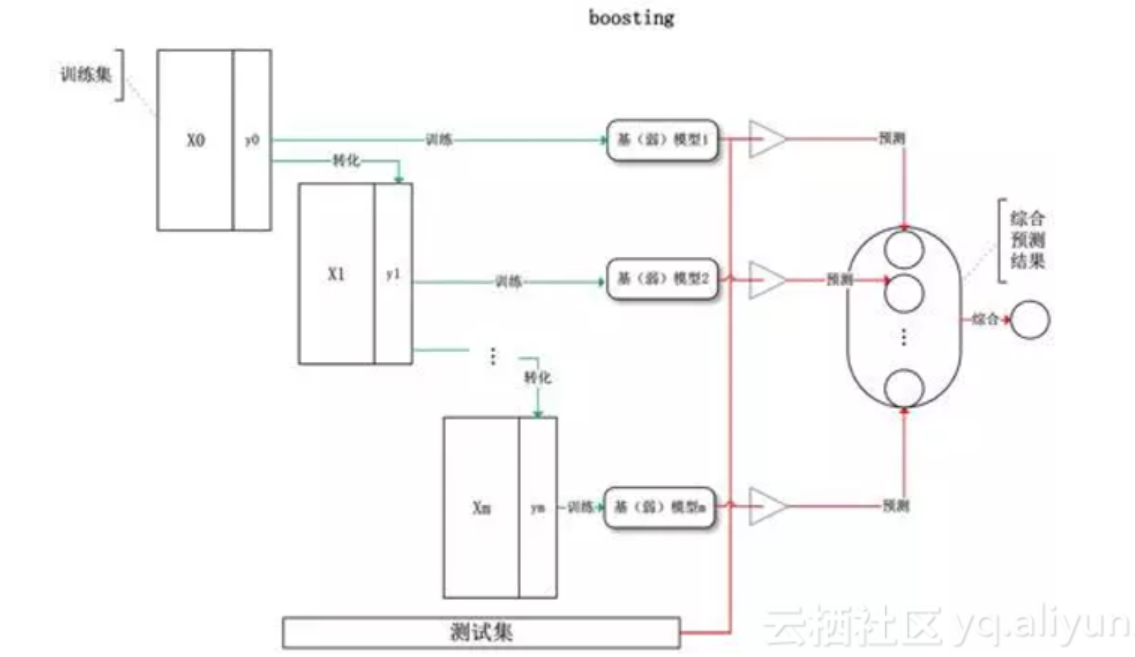
(RF算法实现流程伪代码？？？)

### 3.2 Boosting算法

#### 3.2.1 基本概念

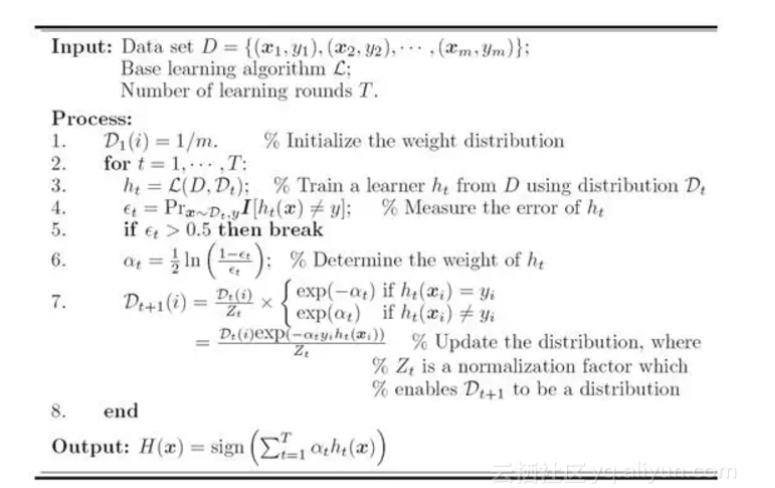
Boosting算法基于错误提升分类器性能的原理，在不断迭代建立新模型的同时，通过调整样本分布特征使新模型更重视上一个模型中被错误分类的样本，提升对数据样本整体的处理精度，迭代执行直到基学习器数目达到指定的值，最后将训练好的基学习器根据按成功度加权组合得到训练结果。

由于引入了逐步改进的思想，重要属性会被加权，这也符合人的直觉。一般来说，它的效果会比Bagging好一些。由于新模型是在旧模型的基本上建立的，是一种典型的串行集成算法。由于对错误样本的关注，也可能造成过拟合。下面给出的是Boosting算法的结构图。



#### 3.2.2 算法介绍

AdaBoost算法是Boosting方法中最具代表性的一种算法，与Bagging算法的不同之处在于AdaBoost模型中子模型必须是串行训练获得，每个新的子模型都是根据已训练出的模型性能来进行训练，每次迭代生成新的子模型使用的训练数据都相同，但是对样本的权重进行了重新设定。AdaBoost会根据当前的错误率，按照增大错误样本权重，减小正确样本权重的原则更新每个样本的权重。它对分类错误属性的给予更大权重，再做下次迭代，直到收敛。不断重复训练和调整权重，直到训练错误率或基学习器的个数满足用户指定的数目为止。Adaboost的最终结果为每个弱学习器加权的结果。如下是AdaBoost算法流程：



AdaBoost是一个相对简单的Boosting算法，可以自己写代码实现，常见的做法是基模型用单层分类器实现，桩对应当前最适合划分的属性值位置。

XGboost是Gradient Boosting的一种高效系统实现

## 模型的评价

当机器学习模型被训练出来之后往往会对模型进行。。。

平均法(加权(个体学习器性能相差较大)，简单(性能相近))，投票法(绝对多数(超过半数标记。否则拒绝预测)，相对多数，加权投票)，学习法(通过另一个学习器来进行结合，Stacking算法)

## 结论

首先，虽然Bagging几乎总是比单个分类器更准确，但有时它远不如升压准确。另一方面，提升可以产生比单个分类器更不精确的集合——尤其是在使用神经网络的时候。分析表明[7]，提升方法的性能取决于被测数据集的特性。事实上，进一步的研究结果表明，增加集合体可能会使噪声数据集过大，从而降低集合体的性能。最后，与之前的研究一致，我们的工作表明，集成电路性能的大部分增益来自前几个分类器的组合;然而，在增强决策树时，最多可以看到25个分类器的较大收益。

## 参考文献

1. LeCun Y, Bengio Y, Hinton G. Deep learning[J]. nature, 2015, 521(7553): 436.
2. Goodfellow I, Bengio Y, Courville A, et al. Deep learning[M]. Cambridge: MIT press, 2016.
3. Robert C. Machine learning, a probabilistic perspective[J]. 2014.
4. Dietterich T G. Ensemble methods in machine learning[C]//International workshop on multiple classifier systems. Springer, Berlin, Heidelberg, 2000: 1-15.
5. Dietterichl T G. Ensemble learning[J]. 2002.
6. Dietterich T G. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization[J]. Machine learning, 2000, 40(2): 139-157.
7. Opitz D, Maclin R. Popular ensemble methods: An empirical study[J]. Journal of artificial intelligence research, 1999, 11: 169-198.
8. Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection[C]//Ijcai. 1995, 14(2): 1137-1145.
9. Breiman L. Bagging predictors[J]. Machine learning, 1996, 24(2): 123-140.
10. 李航. 统计学习方法[J]. 2012.
11. 周志华. 机器学习[M]. Qing hua da xue chu ban she, 2016.
12. Hall P. The bootstrap and Edgeworth expansion[M]. Springer Science & Business Media, 2013.