集成的概念：

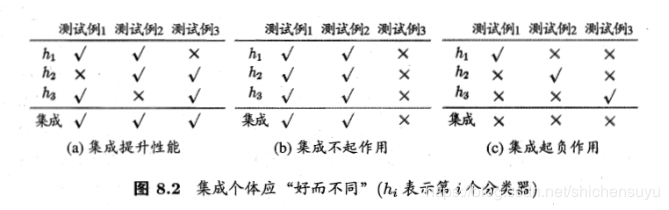
集成学习通过构建并结合多个学习期来完成学习任务，常可获得比单一学习器显著优越的泛化性能．



理论上，集成学习对于弱学习器的集成效果最明显，故许多理论研究都是以若学习器作为基学习器进行的．　　　弱学习器：指泛化性能比其他随机猜测学习器好的学习器

集成如何获得单一学习器更好的性能

若集成的组合策略是投票法，则会出现少数服从多数情况：

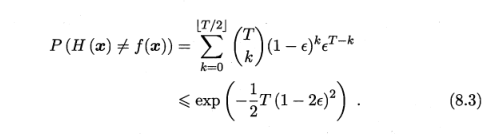


根据此类组合策略，应该保证个体学习器＂好而不同＂，才能获得好的集成．即个体学习器在保持一定准确性的同时，要保证不同个体学习器之间有一定的差异，即多样性

Tip:个体学习器至少不差于弱学习器

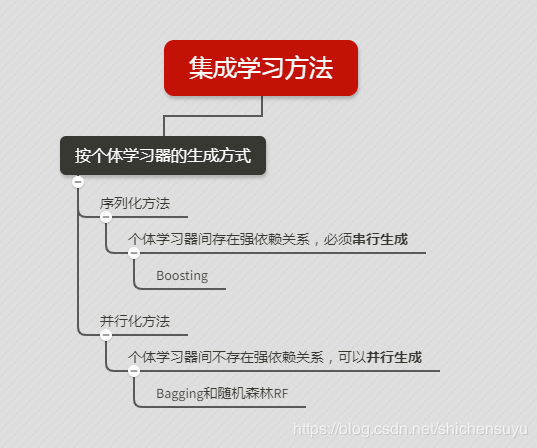
理论分析：

假设基分类器的错误率互相独立，则由hoeffding不等式可知，集成错误率为：



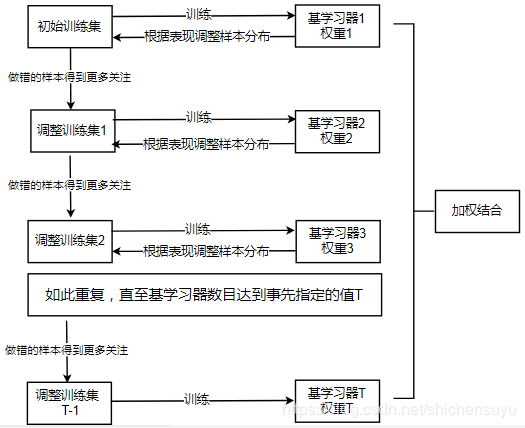
随着学习器数量T的增加，集成错误率将以指数级下降，最终趋向于０，但是以上的公式是基于基学习器的误差相互独立这个假设为前提而进行的，而在现实任务中这显然不可能．（个体学习器都是为了解决同一个问题训练出来的，不可能相互独立）。

集成学习的研究核心就在于如何产生并结合好而不同的个体学习器



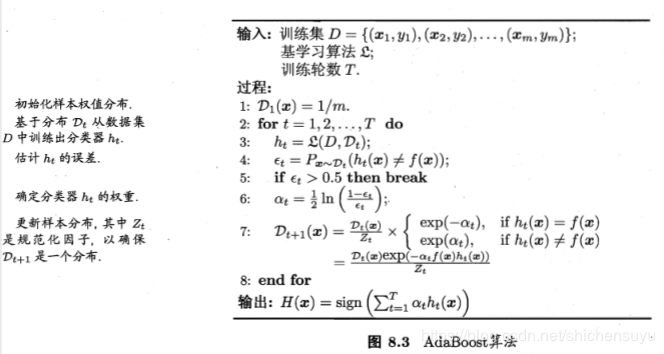
## **Boosting工作机制**

Boosting工作机制如下图所示，其中各个基学习器的对应权重由其误差计算确定，基学习器误差大的则对应的权重小，误差小的则对应权重大

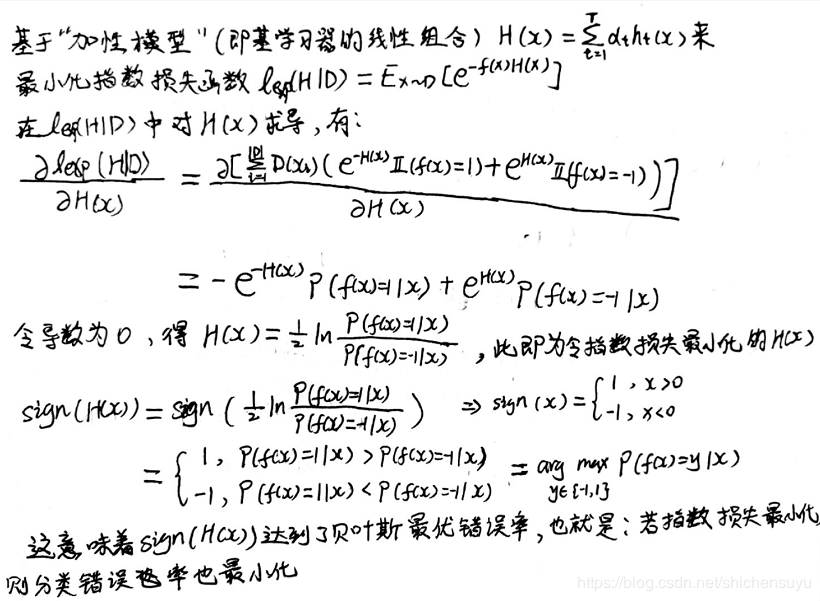


AdaBoost（序列化采样）算法推导

Boosting族算法最著名的代表是AdaBoost：



以下是基于加性模型(additive model)对AdaBoost算法的推导，标准AdaBoost只适用于二分类任务（要运用到其他任务需对AdaBoost进行修改），故以下也是同时基于二分类任务的推导：



## **Boosting算法要点**

重采样

图8.3中的第5步说明当错误率达到0.5以上则会抛弃当前基学习器，且学习过程停止。

为了避免训练过程过早停止，则可采用重采样法(re-sampling)来处理，即根据样本分布对训练集重新进行采样，再用重采样而得的样本集对基学习器进行训练。

使用重采样法，则在抛弃不满足条件的当前基学习器后，可根据当前的数据分布重新对训练样本进行采样，再基于新的采样结果重新训练新的基学习器，从而使得学习过程可以持续到预设的T轮完成。

降低偏差

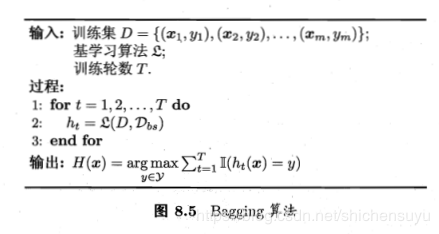
从偏差-方差分解的角度看，Boosting主要关注降低偏差，因此Boosting能基于泛化性能相当弱的学习器构建出很强的集成。

# **Bagging与随机森林**

欲得到泛化性能强的集成，则要尽量使得个体学习器好而不同，“好”和“不同”互有冲突，为缓解这个问题，可以使用互相有交叠的采样子集

## **Bagging（样本扰动）**

用自助法采样出T个含有m个训练样本的采样集，然后基于每个采样集训练出一个基学习器，再将这些基学习器进行结合。全属性训练



### **Bagging的结合策略**

分类任务：简单投票法；若票数相同则随机选择其中一个或者考察学习器投票的置信度

回归任务：简单平均法

### **Bagging的优点**

由Bagging算法的过程可见，若基学习器的计算复杂度为O(m)，则Bagging的复杂度大致为T(O(m)+O(s))，T通常是个不太大的常数，因此训练一个Bagging集成与直接使用基学习算法训练一个学习器的复杂度同阶，着说明Bagging是一个很高效的集成学习算法。

与标准AdaBoost只适用于二分类任务不同，Bagging能不经修改地用于多分类、回归等任务

自助采样使得有剩下约36.8%的样本可用作验证集来对泛化性能进行包外估计

## **随机森林（决策树+样本扰动，属性扰动）**

随机森林是Bagging的一个扩展变体，基学习器采用的是决策树，在决策树的训练过程中引入随机属性选择：在当前结点的d个划分属性中随机选择k个属性，再计算出这k个属性之间的最优划分属性。

注：k值的取值控制了随机性的引入程度：若k=d，则基决策树的构建与传统决策树相同；若k=1，则是随机选择一个属性进行划分；一般情况下，推荐值k=log2d

随机森林的多样性不仅来自样本扰动（自助采样），还来自属性扰动，这就使得最终集成的泛化性能可通过个体学习器间的差异度进一步提升

### **随机森林的特点和优点**

特点：

随机森林在训练的初始阶段（个体学习器少）性能往往较差，这是因为属性扰动带来的影响；但当个体学习器数目逐渐增加，随机森林通常会收敛到更低的泛化误差。

优点：

随机森林的训练效率常优于Bagging，这是因为属性扰动使得随机森林在训练每个个体学习器时，使用的是属性子集，而Bagging使用的是属性全集。

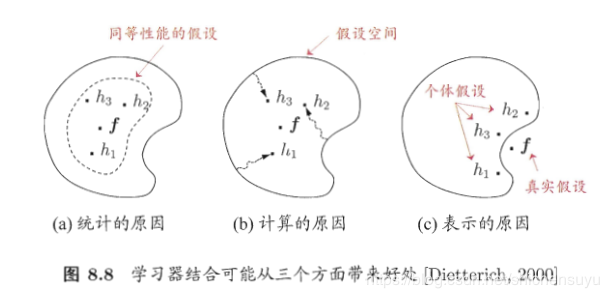
# **结合策略**

## **学习器结合三方面的好处**

统计方面：学习任务的假设空间往往很大，这就使得有可能有多个假设能达到同等性能，使用单学习器可能因误选而导致泛化性能不佳，结合多个学习器则能减小这一风险

计算方面：学习算法往往会陷入局部极小（缓解方法见第5章神经网络的学习笔记），有的局部极小对应的泛化性能可能很糟糕，而通过多次运行之后进行结合，可降低陷入糟糕局部极小点的风险

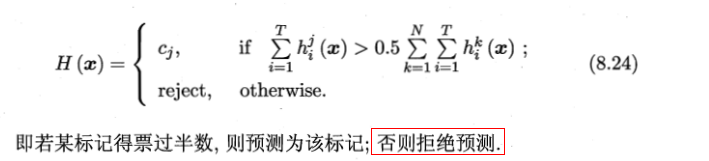
某些学习任务的真实假设可能不在当前算法所考虑的假设空间中，若使用单学习器肯定无效，而结合多个学习器可以扩大假设空间，有可能学得更好的近似。



对数值型输出，最常见的结合策略是使用平均法

* 在个体学习器性能相差较大时宜使用加权平均法
* 在个
* 体学习器性能相差较小时宜使用简单平均法

## **投票法**



在现实任务中，不同的个体学习器可能有不同类型的输出值：

类标记：hji(x)∈{0,1}}，若hi 将样本预测为类别cj则取值为1，否则为0。使用类标记的投票也称硬投票。

注同时能产生分类置信度，则分类置信度可转换为类概率使用（需进行规范化操作“校准”之后才能作为类概率）

类概率：hji(x)∈[0,1]相当于对后验概率P(cj∣x)的一个估计。使用类概率的投票也称软投票

注1：虽然类概率往往不准，但是基于类概率进行结合却往往比直接基于类标记进行结合的性能好

注2：软投票只适用于同质集成中同类型基学习器的场景，若为异质集成，且输出的是类概率，则需将类概率转化为类标记输出（如类概率最大的hji(x)设为1，其他为0）然后再投票

## **学习法（Stacking算法）**

