集成学习

翟婷婷

扬州大学
信息工程（人工智能）学院
zhtt@yzu.edu.cn

2023春

课程目标

- 理解集成学习的基本思想、研究内容和个体对集成性能的影响。
- 掌握经典的集成算法的基本原理，包括AdaBoost算法，Bagging算法和随机森林，并能够编程实现这些算法用于解决具体的模式识别问题。
- 理解多样性对集成分类器性能的影响，了解增强个体分类器的多样性的方法。

引言

- 俗语有云，“三个臭皮匠，抵个诸葛亮”，意思是说三个水平不怎么样的人， 大家一起来商量，共同决策，可能效果会比一个水平很高的人差不了多少。
- 这句话在数学上有依据吗?
- 假设有三个人，他们做出一个正确决策的概率都为p，那么，如果他们把决策意见综合一下，按照多数的意见做出集体决策，那么集体决策的正确率p_{total}是多少?

  集体决策正确有两种可能性:

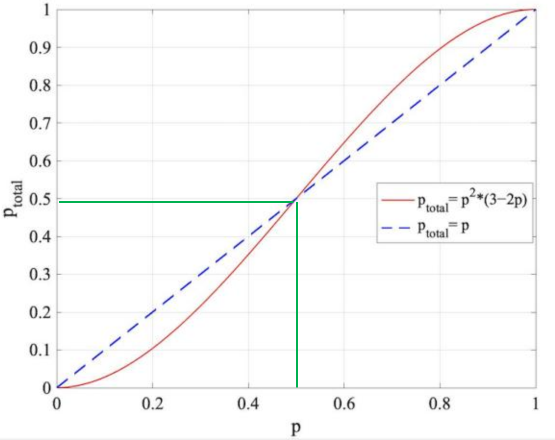
  ①三个人中有两个人决策正确

  ②三个人个体的决策都正确

$$p_{total} = C_3^2 p^2(1-p) + C_3^3 p^3 = p^2(3-2p)$$

引言

- p_{total}与p的关系图：集体决策与个体决策的关系



当个体决策没有任何依据，就是随机猜测，$p = 0.5$，$p_{total} = 0.5$，也就是说集体决策相对于个体决策并没什么长进。

当个体决策的正确率还不如随机猜测，即$p < 0.5$时，有$p_{total} < p$，表明集体决策比个体决策更差。

只有当个体决策稍微有一点依据，即$p > 0.5$时，才有$p_{total} > p$，即集体决策比个体决策更好。

引言

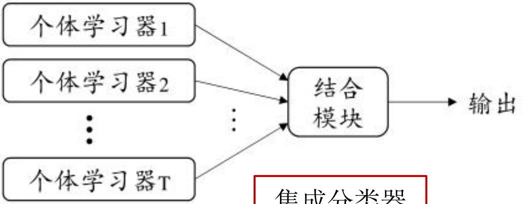
- 把臭皮匠替换成个体分类器，就得到“集成学习(ensemble learning)”的基本思想：将多个泛化性能略优于随机猜测的弱分类器，组合形成一个性能更好的强分类器。

个体分类器

弱分类器

基分类器

相同的概念



集成分类器

- 集成学习研究：(1) 怎样获得不同的弱分类器？(2) 怎样组合弱分类器？

集成学习:个体对集成的影响

- 例子：在二分类问题中，假定3个分类器h_1, h_2, h_3在三个测试样本上的表现如下图所示，其中√ 表示分类正确，X 号表示分类错误，集成的结果通过投票产生。

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	×	√	√
h_3	√	×	√
集成	√	√	√

(a) 集成提升性能

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	√	√	×
h_3	√	√	×
集成	√	√	×

(b) 集成不起作用

	测试例1	测试例2	测试例3
h_1	√	×	×
h_2	×	√	×
h_3	×	×	√
集成	×	×	×

(c) 集成起负作用

要获得好的集成，个体学习器应“好而不同”：

- 个体分类器的准确性不能太低；
- 个体分类器要有多样性 (diversity) / 差异性。

集成学习:个体对集成的影响

- 给定一个二分类问题，有n个不同的分类器能解决该问题，且每个分类器的错误率均为ε。一个集成分类器通过多数表决法结合这n个分类器。假设个体分类器的错误率是相互独立的，则集成分类器的错误率p_{error}为多少?
- n个分类器中有k个预测正确的概率为:

$$C_n^k \varepsilon^{n-k}(1-\varepsilon)^k$$

- 当集成分类器预测错误时，一半以下的分类器预测正确，即当$k=0,1,2,\cdots\lfloor n/2\rfloor$时，集成预测错误:

$$p_{error}=\sum_{k=0}^{\lfloor n/2\rfloor} C_n^k \varepsilon^{n-k}(1-\varepsilon)^k$$

集成学习:个体对集成的影响

➢ 利用Hoeffding不等式，当$\varepsilon \leq \frac{1}{2}$时，可以得到:

$$p_{error} = \sum_{k=0}^{\lfloor n/2 \rfloor} C_n^k \varepsilon^{n-k}(1-\varepsilon)^k \leq exp\left(-\frac{1}{2}(1-2\varepsilon)^2 n\right)$$

上式表明，当个体分类器的错误率$\varepsilon \leq \frac{1}{2}$时，

① 随着集成中个体分类器数目的增加，集成的错误率将指数级下降，最终趋向于0。

② 随着个体错误率ε逐渐减小，集成的错误率也下降，也就是说，当个体分类器的准确率越高，集成的准确率也就越高。

集成学习:个体对集成的影响

- 上面分析的关键假设：个体学习器的误差相互独立。

  但现实任务中，个体分类器是为解决同一个问题训练出来的，不可能互相独立。

  个体分类器的“准确性”和“多样性”存在冲突。

- 如何产生“好而不同”的个体分类器是集成学习研究的核心。

集成学习

- 两种经典的集成学习方法：

  - ✓ 代表算法：Boosting (AdaBoost)
    个体分类器串行训练，存在强依赖关系

  - ✓ 代表算法：Bagging和随机森林
    个体分类器并行训练，不存在依赖关系

Boosting算法基本思想

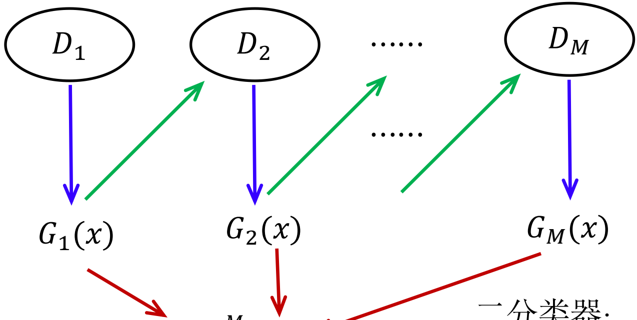
- Boosting是一族可将弱学习器提升为强学习器的算法。这族算法的工作机制类似：

  先从初始训练集训练出一个基分类器，再根据该基分类器的表现对训练样本的权值分布进行调整，使得被基分类器错分类的训练样本在后续学习中受到更多关注，然后基于调整过权值的训练数据集来训练下一个基分类器；如此重复进行，直至基分类器数目达到事先指定的值M, 最终将这M个基分类器进行加权结合。

- 特点：①基分类器串行生成，存在强依赖关系；②每次调整训练数据的权值分布

- Boosting 族算法最著名的代表是 AdaBoost。

AdaBoost算法



$$f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$$

二分类器:

$$H(x) = sign(f(x))$$

- 如何改变训练数据的权值分布?
- 如何对基分类器进行线性组合?

AdaBoost算法

- 两个问题如何解决：

① 每一轮如何改变训练数据的权值分布？

- AdaBoost：提高那些被前一轮弱分类器错误分类样本的权值，降低那些被正确分类样本的权值。

② 如何将基分类器组合成一个强分类器？

- AdaBoost：加权多数表决，加大分类误差率小的基分类器的权值，使其在表决中起较大的作用，减小分类误差率大的基分类器的权值，使其在表决中起较小的作用。

AdaBoost算法

- 输入：二分类训练数据集$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$，其中$\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$，$y_i \in \{-1, +1\}$，弱学习算法
- 输出：集成分类器$G(\boldsymbol{x})$

(1) 初始化训练数据的权值分布：

$$\mathcal{D}_1 = (w_{11}, w_{12}, \cdots w_{1N}), \quad w_{1i} = \frac{1}{N}, \forall i = 1,2, \cdots N$$

(2) 对$m = 1,2, \cdots M$

(a) 在权值分布为$\mathcal{D}_m$的训练数据集上利用弱学习算法训练得到弱分类器：$G_m(\boldsymbol{x}): \mathcal{X} \rightarrow \{-1, +1\}$

(b) 计算$G_m(\boldsymbol{x})$在训练数据集上的分类错误率：

$$e_m = \sum_{i=1}^{N} P(G_m(\boldsymbol{x}_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} \mathbb{I}[G_m(\boldsymbol{x}_i) \neq y_i] = \sum_{G_m(\boldsymbol{x}_i) \neq y_i} w_{mi}$$

AdaBoost算法

(2)(c) 计算$G_m(\boldsymbol{x})$的投票系数：

$$\alpha_m = \frac{1}{2}\ln\frac{1-e_m}{e_m}$$

(d) 更新训练样本的权值分布：

$$\mathcal{D}_{m+1} = (w_{m+1,1}, w_{m+1,2}, \cdots w_{m+1,N})$$

$$w_{m+1,i}' = \begin{cases} w_{m,i}\exp(-\alpha_m) & if\ y_i = G_m(\boldsymbol{x}_i) \\ w_{m,i}\exp(\alpha_m) & if\ y_i \neq G_m(\boldsymbol{x}_i) \end{cases}$$

$$w_{m+1,i} = \frac{w_{m+1,i}'}{\sum_{j=1}^{N} w_{m+1,j}'},\ \forall i = 1,2,\cdots N$$

规范化：使$\mathcal{D}_{m+1}$成为一个概率分布。

(3) 构建弱分类器的线性组合：

$$f(\boldsymbol{x}) = \sum_{m=1}^{M}\alpha_m G_m(\boldsymbol{x})$$

得到最终集成二分类器：$H(\boldsymbol{x}) = sign(f(\boldsymbol{x}))$

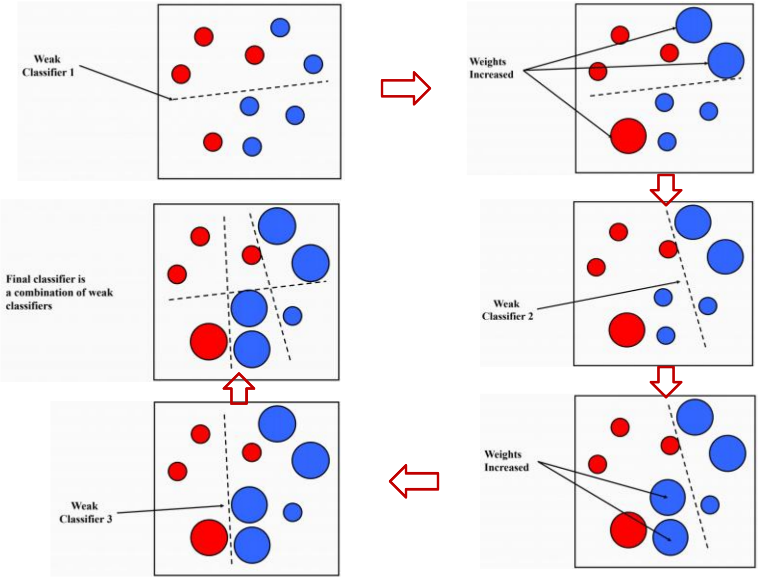
AdaBoost算法说明

- 在(2)(b)处，要检查$e_m < 0.5$吗，即检查$G_m(\boldsymbol{x})$的性能是否比随机猜测好，如果不满足，则学习过程停止，此时，设置的学习轮数M也许远未达到，导致最终集成中只包含很少的基分类器而性能不佳。
- 为避免训练过程过早停止，可在抛弃$e_m \geq 0.5$的$G_m(\boldsymbol{x})$之后，根据当前的权值分布对训练样本进行采样，基于重新采样得到的数据集重新训练基分类器，从而保证能够学到预设的M个基分类器。
- 前面给出的Adaboost算法只适用于二分类任务，为处理多分类或回归任务，需对算法进行修改。
- Adaboost要求弱学习算法能利用带权值的训练样本进行学习。

AdaBoost算法说明

- 如果Adaboost采用BP神经网络为弱学习算法，那么如何修改BP算法使其能利用带权值的训练样本进行学习？
- 当所有训练样本的权重相同时，BP算法的学习目标是：找到一组网络参数，最小化在全体训练样本上的训练误差 $\frac{1}{2}\sum_{t=1}^{N}\|\hat{\boldsymbol{y}}_t-\boldsymbol{y}_t\|^2$，其中，$\hat{\boldsymbol{y}}_t$是网络对$\boldsymbol{x}_t$的预测。
- 当训练样本的权重不一样时，假设对于$\forall t=1,2,\cdots,\mathrm{N}$，样本$(\boldsymbol{x}_t,\boldsymbol{y}_t)$的权重为$w_t$，则此时BP算法的学习目标是：找到一组网络参数，最小化在全体训练样本上的加权训练误差 $\frac{1}{2}\sum_{t=1}^{N}w_t\|\hat{\boldsymbol{y}}_t-\boldsymbol{y}_t\|^2$。利用随机梯度下降法，最终转化为在每一轮中，最小化当前的加权误差 $E_t=\frac{1}{2}w_t\|\hat{\boldsymbol{y}}_t-\boldsymbol{y}_t\|^2$。通过推导发现，如果$u\leftarrow u-\eta g_t$是不加权重的更新公式，则$u\leftarrow u-\eta w_t g_t$是带权重的更新公式。

AdaBoost算法图示



Bagging算法 (Bootstrap AGGregatING)

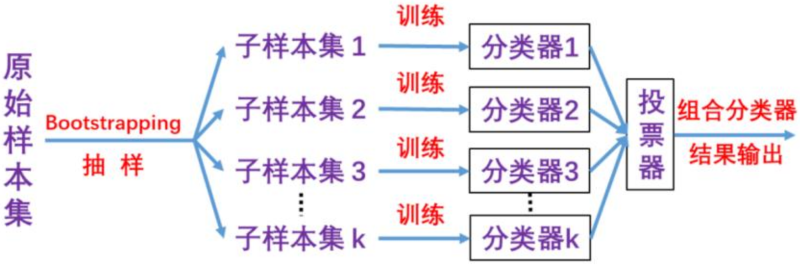
- Bagging是**并行式**集成学习方法最著名的代表，它的基础是统计学中的自助抽样(Bootstrapping)。
- **自助抽样**：给定包含m个样本的数据集D，对该数据集进行m次**有放回抽样**，得到新的样本集D'。显然，D 中有一部分样本会在D'中多次出现，而另一部分样本则在D'中不出现。
- 样本在m次采样中始终不被采到的概率是$\left(1-\frac{1}{m}\right)^m$，取极限得到

$$\lim_{m \mapsto \infty}\left(1-\frac{1}{m}\right)^m \mapsto \frac{1}{e} \approx 0.368$$

即通过自助采样，初始数据集D中约有36.8% 的样本未出现在自助采样得到的数据集D'中。

Bagging算法

- 使用Bootstrapping抽样从原始样本集中获得若干独立同分布的子样本集，然后使用这些样本集分别训练出一组相同形式的弱分类器，再通过投票多数表决的方式将弱分类器组合成一个强分类器。



并行式训练

随机森林 (Random Forest)

- 随机森林是Bagging的一个扩展变体，它在以决策树为弱学习算法构建 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。

- 样本选择的随机性：bootstrap采样引入的

- 属性选择的随机性：决策树学习引入的

随机森林 (Random Forest)

- 传统决策树在选择划分属性时是在当前结点的可利用的属性集合中选择一个最优划分属性。

- 随机森林在构建决策树时，随机选择属性：

  对每棵决策树的每个结点，先从该结点的可利用属性集合中随机选择一个包含k个属性的子集，再从这个子集中选择一个最优的划分属性。

  假设一个结点包含的属性个数为d

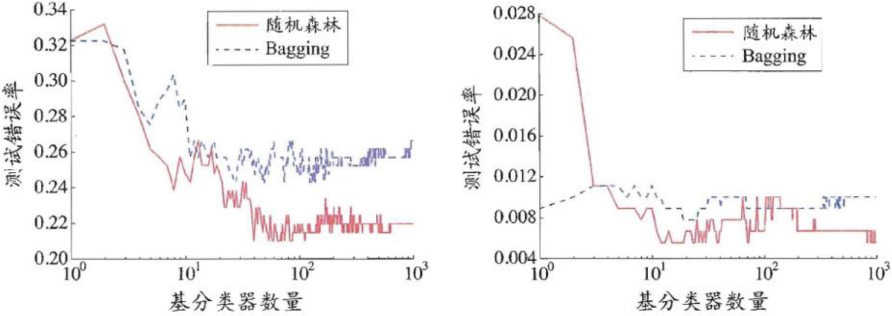
  若$k = d$，则个体决策树的构建与传统决策树相同

  若$k = 1$，则是随机选择1个属性用于划分

  一般情况下，推荐$k = \log_2 d$。

随机森林 (Random Forest)

在两个 UCI 数据上，集成规模对随机森林与 Bagging 的影响



(a) glass 数据集　　(b) auto-mpg 数据集

随机森林的起始性能往往相对较差，随着个体学习器数目的增多，通常会收敛到更低的泛化误差；随机森林的训练效率优于Bagging,因为Bagging使用的是确定型的决策树，而随机森林使用的随机型决策树。

增强个体间多样性的方法

- 常见的增强个体学习器之间多样性的方法:
  - 数据样本扰动　　输入属性扰动
  - 输出表示扰动　　算法参数扰动
- 数据样本扰动通常是基于采样法
  - ✓ Bagging中的自助采样法
  - ✓ Adaboost中的序列采样法

数据样本扰动对“不稳定基学习器”很有效

对样本的扰动敏感的基学习器(不稳定基学习器)

- ✓ 决策树，神经网络等　(训练样本集稍加变化就会导致学习器有显著变动)

对样本的扰动不敏感的基学习器(稳定基学习器)

- ✓ 支持向量机，朴素贝叶斯，k近邻等

增强个体间多样性的方法

- 输入属性扰动：训练样本通常由一组属性描述，不同的"子空间" (即属性子集)，提供了观察数据的不同视角。让个体学习器在不同的子空间中训练得到。
- 输出表示扰动：对输出表示进行操纵以增强多样性。
  - ✓ 翻转法 (Flipping Output)：随机改变一部分训练样本的标记
  - ✓ 输出调剂法(Output Smearing)：将分类输出转化为回归输出等
  - ✓ ECOC法：利用纠错输出码将多分类问题拆解为一系列的二分类任务
- 算法参数扰动：通过随机设置不同的参数产生差别较大的个体学习器。

集成分类器设计要求

- 个体分类器的准确率要尽量高，最坏的情况下，也要大于随机猜测的准确率，否则集成分类器的性能会比单个分类器更差。
- 个体分类器要有差异，保证它们的预测结果也有差异，在进行集成时能各自提供不同的信息，从而提升集成分类器的性能。
- 个体分类器的数量并不是越多越好，要权衡。虽然个体分类器数量越多，集成分类器的性能会越好，但是集成分类器的训练代价也越来越高，高的训练代价可能只能带来小幅度的性能提升，因此要权衡。

小结

- 集成学习的研究动机：将多个泛化性能略优于随机猜测的弱分类器，组合形成一个性能更好的强分类器。
- 集成学习研究的2个关键问题：(1) 怎样获得不同的个体分类器？(2) 怎样组合个体分类器？其中，如何产生“好而不同”的个体分类器是集成学习研究的核心。
- 两种经典的集成学习方法：一是串行训练个体分类器，使得个体分类器之间存在强依赖关系的Boosting族算法，其代表算法是 AdaBoost。二是并行训练个体分类器，使得个体分类器之间不存在依赖关系的Bagging算法，及其变种算法随机森林。

重点

小结

- AdaBoost先从初始训练集训练出一个弱学习器，再根据弱学习器的表现对训练样本的权值分布进行调整，使得被该弱学习器错分类的样本在后续受到更多关注，然后基于调整后的样本分布来训练下一个弱学习器；如此重复进行，直至弱学习器数目达到事先指定的值，最终将所有的弱学习器进行加权组合。

重点

- Bagging算法是在bootstrap采样得到的不同训练集上训练多个弱分类器，通过多数投票组合这些分类器。

重点

- 随机森林使用Bagging算法组合若干弱学习器-决策树，其中，决策树构建时在每个结点处并不是从所有可能的属性集合中，而是从随机选择的一个属性子集中选择最优的划分属性。

重点

第8页错误率的证明

假设抛硬币正面朝上的概率为 p, 反面朝上的概率为 $1-p$. 令 $H(n)$ 代表抛 n 次硬币所得正面朝上的次数, 则最多 k 次正面朝上的概率为

$$P(H(n) \leqslant k) = \sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-i} .$$

对 $\delta > 0$, $k = (p-\delta)n$, 有 Hoeffding 不等式

$$P(H(n) \leqslant (p-\delta)n) \leqslant e^{-2\delta^2 n} .$$

使用上述定理推导出下式:

$$p_{error} = \sum_{k=0}^{\lfloor n/2 \rfloor} C_n^k \varepsilon^{n-k} (1-\varepsilon)^k \leq exp\left(-\frac{1}{2}(1-2\varepsilon)^2 n\right)$$