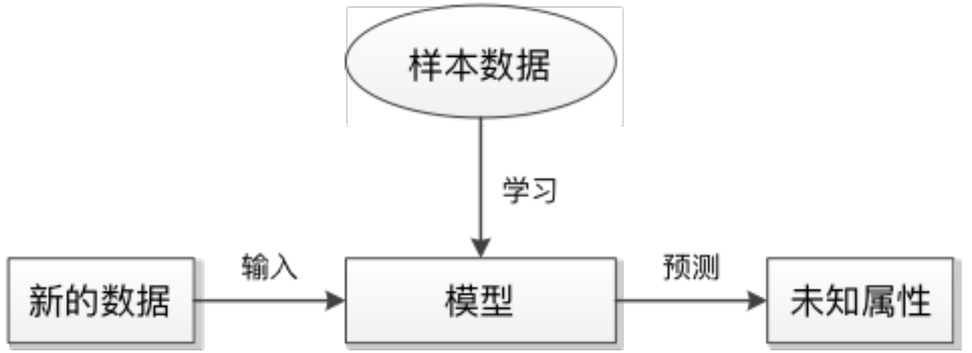
数学归纳法，树和机器学习

学习

- 学习是人类以及各种动物与生俱来的基本能力，自从人们试图在计算机上表现人类智能之日起，学习就成为研究的主要课题。
- H.Simon曾于1983年给出了一个关于学习的哲学式说明：如果一个系统能够通过执行某种过程而改进其性能，这就是学习。以常见的有监督学习为例，机器学习可以理解为针对所给定的样本集{(xi,yi): i=1,2,…,n}，该样本集来自实际问题y=F(x)（称为自然模型）的独立同分布采样，我们需要设计算法以得到函数y=f(x)，使得它对自然模型在一定的统计指标下为真，即f(x)是自然模型的一个近似模型



从IF…THEN开始

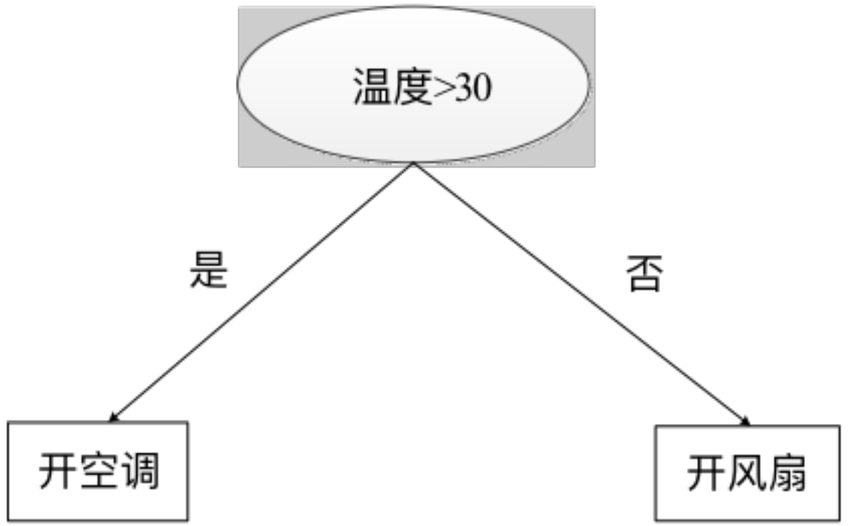
- 天热决定如何降温时，收集了一些以前的样例：

温度	是否开空调
29	否
31	是
33	是
26	否
32	是

- 观察分析这些数据，可以很容易得到这样的决策：

  IF 温度>30
  THEN 开空调
  ELSE 开风扇

图形的方式来描述



- 这就是最简单的决策树，根节点表示在一个属性上的测试或判断，每个分枝代表一个测试输出，而每个树叶节点存放一个相应的决策。

另一个例子

IF 天气晴朗 THEN

IF 湿度<=75

THEN 打球

ELSE 不去打球

ELSEIF 天气多云THEN 去打球

ELSEIF 下雨 THEN

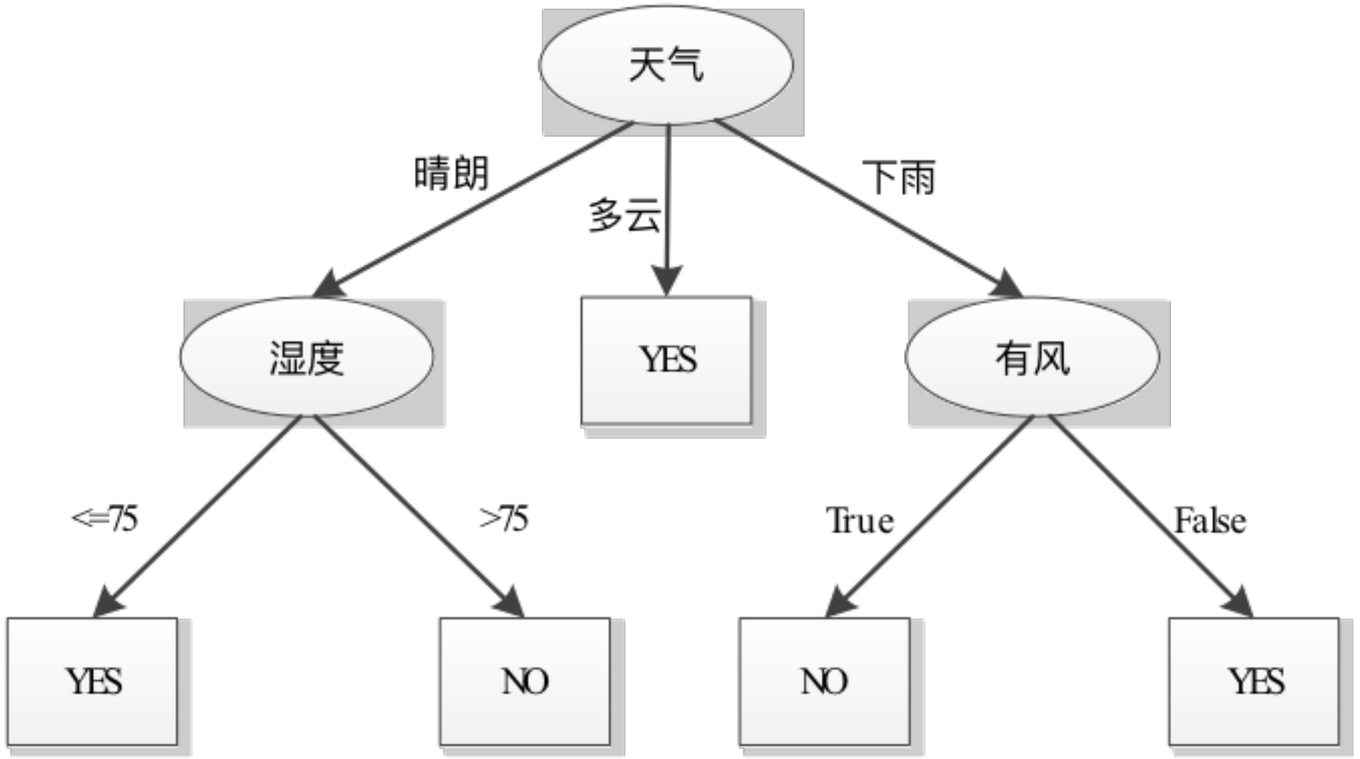
IF 有风

THEN 不去打球

ELSE 去打球

- 这是一条能够帮助人们根据天气情况决定是否去打高尔夫球的决策规则，与前面天热决定如何降温的规则相比要复杂多了，通过将多个IF…THEN复合嵌套，在增加复杂性的同时我们获得了一条相当“聪明”的规则，可以处理多种复杂情况。
- 但是比较难理解。
- 将此规则用另一种容易阅读并理解的形式表示出来

另一个例子



决策树

- 决策树可以看成是一个IF…THEN规则的集合，是一种类似流程图的树结构，其中每个内部节点（非树叶节点）表示在一个属性上的测试，每个分枝代表一个测试输出，而每个树叶节点存放一个类标号。
- 一旦建立好了决策树，对于一个需要决策的输入，跟踪一条从根节点到叶节点的路径，该叶节点就存放着该输入的预测（或者分类、决定等）。
- 因此决策树可以看作由条件IF（内部节点）和满足条件时对应的规则THEN（边）组成的。

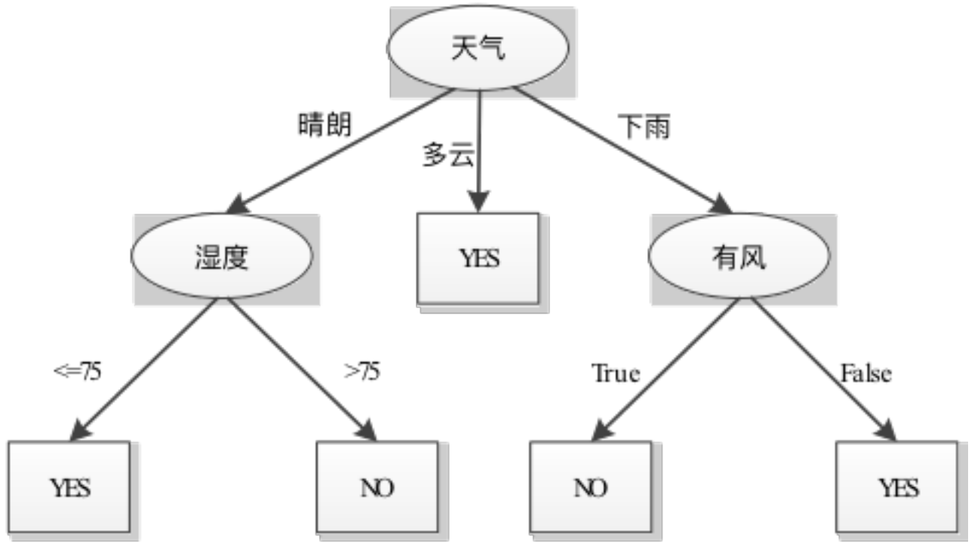
这棵决策树是怎么得到的呢?

- 通过学习，通过学习事先收集的实例，就可以得到决策树。

Day	Outlook	Temperature	Humidity	Windy	Play?
1	Sunny	85	85	False	No
2	Sunny	80	90	True	No
3	Overcast	83	78	False	Yes
4	Rainy	70	96	False	Yes
5	Rainy	68	80	False	Yes
6	Rainy	65	70	True	No
7	Overcast	64	65	True	Yes
8	Sunny	72	95	False	No
9	Sunny	69	70	False	Yes
10	Rainy	75	80	False	Yes
11	Sunny	75	70	True	Yes
12	Overcast	72	90	True	Yes
13	Overcast	81	75	False	Yes
14	Rainy	71	80	True	No

怎样从数据集得到决策树呢

- 数据集本身有很多属性，我们怎么知道首先要对哪个属性进行判断，接下来要对哪个属性进行判断?
- 在图中，我们怎么知道第一个要测试的属性是Outlook，而不是Windy？而且，学习实例中包含了温度信息，但是决策树中却没有涉及到，这又是为什么呢?



决策树学习算法

- 常见的决策树学习算法有ID3、C4.5、CART等
- 这些不同的学习算法主要差别在于它们选择特征的依据不同，而决策树的生成过程都是一样的，都是根据当前环境对特征进行贪婪的选择，从而将学习样本集分成不同的子集，得到规模小一些的分类问题。

ID3算法

- ID3算法也采用数学归纳法的思想，将原始问题进行分解，降低问题规模。
- 最容易解决的问题是所有训练样例都属于同一类的情况，这时不需要做任何测试判断，直接就可以输出结果了
- 如果训练样例不属于同一类，则通过自顶向下构造决策树来进行学习。第一个需要解决的问题是树的根节点测试哪一个属性?
- 为了回答这个问题，需要利用统计测试来求出每一个实例属性单独分类训练样例的能力，分类能力最好的属性将被选作根节点的测试属性。然后为根节点属性的每个可能值产生一个分支，并将训练样例分配到适当的分支之下。
- 通过以上操作，原始的决策树构造问题就被分解成数个小规模的决策树构造问题。重复前面的过程，用每个分支节点关联的训练样例来选取在该分支节点测试的属性，直至每个分支下的训练样例属于同一类为止。

ID3算法

算法：ID3（学习布尔函数）
输入：学习样本集合Example，样例属性集
输出：决策树

1. 创建树的根节点
2. 如果样例都为正，则返回标签为正的单节点树
3. 如果样例都为负，则返回标签为负的单节点树
4. 求出属性中对样本分类能力最好的属性A
5. 对于属性A的每一个可能值v_i
    1）在根节点下增加分支对应$A=v_i$的测试结果
    2）设Example(i)为Example中满足属性A的值为v_i的子集
    3）递归调用ID3(Example(i),样本属性集-{A})

如何求对样本分类能力最好的属性

- ID3算法使用信息增益来衡量给定的属性区分训练样例的能力，从而在候选属性中选择
- 熵是表示随机变量不确定性的度量。设X是一个取有限个值的离散随机变量，其概率分布为：$Prob(X=x_i)=p_i$
- 则随机变量X的熵定义为：$H(X)=-\sum p_i \log p_i$ 熵越大，不确定性越大
- 熵只依赖X的分布，和X的取值没有关系。熵是用来度量不确定性，当熵越大，意味着随机变量X的不确定性越大，反之越小，相应的在机器学习分类中，熵越大即表示这个类别的不确定性更大，反之越小。
- 条件熵表示在已知随机变量X的条件下随机变量Y的不确定性度量。设有两个随机变量X和Y，在随机变量X给定的条件下随机变量Y的条件熵H(Y|X)，定义为X给定条件下Y的条件概率分布的熵对X的数学期望：$H(Y|X)=\sum Prob(X=x_i)H(Y|X=x_i)$
- 信息增益表示得知特征X的信息而使得类Y的信息的不确定性减少的程度。特征A对训练数据集D的信息增益g(D, A)，定义为集合D的熵H(D)与特征A给定条件下D的条件熵H(D|A)之差，即：$g(D,A)=H(D)-H(D|A)$
- 信息增益大的特征具有更强的分类能力。

如何求对样本分类能力最好的属性

- 14个样本，其中9个为yes，5个为no，因此该训练样本集合S的熵为H(S)=-9/14log(9/14)-5/14log(5/14)=0.940
- 考虑风力windy的信息增益H(S|Windy)= Prob(Windy=True)H(S | Windy=True)+Prob(Windy=False)H(S | Windy=False)=6/14*(-3/6log(3/6)-3/6log(3/6)+8/14*(-2/8log(2/8)-6/8log6/8)=0.892
- 因此g(S,Windy)=H(S)-H(S | Windy)=0.048
- 类似的，我们可以求出其它三个属性的信息增益分别为：
- g(S,Outlook)=0.246
- g(S,Humidity)=0.151
- g(S,Temperature)=0.029
- 由于Outlook的信息增益最大，即属性outlook在训练样本上提供了对决策目标的最佳预测，因此outlook被选作根节点的决策属性

如何求对样本分类能力最好的属性

- 当outlook被选作根节点的决策属性后，要为其每一个可能取值（sunny，overcast，rainy）在根节点下创建分支，所提供的学习样本根据其outlook属性的取值情况分为3组供后续决策树的构造。
- outlook属性取值为overcast的分支，由于其分配到的样本其决策值都为yes，所以该分支下面的节点就是一个值为yes的叶节点
- 其余两个分支，由于分配到的样本决策值同时存在yes和no，因此需要重复刚才的过程，选择新的属性来分割学习样本，并且仅使用与这个分支相关的样本，如outlook属性取值为sunny的分支，继续学习时只使用outlook属性为sunny的5个样本。

学习过程结束的标准

- 对于一条从根节点开始的路径，如果所有属性都在这条路径中测试过，则该路径不再扩展。
- 对于一个测试节点而言，如果其某个分支分配到的样本其决策值一致，则该分支下面的节点为叶节点。

缺点

- 在上述ID3算法中，决策树的学习是根据信息增益来进行属性的选择的
- 但是会偏向于具有大量值的属性。
- 在训练集中，某个属性所取的不同值的个数越多，那么越有可能拿它来作为分裂属性。
- 例如在上述的例子中增加一个日期属性，每个样本的日期都不同，则在所有属性中，日期属性具有最大的信息增益，单独的日期就可以完全预测训练数据的目标属性，于是日期属性就会被选中作为根节点的决策属性并形成一棵深度为1但是非常宽的树，这棵树可以百分之百地正确分类训练数据，但对于不在训练集中的数据，由于日期不一致，因此无法进行分类，因而不是一个好的预测器。

改进

- C4.5采用了信息增益率
  - 信息增益率$g_R(D, A)$定义为其信息增益$g(D, A)$与训练数据集D关于特征A的值的熵$H_A(D)$之比，即

$$g_R(D,A) = \frac{g(D,A)}{H_A(D)}$$

  - 其中分裂信息为

$$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \log \frac{|D_i|}{|D|}$$

  - 在这些公式中，$D_1,D_2,\ldots,D_n$是根据属性A对样本集分割形成的n个样本子集，每个子集中的样本属性A取值相同。

改进

- CART算法则以基尼指数（gini index）做为属性选择的依据。
  - 在分类问题中，假设有K个类，样本属于第k类的概率为p_k，则概率分布的基尼指数定义为：

$$\mathrm{Gini}(p)=\sum p_k(1-p_k)=1-\sum p_k^2$$

  - 对于二分类问题，若样本点属于第一个类的概率是p，则概率分布的基尼指数为Gini(p)=2p(1-p)
  - 对于给定的样本集合D，其基尼指数为：

$$\mathrm{Gini}(D)=1-\sum\left(\frac{|D_k|}{|D|}\right)^2$$

  - 这里，D_k是D中属于第k类的样本子集，k是类的个数。
  - 如果样本集合D根据特征A是否取到某一可能值a被分割成D1和D2两部分，则在特征A的条件下，集合D的基尼指数定义为：

$$\mathrm{Gini}(D,A)=\frac{|D_1|}{|D|}*\mathrm{Gini}(D_1)+\frac{|D_2|}{|D|}*\mathrm{Gini}(D_2)$$

  - 基尼指数Gini(D)表示集合D的不确定性，基尼指数越大，样本集合的不确定性也就越大，这一点与熵相似。CART用作分类树时采用基尼指数最小化原则，进行特征选择，递归地生成决策树。

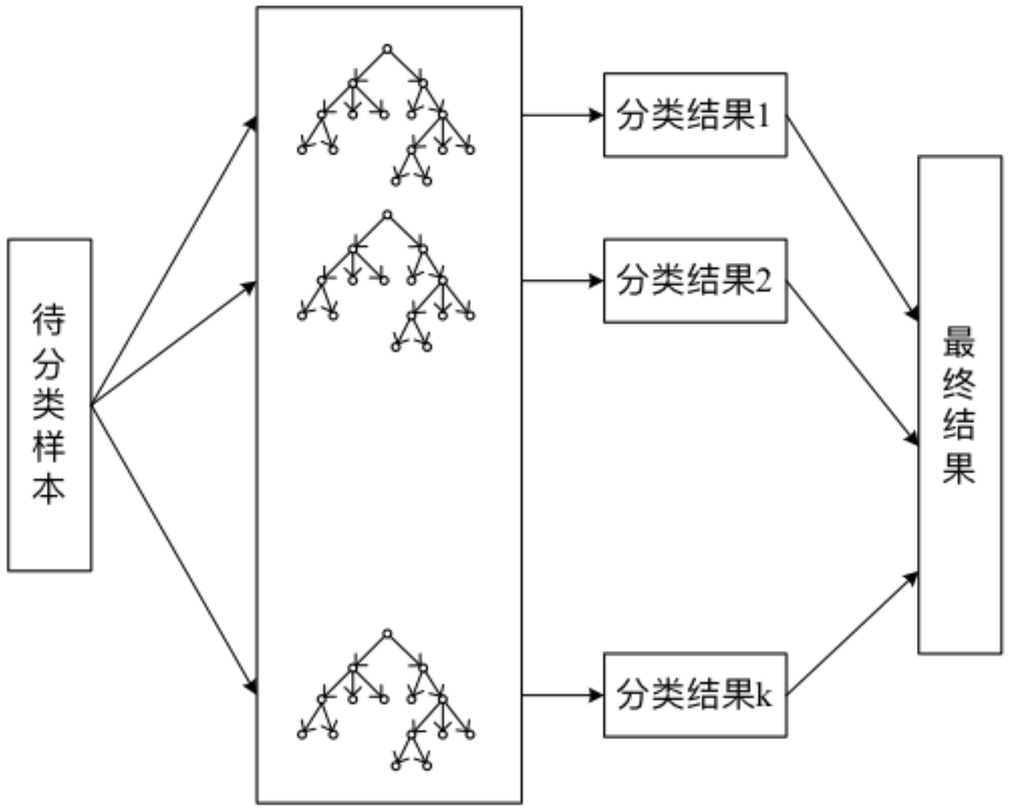
小结一下

- 不管是采用哪一种学习算法，其基本思路都是利用统计测试来计算每一个实例属性单独分类训练样例的能力，分类能力最好的属性将被选作根节点的测试属性。然后为根节点属性的每个可能值产生一个分支，并将训练样例分配到适当的分支之下。通过这样的操作，原始的决策树构造问题就被分解成数个小规模的决策树构造问题。重复前面的过程，用每个分支节点关联的训练样例来选取在该分支节点测试的属性，直至每个分支下的训练样例属于同一类为止
- 性能良好的决策树是一棵与学习样本矛盾较小的决策树，同时具有良好的泛化能力，即好的决策树不仅对学习样本有着很好的分类效果，对于新的数据也有着较低的错误率。

如果性能不够好怎么办?

随机森林

- 随机森林是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树



集成分类器性能与个体关系

	样本1	样本2	样本3
分类器1	F	T	T
分类器2	T	F	T
分类器3	T	T	F
集成结果	T	T	T

a

	样本1	样本2	样本3
分类器1	T	T	F
分类器2	T	T	F
分类器3	T	T	F
集成结果	T	T	F

b

	样本1	样本2	样本3
分类器1	T	F	F
分类器2	F	T	F
分类器3	F	F	T
集成结果	F	F	F

c

分类器服从多数.

100%

66.7%

0%

三个分类器性能差量不同.

三个分类器性能要达到一定程度.

集成分类器性能与个体关系

	样本1	样本2	样本3
分类器1	F	T	T
分类器2	T	F	T
分类器3	T	T	F
集成结果	T	T	T

a

	样本1	样本2	样本3
分类器1	T	T	F
分类器2	T	T	F
分类器3	T	T	F
集成结果	T	T	F

b

	样本1	样本2	样本3
分类器1	T	F	F
分类器2	F	T	F
分类器3	F	F	T
集成结果	F	F	F

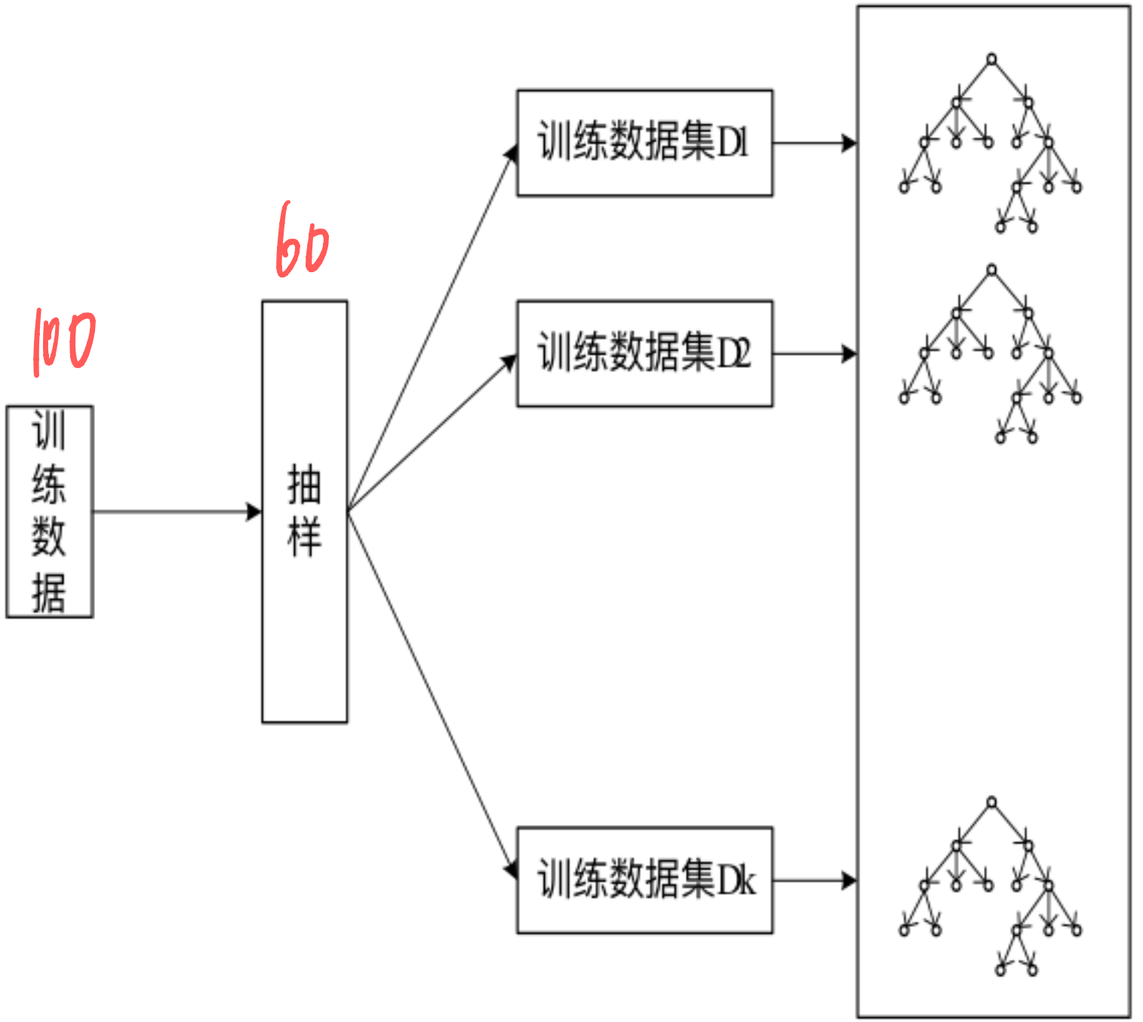
c

结论：要获得好的集成结果，一方面每个个体分类器都要有一定的准确率，同时每个分类器的表现要尽量有差异性

常用的两类方法：bagging和boosting

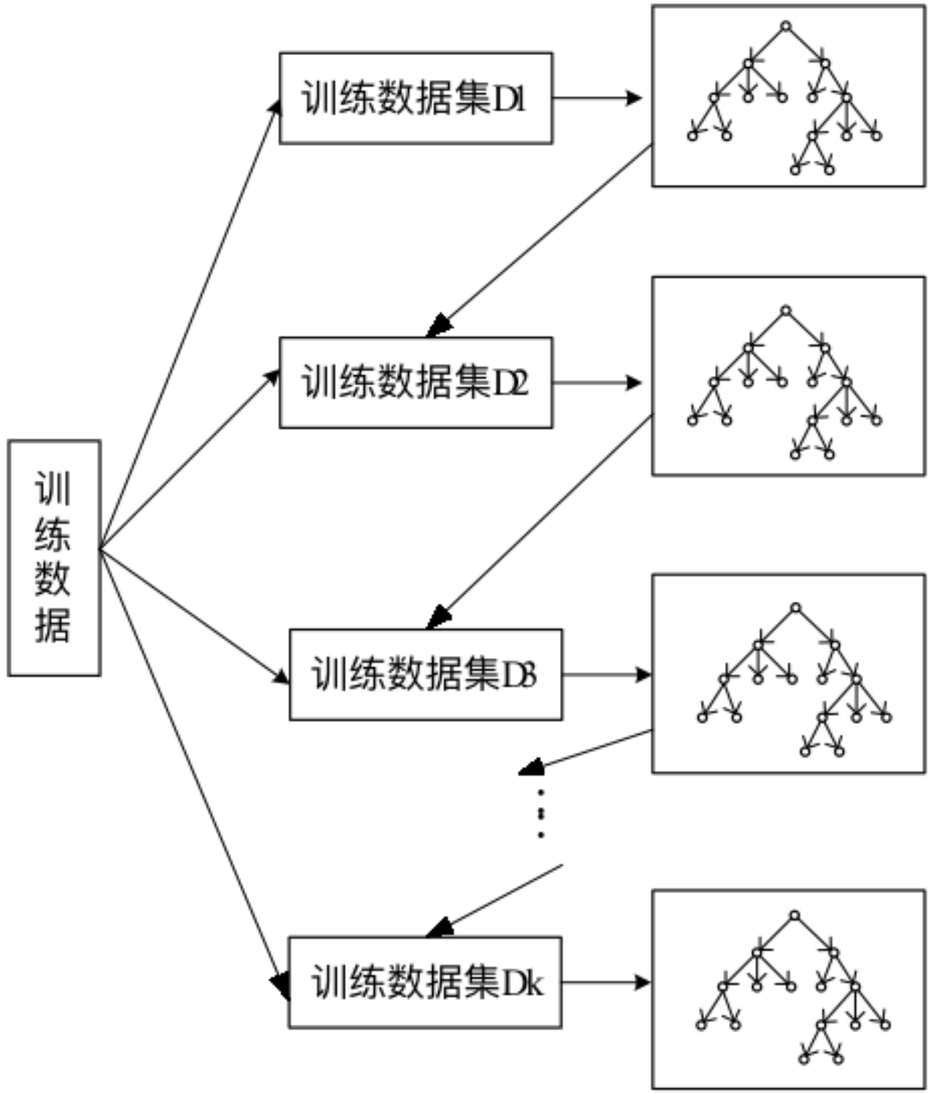
Bagging

- 对于初始给定的训练集，每次从中随机取出n个训练样本组成每轮的训练集，某个初始训练样本在某轮训练集中可以出现多次或根本不出现，训练之后可得到一个决策树序列h_1，……h_n 构成最终的随机森林H，H对分类问题采用投票方式决定最后的结果
- 决策树在进行节点分裂时，不是所有的属性都参与属性指标的计算，而是随机地选择某几个属性参与比较，参与的属性个数就称之为随机特征变量。随机特征变量是为了使每棵决策树之间的相关性减少，同时提升每棵决策树的分类精度，从而提升整个森林的性能。



Boosting

- boosting的训练则是串行进行的，第k个分类器训练时关注对前k-1分类器中错分的样本，即不是随机取，而是加大取这些样本的概率。初始化时对每一个训练样本赋相等的权重1／n，然后用该学算法对训练集训练t轮，每次训练后，对训练失败的训练样本赋以较大的权重，也就是让学习算法在后续的学习中集中对比较难的训练样本进行学习，从而得到一个决策树序列h_1，……h_n构成最终的随机森林H, 其中每个h_i也有一定的权重，预测效果好的决策树权重较大，反之较小，H对分类问题采用有权重的投票方式。



树 → 森林 → 深度随机森林

- 深度随机森林通过对树组成的森林来集成并前后串联起来达到表征学习的效果。它的学习能力可以通过对高维输入数据的多粒度扫描而进行加强，串联的层数也可以通过自适应的决定从而使得模型复杂度不需要成为一个自定义的超参数，而是一个根据数据情况而自动设定的参数。深度随机森林采用了深度网络的串联结构，从前层输入数据，输出结果及源输入作为下层的输入。