1. C4.5

C4.5是对ID3的一个扩张，可用作分类。C4.5使用信息熵的概念。

C4.5是一系列用在机器学习和数据挖掘的分类问题中的算法。它的目标是监督学习：给定一个数据集，其中的每一个元组都能用一组属性值来描述，每一个元组属于一个互斥的类别中的某一类。C4.5的目标是通过学习，找到一个从属性值到类别的映射关系，并且这个映射能用于对新的类别未知的实体进行分类。

决策树的优势在于不需要任何领域知识或参数设置，适合于探测性的知识发现。

1. 相对于ID3的改进：

用信息增益率来选择属性。ID3选择属性用的是子树的信息增益，这里可以用很多方法来定义信息，ID3使用的是熵（entropy， 熵是一种不纯度度量准则），也就是熵的变化值，而C4.5用的是信息增益率。

在决策树构造过程中进行剪枝，因为某些具有很少元素的结点可能会使构造的决策树过适应（Overfitting），如果不考虑这些结点可能会更好。

对非离散数据也能处理。

能够对不完整数据进行处理

1. 优缺点

优点：产生的规则易于理解，准确率较高

缺点：构造树的时候，需要对数据进行多次顺序扫描和排序，算法效率低；适合于能驻留在内存中的数据集，数据集过大时不能运行。

1. 分裂规则

分裂规则决定给定节点上的元组如何分裂。属性选择度量提供了每个属性描述给定训练元组的秩的评定，具有最好度量得分的属性被选作给定元组的分裂属性。流行的属性选择度量有信息增益、增益率和Gini指标

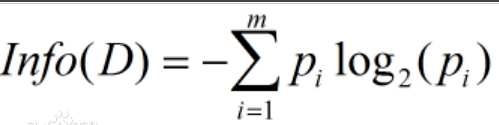
ID3使用信息增益，C4.5使用信息增益率，CART使用Gini指数。

先做一些假设，设D是类标记元组训练集，类标号属性具有m个不同值，m个不同类Ci(i=1,2,…，m)，CiD是D中Ci类的元组的集合，|D|和|CiD|分别是D和CiD中的元组个数。

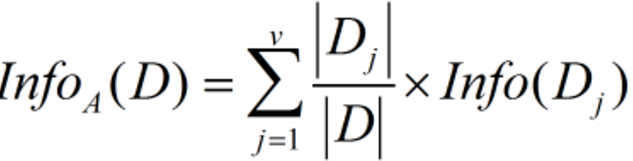
1. 信息增益

信息增益选择具有最高信息增益属性作为节点的分裂属性。该属性使结果划分后的信息增益最大，或者熵降低最多。

训练集D的熵为：



假定按照属性A将D中元组划分成v个不同的类之后，集合的熵为：



那么分类之后的信息增益定义为原来的熵与分类之后的熵之间的差，即：

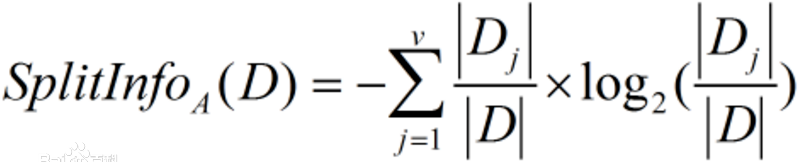


使用信息增益有一个缺点，就是偏向具有大量值得属性，即某个属性可以去的值越多，那么越有可能以这个属性进行分裂。

1. 信息增益率

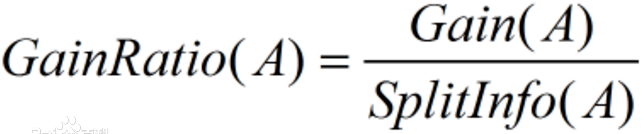
基于此，C4.5采用了信息增益率这样一个概念。信息增益率使用分裂信息值将信息增益规范化。

分裂信息的定义如下：



这个式子和熵的计算类似，分类越多，分裂信息越大。

信息增益率的定义为：

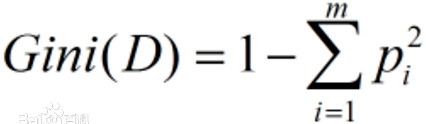


信息增益率可以看作单位分类带来的信息增益大小。

选择具有最大信息增益率的属性作为分裂属性。

1. Gini指数

Gini指标在CART中使用，度量数据划分或者元组集D的不纯度，定义为：



可以看出各分类的数量越平均，Gini指数越大。

1. 剪枝方法

剪枝是避免训练过拟合的方法。通常剪枝是根据度量，剪去最不可靠的分枝。

剪枝一般分为先剪枝和后剪枝：

1. 先剪枝方法通过提前停止树的构造来实现对树的剪枝。一旦停止，这个节点就变成叶子节点，选择该节点子集最频繁的类作为节点的类。先剪枝的方法包括树到达一定深度就停止生长；节点的实例具有相同的特征向量，不必属于同一类，就停止生长；节点的实例数量小于某个阈值的时候，停止生长；节点的分裂对系统性能的增益小于某个阈值，停止生长。先剪枝的缺点是视野受限，导致过早停止决策树的生长。

另一种更常用的方法是后剪枝，树先完全生长，在测试集上通过删除节点的分枝来降低树的过拟合。

1. K-means

K-means是一个聚类算法，把n个对象根据属性分为k个分割，k<n。之所以称为K-均值是因为它可以发现k个不同的簇，且每个簇的中心采用簇中所含值得均值计算而成。

1. K-means实现聚类的思想：
2. 随机选择K个中心值
3. 根据样本数据各点与中心点的距离来进行归簇
4. 通过整个簇的均值，重置每个簇的中心值
5. 重复2、3，直至达到最大的迭代次数（例如：我这里的条件设置为本次与上次的平方误差相等）

K-means算法简单，但是会收敛导局部最小值。因此需要使用后处理来提高聚类性能，一种度量聚类效果指标是SSE误差平方和，由于取了平方，因此距离簇中心越远的点，影响越大。对于SSE较大的簇进行二分，即在该簇上运用k-means算法，k=2。

1. 二分k-means

首先将所有点作为一个簇，然后将该簇一分为二。之后选择一个簇继续进行划分，选择哪一个簇进行划分取决于对其划分是否可以最大程度降低SSE的值。而划分就是上面提到的K-均值的思想了，利用上面的函数k设为2来划分。通过不断重复的操作，直到达到需要的簇数量。

1. svm

支持向量机（Support Vector Machine, SVM）是一类按监督学习（supervised learning）方式对数据进行二元分类的广义线性分类器（generalized linear classifier），其决策边界是对学习样本求解的最大边距超平面（maximum-margin hyperplane）

支持向量机（support vector machines, SVM）是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；SVM还包括核技巧，这使它成为实质上的非线性分类器。SVM的的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小化问题。SVM的的学习算法就是求解凸二次规划的最优化算法。

详细参考<https://blog.csdn.net/qq_35992440/article/details/80987664>

1. Apriori

Apriori是继K-means之后第二个无监督学习算法。这个算法是用先验知识来预测数据的关联规则的。

1. 概念

支持度：数据集中包含该项集的数据所占数据集的比例，度量一个集合在原始数据中出现的频率；

置信度：是针对一条关联规则来定义的，a->b的置信度=支持度{a|b}/支持度{a}，a|b表示ab的并集

1. 关联分析

发现频繁项集（频繁项集是满足最小支持度要求的项集，它给出经常在一起出现的元素项）

发现关联规则（关联规则意味着元素项之间“如果…那么…”的关系）

1. 算法原理

如果某个项集是频繁的，那么它的所有子集也是频繁的

如果某个项集是非频繁的，那么它的所有超集也是非频繁的

基于此，Apriori算法从单元素项集开始，通过组合满足最小支持度的项集来形成更大的集合

1. 缺点

Apriori算法的缺点——每次增加频繁项集大小时(即Ck->Lk时)，算法需要重新扫描整个数据集，当数据集很大时，算法效率很低。

1. 解决方法

FP-Growth

1. 最大期望EM

我们知道最大似然估计的根本目的是根据抽样的到的样本（即数据），反推出最有可能的分布参数（即模型），这是一个非常典型的机器学习的思想。所以在很多领域最大似然估计有着极为广泛的应用。然而，如果已知的数据中含有某些无法观测的隐藏变量时，直接使用最大似然估计是不足以解决问题的。这个时候就要依靠最大化期望（EM）算法了。简单的说，EM算法是在依赖于无法观测的隐藏变量的概率模型中，寻找参数最大似然估计或者最大后验估计的算法。

直观考虑这种隐藏变量的问题，你会发现它很麻烦，因为它使得人们陷入了一种两难的境地：我只有知道了哪些样本是属于同一个类别的，才能根据最大似然函数估计这个类别样本的分布参数；同样，我只有知道了不同类别样本的分布参数，才有可能判断现某个样本到底属于哪个类别的可能性更大。

也就是说，你不确定，我就确定不了；而我不确定，你也确定不了。那怎么办？我们可以先让其中一方随便确定一个值，然后用根据这个值看看对方如何变化，再根据对方的变化调整己方，这样你根据我调整，我再根据你调整，循环往复，最终双方都几乎不变了（也就是收敛了），那就可以确定相关的值了。百度百科上有一个形象的例子，我抄过来，大家可以理解一下：

“比如说食堂的大师傅炒了一份菜，要等分成两份给两个人吃，显然没有必要拿来天平一点的精确的去称分量，最简单的办法是先随意的把菜分到两个碗中，然后观察是否一样多，把比较多的那一份取出一点放到另一个碗中，这个过程一直迭代地执行下去，直到大家看不出两个碗所容纳的菜有什么分量上的不同为止。”

EM的求解思路就是我上面所描述的这样。（1）我们先根据经验为每个类别（即隐藏变量）赋予一个初始分布，这相当于是假定了分布参数。然后根据分布的参数可以求取每个数据元组的隐藏变量的期望（相当于实施了归类操作）；（2）再根据归类结果计算分布参数（向量）的最大似然值，然后根据这个最大似然值在反过来重新计算每个元组的隐藏变量的期望。这样循环往复，最终如果隐藏变量的期望与参数的最大似然值趋于稳定了，EM算法就算是执行完毕了。

综上，我们也就能理解为什么EM算法要叫“最大化期望”算法了，它是由两步组成，第一步是E步，就是求期望；第二步是M步，就是最大化：

E步(Expectation)：根据当前的参数值，计算样本隐藏变量的期望；

M步(Maximum)：根据当前样本的隐藏变量，求解参数的最大似然估计。

1. PageRank
2. AdaBoost
3. Knn
4. Naïve Bayes
5. CART