# Name Convention

1. 变量符号及含义

|  |  |
| --- | --- |
|  | 表示 |
| Dataset数据集 | 是samples（ features every sample）  是分类结果 (K classes in total) |
| Feature/Attribute特征 |  |
| Node |  |
| Directed Edge |  |

1. 常用词汇

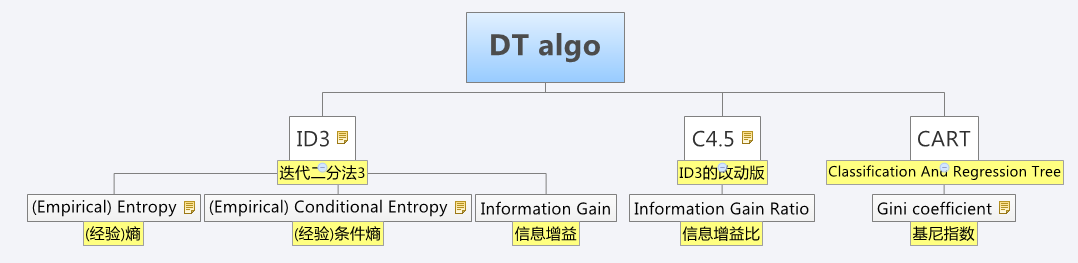
* Decision Tree
* Node
* Internal Node: feature/attribute
* Leaf node: a specific class label
* Directed edge

# 分类树概况

## 理解分类树

1. 逻辑角度上，decision tree是一组if-then规则的集合，if是feature层面的，then是分类结果层面的。
   1. Root node到leaf node的每一条路径都代表一种if规则，且路径上每一个内部节点都代表一种feature的if条件 # 决策树上的路径或者if规则集合满足特性：互斥且完备， 也就是说每个instance分类结果（then）都有且仅被一条规则（if）覆盖
   2. Leaf node代表then，是instance 分类的结果
2. 概率论上，decision tree是指特定条件下，类的条件概率分布。这一条件概率分布是定义在特征空间（feature space）上的一个划分（partition）， 被划分成一个个互不相交的单元（cell）或区域（region），并在每个cell/region定义概率分布就构成了条件概率分布。

## 分类树思维导图



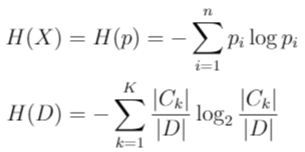
## 分类树步骤

1. Feature selection
2. Decision Tree Generation/Building/Construction
3. Decision Tree Pruning

# 分类树算法

## ID3

### (Empirical) Entropy (经验)熵



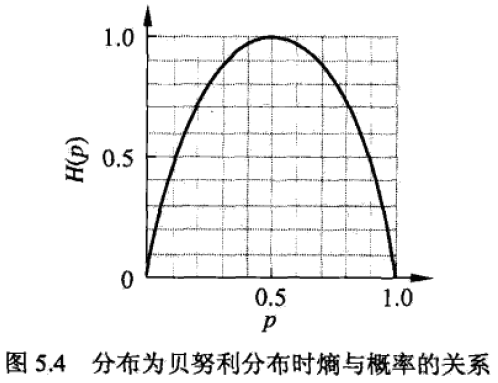
* 熵只与X的分布有关，与X取值无关
* empirical 一次表示 熵中的概率估计是直接用数据分布得到的，而不是原始概率分布函数，有一定的经验性

为什么熵能有效地判断一个特征的分类能力呢？用伯努力0/1二分类(抛硬币统计正反面)为例，



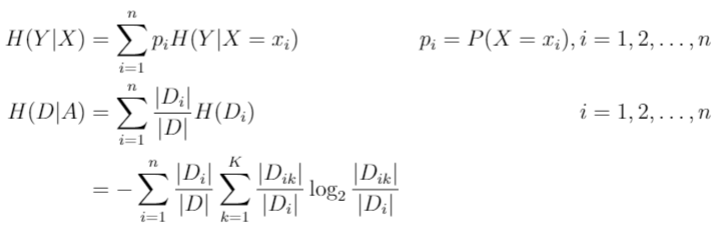
这时随p变化的曲线如下图：

* 当p=0.5, 熵最大，和实际情况一致，0，1完全随机分布，很难预测，预测准确全凭猜测就是一半一半。
* 当p<0.5时，熵变小，结果偏向0的概率大一些了
* 当p>0.5时，熵变小，结果偏向1的概率大一些了



### (Empirical) Conditional Entropy (经验)条件熵

条件熵$H(Y|X)$表示在已知随机变量$X$的条件下随机变量$Y$的不确定性



### Information Gain信息增益



### Pseudocode

ID3是一种贪婪的 启发式算法，其执行对局部最优熵值的局部优先搜索。每次都选择能使得当前root node 分类后熵最小的feature & partition，也就是信息增益最的feature & partition。

Samples/Dataset

Target Label/Class

Feature/Attribute Set

Threshold

tree

ID3 (Samples/Dataset , Features , Threshold )

(0) Create output tree with an empty root node

(1) If all , then is single-node tree Root, return with label = .

(2) If number of predicting Features is empty(), then is single-node tree Root, return T

with label = most common value of the target label (label = ) in the samples.

(3) Otherwise Begin

Compute the information gain for all features in current Dataset

← The Attribute that best classifies examples(minimum Entropy / maximum info gain).

(4) If , return with label =

(5) Else

For each value of , Add a new tree branch below , get Dataset slice by

(6) subtree = ID3(Dataset, Features , Threshold )

End

Return Root tree

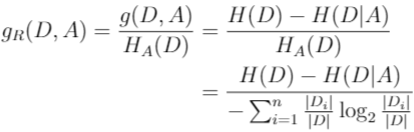
### Reference

1. <https://en.wikipedia.org/wiki/ID3_algorithm>
2. 统计学习方法

## C4.5

### Information Gain Ratio (normalized information gain)信息增益比(归一化信息增益)

之前“信息增益”概念，会更倾向于分类数多的特征，增加比值来归一化这个数。



Where *HA(D)* is also called as “*split information*”, is the entropy of splitting dataset *D* over attribute *A*.

### Pseudocode of generating tree(discrete) C4.5树的生成伪码

C4.5 基本上只是对ID3的一种改进算法，criterion由信息增益改变为信息增益比

Samples/Dataset

Target Label/Class ,

Threshold

tree

C4.5 (Samples/Dataset , Features , Threshold )

(0) Create output tree with an empty root node

(1) If all , then is single-node tree Root, return with label = .

(2) If number of predicting Features is empty(), then is single-node tree Root, return T

with label = most common value of the target label (label = ) in the samples.

(3) Otherwise Begin

Compute the information gain for all features in current Dataset

← The Attribute that best classifies examples(maximum info gain ratio ).

(4) If , return with label =

(5) Else

For each value of , Add a new tree branch below , get Dataset slice by

(6) subtree = C4.5(Dataset, Features , Threshold )

End

Return Root tree

### Generating tree(based on continuous attributes) 基于连续变量的C4.5树的生成

Refer to *Improved Use of Continuous Attributes in C4.5* [4], there are two changes from the C4.5 based on the discrete attributes.

1. Use the Minimum Description Length (MDL) principle, reduce the continuous attributes’ info gain from into , because:
   1. For discrete attributes, Ai=?, A can just be specified by traverse the Ai, j involved
   2. For continuous attributes, Ai<=t, there is a threshold t to be determined, and threshold t has n-1 possible values if Ai has n unique values.
2. When to select threshold t, we use the *info gain* as criterion rather than *info gain ratio* , but after the threshold t is determined, we still use info gain ratio as the *splitting criterion* to compare with other attributes to decide the best one or . Furtherly,
   1. Because if still use *info gain ratio* to select threshold, then both Denominator (*info gain*) and Numerator(*split information*) all will vary with thresh t, this seems to be an unnecessary complication.

### Pseudocode of generating tree(continuous) C4.5树的生成伪码

Refer to C4.5[3], [4], if there are N distinct values of attribute A in the set of cases D, there are N -1 thresholds that could be used for a test on A. Each threshold gives unique subsets D1 and D2 and so the value of the splitting criterion is a function of the threshold.

C4.5 (Samples/Dataset , Features , Threshold )

Input: Train Dataset , Feature/Attribute Set , Threshold

Output: Tree

(0) Create output tree with an empty root node

(1) If all , then is single-node tree Root, return with label = .

(2) If number of predicting Features is empty(), then is single-node tree Root, return T

with label = most common value of the target label (label = ) in the samples.

(3) Otherwise Begin

Compute the information gain for all features in current Dataset

Sort Dataset in feature (continuous) in ascending order

Iteratively pick the threshold as (ai,j+ai,j+1)/2, and compute the info gain with penalty

← The Attribute that best classifies examples(maximum info gain ratio ).

(4) If , return with label =

(5) Else

For each value of , Add a new tree branch below , get Dataset slice by

(6) subtree = C4.5(Dataset, Features , Threshold )

End

Return Root tree

### C4.5’s Improvements from ID.3 algorithm

C4.5 made a number of improvements to ID3. Some of these are:

* Handling both continuous and discrete attributes - In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it.[[5]](https://en.wikipedia.org/wiki/C4.5_algorithm#cite_note-5)
* Handling training data with missing attribute values - C4.5 allows attribute values to be marked as ? for missing. Missing attribute values are simply not used in gain and entropy calculations.
* Handling attributes with differing costs.
* Pruning trees after creation - C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes

Another comparison from sklearn[5]:

C4.5是ID3的后继者， ID3不能处理连续特征，C4.5通过一些离散的分隔点将连续特征转换为有几个特定离散分段区间的离散特征，消除了这个限制。C4.5将训练的树（即ID3算法的输出）转换为if-then规则集。然后评估每个规则的这些准确性以确定它们应该应用的顺序。如果规则的准确性在没有它的情况下还能得到改善，则通过删除此规则来完成修剪。

### Reference

1. https://en.wikipedia.org/wiki/C4.5\_algorithm
2. 统计学习方法
3. <https://stackoverflow.com/questions/15629398/how-does-the-c4-5-algorithm-handle-continuous-data>
4. Improved Use of Continuous Attributes in C4.5, <https://arxiv.org/pdf/cs/9603103>
5. https://scikit-learn.org/stable/modules/tree.html#tree-algorithms

## CART

CART(Classification And Regression Tree), 是一种广泛应用的决策树学习方法。

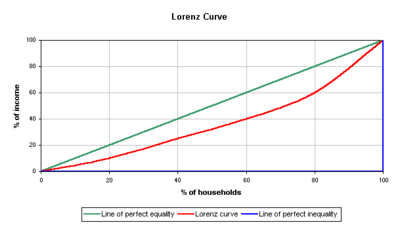
[数据挖掘](https://en.wikipedia.org/wiki/Data_mining)中使用的决策树有两种主要类型：

* [分类树](https://en.wikipedia.org/wiki/Classification_tree)分析是指预测结果是数据所属的类（离散）。
* 回归树分析是指预测结果可以被视为实数（例如房屋价格或患者在医院中停留的时间）。

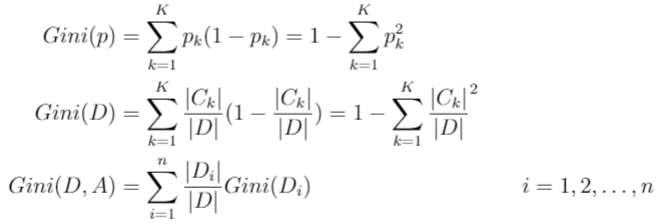
术语分类和回归树（CART）分析是用于指代上述两种程序的[总称](https://en.wikipedia.org/wiki/Umbrella_term)，由[Breiman](https://en.wikipedia.org/wiki/Leo_Breiman)等人首先提出

### Gini coefficient

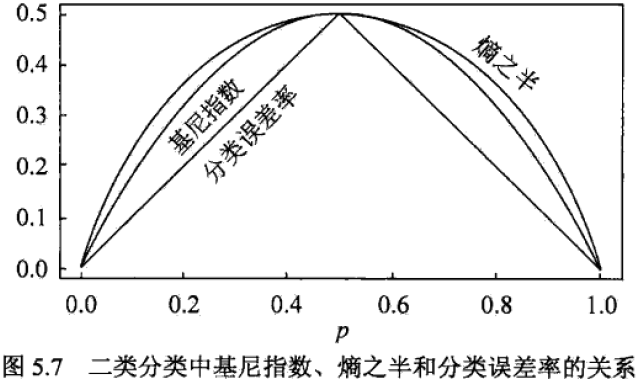
Wiki也叫 Gini impurity(基尼混乱度)，以免与经济学上的Gini coefficient混淆起来。



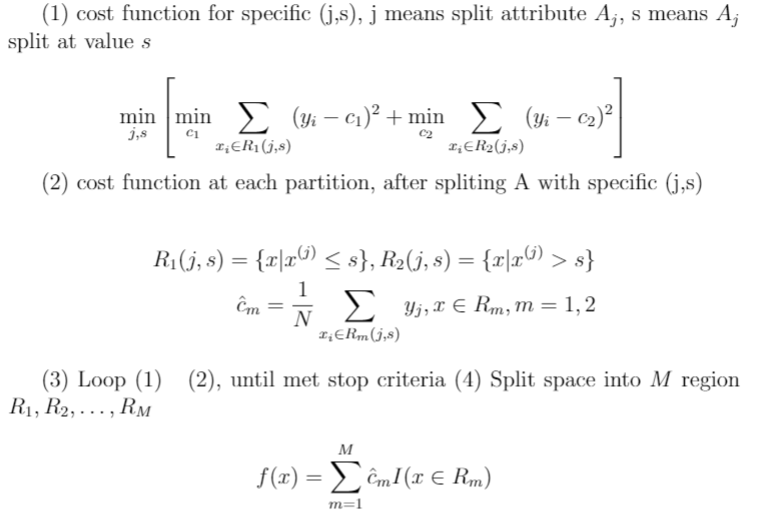
经济学中的表征收入均等程度的基尼系数如上图，图中横轴为人口累计百分比，纵轴为该部分人的收入占人口总收入的百分比，三条色线各表示不同情况下后者和前者的比例。绿线表示人口收入分配处于绝对平均状态，蓝线表示绝对不平均（即所有收入由一人独占），红线则表示实际情况。红线和绿线之间的面积越小，则收入分配越平等。



Gini coefficient, similar with entropy, also describe the uncertainty of a dataset. The larger Gini coefficient, the larger uncertainty.



### Pseudocode of least square regression tree



### Pseudocode of generating CART CART的生成伪码

CART和C4.5的伪代码流程基本一样，借鉴wiki上一段来做总结：

[Decision trees](https://en.wikipedia.org/wiki/Decision_trees) are formed by a collection of rules based on variables in the modeling data set:

1. Rules based on variables' values are selected to get the best split to differentiate observations based on the dependent variable
2. Once a rule is selected and splits a node into two, the same process is applied to each "child" node (i.e. it is a recursive procedure)
3. Splitting stops when CART detects no further gain can be made, or some pre-set stopping rules are met. (Alternatively, the data are split as much as possible and then the tree is later [pruned](https://en.wikipedia.org/wiki/Pruning_(decision_trees)).)

This process of top-down induction of decision trees (TDIDT) [[2]](https://en.wikipedia.org/wiki/Decision_tree_learning#cite_note-Quinlan86-2) is an example of a [greedy algorithm](https://en.wikipedia.org/wiki/Greedy_algorithm), and it is by far the most common strategy for learning decision trees from data

### Compare CART with C4.5 and sklearn’s improvement

[CART](https://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees_.28CART.29) (Classification and Regression Trees) is very similar to C4.5, but it differs in that it supports numerical target variables (regression) and does not compute rule sets. CART constructs binary trees using the feature and threshold that yield the largest information gain at each node.

scikit-learn uses an optimised version of the CART algorithm; however, scikit-learn implementation does not support categorical variables for now.

CART（分类和回归树）与C4.5非常相似，但它的不同之处在于它支持数值目标变量（回归）并且不计算规则集。CART使用在每个节点处产生最大信息增益的特征和阈值来构造二叉树。

scikit-learn使用CART算法的优化版本; 但是，scikit-learn实现目前不支持分类变量。

Scikit-learn offers a more efficient implementation for the construction of decision trees. A naive implementation (as above) would recompute the class label histograms (for classification) or the means (for regression) at for each new split point along a given feature. Presorting the feature over all relevant samples, and retaining a running label count, will reduce the complexity at each node

Scikit-learn为决策树的构建提供了更有效的实现。一个简单的实现将重新计算在给定feature的每个partition点上的class label histogram（用于分类）或 the means（用于回归）。在所有相关样本上预先对feature进行排序，and retaining a running label count，将降低每个节点的复杂性.

### Reference

1. <https://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees_(CART)>
2. https://en.wikipedia.org/wiki/Decision\_tree\_learning
3. 统计学习方法

## Pruning 剪枝

由于decision tree生成过程greedy recursive algorithm的本质，the decision tree might cause an overfitting effect, because the resulting tree adapts itself to the *noise* in the sample set rather than the true underlying structure, and may fail to classify the unseen instances.

So we need Pruning, to prune the unnecessary nodes of the resulting decision tree. This is an model selection problem, trade-off between model complexity(*the tree size*) and accuracy (the classify error).

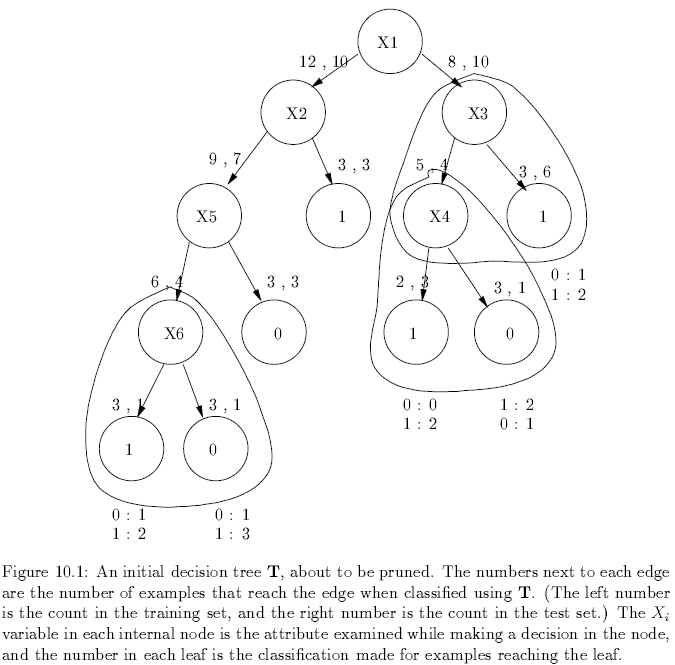
### Reduce Error Pruning

Use cross-validation method, training set to build the decision tree **T**, test set to prune **T** and get pruned tree **P**.

Why use test set for pruning, because the true error is approximated by using the test set.

If number of classification true error over the test set of *v* as a pruned leaf is significantly smaller than the true error of subtree rooted at *v*, then we can choose to prune *v*.

This is a method from bottom to up method, Left hand order or Right hand order visiting a Tree, as Figure 10.1.



### Structural Risk Minimization

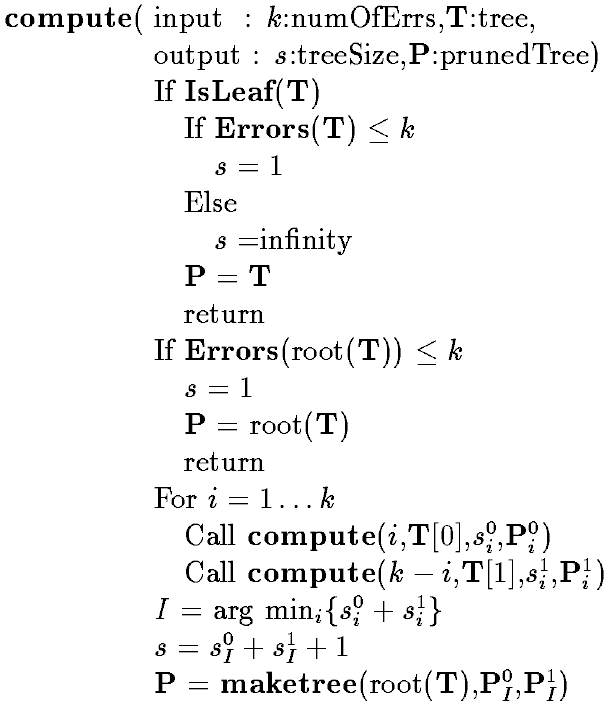
Based on the struct of the tree, we can also use recursion to prune.

#### Pseudocode

tree

root() the root node of tree

and the left and right child subtrees of respectively

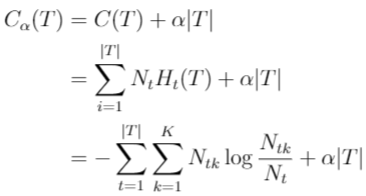


For *I=arg mini{si0 + si1}* , we can also use Dynamic Programing method to memorize the smallest size of left and right subtree at each node, so the parent node just need to check the memory table.

### Cost complexity pruning

#### DT cost function 决策树整体代价函数

ID3为例，树 的叶结点个数为，是树的叶结点，该结点有 个样本点，其中 类的样本点有个，为叶结点上的经验熵，为参数，决策树学习的损失函数可以定义为:



其中表示模型对训练数据的误差，表示模型复杂度，参数控制两者之间的影响。

带入到Reduce Error Pruning中, 将cost function由Tree Error变换为Overall cost 就可以得到一种新的pruning 方法。

### Reference

1. [Introduction to Decision tree pruning](http://www.math.tau.ac.il/~mansour/ml-course/scribe11.ps)
2. 统计学习方法

# Practice View

## DT pros & cons

Pros：

1. White-box model, easy to visualize, easy to interpret, fast to apply model

Cons：

1. DT construction algo use greedy, easy for overfitting model, not has good generalization capability # for sklearn, we can just use maxdepth & min\_samples\_leaf to regularize ⇨ beside sklearn, we can perform tree pruning
2. not robust to noise, unstable to small variation, sensitive to data, and order; Greedy method to generate tree is NP-complete question, only can get local minimum, cannot guarantee to return global minima ⇨ both need to be mitigated by assemble algorithm, like Random Forest, multiple DT with random subset of data (with replacement)
3. DT results in biased tree if some classes dominate, biased tree means only good cover main feature, but regardless the secondary feature in training set, this may also lead to bad generalization capability ⇨ recommended to balance the weight of dataset before training

## Practice Tips

1. parameter to enhance generalization capability:
   1. maxdepth=3, for start
   2. min\_samples\_split=5 for start, int/fraction, min #samples @ internal node,
   3. min\_samples\_leaf =2 for start, int/fraction, min #samples @ leaf node, (default =1 for few classes)
   4. min\_weight\_fraction\_leaf, consider weight fraction @leaf
   5. min\_impurity\_decrease, Gini impurity
   6. Pruning
2. Modeling and Visualization, export to grahviz
3. Avoid biased tree only for dominant class
   1. Either equally select sample from the two classes
   2. Or use sample\_weight / class\_weight to balance
4. #samples >> #features

# Glossary

Feature selection

Tree Building

Tree Pruning

DT Algorithms(split criterion)

ID3

(Empirical) Entropy

(Empirical) Conditional Entropy

Information Gain

C4.5

Information Gain Ratio

CART

Gini coefficient