逻辑回归正则化

1. 三种优化问题

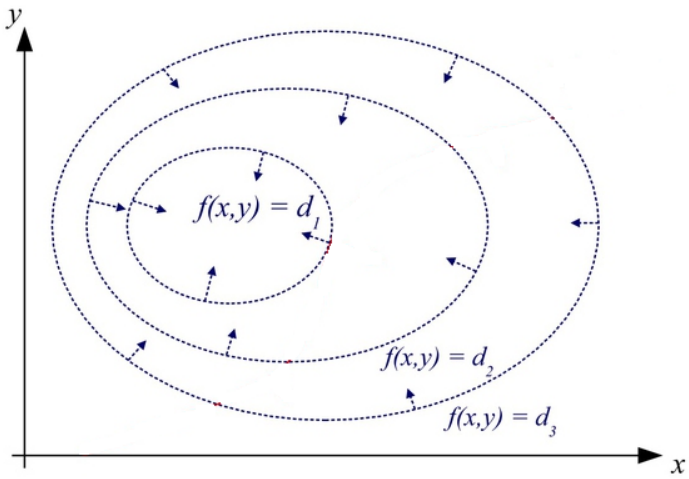
通常我们求解的最优化问题可以分为以下三类：

- 无约束的优化问题：

  $min\ f(X)$

  这是最简单的情况，解决方法通常是 $f(X)$ 的导数 $f'(X)$ ，然后令 $f'(X)=0$ ，可以求得极值点，再将这些极值点带入原函数验证即可；如果是凸函数，极值点即为最优解。

  其几何含义是：



- 有等式约束的优化问题

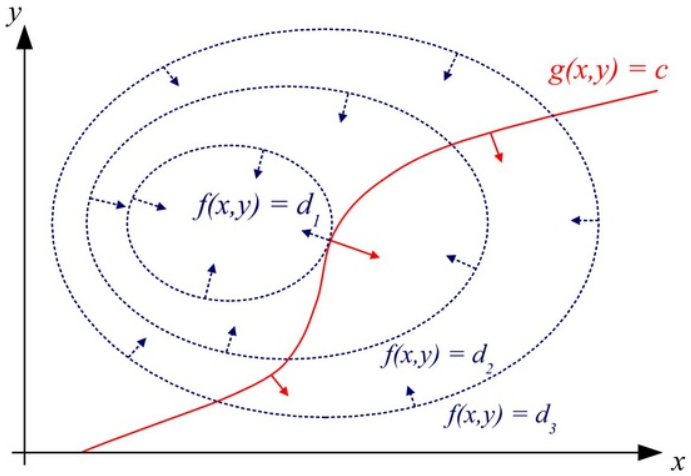
  $min\ f(X)$

  $s.t.\ g(X)=0$

  注：$s.t.$ 表示约束条件。

常常使用的方法就是拉格朗日乘子法（Lagrange Multiplier），即写成 $L(\lambda, X) = f(X) + \lambda g(X)$，称为拉格朗日函数，系数 λ 称为拉格朗日乘子。通过 $L(\lambda, X)$对各个变量 (X, λ) 求偏导，令其为零，可以求得极值点集合，然后验证求得最优值。

其几何含义是：



- 等式和不等式约束的优化问题

  $min\ f(X)$

  $s.t.\ g(X) = 0$

  $h(X) \leq 0$

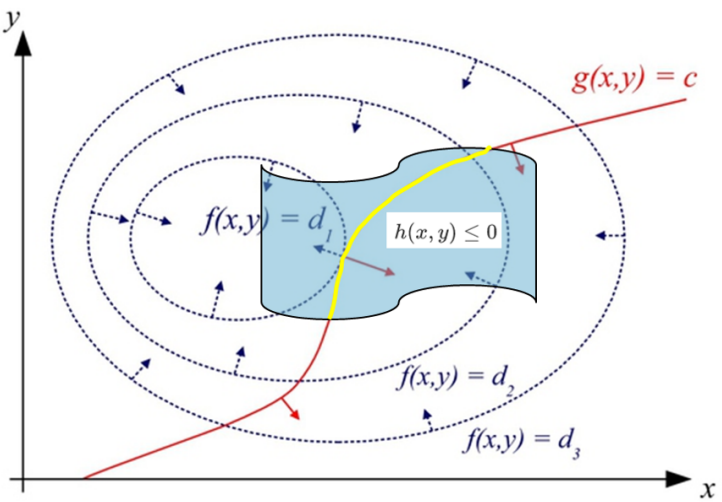
注：$s.t.$ 表示约束条件。

常常使用的方法就是 KKT 条件。$L(\mu, \lambda, X) = f(X) + \lambda g(X) + \mu h(X)$，利用最优值的必要条件 (KKT条件)：

- $L(\mu, \lambda, X)$ 对变量的导数为 0
- $g(X) = 0$
- $\mu h(X) = 0$

求取这些等式之后就能得到候选最优值。其中第三个式子非常有趣，因为 $h(X) \leq 0$，如果要满足这个等式，必须 $\mu = 0$ 或者 $h(X) = 0$。

那么KTT的几何含义是什么呢？



2. 逻辑回归正则化的几何含义

L1 正则化通常称为 Lasso 正则化：

$$J(\theta) = -\sum_{i=1}^{m}(y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))) + \frac{\lambda}{m}\sum_{j=1}^{n}|\theta_j|$$

L2 正则化通常称为 Ridge 正则化：

$$J(\theta) = -\sum_{i=1}^{m}(y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))) + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

我们可以写成统一的形式：

$$J(\theta, \mu) = f(\theta) + \mu h(\theta)$$

可以还原为：

$$min\ f(\theta)$$

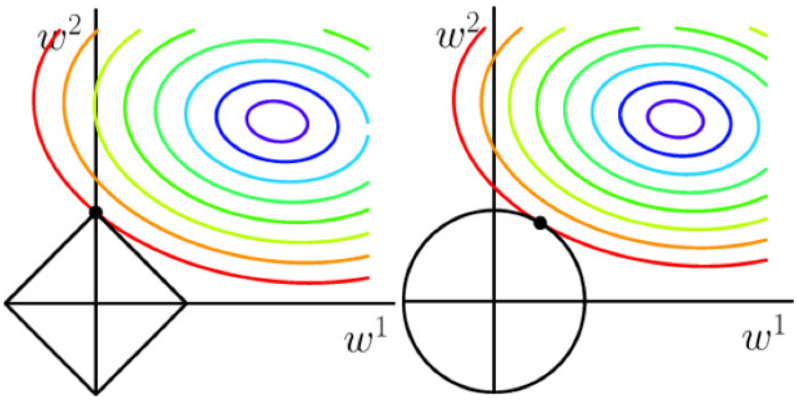
$$s.t.\ h(\theta) = 0$$

那么他们的几何含义是：

对于 L1 正则化 (Lasso 正则化)：$h(\theta) = \sum_{j=1}^{n}|\theta_j|$

对于 L2 正则化 (Ridge 正则化)：$h(\theta) = \sum_{j=1}^{n}\theta_j^2$

Q0：以下哪个图形是 L1 正则化，哪个图形是 L2 正则化？



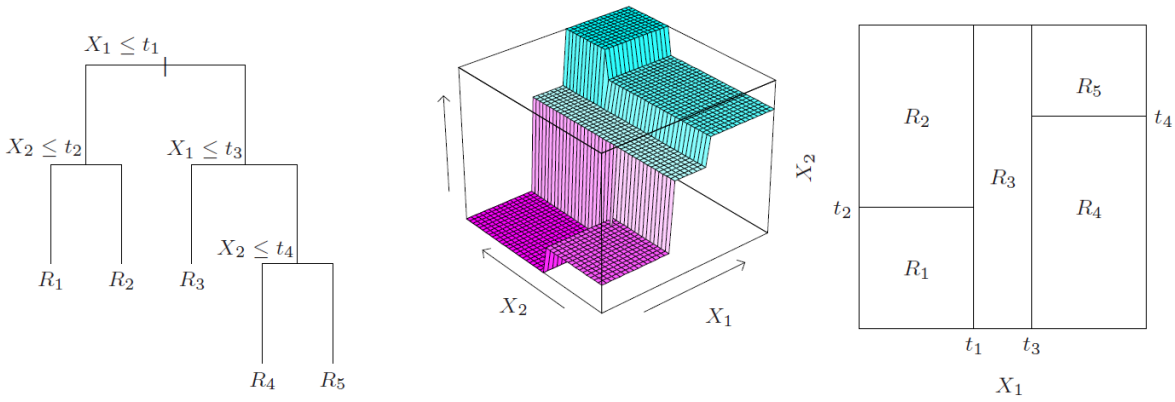
决策树

简介

决策树是一种基本的分类和回归方法。

物理含义：表示定义在特征空间上的条件概率分布，也就是把 $P(y)$ 转化为 $P(y|X)$ 。同时也可以理解为 if-then 规则的集合。

几何含义：“分而治之”：把特征空间划分为一系列的矩形区域，然后再每一个区域拟合一个简单的模型。



类比：中国的省市县的划分，公司部门科室的划分。

步骤：特征选择，决策树的生成和决策树的修剪。特征选择在于选取对训练数据具有分类能力的特征，这样可以提高决策树的学习效率。决策树生成对应于提高决策树在训练集上的性能，减小模型的bias。决策树的修剪对应于提高模型在验证集上性能，减小模型的 variance，提高模型的泛化能力。

输入的是 m 个样本，样本输出集合为 D，每个样本有 n 个离散特征，特征集合即为 A，输出为决策树 T，特征选择标准为 Criterion。

1) 初始化 Criterion 的阈值 ϵ。

2) 判断样本是否为同一类输出 D_i，如果是则返回单节点树T。标记类别为 D_i

3) 判断特征是否为空，如果是则返回单节点树 T，标记类别为样本中输出类别 D 实例数最多的类别。

4) 计算 A 中的各个特征（一共 n 个）对输出 D 的 Criterion ，选择 Criterion 值最大的特征 A_g

5) 如果 A_g 的 Criterion 值小于阈值 ϵ，则返回单节点树 T，标记类别为样本中输出类别 D 实例数最多的类别。

6) 否则，按特征 A_g 的不同取值 A_{gi} 将对应的样本输出 D 分成不同的类别 D_i。每个类别产生一个子节点。对应特征值为 A_{gi}。返回增加了节点的数 T。

7) 对于所有的子节点，令 $D = D_i$，$A = A - A_g$ 递归调用 2-6 步，得到子树 T_i 并返回。

省市县镇村市按照空间地理位置进行划分，部门科室是按照职能进行划分，那么按照什么对空间进行划分呢？Quinlan 在 1986 年提出的 ID3 算法和1993年提出的 C4.5 算法，以及 Breiman 等人在1984年提出的 CART 算法。下面我们对这些算法进行一一介绍。

1. ID3

特征选择：信息增益

首先，我们需要熟悉信息论中熵的概念。熵度量了事物的不确定性，越不确定的事物，它的熵就越大。随机变量 X 熵的表达式如下：

$$Ent(D) = -\sum_{i=1}^{n} p_i \ log \ p_i$$

其中 $p_i = P(X = x_i), i = 1, 2, \cdots, n$，log 以 2 或者 e 为底。若 $p_i = 0$ ，则定义 $0 \ log \ 0 = 0$ 。

Q1：与逻辑回归的代价函数之间有什么关系？

Q2：计算以下3种情况的熵，从中得出什么结论？

- X 有 2 个可能的取值，而这两个取值的概率各为 1/2 时 。

- X 有 2 个可能的取值，一个概率 1/3，一个概率 2/3。
- X 有 2 个可能的取值，一个值概率为 0，另一个值概率为 1。

  $Ent(D) = -(1log1 + 0log0) = 0$

多个变量的联合熵，这里给出两个变量 X 和 Y 的联合熵表达式：

$$Ent(X,Y) = -\sum_{i=1}^{n} p(x_i,y_i)logp(x_i,y_i)$$

有了联合熵，我们可以得到条件熵的表达式 $H(Y|X)$，条件熵类似于条件概率, 它度量了我们已知X的情况下Y的不确定性。表达式如下：

$$Ent(D|X) = \sum_{i=1}^{n} p(x_i)Ent(Y|X=x_i) = \sum_{i=1}^{V} \frac{|D^i|}{D} Ent(D^i)$$

Q3：我们刚才提到 Ent(D)度量了 Y 的不确定性，条件熵 Ent(D|X) 度量了我们在知道X以后Y的不确定性，那么 Ent(D)-Ent(D|X) 表示什么呢？

$$Gain(D,X) = Ent(D) - Ent(D|X) = Ent(D) - \sum_{i=1}^{V} \frac{|D^i|}{D} Ent(D^i)$$

ID3 算法就是用信息增益来判断当前节点应该用什么特征来构建决策树。不同的特征具有的信息增益不同，信息增益大表示该特征具有更强的分类能力。选择信息增益最大的特征来建立决策树的当前节点。

Q4：计算以下几个特征的信息增益，并按照信息增益选择特征，设计出最合理的决策树。(周志华 《机器学习》 P76)

表 4.1 西瓜数据集 2.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

ID3算法不足：

ID3算法虽然提出了新思路，但是还是有很多值得改进的地方。

1. ID3没有考虑连续变量，比如长度，密度都是连续值，无法在 ID3 运用。这大大限制了 ID3 的用途。
2. ID3采用信息增益大的特征优先建立决策树的节点。在相同条件下，取值比较多的特征比取值少的特征信息增益大。
3. ID3算法对于缺失值的情况没有做考虑
4. 没有考虑过拟合的问题

Q5：能否举例说明第二个问题吗？第二个问题和第三个问题有什么关系？

昆兰基于上述不足，对ID3算法做了改进，这就是 C4.5 算法。下面我们介绍一下 C4.5 算法。

2. C4.5

C4.5算法的改进

昆兰在 C4.5 算法中就以上四个问题做了改进。

1. 对于第一个问题，不能处理连续特征， C4.5 的思路是将连续的特征离散化。比如 m 个样本的连续特征 A 有 m 个:
   - 从小到大排列为 $a_1, a_2, \ldots, a_m$
   - 取相邻两样本值的平均数，一共取得 m-1 个划分点，其中第 i 个划分点 T_i 表示为：$T_i = \frac{a_i + a_{i+1}}{2}$。
   - 对于这 m-1 个点，分别计算以该点作为二元分类点时的信息增益。
   - 选择信息增益最大的点作为该连续特征的二元离散分类点。比如取到的增益最大的点为 a_t，则小于 a_t 的值为类别 1，大于 a_t 的值为类别 2，这样我们就做到了连续特征的离散化。
   - 计算离散之后的特征的信息增益比。
   - 注意，则该属性后面还可以参与子节点的产生选择过程。

Q6：计算以下密度和含糖率的信息增益比，并将密度和含糖率添加进生成的决策树中。(周志华《机器学习》P84)

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

2. 对于第二个问题，我们引入了信息增益比的，它是信息增益和特征熵的比值，其实质是信息增益的标准化 (无量纲化)。表达式如下：

$$Gain\_ratio(D,A)=\frac{Gain(D,A)}{IV(A)}$$

其中 D 为样本特征输出的集合，A 为样本特征，对于特征熵 $IV(A)$，表达式如下：

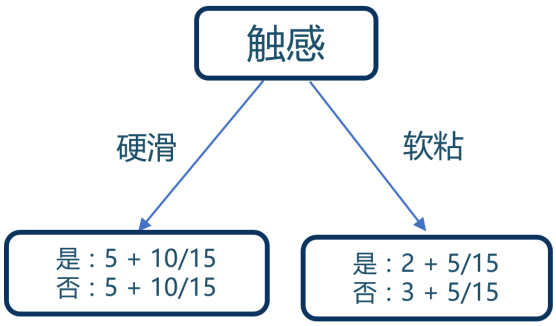
$$IV(A)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}log_2\frac{|D_i|}{|D|}$$

其中 n 为特征 A 的类别数， D_i 为特征 A 的第 i 个取值对应的样本个数。D 为样本个数。特征数越多的特征 对应的特征熵 $IV(A)$ 越大，它作为分母，可以校正信息增益容易偏向于取值较多的特征的问题。

3. 对于第三个缺失值处理的问题，主要需要解决的是两个问题
    - 一是在样本某些特征缺失的情况下选择划分的属性，

      C4.5 的思路是将数据分成两部分，一部分是非缺失值的数据 D1，另一部分是缺失值的数据 D2。然后对于 D1 计算信息增益比，最后乘上一个权重，这个权重是非缺失值的样本占总样本数的比例。
    - 二是选定了划分属性，对于在该属性上缺失特征的样本的处理。

      将缺失特征的样本按各个子节点样本的数量比例来分配同时划分入所有的子节点。



Q7：计算以下几个特征的信息增益，并按照信息增益选择特征，设计出最合理的决策树。(周志华《机器学习》P76)

表 4.4 西瓜数据集 2.0α

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	–	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	–	是
3	乌黑	蜷缩	–	清晰	凹陷	硬滑	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	–	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	–	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
8	乌黑	稍蜷	浊响	–	稍凹	硬滑	是
9	乌黑	–	沉闷	稍糊	稍凹	硬滑	否
10	青绿	硬挺	清脆	–	平坦	软粘	否
11	浅白	硬挺	清脆	模糊	平坦	–	否
12	浅白	蜷缩	–	模糊	平坦	软粘	否
13	–	稍蜷	浊响	稍糊	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	–	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	–	沉闷	稍糊	稍凹	硬滑	否

4. C4.5 引入了正则化系数进行初步的剪枝。

C4.5算法的不足与思考

- 由于决策树算法非常容易过拟合，因此对于生成的决策树必须要进行剪枝。剪枝的算法有非常多，C4.5 的剪枝方法有优化的空间。思路主要是两种，一种是预剪枝，即在生成决策树的时候就决定是否剪枝。另一个是后剪枝，即先生成决策树，再通过交叉验证来剪枝。
- C4.5 生成的是多叉树，即一个父节点可以有多个节点。很多时候，在计算机中二叉树模型会比多叉树运算效率高。如果采用二叉树，可以提高效率（多叉树会导致我们的数据更快的分裂成多个碎片，从而能导致下一层每个节点的数据量不足，当然，多叉树也可以通过多个二叉树实现）。
- C4.5 只能用于分类，如果能将决策树用于回归的话可以扩大它的使用范围。
- C4.5 由于使用了熵模型，里面有大量的耗时的对数运算，如果是连续值还有大量的排序运算。

3. CART

特征选择：Gini 指数

无论是 ID3 还是 C4.5 都会涉及大量的对数运算。CART 分类树算法使用基尼系数，基尼系数代表了模型的不纯度，基尼系数越小，则不纯度越低，特征越好。

在分类问题中，假设有 K 个类别，第 k 个类别的概率为 p_k, 则基尼系数的表达式为：

$$Gini(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$$

如果是二类分类问题，计算就更加简单了，如果属于第一个样本输出的概率是 p，则基尼系数的表达式为：

$$Gini(p) = 2p(1-p)$$

对于个给定的样本D，假设有 K 个类别, 第 k 个类别的数量为 C_k,则样本 D 的基尼系数表达式为：

$$Gini(D) = 1 - \sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2$$

特别的，对于样本 D，如果根据特征 A 的某个值 a，把 D 分成 D1 和 D2 两部分，则在特征 A 的条件下，D 的基尼系数表达式为：

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

Q8：计算以下特征的Gini系数，生成最优决策树。（周志华《机器学习》）

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

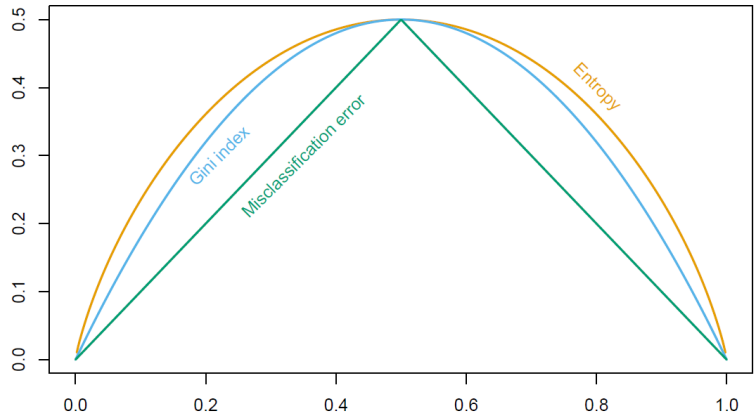
下面我们对比一下不同的非纯度的度量方式：

误分类误差：$\frac{1}{N_m}\sum_{i\in R_m} Count(y_i \neq k(m)) = 1 - p_k$

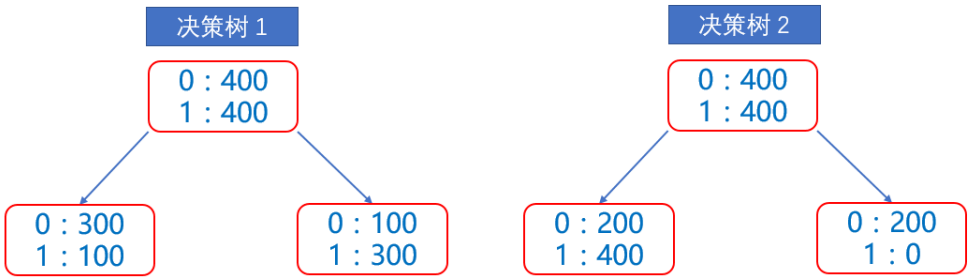
Gini 系数：$Gini(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$

互熵：$-\sum_{k=1}^{K} p_k log p_k$

- 对于二分类，如果 P 是 label 为 0 的类比例，这三个度量分别为
  $1 - \max(p, 1-p)$，$2p(1-p)$，$-p\,log(p) - (1-p)log(1-p)$
  这三种度量相似，但是互熵和 Gini 系数是可导的，更适合数值优化。



- 互熵和Gini指数对节点中概率的变化更敏感。举例来说，一个二分类模型，每个类各含有400个样本下面有两颗决策树：



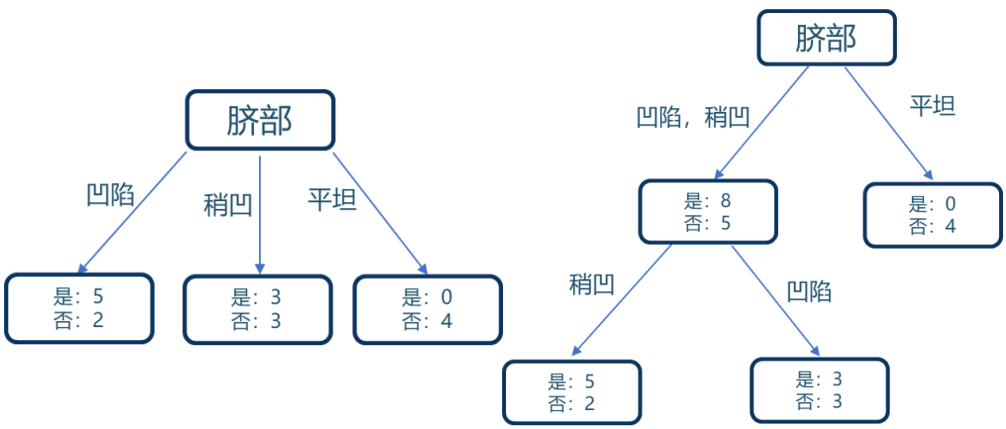
第二种分裂方式产生了纯净的节点，显然更加合理。

然而两种分裂方式的误分类率均为 0.25，通过误分类率无法分辨两种分裂方式的优劣，而第二种方式的 互熵 和 Gini 指数较低能够较好区分两种分裂方式的优劣。因此树增长应该使用 互熵 和 Gini 指数来度量。

- 假设我们把某个节点的结果设置为概率的 p_k 的类 k 中，则该节点上的训练误差为 $\sum_{k \neq k'} p_k(1-p_k)$ ，与 Gini 指数的形式类似 。

CART分类树的改进：

- 采用基尼系数可以降低计算量
- 连续值处理：其思想和 C4.5 是相同的，都是将连续的特征离散化。唯一的区别在于在选择划分点时的度量方式不同，C4.5 使用的是信息增益比，则 CART 分类树使用的是基尼系数。
- 采用二叉树：ID3 或者 C4.5采用的是多叉树；CART树采用的是二叉树。在 ID3 或者 C4.5 的一棵子树中，离散特征只会参与一层节点的建立；CART树中离散的特征可能参与多层节点的建立。

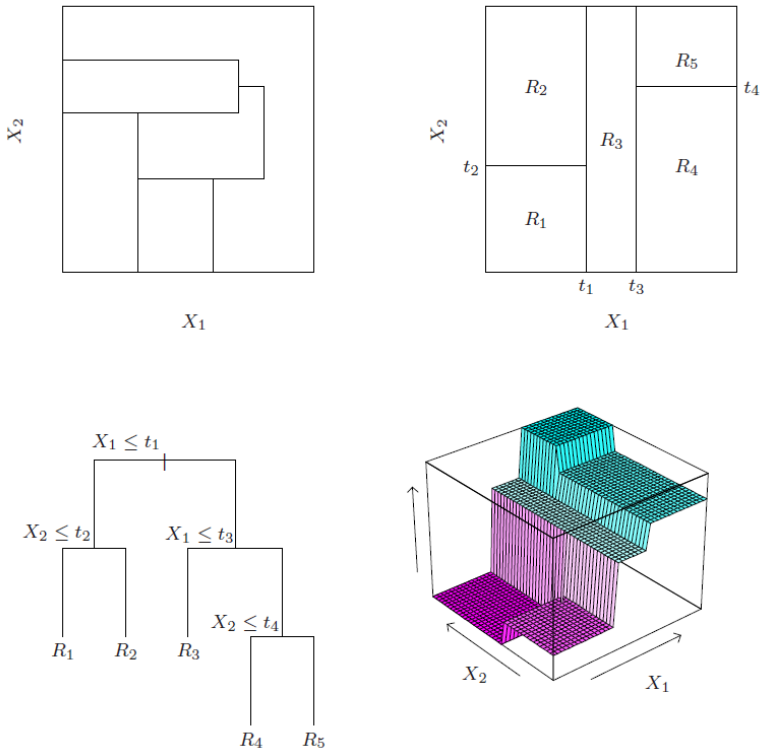


对于生成的决策树做预测的时候，假如测试集里的样本A落到了某个叶子节点，而节点里有多个训练样本。则对于A的类别预测采用的是这个叶子节点里概率最大的类别。

CART回归树：

首先，什么是回归树，什么是分类树。两者的区别在于样本输出，如果样本输出是离散值，那么这是一颗分类树。如果样本输出是连续值，那么这是一颗回归树。

如下图所示，输入为 X1，X2， 输出为连续变量 Y，左上图显示的为一般划分，右上图显示了二叉分裂对特征空间的划分（与坐标轴平行的直线）。



CART 回归树和 CART 分类树的建立和预测的区别主要有下面两点：

Q9：请列举回归树和分类树之间的不同点？

1. 对于连续值的分割：CART 分类树采用的是用基尼系数来度量特征的各个划分点的优劣情况。但是对于回归模型，我们使用了均方差的度量方式，CART 回归树的度量目标是，对于任意分裂变量 j，对应的任意划分点 s 两边划分成的数据集 R1 和 R2，求出使 R1 和 R2 各自集合的均方差最小，同时 R1 和 R2 的均方差之和最小所对应的变量和变量划分点。表达式为：

$$\underbrace{min}_{j,s}\left[\underbrace{min}_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\underbrace{min}_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right]$$

根据平方和的极小值来发现最好的二叉划分通常在计算上是不可行的。对于任意待选择变量 j 和分裂点 s，内部极小化可以用下式子求解。

$$\hat{c}_1 = ave(y_i|x_i \in R_1(j,s))$$
$$\hat{c}_2 = ave(y_i|x_i \in R_2(j,s))$$

其中，c_1 用 R1 区域的输出均值 $\hat{c}_1$ 估计，c_2 用 R2 区域的样本输出均值 $\hat{c}_2$ 估计。

2. 决策树建立后的预测方式：假设我们已经将空间划分为 M 个区域 $R_1, R_2, \cdots, R_M$ ， 之后我们需要求解每个区域上的预测结果。

   CART 分类树采用叶子节点里概率最大的类别作为当前节点的预测类别。即：

   $$k(m) = \mathbf{argmax}_k \hat{p}_{mk}$$

   表示区域 m 上的多数类。

   而回归树它采用的是每个区域的均值或者中位数来预测输出结果，即：

   $$f(x) = \sum_{m=1}^{M} I(x \in R_m)$$

函数 $I()$ 表示求均值或者求中位数。

除了上面提到了以外，CART 回归树和 CART 分类树的建立算法和预测没有什么区别。

4. 决策树小结

Q9：下表给出了ID3，C4.5和CART的一个比较，请补充表格内容

算法	支持模型	树结构	特征选择标准	连续值处理	缺失值处理	剪枝
ID3						
C4.5						
CART						

那么 CART 算法还有没有什么缺点呢？

- 目前我们学到的特征选择都是选择最优的一个特征来做分类决策 $x_j \leq s$，但大多数情况下分类决策不应由某一个特征决定的，而是应该由一组特征决定的 $\sum \alpha_j x_j \leq s$。可以参考 HME 模型 (hierarchical mixtures of experts)。
- 树模型一般都具有较高的方差，数据的一个较小的变化将导致完全不同的分裂结果（出现中这问题的本质是顶层的分裂错误的影响被传播到下面所有的分裂）。可以通过 bagging 可以减轻不稳定性，代价是增加了计算量。

决策树算法的优缺点 (借鉴 sklearn)

首先我们看看决策树算法的优点：

1）简单直观，决策树可以可视化。

2）需要很少的数据准备。不需要提前归一化，处理缺失值。

3）使用决策树预测的代价是O(log2m)。 m为样本数。

4）既处理分类问题，也可以处理回归问题。很多算法只是专注于分类或者回归。

5）能够处理多分类问题。

6) 相比于神经网络之类的黑盒分类模型，决策树在逻辑上可以得到很好的解释

7) 可以交叉验证的剪枝来选择模型，从而提高泛化能力。

8) 对于异常点的容错能力好，健壮性高。

我们再看看决策树算法的缺点:

1) 决策树算法非常容易过拟合，导致泛化能力不强。可以通过设置节点最小样本数和限制决策树深度来改进。

2) 决策树会因为数据的一个较小的波动导致树结构的剧烈改变。可以通过集成学习之类的方法缓解。

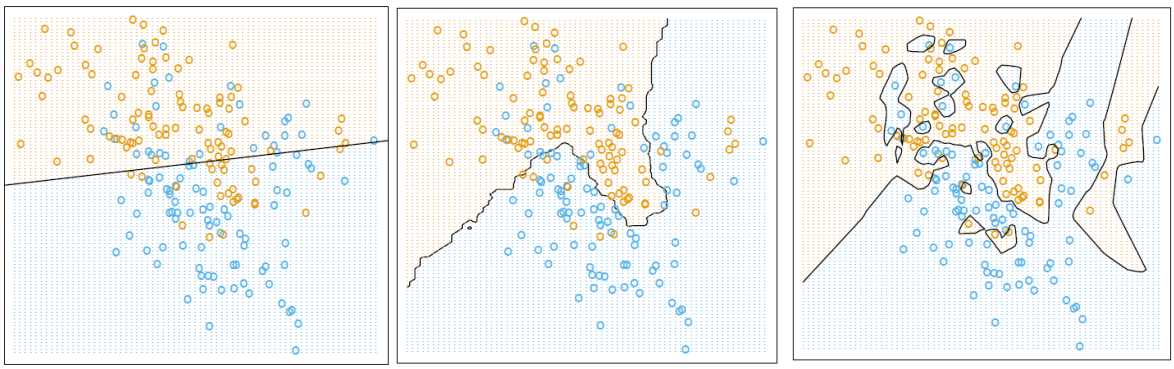
3) 有些比较复杂的关系，比如异或决策树很难学习。这种关系可以换神经网络分类方法来解决。

4) 对于非均衡的数据，生成决策树容易偏向于比例较大的类。可以通过调节样本权重，欠采样，过采样等方法来改善。

怎样运用决策树

首先我们回忆一下欠拟合 (high bias) 和过拟合 (high variance) 的概念。

Q10 ：以下哪个是欠拟合？哪个性能较好？哪个是过拟合



(a) (b) (c)

Q11 ：以下哪个是欠拟合？哪个性能较好？哪个是过拟合

CASE	训练集误差	验证集误差
A	0.1%	1%
B	1%	11%
C	15%	16%

我们来研究一下sklearn中相关的参数：

```
DecisionTreeClassifier(criterion='gini', splitter='best', max_depth=None, min_samples_split=2,
                       min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None,
                       random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0,
                       min_impurity_split=None, class_weight=None, presort=False)

DecisionTreeRegressor(criterion='mse', splitter='best', max_depth=None, min_samples_split=2,
                      min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None,
                      random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, presort=False)
```

参数	DecisionTreeClassifier	DecisionTreeRegressor
特征选择标准 criterion	可以使用"gini"或者"entropy"，前者代表基尼系数，后者代表信息增益。一般说使用默认的基尼系数"gini"就可以了，即CART算法。除非你更喜欢类似ID3, C4.5的最优特征选择方法。	可以使用"mse"或者"mae"，前者是均方差，后者是和均值之差的绝对值之和。推荐使用默认的"mse"。一般来说"mse"比"mae"更加精确。除非你想比较二个参数的效果的不同之处。
特征划分点选择标准 splitter	可以使用"best"或者"random"。前者在特征的所有划分点中找出最优的划分点。后者是随机的在部分划分点中找局部最优的划分点。 默认的"best"适合样本量不大的时候，而如果样本数据量非常大，此时决策树构建推荐"random"	
划分时考虑的最大特征数 max_features	可以使用很多种类型的值，默认是"None",意味着划分时考虑所有的特征数；如果是"log2"意味着划分时最多考虑$\log_2 N$、个特征；如果是"sqrt"或者"auto"意味着划分时最多考虑$\sqrt{N}$个特征。如果是整数，代表考虑的特征绝对数。如果是浮点数，代表考虑特征百分比，即考虑（百分比xN）取整后的特征数。其中N为样本总特征数。 一般来说，如果样本特征数不多，比如小于50，我们用默认的"None"就可以了，如果特征数非常多，我们可以灵活使用刚才描述的其他取值来控制划分时考虑的最大特征数，以控制决策树的生成时间。	
决策树最大深 max_depth	决策树的最大深度，默认可以不输入，如果不输入的话，决策树在建立子树的时候不会限制子树的深度。一般来说，数据少或者特征少的时候可以不管这个值。如果模型样本量多，特征也多的情况下，推荐限制这个最大深度，具体的取值取决于数据的分布。常用的可以取值10-100之间。	
内部节点划分所需最小样本数 min_samples_split	这个值限制了子树继续划分的条件，如果某节点的样本数少于min_samples_split，则不会继续再尝试选择最优特征来进行划分。 默认是2.如果样本量不大，不需要管这个值。如果样本量数量级非常大，则推荐增大这个值。我之前的一个项目例子，有大概10万样本，建立决策树时，我选择了min_samples_split=10。可以作为参考。	
叶子节点最少样本数 min_samples_leaf	这个值限制了叶子节点最少的样本数，如果某叶子节点数目小于样本数，则会和兄弟节点一起被剪枝。 默认是1,可以输入最少的样本数的整数，或者最少样本数占样本总数的百分比。如果样本量不大，不需要管这个值。如果样本量数量级非常大，则推荐增大这个值。之前的10万样本项目使用min_samples_leaf的值为5，仅供参考。	
叶子节点最小的样本权重和 min_weight_fraction_leaf	这个值限制了叶子节点所有样本权重和的最小值，如果小于这个值，则会和兄弟节点一起被剪枝。 默认是0，就是不考虑权重问题。一般来说，如果我们有较多样本有缺失值，或者分类树样本的分布类别偏差很大，就会引入样本权重，这时我们就要注意这个值了。	
最大叶子节点数 max_leaf_nodes	通过限制最大叶子节点数，可以防止过拟合，默认是"None”，即不限制最大的叶子节点数。如果加了限制，算法会建立在最大叶子节点数内最优的决策树。如果特征不多，可以不考虑这个值，但是如果特征分成多的话，可以加以限制，具体的值可以通过交叉验证得到。	
类别权重class_weight	指定样本各类别的的权重，主要是为了防止训练集某些类别的样本过多，导致训练的决策树过于偏向这些类别。这里可以自己指定各个样本的权重，或者用“balanced”，如果使用“balanced”，则算法会自己计算权重，样本量少的类别所对应的样本权重会高。当然，如果你的样本类别分布没有明显的偏倚，则可以不管这个参数，选择默认的"None"	不适用于回归树
节点划分最小不纯度 min_impurity_split	这个值限制了决策树的增长，如果某节点的不纯度(基尼系数，信息增益，均方差，绝对差)小于这个阈值，则该节点不再生成子节点。即为叶子节点 。	
数据是否预排序 presort	这个值是布尔值，默认是False不排序。一般来说，如果样本量少或者限制了一个深度很小的决策树，设置为true可以让划分点选择更加快，决策树建立的更加快。如果样本量太大的话，反而没有什么好处。问题是样本量少的时候，我速度本来就不慢。所以这个值一般懒得理它就可以了。	

决策树常用的一些参数：

max_features，max_depth，min_samples_split，min_samples_leaf，class_weight

Q12：针对一下几种情况，分析怎样调整相关参数：

模型过拟合（高方差）：max_features，max_depth，min_samples_split，min_samples_leaf

模型欠拟合（高偏差）：max_features，max_depth，min_samples_split，min_samples_leaf

数据不平衡：class_weight