**西瓜的选择**

在实际使用机器学习算法的过程中，往往在[特征选择](https://so.csdn.net/so/search?q=%E7%89%B9%E5%BE%81%E9%80%89%E6%8B%A9&spm=1001.2101.3001.7020)这一块是一个比较让人模棱两可的问题，有时候可能不知道如果想要让当前的模型效果更好，到底是应该加还是减掉一些特征，加又是加哪些，减又是减哪些。

**一、子集搜索与评价**

一般地，我们可以用很多属性/特征来描述一个示例，例如对于一个人可以用性别、身高、体重、年龄、学历、专业、是否吃货等属性来描述，那现在想要训练出一个学习器来预测西瓜的好坏。根据生活经验易知：并不是所有的特征都与学习任务相关。因此我们只需要那些与学习任务紧密相关的特征，特征选择便是从给定的特征集合中选出相关特征子集的过程。

与上篇中降维技术有着异曲同工之处的是，特征选择也可以有效地解决维数灾难的难题。具体而言：降维从一定程度起到了提炼优质低维属性和降噪的效果，特征选择则是直接剔除那些与学习任务无关的属性而选择出最佳特征子集。若直接遍历所有特征子集，显然当维数过多时遭遇指数爆炸就行不通了；若采取从候选特征子集中不断迭代生成更优候选子集的方法，则时间复杂度大大减小。这时就涉及到了两个关键环节：

1.如何生成候选子集；

2.如何评价候选子集的好坏，这便是早期特征选择的常用方法。

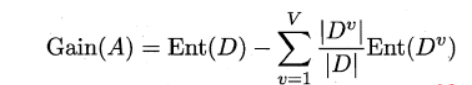
这便是早期特征选择的常用方法。书本上介绍了贪心算法，分为三种策略：

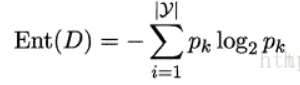
前向搜索：初始将每个特征当做一个候选特征子集，然后从当前所有的候选子集中选择出最佳的特征子集；接着在上一轮选出的特征子集中添加一个新的特征，同样地选出最佳特征子集；最后直至选不出比上一轮更好的特征子集。

后向搜索：初始将所有特征作为一个候选特征子集；接着尝试去掉上一轮特征子集中的一个特征并选出当前最优的特征子集；最后直到选不出比上一轮更好的特征子集。

双向搜索：将前向搜索与后向搜索结合起来，即在每一轮中既有添加操作也有剔除操作。

对于特征子集的评价，书中给出了一些想法及基于信息熵的方法。假设数据集的属性皆为离散属性，这样给定一个特征子集，便可以通过这个特征子集的取值将数据集合划分为V个子集，其中每个子集的取值完全相同。这时我们就可以像决策树选择划分属性那样，通过计算信息增益来评价该属性子集的好坏。





此时，信息增益越大表示该属性子集包含有助于分类的特征越多，使用上述这种子集搜索与子集评价相结合的机制，便可以得到特征选择方法。事实上，决策树也可以用于特征选择，树节点划分属性组成的集合便是选择出的特征子集。用以上策略可以对西瓜进行特征选择。

常见的特征选择方法大致可分为3类；过滤式、包裹式和嵌入式

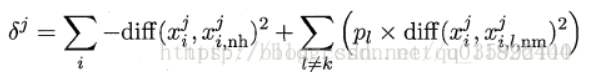
**1.1过滤式特征选择**

过滤式方法是一种将特征选择与学习器训练相分离的特征选择技术，即首先将相关特征挑选出来，再使用选择出的数据子集来训练学习器。Relief是其中著名的代表性算法，它使用一个“相关统计量”来度量特征的重要性，该统计量是一个向量，其中每个分量代表着相应特征的重要性，因此我们最终可以根据这个统计量各个分量的大小来选择出合适的特征子集。

易知Relief算法的核心在于如何计算出该相关统计量。对于数据集中的每个样例xi，Relief首先找出与xi同类别的最近邻与不同类别的最近邻，分别称为猜中近邻（near-hit）xi,nhxi,nh与猜错近邻（near-miss）xi,nmxi,nm，接着便可以分别计算出相关统计量中的每个分量。对于j分量：

直观上理解：对于猜中近邻，两者j属性的距离越小越好，对于猜错近邻，j属性距离越大越好。更一般地，若xi为离散属性，diff取海明距离，即相同取0，不同取1；若xi为连续属性，则diff为曼哈顿距离，即取差的绝对值。分别计算每个分量，最终取平均便得到了整个相关统计量。

标准的Relief算法只用于二分类问题，后续产生的拓展变体Relief-F则解决了多分类问题。对于j分量，新的计算公式如下：



**1.2包裹式特征选择**

与过滤式选择不同的是，包裹式选择将后续的学习器也考虑进来作为特征选择的评价准则。因此包裹式选择可以看作是为某种学习器量身定做的特征选择方法，由于在每一轮迭代中，包裹式选择都需要训练学习器，因此在获得较好性能的同时也产生了较大的开销。下面主要介绍一种经典的包裹式特征选择方法 –LVW（Las Vegas Wrapper），它在拉斯维加斯框架下使用随机策略来进行特征子集的搜索。

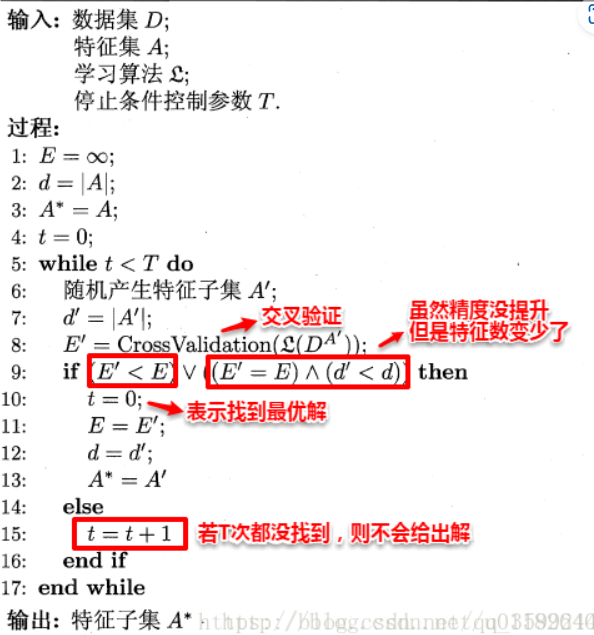
蒙特卡罗算法：采样越多，越近似最优解，一定会给出解，但给出的解不一定是正确解；

拉斯维加斯算法：采样越多，越有机会找到最优解，不一定会给出解，且给出的解一定是正确解。

举个例子，假如筐里有100个苹果，让我每次闭眼拿1个，挑出最大的。于是我随机拿1个，再随机拿1个跟它比，留下大的，再随机拿1个……我每拿一次，留下的苹果都至少不比上次的小。拿的次数越多，挑出的苹果就越大，但我除非拿100次，否则无法肯定挑出了最大的。这个挑苹果的算法，就属于蒙特卡罗算法——尽量找较好的，但不保证是最好的。

而拉斯维加斯算法，则是另一种情况。假如有一把锁，给我100把钥匙，只有1把是对的。于是我每次随机拿1把钥匙去试，打不开就再换1把。我试的次数越多，打开（正确解）的机会就越大，但在打开之前，那些错的钥匙都是没有用的。这个试钥匙的算法，就是拉斯维加斯的——尽量找最好的，但不保证能找到。

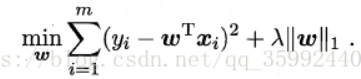
LVW算法的具体流程如下所示，其中比较特别的是停止条件参数T的设置，即在每一轮寻找最优特征子集的过程中，若随机T次仍没找到，算法就会停止，从而保证了算法运行时间的可行性。



可以实现对西瓜最优特征子集的筛选。

**1.3 嵌入式选择与正则化**

前面提到了的两种特征选择方法：过滤式中特征选择与后续学习器完全分离，包裹式则是使用学习器作为特征选择的评价准则；嵌入式是一种将特征选择与学习器训练完全融合的特征选择方法，即将特征选择融入学习器的优化过程中。在之前《经验风险与结构风险》中已经提到：经验风险指的是模型与训练数据的契合度，结构风险则是模型的复杂程度，机器学习的核心任务就是：在模型简单的基础上保证模型的契合度。例如：岭回归就是加上了L2范数的最小二乘法，有效地解决了奇异矩阵、过拟合等诸多问题，下面的嵌入式特征选择则是在损失函数后加上了L1范数。



总的来说：L1范数会趋向产生少量的特征，其他特征的权值都是0；L2会选择更多的特征，这些特征的权值都会接近于0。这样L1范数在特征选择上就十分有用，而L2范数则具备较强的控制过拟合能力。可以从下面两个方面来理解：

（1）下降速度：L1范数按照绝对值函数来下降，L2范数按照二次函数来下降。因此在0附近，L1范数的下降速度大于L2范数，故L1范数能很快地下降到0，而L2范数在0附近的下降速度非常慢，因此较大可能收敛在0的附近。

(2）空间限制：L1范数与L2范数都试图在最小化损失函数的同时，让权值W也尽可能地小。

**二、稀疏表示与字典学习**

当样本数据是一个稀疏矩阵时，对学习任务来说会有不少的好处，例如很多问题变得线性可分，储存更为高效等。这便是稀疏表示与字典学习的基本出发点。稀疏矩阵即矩阵的每一行/列中都包含了大量的零元素，且这些零元素没有出现在同一行/列，对于一个给定的稠密矩阵，若我们能通过某种方法找到其合适的稀疏表示，则可以使得学习任务更加简单高效，我们称之为稀疏编码（sparse coding）或字典学习（dictionary learning）。

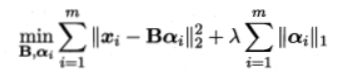
给定一个数据集，字典学习/稀疏编码指的便是通过一个字典将原数据转化为稀疏表示，因此最终的目标就是求得字典矩阵B及稀疏表示α，书中使用变量交替优化的策略能较好地求得解。

特征选择考虑的问题是特征具有稀疏性，即矩阵中的许多列与当前学习任务无关。

另一种稀疏性：D所对应的矩阵存在很多零元素，但这些零元素并不是以整列、整行的形式存在。当样本具有这样的稀疏表达形式时，对学习任务来说会有不少好处，例如，支持向量机之所以能在文本数据上有很好的性能恰是由于文本数据在使用上述字频表示时具有高度的稀疏性，使大多数问题变得线性可分，同时稀疏矩阵不会造成存储上的巨大负担。

字典学习：为普通稠密表达的样本找到合适的字典，将样本转换为合适的稀疏表示形式，从而使学习任务得以简化，模型复杂度得以降低，通常称为“字典学习”，亦称“稀疏编码”。

字典学习最简答的形式为：



是的稀疏表示，B是字典矩阵，第一项希望 能很好的重构，第二项希望 尽量稀疏。

**2.1压缩感知**

压缩感知在前些年也是风风火火，与特征选择、稀疏表示不同的是：它关注的是通过欠采样信息来恢复全部信息。在实际问题中，为了方便传输和存储，我们一般将数字信息进行压缩，这样就有可能损失部分信息，如何根据已有的信息来重构出全部信号，这便是压缩感知的来历，压缩感知的前提是已知的信息具有稀疏表示。

**三、西瓜数据集的CART算法实现**

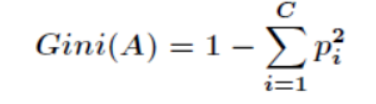
3.1、CART算法简介

CART算法是一种二分递归分割技术，把当前样本划分为两个子样本，使得生成的每个非叶子结点都有两个分支，因此CART算法生成的决策树是结构简洁的二叉树。由于CART算法构成的是一个二叉树，它在每一步的决策时只能是“是”或者“否”，即使一个feature有多个取值，也是把数据分为两部分。

3.2、步骤简叙

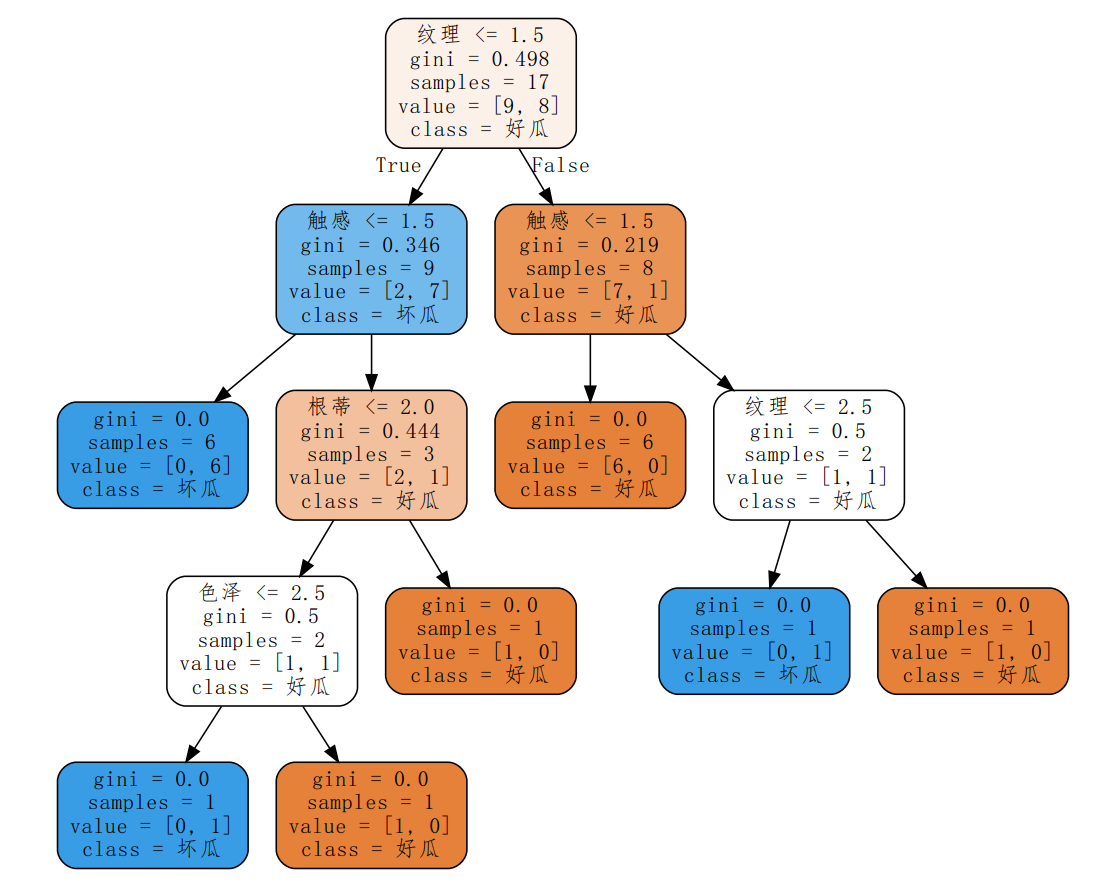
首先选一个自变量，再选取的一个值，把维空间划分为两部分，一部分的所有点都满足，另一部分的所有点都满足，对非连续变量来说属性值的取值只有两个，即等于该值或不等于该值。

然后递归处理，将上面得到的两部分按步骤一重新选取一个属性继续划分，直到把整个维空间都划分完。在划分时候有一个问题，它是按照什么标准来划分的 ？ 对于一个变量属性来说，它的划分点是一对连续变量属性值的中点。假设个样本的集合一个属性有个连续的值，那么则会有个分裂点，每个分裂点为相邻两个连续值的均值。每个属性的划分按照能减少的杂质的量来进行排序，而杂质的减少量定义为划分前的杂质减去划分后的每个节点的杂质量划分所占比率之和。而杂质度量方法常用Gini指标，假设一个样本共有类，那么一个节点的Gini不纯度可定义为：



其中pi表示属于第i类的概率，当Gini(A)=0时，所有样本属于同类，所有类在节点中以等概率出现时，Gini(A)最大化，此时Pi 有了上述理论基础，实际的递归划分过程是这样的：如果当前节点的所有样本都不属于同一类或者只剩下一个样本，那么此节点为非叶子节点，所以会尝试样本的每个属性以及每个属性对应的分裂点，尝试找到杂质变量最大的一个划分，该属性划分的子树即为最优分支。

通过python实现对西瓜数据集3.0构建的CART决策树，并通过graphviz可视化输出如下：



**总结：通过以上策略可以对西瓜的特征进行筛选，从而得到影响西瓜好坏最大的特征集，便于我们了解拥有什么样特征的西瓜才是好瓜。**