**'''  
函数说明：计算根节点的“信息熵”、以及其他输入数据的“信息熵”  
输入：data（DataFrame类型变量，最后一列是 label，最后一列之前有多少列没有要求）  
返回值：“信息熵”  
'''**  
**def** root\_information\_gain(data):

**'''  
函数说明：计算输入数据集中，下一次该选择哪个特征来分裂数据（这个只可以处理 离散型特征）  
输入：data（DataFrame类型变量，最后一列是 label，最后一列之前“全部”是N列特征列，我们就是要从这N列特征中选择下一次该选择哪个特征来分裂）  
返回值：1个list变量，里面存有两个值，第一个是 选择好的特征名，第二个是 对应的信息增益  
'''  
def** nextChooseWhichFeature\_lisan(data):

**'''  
函数说明：计算输入数据集中，下一次该选择哪个特征来分裂数据（这个只可以处理 连续性特征）  
输入：data（DataFrame类型变量，最后一列是 label，最后一列之前“全部”是N列特征列，我们就是要从这N列特征中选择下一次该选择哪个特征来分裂）  
返回值：最好的特征“name”  
'''  
def** nextChooseWhichFeature\_lianxu(data):

**'''  
函数说明：分裂结束的时候，程序遍历完所有划分数据集的属性，但是每一个分支下的所有实例如果不具有同一个“标签”，就得用“投票选择”。下面这个函数就是这个  
输入：data（list类型，每一个索引对应的值是 这个实例的分类）  
返回值：“投票选择”思想，投出来的“分类”  
'''  
def** vote(data):

**'''  
函数说明：创建一棵树，具体的分裂信息 用 一个字典嵌套一个字典 表示出来  
输入：data（DataFrame类型变量，最后一列是 label，最后一列之前“全部”是N列特征列）  
返回值：一个字典（表示树的分裂情况）  
补充说明：分裂停止的条件 1：要么一个结点所有的实例都属于了一个分类 2：要么就是达到了预先设定好的树深（或者 遍历完了所有划分数据集的属性）  
'''  
def** create\_tree(data):