```
import numpy as np
from math import log

# 信息熵

def calcShang(inputD):
    """
    计算给定数据集的信息熵
    """
    numEntries = len(inputD)
    labelCounts = {}
    for featVec in inputD:
        nowLabel = str(featVec[-1])
        if nowLabel not in labelCounts:
            labelCounts[nowLabel] = 0;
        labelCounts[nowLabel] += 1
    shang = 0.0
    for key in labelCounts:
        prob = float(labelCounts[key]) / numEntries
        print(str(key) + "类别的概率：" + str(prob))
        print(prob * log(prob, 2) )
        shang -= prob * log(prob, 2)
        print("熵值：" + str(shang))
    return shang

def normalData(dataIn, axis, value):
    """
    划分数据集,处理离散变量 提取所有满足一个特征的值
    @ dataIn: 数据集
    @ axis: 划分依据
    @ value: 提取出来满足某特征的list
    """
    dataOut = []
    for i in dataIn:
        if i[axis] == value:
            dataOut0 = list(i[:axis])
            dataOut0.extend(i[axis+1:])
            dataOut.append(dataOut0)
    return dataOut




def continueData(dataIn, axis, value,direction):
```

```
'''
处理连续特征返回特征取值大于/小于value的所有样本
@ dataIn: 数据集
@ axis: 划分依据
@ value: 提取出来满足某特征的list
'''
output=[]
for i in dataIn:
    if direction==0:
        if i[axis] > value:
            dataOutS0 = i[:axis]
            dataOutS0.extend(i[axis+1:])
            output.append(dataOutS0)
    else:
        if i[axis]<=value:
            dataOutL0 = i[:axis]
            dataOutL0.extend(i[axis+1:])
            output.append(dataOutL0)
return output
```