```
import calculate as ca
from numpy import *
import operator


def chooseAxis(dataIn, labels):
    """
    选择最优的划分属性
    @ param dataIn: 数据集
    @ return bestAxis: 最佳划分属性
    """
    # NomalShang convert SE
    # InfoGain convert IG
    numAxiss = len(dataIn[0])-1
    baseShang = ca.calcShang(dataIn)
    bestIG = 0.0
    bestAxis = -1
    bestSplitDict = {}
    for i in range(numAxiss):
        featList = [example[i] for example in dataIn]
        if type(featList[0]).__name__ == 'float' or type(featList[0]).__name__ == 'int':
            sortfeatList = sorted(featList)
            splitList = []
            for j in range(len(sortfeatList)-1):
                splitList.append((sortfeatList[j]+sortfeatList[j+1])/2.0)
            bestSE = 10000
            slen = len(splitList)
            for j in range(slen):
                value = splitList[j]
                newShang = 0.0
                subDataSet0 = ca.continueData(dataIn, i, value, 0)
                subDataSet1 = ca.continueData(dataIn, i, value, 1)
                prob0 = len(subDataSet0)/float(len(dataIn))
                newShang += prob0*ca.calcShang(subDataSet0)
                prob1 = len(subDataSet1)/float(len(dataIn))
                newShang += prob1*ca.calcShang(subDataSet1)
                if newShang < bestSE:
                    bestSE = newShang
                    bestSplit = j
            bestSplitDict[labels[i]] = splitList[bestSplit]
            iG = baseShang-bestSE
        else:
            uniqueVals = set(featList)
            newShang = 0.0
```

```
            for value in uniqueVals:
                subDataSet = ca.normalData(dataIn, i, value)
                prob = len(subDataSet)/float(len(dataIn))
                newShang += prob*ca.calcShang(subDataSet)
            iG = baseShang-newShang
        if iG > bestIG:
            bestIG = iG
            bestAxis = i
    if type(dataIn[0][bestAxis]).__name__ == 'float' or
type(dataIn[0][bestAxis]).__name__ == 'int':
        bestSplitValue = bestSplitDict[labels[bestAxis]]
        labels[bestAxis] = labels[bestAxis]+'<='+str(round(bestSplitValue, 4))
        for i in range(shape(dataIn)[0]):
            if dataIn[i][bestAxis] <= bestSplitValue:
                dataIn[i][bestAxis] = 1
            else:
                dataIn[i][bestAxis] = 0
    return bestAxis


def extraChoose(listin):
    '''
    意外：vote函数
    '''
    lableDict = {}
    for i in listin:
        if i not in lableDict.keys():
            lableDict[i] = 0
        lableDict[i] += 1
    return max(lableDict)
```