**作业7.3**

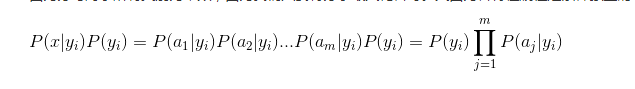
1. **实验要求**

试编程实现拉普拉斯修正的朴素贝叶斯分类器，并以西瓜数据集3.0为训练集，对p.151”测1”样本进行判别。

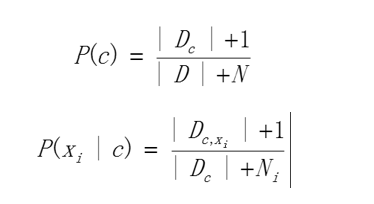
1. **实验原理**

朴素贝叶斯分类器采用了“属性条件独立性假设”：对于已知类别，假设所有属性相互独立，即每个属性独立的对分类结果发生影响。

所以有：



当P(a|y)=0时，即某个类别下某个特征项划分没有出现时，就是产生这种现象，这会令分类器质量大大降低。为了解决这个问题，我们引入Laplace校准，它的思想非常简单，就是对没类别下所有划分的计数加1，这样如果训练样本集数量充分大时，并不会对结果产生影响。即:



判别函数如下：



根据此式计算结果进行判别。

1. 实验过程及代码

本题的处理思路是，先列出数据，然后再分别对两个类的各个离散属性和连续属性建立类条件概率表达式，最后代入测试样例给出其属于两种类的概率，并进行判别，过程如下：

1. 准备数据，设立标记
2. 为两个类分别建立字典，分别统计每个类的比例。
3. 利用拉普拉斯方法分别统计两个类中各种离散属性分别占本类中该属性值的比例。对连续值属性也进行类似计算。
4. 代入测试集，分别计算其被分到两个类的概率，并给出结论。

相关代码如下：

import numpy as np

D = np.array([

[1, 1, 1, 1, 1, 1, 0.697, 0.460, 1],

[2, 1, 2, 1, 1, 1, 0.774, 0.376, 1],

[2, 1, 1, 1, 1, 1, 0.634, 0.264, 1],

[1, 1, 2, 1, 1, 1, 0.608, 0.318, 1],

[3, 1, 1, 1, 1, 1, 0.556, 0.215, 1],

[1, 2, 1, 1, 2, 2, 0.403, 0.237, 1],

[2, 2, 1, 2, 2, 2, 0.481, 0.149, 1],

[2, 2, 1, 1, 2, 1, 0.437, 0.211, 1],

[2, 2, 2, 2, 2, 1, 0.666, 0.091, 0],

[1, 3, 3, 1, 3, 2, 0.243, 0.267, 0],

[3, 3, 3, 3, 3, 1, 0.245, 0.057, 0],

[3, 1, 1, 3, 3, 2, 0.343, 0.099, 0],

[1, 2, 1, 2, 1, 1, 0.639, 0.161, 0],

[3, 2, 2, 2, 1, 1, 0.657, 0.198, 0],

[2, 2, 1, 1, 2, 2, 0.360, 0.370, 0],

[3, 1, 1, 3, 3, 1, 0.593, 0.042, 0],

[1, 1, 2, 2, 2, 1, 0.719, 0.103, 0]])

m, n = D.shape[0], D.shape[1] - 1 # number of instances,attributes

label = np.unique(D[:, -1]) #返回类别

class\_dict = {int(l): 0 for l in label}#建立类别字典分别为0,1

for i in range(m):

class\_dict[D[i, -1]] += 1 #为0和1两个不同的类别分别进行累加统计0和1类的数目

p\_class = {l: (class\_dict[l] + 1) / (m + 2) for l in class\_dict}#得到各类的比例

DICT0 = [{} for item in range(n)] # list of dicts that contain their own samples with class 0

DICT1 = [{} for item in range(n)] # list of dicts that contain their own samples with class 1

for i, d in enumerate(DICT0[:-2]): #i的范围是前六个属性,d为该属性的值

DICT0[i] = {int(a): 0 for a in np.unique(D[:, i])} #统计第i个属性的值有哪些

d = DICT0[i] #建立这个属性的值的字典

k = len(np.unique(D[:, i])) # 第i个属性有几种值

for j in range(8):

d[D[j, i]] += 1 #统计该属性不同值的数目

DICT0[i] = {l: (d[l] + 1) / (8 + k) for l in d}

for i, d in enumerate(DICT1[:-2]):

DICT1[i] = {int(a): 0 for a in np.unique(D[:, i])}

d = DICT1[i]

k = len(np.unique(D[:, i])) # number of attributes in column i

for j in range(8, m):

d[D[j, i]] += 1

DICT1[i] = {l: (d[l] + 1) / (9 + k) for l in d}

def prob\_continuous(x, data\_n): # probability of continuous variables

mean = np.mean(data\_n) #计算均值

var = np.var(data\_n) #计算方差

p = np.exp(-(x - mean) \*\* 2 \* 0.5 / var) / (np.sqrt(2 \* np.pi \* var))

return p

test = [1, 1, 1, 1, 1, 1, 0.697, 0.46] # the predict sample

result = [p\_class[0], p\_class[1]]

DICT0[6], DICT0[7] = prob\_continuous(test[-2], D[:8, 6]), prob\_continuous(test[-1], D[:8, 7]) #计算概率密度

DICT1[6], DICT1[7] = prob\_continuous(test[-2], D[8:, 6]), prob\_continuous(test[-1], D[8:, 7])

for i, t in enumerate(test[:-2]):

result[0] \*= DICT0[i][t] #分别乘上各个属性的对应的计算值

result[1] \*= DICT1[i][t]

result[0] \*= DICT0[6] \* DICT0[7]

result[1] \*= DICT1[6] \* DICT1[7]

print(result)

1. **实验结果**

被分类到好瓜和坏瓜的概率分别如下：

[0.024223607117715082, 4.4242506192749345e-05]

很显然分类到好瓜的概率更大，因此判别结果为好瓜。