朴素贝叶斯

## 朴素贝叶斯原理：

## 朴素贝叶斯分类是一种十分简单的分类算法，叫它朴素贝叶斯分类是因为这种方法的思想真的很朴素，朴素贝叶斯的思想基础是这样的：对于给出的待分类项，求解在此项出现的条件下各个类别出现的概率，哪个最大，就认为此待分类项属于哪个类别。

朴素贝叶斯分类的正式定义如下：

1、设为一个待分类项，而每个a为x的一个特征属性。

2、有类别集合。

**3、计算。**

4、如果，则。

那么现在的关键就是如何计算第3步中的各个条件概率。我们可以这么做：

1、找到一个已知分类的待分类项集合，这个集合叫做训练样本集。

2、统计得到在各类别下各个特征属性的条件概率估计。即

。

3、如果各个特征属性是条件独立的，则根据贝叶斯定理有如下推导：



因为分母对于所有类别为常数，因为我们只要将分子最大化皆可。又因为各特征属性是条件独立的，所以有：



### 朴素贝叶斯流程

整个朴素贝叶斯分类分为三个阶段：

第一阶段——准备工作阶段，这个阶段的任务是为朴素贝叶斯分类做必要的准备，主要工作是根据具体情况确定特征属性，并对每个特征属性进行适当划分，然后由人工对一部分待分类项进行分类，形成训练样本集合。这一阶段的输入是所有待分类数据，输出是特征属性和训练样本。这一阶段是整个朴素贝叶斯分类中唯一需要人工完成的阶段，其质量对整个过程将有重要影响，分类器的质量很大程度上由特征属性、特征属性划分及训练样本质量决定。

第二阶段——分类器训练阶段，这个阶段的任务就是生成分类器，主要工作是计算每个类别在训练样本中的出现频率及每个特征属性划分对每个类别的条件概率估计，并将结果记录。其输入是特征属性和训练样本，输出是分类器。这一阶段是机械性阶段，根据前面讨论的公式可以由程序自动计算完成。

第三阶段——应用阶段。这个阶段的任务是使用分类器对待分类项进行分类，其输入是分类器和待分类项，输出是待分类项与类别的映射关系。这一阶段也是机械性阶段，由程序完成。

## 题目

价格A 课时B 销量C

低 多 高

高 中 高

低 少 高

低 中 低

中 中 中

高 多 高

低 少 中

现在学校出了一门新的课程，课程价格A=高，课时B=多，需要预测这个课程的销量

这个问题提出了预测之后的结果，而朴素贝叶斯正好可以满足这一点，网上大多是直接调用API进行预测，实际上最好还是自己实现朴素贝叶斯，朴素贝叶斯公式：P（B|A）=P（A|B）P（B）/P（A），而本文中，公式即为，P（C|AB）=P（AB|C）P（C）/P（AB）=P（A|C）P（B|C）P（C）/P（AB），方法就是分别推算出C为低销量，中销量，高销量时候的概率，然后进行比较，反馈出最大的概率为预测的结果

## 代码：

#coding=utf-8

from \_\_future\_\_ import division

from numpy import array

def set\_data(price,time,sale):

price\_number =[]

time\_number= []

sale\_number =[]

for i in price:

if i=="低":

price\_number.append(0)

elif i=="中":

price\_number.append(1)

elif i=="高":

price\_number.append(2)

for j in time:

if j=="少":

time\_number.append(0)

elif j=="中":

time\_number.append(1)

elif j=="多":

time\_number.append(2)

for k in sale:

if k=="低":

sale\_number.append(0)

elif k=="中":

sale\_number.append(1)

elif k=="高":

sale\_number.append(2)

return price\_number,time\_number,sale\_number

def naive\_bs(price\_number,time\_number,sale\_number,expected\_price,expected\_time):

price\_p=[]

time\_p=[]

sale\_p=[]

m = array(zip(price\_number,time\_number,sale\_number)).T

for i in range(3):

price\_p.append(price.count(i)/len(price\_number)) #计算各项概率

time\_p.append(time.count(i)/len(time\_number))

sale\_p.append(sale.count(i)/len(sale\_number))

advance\_sale=[]

p\_ex\_price = price.count(expected\_price)/len(price\_number)

p\_ex\_time = time.count(expected\_time)/len(time\_number)

low\_ex\_sale=0

middle\_ex\_sale=0

high\_ex\_sale=0

for i in range(0,len(sale\_number)):

if sale\_number[i]==0:

low\_ex\_sale=low\_ex\_sale+1

elif sale\_number[i]==1:

middle\_ex\_sale=middle\_ex\_sale+1

elif sale\_number[i]==2:

high\_ex\_sale=high\_ex\_sale+1

#统计p(c)出现的概率

#计算不同情况

aa=0

bb=0

cc=0

for i in range(0,len(price\_number)):

if expected\_price==price\_number[i] and sale\_number[i]==0:

aa=aa+1

elif expected\_price==price\_number[i] and sale\_number[i]==1:

bb=bb+1

elif expected\_price==price\_number[i] and sale\_number[i]==2:

cc=cc+1

p\_aa = aa/low\_ex\_sale

p\_bb =bb/middle\_ex\_sale

p\_cc = cc/high\_ex\_sale

print "p(a|c):%s ,%s,%s"%(p\_aa,p\_bb,p\_cc)

aaa=0

bbb=0

ccc=0

for i in range(0,len(time\_number)):

if expected\_time==time\_number[i] and sale\_number[i]==0:

aaa=aaa+1

elif expected\_time==time\_number[i] and sale\_number[i]==1:

bbb=bbb+1

elif expected\_time==time\_number[i] and sale\_number[i]==2:

ccc=ccc+1

p\_aaa=aaa/low\_ex\_sale

p\_bbb=bbb/middle\_ex\_sale

p\_ccc=ccc/high\_ex\_sale

print "p(b|c): %s,%s,%s"%(p\_aaa,p\_bbb,p\_ccc)

final\_low\_p = p\_aa\*p\_aaa\*low\_ex\_sale/len(sale\_number)\*1000

final\_midd\_p = p\_bb\*p\_bbb\*middle\_ex\_sale/len(sale\_number)\*1000

final\_high\_p = p\_cc\*p\_ccc\*high\_ex\_sale/len(sale\_number)\*1000

final\_list=[final\_low\_p,final\_midd\_p,final\_high\_p]

final\_index= final\_list.index(max(final\_list))

print final\_list

if final\_index==0:

print "销量预测销量为低"

elif final\_index==1:

print "销量预测销量为中"

else:

print "销量预测销量为高"

if \_\_name\_\_=="\_\_main\_\_":

price = ["低","高","低","低","中","高","低"]

time = ["多","中","少","中","中","多","少"]

sale = ["高","高","高","低","中","高","中"]

expected\_price="高" #新课程价格高

expected\_time="高" #新课程课时多

if expected\_price=="低":

expected\_price\_id=0

elif expected\_price=="中":

expected\_price\_id=1

else:

expected\_price\_id=2

if expected\_time=="少":

expected\_time\_id=0

elif expected\_time=="中":

expected\_time\_id=1

else:

expected\_time\_id=2

price\_number,time\_number,sale\_number= set\_data(price, time, sale)

print price\_number,time\_number,sale\_number

naive\_bs(price\_number, time\_number, sale\_number, expected\_price\_id, expected\_time\_id).

### 实验结果截图

