电子商务导论

第四章 商务数据挖掘

朱桂祥 (9120201070@nufe.edu.cn)
南京财经大学信息工程学院
江苏省电子商务重点实验室
电子商务信息处理国家级国际联合研究中心
电子商务交易技术国家地方联合工程实验室

南京理工大学校训

进德修业　志道鼎新

《周易·乾》有云：“君子进德修业，忠信，所以进德也，修辞立其诚，所以居业也。”以“德”为首，体现了学校“立德树人”“以德为先”的办学前提，而“修业”则体现了学校育人的追求与境界，即教师诲人不倦，勤业精业乐业；学生孜孜以求，创新创业创优。

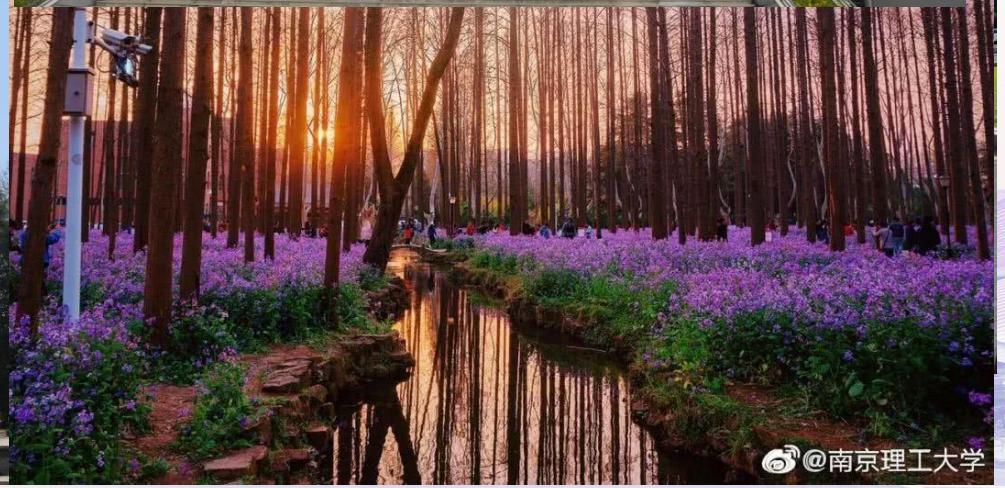
“志道鼎新”，取意“探究道理，创造新知”，既是南京理工人追求科学真理、矢志技术创新的真实写照，也是他们勇立潮头、披荆斩棘的责任担当和精神源泉

南京理工大学校风

20世纪80年代，学校以1953年8月毛泽东主席为哈军工颁发的《中央人民政府人民革命军事委会训词》为指导，经过征集、提炼，将**“团结 献身 求是 创新”**确定为学校校风。

“团结” 是包容，是协作，是团队合作的凝聚力量；
“献身” 是奉献，是追求，是执着进取的精神境界；
“求是” 是探索，是求真，是理性务实的科学素养；
“创新” 是批判，是创造，是成就进步的不竭源泉。

南京理工大学校风风光



目 录
Contents

第一节 关联规则挖掘

第二节 分类

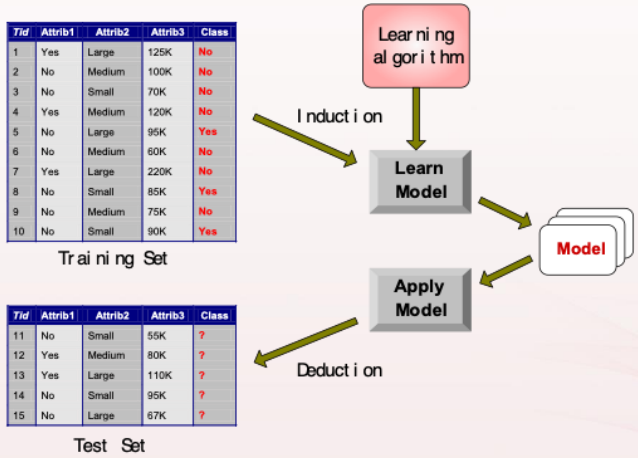
第三节 聚类

二、分类

1.背景

标签离散

基于训练数据，构建面向**类别属性(label)/应变量(特征属性)**的分类预测模型；对未知类别的实例，由输入的应变量可获得预测的类别。
训练分类器属于**监督学习(Supervised Learning)**



例子：预测社交账号是否真实

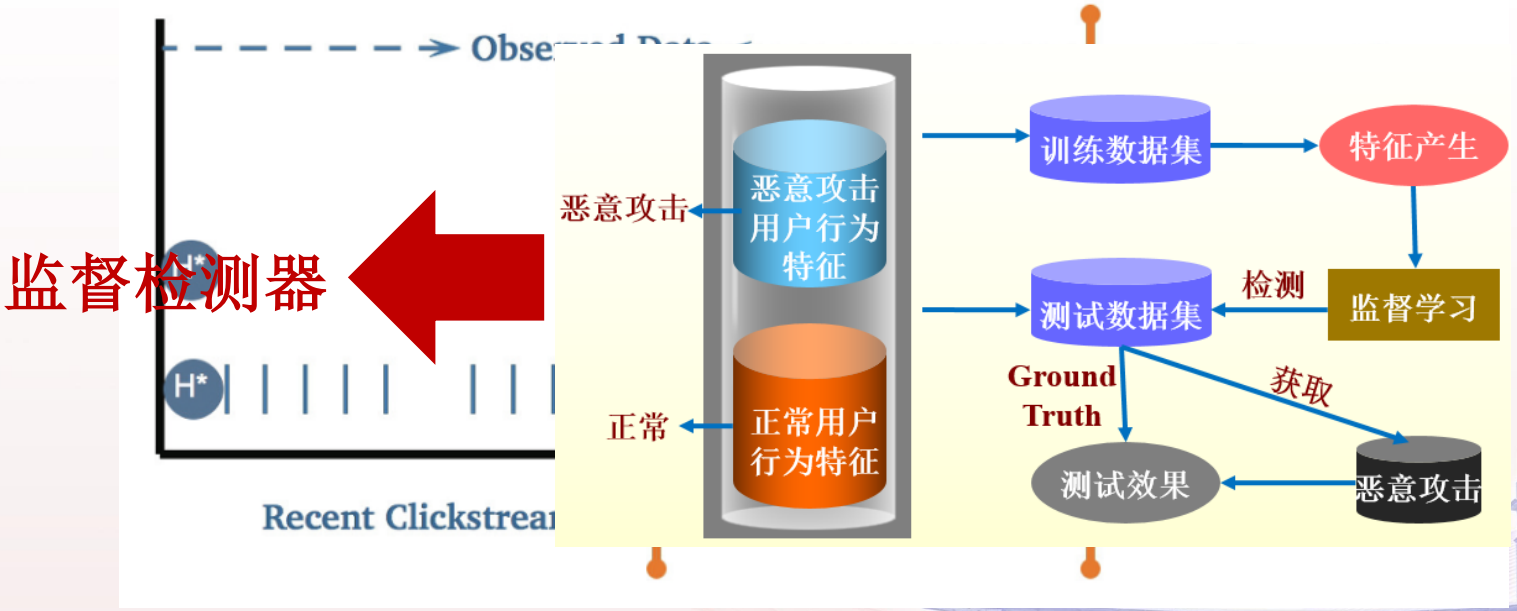
- 类别: 0-虚假，1-真实
- 特征：F1: 日志数量/注册天数；F2: 好友数量/注册天数；F3: 是否使用真实头像（真实头像为1，非真实头像为0）

输入变量可以是离散的，也可以是连续的。

二、分类

1.背景

分类模型可以用于：



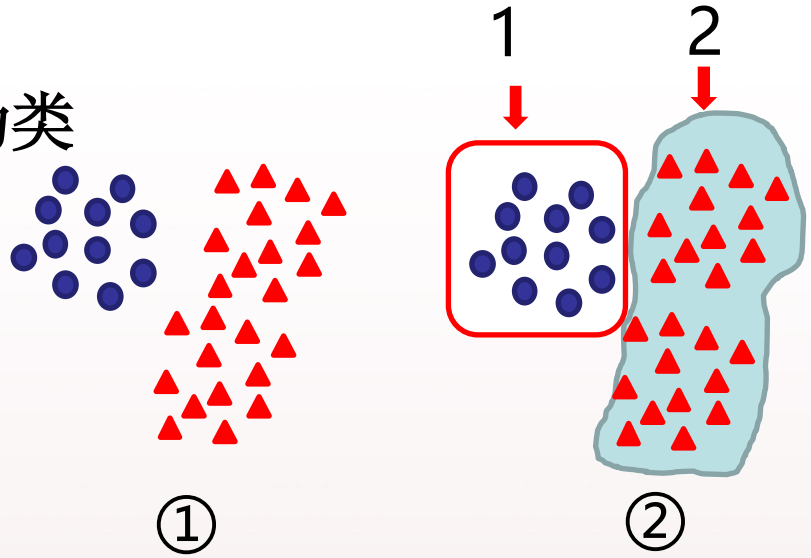
二、分类

2. 二分类

我们先从用蓝色圆形数据定义为类型1，其余数据为类型2；

只需要分类1次

步骤：①->②



二分类

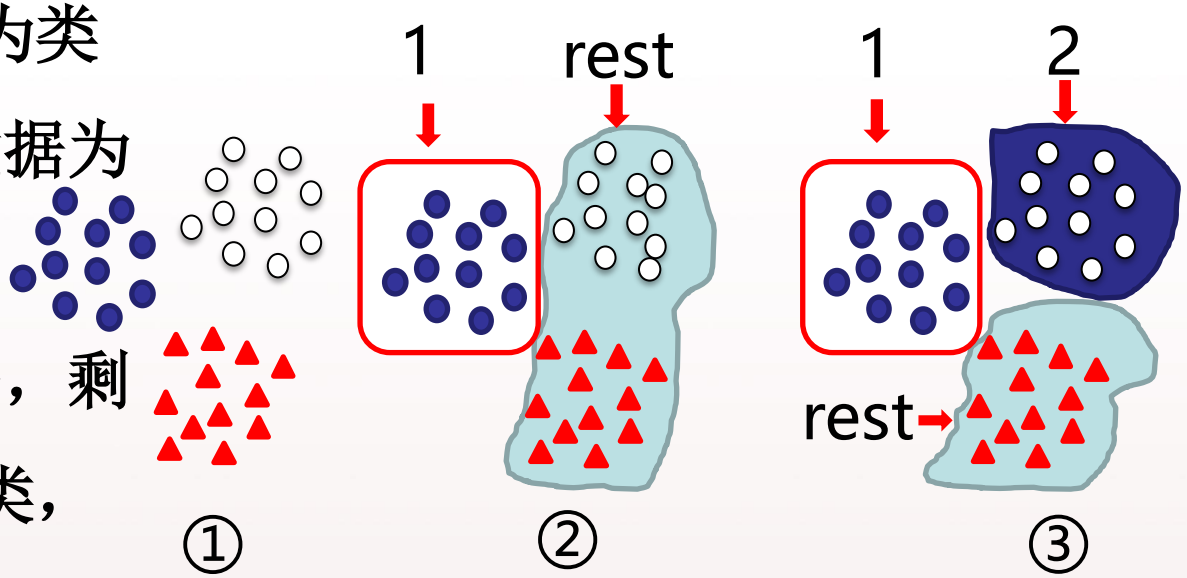
二、分类

3. 多分类

我们先定义其中一类为类型1（正类），其余数据为负类（rest）；

接下来去掉类型1数据，剩余部分再次进行二分类，分成类型2和负类；如果有n类，那就需要分类n-1次

步骤：①->②->③->……



One-vs-All (One-vs-Rest)

一对多 (一对余)

二、分类

4. 经典的分类算法

✓ **K近邻（KNN，K Nearest Neighbors）**

K近邻算法，即是给定一个训练数据集，对新的输入实例，在训练数据集中找到与该实例最邻近的K个实例，这K个实例的多数属于某个类，就把该输入实例分类到这个类中。

✓ **决策树（Decision Tree）**

决策树算法根据历史数据提炼出规则，并以现有信息为基础形成决策。决策树算法通过学习这些样本，得到一个决策树，这个决策树能够对新的数据给出合适的分类。

✓ **朴素贝叶斯（Naive Bayes）**

朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法，使用概率统计的知识对样本数据集进行分类。

二、分类

5. K近邻（KNN）

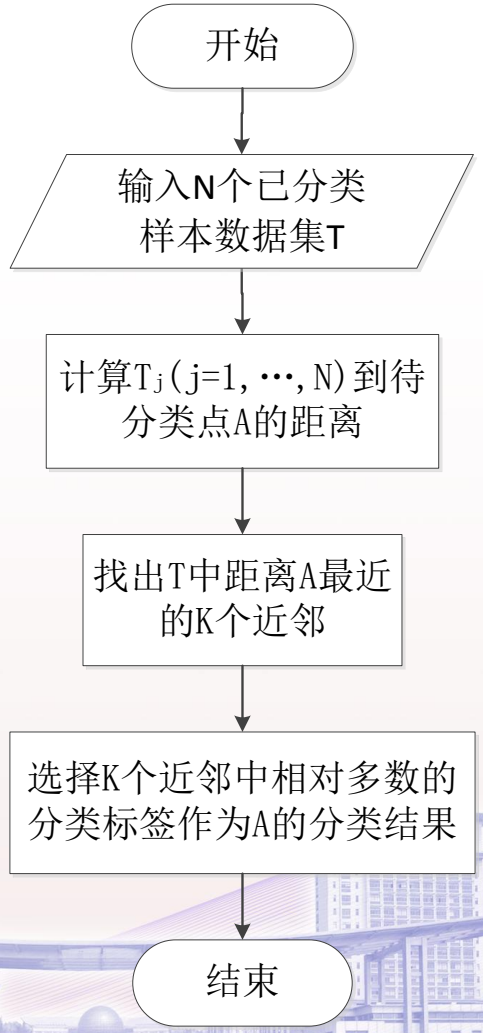
- ✓ K近邻分类算法的逻辑非常直观：待分类的这个数据点归属到哪一类，由它的K个近邻样本点的分类情况决定。
- ✓ 每个样本数据点都有若干个属性，例如一个手机网银用户的信息构成这样一个属性集合：{年龄，学历，收入，……}。
- ✓ 通过某种规则，将这些属性值转换成坐标值，即将用户转换成n维空间中的一个点。为了简单起见，后文阐述以二维空间的点为例。
- ✓ 如图 所示，待分类的数据点为A，图中用圆表示；其他的数据分类结果是已知的（所谓有监督的学习，表示监督者已经分好类，打好标签了），被分为1和2两类，分别用三角形和矩形表示。

二、分类

5. K近邻（KNN）

K近邻算法流程图：

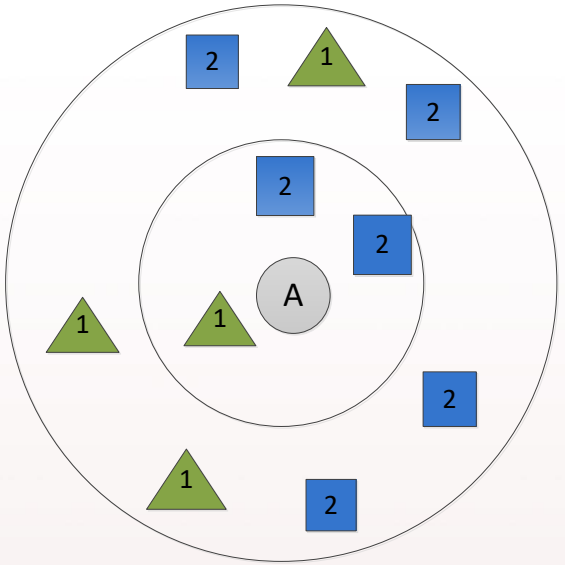
- ✓ K近邻分类算法的逻辑非常直观：待分类的这个数据点归属到哪一类，由它的K个近邻样本点的分类情况决定。
- ✓ 每个样本数据点都有若干个属性，例如一个手机网银用户的信息构成这样一个属性集合：{年龄，学历，收入，……}。
- ✓ 通过某种规则，将这些属性值转换成坐标值，即将用户转换成n维空间中的一个点。为了简单起见，后文阐述以二维空间的点为例。



二、分类

5. K近邻（KNN）

K近邻算法示意图：



例如，待分类的数据点为A，图中用圆表示；其他的数据分类结果是已知的（所谓有监督的学习，表示监督者已经分好类，打好标签了），被分为1和2两类，分别用三角形和矩形表示。

- ✓ A点的分类结果可以简单地选择K近邻中相对多数的分类标签，例如，图 中A的3个近邻中有2个分类结果为2，占据多数，因此A的分类结果为2。
- ✓ 也可以相对复杂地按照距离远近进行权重投票，图 中分类标签为1的近邻虽然只有1个，但是它离A最近，如果权重比例足够大，也有可能影响到A的最终分类结果。

二、分类

5. K近邻（KNN）

K近邻算法Python实现：

K近邻算法有比较直观的解释（特别是在低维空间中），其欧氏距离的计算量也相对比较小，是一个应用很广的基础分类算法。

- ✓ **下面代码首先新建了两个数据点testX和testY，再分别调用kNNClassify函数计算其分类归属。**

(1)定义一个用于计算欧氏距离的函数euclDistance，使用numpy中的函数计算由列表形式存储的两个向量的欧式距离。

```
import pandas as pd
import numpy as np

import matplotlib as mpl
import matplotlib.pyplot as plt

# 计算欧式距离，即两点间的直线距离
#参数：vector1-List列表，n维属性坐标值构成的向量
#      vector2-List列表，n维属性坐标值构成的向量
#返回值：浮点数，欧式距离
def euclDistance(vector1, vector2):
    return np.sqrt(np.sum(np.power(vector2 - vector1, 2)))
```

二、分类

5. K近邻（KNN）

K近邻算法Python实现：

(2)创建一个数据集，包含2个类别共8个样本：

```
# 创建一个数据集，包含2个类别共8个样本
def createDataSet():
# 生成一个矩阵，每行表示一个样本
    group = np.array([[1.0,0.9], [1.0,1.0], [0.8,0.9], [0.6,0.65],
    [0.1, 0.2], [0.3,0.4], [0.2,0.3], [0.0, 0.1]])
# 监督学习，手工设置8个样本所属的类别标签
    labels = ['A', 'A','A', 'A', 'B', 'B','B', 'B']
    return group, labels
```

二、分类

5. K近邻（KNN）

K近邻算法Python实现：

(3) KNN分类算法函数实现：定义一个kNNClassify函数。根据dataSet和labels的输入，选择待分类点newInput的k个近邻，决定其分类归属。

```
# KNN分类算法函数实现
#参数：newInput-List列表，待分类的数据点
#      dataSet-List列表，已分类点坐标
#      lables-List列表，分类标签
#      k-整数，近邻数量
#返回值：maxIndex-字符，分类结果
def kNNClassify(newInput, dataSet, labels, k):
    numSamples = dataSet.shape[0]   # shape[0]表示行数
    distance = []
#计算newInput与dataSet中个点的距离，放入distance列表内
    for vec in dataSet:
        distance.append(euclDistance(newInput,vec))
#对距离排序
    sortedDistIndices = np.argsort(distance)
    classCount = {}
#选择k个最近邻
    for i in range(k):
        voteLabel = labels[sortedDistIndices[i]]
        classCount[voteLabel] = classCount.get(voteLabel, 0) + 1
    maxCount = 0
    for key, value in classCount.items():
        if value > maxCount:
            maxCount = value
            maxIndex = key
    return maxIndex
```

二、分类

5. K近邻（KNN）

K近邻算法Python实现：

(4)生成数据集和类别标签：

```
# 生成数据集和类别标签
dataSet, labels = createDataSet()
#K取值3，调用K近邻分类算法
k = 3
#对testX进行分类
testX = np.array([1.2, 1.0])
outputLabel = kNNClassify(testX, dataSet, labels, 3)
print( "Your input is:", testX, "and classified to class: ", outputLabel)
#对testY进行分类
testY = np.array([0.1, 0.3])
outputLabel = kNNClassify(testY, dataSet, labels, 3)
print("Your input is:", testY, "and classified to class: ", outputLabel)
```

```
Your input is: [1.2 1. ] and classified to class:  A
Your input is: [0.1 0.3] and classified to class:  B
```

二、分类

6. 贝叶斯（Naive Bayes）

贝叶斯（Naive Bayes）分类算法是一种有监督的分类算法，以坚实的数学理论（即贝叶斯公式）作为支撑，实现简单，在大量样本下有较好的表现。

要理解贝叶斯公式，需要先明确条件概率、全概率、先验概率和后验概率的定义。另外，因为条件概率、全概率的相关计算需要被研究的事件满足相互独立的前提条件，所以在算法前面加上“朴素”两个字。

- ✓ 条件概率:指在B事件发生的前提下，A事件发生的可能性，记为：

$$P(A|B) = \frac{P(AB)}{P(B)}.$$

例子：统计结果表明，每年52周，其中有40周因为学生要上课，家长周一早上接送，会导致这40周有80%的可能性发生早高峰拥堵。如果用A表示周一早上堵车，B1表示这个周一处在学期中，那么$P(A|B1) = 0.8$，就表示了学期中周一早上堵车的概率是80%；而剩下的寒暑假期间的12周的周一，发生早高峰拥堵的可能性仅有30%，用B2表示这个周一处在寒暑假中，那么$P(A|B2) = 0.3$。

二、分类

6. 贝叶斯（Naive Bayes）

- ✓ 如果将全年所有的周一划分成两部分：B1表示处在学期中的周一和B2表示处在假期中的周一，就可以完全覆盖所有的周一。任意一个周一处在学期中的概率$P(B1)=\frac{40}{52}$，处在假期中的概率$P(B2)=\frac{12}{52}$。可以算出周一堵车事件发生的概率的堵车概率$P(A)=\frac{40}{52}*80\%+\frac{12}{52}*30\%=68.46\%$。这个概率称为全概率。其数学定义为：设$\Omega$是样本空间，B1，B2，……Bn是样本$\Omega$的一个划分，全概率$P(A)=\sum_{j=1}^{n}P(B_j)P(A|B_j)$。
- ✓ “学期中周一早上堵车的概率是80%”和“寒暑假周一早上堵车的概率是30%”这两个结论，是根据以往数据统计或者是分析得到的概率，称之为先验概率。最简单的掷骰子实验，掷一次，得到1~6中任意一个数的概率是相等的。那么掷之前就可以判定，得到数字1的概率是1/6。

二、分类

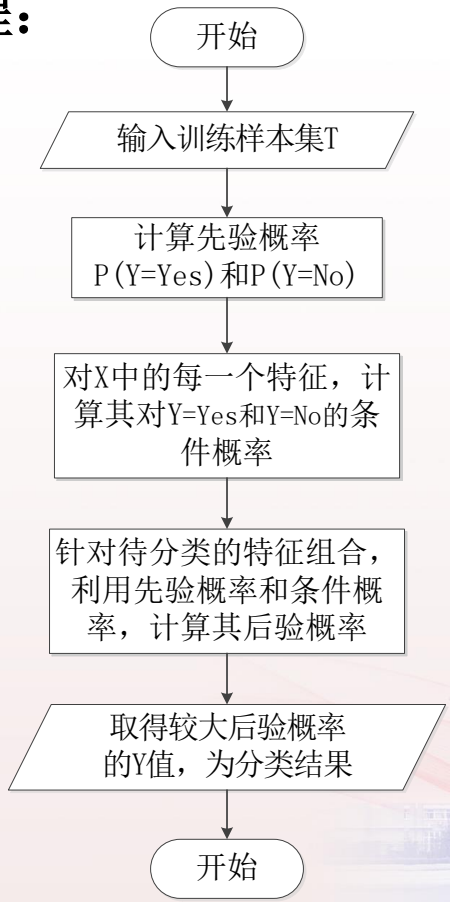
6. 贝叶斯（Naive Bayes）

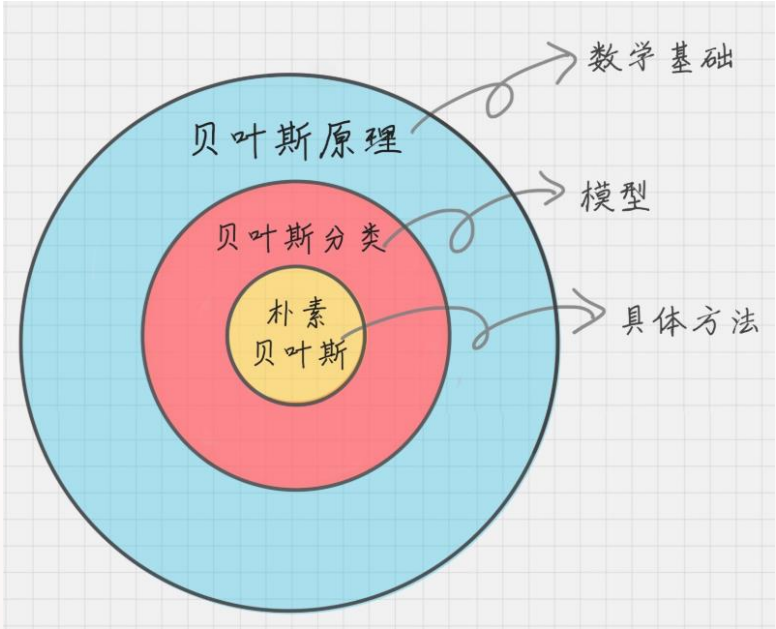
- ✓ 后验概率是指在得到“结果”的信息后重新修正的概率，后验概率的计算要以先验概率为基础。从计算角度看事情还没有发生，要求这件事情发生的可能性的大小，是先验概率。事情已经发生，要求这件事情发生的原因是由某个因素引起的可能性的大小，是后验概率。回到上面周一堵车的例子，根据历史数据，$P(B1), P(A|B1), P(B2), P(A|B2)$均为已知的先验概率，如果某个周一早上出行，发现堵车，要估算这个周一是处在学期中的可能性$P(B1|A)$，就是一个后验概率的问题。$P(B1|A) = \frac{\text{学期中周一堵车概率}}{\text{任意周一堵车概率}} = \frac{\frac{40}{52}*80\%}{\frac{40}{52}*80\%+\frac{12}{52}*30\%} = 89.89\%$。这个结论和经验相符：既然学期中的周一容易堵车，那么当某一个周一堵车时，它处在学期中的概率也较大。后验概率计算的数学定义为：设Ω是样本空间，B1，B2，……Bn是样本Ω的一个划分，A是一个事件，而且P(A)>0，$P(B_i)>0$，i=1，2，……n，则 $P(B_i|A) = \frac{P(B_iA)}{P(A)} = \frac{P(A|B_i)P(B_i)}{\sum_{j=1}^{n} P(A|B_j)P(B_j)}$。这个公式，又称为贝叶斯公式。它实现了先验概率和后验概率之间的相互演算。

二、分类

6. 贝叶斯(Naive Bayes)

贝叶斯分类流程：





二、分类

6. 贝叶斯（Naive Bayes）

Python实现朴素贝叶斯算法:

某银行打算对客户进行某项理财产品促销。类似产品的促销记录表明：促销成功与否和客户的工龄、工作性质、教育程度和婚姻状况等因素有关。促销记录如下表所示:

ID	工龄	工作性质	教育程度	婚姻状况	销售结果
10001	1	2	1	1	N
10002	1	2	3	1	N
10003	1	3	4	1	Y
10004	4	2	2	1	Y
10005	1	1	3	2	Y
10006	1	1	3	1	N
10007	1	3	3	1	N
10008	4	3	2	2	Y
			.....		
10132	1	3	1	1	N

二、分类

6. 贝叶斯（Naive Bayes）

Python实现朴素贝叶斯算法:

(1) 数据预处理：

其中ID列为客户识别号，其他列的取值规则如下：

- ✓ **工龄：小于3年取值为1；3-5年取值为1；6-10年取值为2；大于10年取值为3。**
- ✓ **工作性质：机关事业单位取值为1；国企取值为2；私营企业取值为3。**
- ✓ **教育程度：高中及以下取值为1；大专取值为2；本科取值为3；研究生取值为4。**
- ✓ **婚姻状况：未婚取值为1；已婚取值为2。**
- ✓ **销售结果：该客户未购买产品记为N；购买产品记为Y。**

要求通过以上数据集构造客户购买产品的分类模型，从而有针对性地对其他客户进行产品促销。

二、分类

6. 贝叶斯（Naive Bayes）

Python实现朴素贝叶斯算法:

(2) 数据读取：

import numpy as np

import pandas as pd

```
import numpy as np
import pandas as pd

#读取excel文件，将原来二值表格转换为2维列表，用于后续处理
#参数：fname-excel文件名
#返回值：data2Dlist-2维列表
def loadExcel(fname):
    data2Dlist=[]
    pd1 = pd.read_excel(fname,'Sheet1')
    print('读入的数据文件首5行：')
    print(pd1.head())
    for index ,row in pd1.iterrows():
        list1 = row.tolist()
        list1.pop(0)
        data2Dlist.append(list1)
#显示2维列表前5项
    print('转化后的2维列表前5项：')
    print(data2Dlist[:4])
#返回该2维列表
    return data2Dlist

BMData=loadExcel('bankMarketing_info.xlsx')
BMLabel= ['工龄', '工作性质', '教育程度', '婚姻状况']
print(BMLabel)
```

```
读入的数据文件首5行：
    客户ID  工龄  岗位性质  教育程度  婚姻状态 销售结果
0  10001   1     2     1     1    N
1  10002   1     2     3     1    N
2  10003   1     3     4     1    Y
3  10004   4     2     2     1    Y
4  10005   1     1     3     2    Y
转化后的2维列表前5项：
[[1, 2, 1, 1, 'N'], [1, 2, 3, 1, 'N'], [1, 3, 4, 1, 'Y'], [4, 2, 2, 1, 'Y']]
['工龄', '工作性质', '教育程度', '婚姻状况']
```

二、分类

6. 贝叶斯（Naive Bayes）

Python实现朴素贝叶斯算法:

(3) 构建贝叶斯预测函数:

```
#利用后验概率计算先验概率
#参数：dataset-List列表，训练集，包含了样本数据和分类结果
#      test-List列表，测试样本属性列表
#      cls_y,cls_n-字符，分类标签Y，N
#返回值：{cls_y:PY,cls_n:PN}-浮点数，先验概率公式1-3的分子部分
def NB(dataset,test,cls_y,cls_n):
    PY = Prob(dataset,cls_y)
    PN = Prob(dataset,cls_n)
#对于测试样本test属性取值，计算其公式1-3的分子部分
#较大的值对应较大的分类可能
    for i, val in enumerate(test):
        PY *= ConditionP(dataset,cls_y,i,val)
        PN *= ConditionP(dataset,cls_n,i,val)
    return {cls_y:PY,cls_n:PN}
```

二、分类

6. 贝叶斯（Naive Bayes）

Python实现朴素贝叶斯算法:

(4) 测试预测结果：

两个测试样本（客户）分别为[2,2,3,2]、[1,3,1,1]，物理含义为：

客户1的工龄：6-10年，工作单位性质：国企，教育程度：本科，婚姻状况：已婚；

客户2的工龄：小于3年，工作单位性质：私企，教育程度：高中及以下，婚姻状况：未婚。通过朴素贝叶斯分类算法进行预测：

```
print('理财产品促销客户1预测结果:')
BMVec = [2, 2, 3, 2]
prob=NB(BMData,BMVec,'Y','N')
print(prob)

print('理财产品促销客户2预测结果:')
BMVec = [1,3,1,1]
prob=NB(BMData,BMVec,'Y','N')
print(prob)
```

```
理财产品促销客户1预测结果:
{'N': 0.0, 'Y': 0.000943}
理财产品促销客户2预测结果:
{'N': 0.045776, 'Y': 0.002475}
```

谢谢观赏 下节课见