第 4 章 概率论

概率论是研究随机现象统计规律性的一门学科，为数据建模和处理不确定性提供了强大的分析工具。

4.1 概率论基础

假设有某种随机试验 E（例如掷硬币、掷骰子），实验产生一组固定可能的结果集合，此集合称为随机试验 E 的样本空间，记为 Ω={ω}，其中每一个结果 ω 是 Ω 中的一个元素，称为样本点。

随机试验 E 的一些样本点构成的集合称为随机事件。单个样本点构成的事件称为基本事件。如果将 Ω 也看作事件，则每次试验中，Ω 总会发生，通常称样本空间 Ω 为必然事件。空集∅是样本空间样本空间的子集，不包含任何样本点，称为不可能事件。

定义 4-1（概率）：给定随机试验 E 及其样本空间 Ω，对于 E 的每一事件 A 赋予一个实数，记为 P(A)，如果集合函数满足以下条件，P(A)称为事件 A 的概率

(1) 非负性：对于每一个事件 A，P(A)≥0。

(2) 规范性：对于必然事件 Ω，P(Ω)=1。

(3) 可列可加性：设$A_1, A_2, \ldots$是两两不相容的事件，即对于$A_iA_j = \emptyset, i \neq j\ (i, j = 1,2, \ldots)$

$$P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$$

性质：

(1) P(∅)=0

(2) 有限可加性: 若$A_1, A_2, \ldots, A_n$是两两互不相容，则有

$$P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$$

(3) 事件 A,B,若 A⊂B,则 P(B–A)=P(B)-P(A),P(B)≥P(A)

(4) 对任一事件 A, P(A)≤1

(5) 逆事件的概率: 对任一事件 A,$P(\bar{A}) = 1 - P(A)$

(6) 加法公式: 对任意事件 A, B, 有P$(A \cup B) = P(A) + P(B) - P(AB)$

(7) Boole 不等式：对于可数的事件$A_1, A_2, \cdots, A_n$，无论是否相交，都有

$$P\left(\bigcup_i A_i\right) \leq \sum_i P(A_i)$$

此不等式也称为联合界，常用于估计和事件的概率上界。

定义 4-2（条件概率）： 设 P(B)>0，给定事件 B 已经发生的条件下,事件 A 发生的条件概率记为 P(A|B)，定义为

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

定义 4-3（乘法公式/链式法则）： 设 P(A)＞0，则$P(AB) = P(A)P(B|A)$

推广至多个事件的积事件

$$P(ABC) = P(A)P(B|A)P(C|AB)$$

$$P(A_1A_2 \ldots A_n) = P(A_1)P(A_2|A_1)P(A_3|A_1A_2) \ldots P(A_n|A_1A_2 \ldots A_{n-1})$$

定义 4-4（贝叶斯公式）： 设 P(B)>0，给定事件 B 已经发生的条件下,事件 A 发生的后验概率为

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

有时可以把公式表示成以下形式

$$P(A|B) \propto P(A)P(B|A)$$

其中，P(A)称为先验概率，P(A|B)是后验概率，P(B|A)称为似然概率。

设随机试验的样本空间为$S = \{e\}$。$X = X(e)$是定义在样本空间 S 上的实值单值函数，则称$X = X(e)$为随机变量（Random Variable）。

定义 4-5（分布函数）： 设 X 是一个随机变量，x 是任意实数，函数

$$F(x) = P(X \leq x), \quad -\infty < x < \infty$$

称为 X 的分布函数，也称累积分布函数（Cumulative Distribution Function, CDF），确定了随机变量的数值落在小于等于 x 一侧的概率。

已知随机变量 X 的分布函数，则可知 X 落在任意区间$(x_1, x_2]$ 的概率：

$$P(x_1 < X \leq x_2) = P(X \leq x_2) - P(X \leq x_1) = F(x_2) - F(x_1)$$

常见的随机变量通常分为离散型随机变量和连续型随机变量两大类。离散型随机变量全部可能取到的值是有限个或可列无限个。连续型随机变量的取值为无限个（或数值无法一一列出）。还有些随机变量既不是离散的也不是连续的，这些随机变量需要测度论知识来分析，不在本书讨论范围内。

离散型随机变量使用分布律来描述它的概率分布，即给出离散型随机变量的全部取值及每个值的概率。常见的离散型随机变量的分布有：0-1 分布、几何分布、二项分布、泊松分布等。

连续型随机变量的概率分布常用概率密度来描述。

定义 4-6（概率密度）：设 X 是随机变量，其分布函数为$F(x)$，如果存在非负函数$f(t)$，使得对于任意实数 x 有

$$F(x) = \int_{-\infty}^{x} f(t)\mathrm{d}t$$

则称 X 为连续型随机变量，其中函数$f(x)$称为 X 的概率密度函数，简称概率密度。

常见的连续型随机变量的分布有：高斯分布、均匀分布、指数分布、χ^2分布、t 分布、F 分布等。

随机变量的数学期望（Expectation）是随机实验中每次可能结果的概率乘以其结果的总和。期望是随机变量最基本的数学特征之一，反映随机变量的平均值。

定义 4-7（数学期望）：假设 X 是一个离散型随机变量，可能的取值有$x_1, x_2, \cdots, x_n$，各取值对应的概率取值为$P(x_k), k = 1,2, \cdots, n$。如果级数$\sum_{k=1}^{n} x_k\, P(x_k)$绝对收敛，则随机变量 X 的数学期望定义为：

$$E(X) = \sum_{k=1}^{n} x_k\, P(x_k)$$

如果 X 是连续型随机变量，概率密度函数为$f(x)$。若积分$\int_{-\infty}^{+\infty} xf(x)dx$绝对收敛，则随机变量 X 的数学期望定义为：

$$\mathrm{E(X)} = \int_{-\infty}^{+\infty} xf(x)\mathrm{d}x$$

随机变量的方差（Variance）用来衡量随机变量取值的偏离程度。

定义 4-8（方差）：设 X 是一个随机变量，若$E\{[X-E(X)]^2\}$存在，则称$E\{[X-E(X)]^2\}$为 X 的方差，记为 D(X)或 DX 或 Var(X),即

$$D(X) = E\{[X-E(X)]^2\}$$

设离散型随机变量 X 的分布律为$P(X=x_k)=p_k\ (k=1,2,3,\cdots)$, 则 X 的方差为

$$D(X) = \sum_k [x_k - E(X)]^2\, p_k$$

对于连续型随机变量，设随机变量 X 的概率密度为$f(x)$，则 X 的方差为

$$D(X) = \int_{-\infty}^{+\infty} [x-E(X)]^2\, f(x)\mathrm{d}x$$

此外，计算方差可采用以下简便计算公式：

$$D(X) = E(X^2) - [E(X)]^2$$

二维随机变量(X,Y)除讨论 X 与 Y 的期望和方差外，通常还需讨论 X 与 Y 之间关系。

定义 4-9（协方差）：量$E\{[X-E(X))][Y-E(Y)]\}$称为随机变量 X 与 Y 的协方差，记为$Cov(X,Y)$

$$Cov(X,Y) = E\{[X-E(X))][Y-E(Y)]\}$$

实际中可以采用下式计算：$Cov(X,Y) = E(XY) - E(X)E(Y)$

定义 4-10（协方差矩阵）：n 维随机变量$(X_1, X_2, \cdots, X_n)$的协方差矩阵是

$$\Sigma_n = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix}$$

其中$c_{ij} = cov(X_i, X_j) = E\{[X_i - E(X_i)][X_j - E(X_j)]\},\ i,j = 1,2,\cdots,n$

定义 4-11（相关系数）：设 X、Y 是随机变量，若$D(X)>0$，$D(Y)>0$，则

$$\rho_{XY} = \frac{Cov(X,Y)}{\sqrt{D(X)D(Y)}}$$

称为随机变量 X 与 Y 的相关系数，也称为皮尔逊(Pearson)相关系数。

相关系数$|\rho_{XY}| \leq 1$测量两个变量之间的线性相关程度。如果$\rho_{XY} = 0$，两个变量不相关,没有线性关系。 $|\rho_{XY}|$越大，两个变量线性相关程度越好。如果两个变量相互独立，则不相关，但反过来不一定成立。

例 4-* 异常检测：异常检测（或离群值检测）是从事件或观察结果中识别出罕见现象。如何确定数据点是正常还是异常？对于高维数据，无法只查看一个纬度的变量来识别异常，需要综合考虑各个维度的变量组合。常用的方法是首先使用主成分分析（PCA）进行降维，将数据映射到低维空间。实践中会构建数据的协方差矩阵以及计算该矩阵的特征向量。最大特征值（主成分）相对应的特征向量可以用于重构原始数据方差的主要部分。

此后，需要估计数据点是否属于某个分布来确定是否为离群点。由于采样点的分布一般为非正球体，如果采用欧式距离来度量数据点与分布中心的距离，由于数据不同维度的方差不同，会导致估计出现偏差。通常采用马氏距离（Mahalanobis Distance）来规避欧式距离对于数据特征方差不同的风险，使距离更加符合数据分布特征。如果待估计数据点的距离超过某个阈值，则将数据点分类为异常。

向量$\mathbf{x}$到一个样本均值为 μ、样本协方差矩阵为Σ_n的样本分布的马氏距离为：

$$d_{MD} = \sqrt{(\mathbf{x} - \mu)\Sigma_n^{-1}(\mathbf{x} - \mu)'}$$

对于两组数据样本$\mathbf{x} = (x_1, x_2, \cdots, x_n)$和$\mathbf{y} = (y_1, y_2, \cdots, y_n)$，样本协方差为：

$$S(\mathbf{x}, \mathbf{y}) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

其中，$\bar{x}$和$\bar{y}$分别为$\mathbf{x}, \mathbf{y}$的样本均值。

表 4-* 数据样本

#	X（身高）	Y（体重）
1	175	72
2	180	78
3	168	62
4	178	70
5	182	77
6	185	88

对于以上数据集，可以得到样本均值 μ、样本协方差矩阵Σ_n为：

$$\mu = (\bar{x}, \bar{y}) = (178, 74.5)$$

$$\Sigma_n = \begin{bmatrix} S(\mathbf{x},\mathbf{x}) & S(\mathbf{x},\mathbf{y}) \\ S(\mathbf{x},\mathbf{y}) & S(\mathbf{y},\mathbf{y}) \end{bmatrix} = \begin{bmatrix} 35.6 & 48.8 \\ 48.8 & 76.6 \end{bmatrix}$$

4.2 常用随机变量及其分布

1、0-1 分布与二项分布

设随机变量 X 只取 0 与 1 两个值，其分布律为

$$P(X = k) = p^k(1-p)^{1-k} \quad (k = 0,1)$$

则称 X 服从以$p \in (0,1)$为参数的 0-1 分布。0-1 分布也称为伯努利分布，是一次伯努利试验两种可能结果的分布。

如果重复 n 次伯努利试验，各次试验之间相互独立，则事件 A 在 n 重伯努利试验中发生的次数 X$(X = 0,1,2,\cdots,n)$服从二项分布，记为$X \sim B(n,p)$，分布律表达公式为：

$$P(X = k) = C_n^k p^k (1-p)^{n-k}, \qquad k = 0,1,2,\cdots,n$$

2、泊松分布

若随机变量 X 的分布律为

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \qquad k = 0,1,2,\cdots, \qquad \lambda > 0$$

称 X 服从参数为 λ $(\lambda > 0)$的泊松分布，记为X $\sim \pi(\lambda)$。

3、几何分布

若随机变量 X 的分布律为

$$P(X = k) = (1-p)^{k-1}p, \qquad k = 1,2,\cdots \ 0 < p < 1$$

则称 X 服从参数为 p 的几何分布，记为$X \sim G(p)$，表示在独立重复的伯努利试验中，试验了 k 次才得到第一次成功的概率。

4、均匀分布

若连续型随机变量 X 具有概率密度

$$f(x) = \frac{1}{b-a}, \qquad a < x < b$$

则称 X 在区间(a,b)上服从一维均匀分布，记为$X \sim U(a,b)$。X 落在(a,b)区间中任意等长度子区间的概率相同

5、指数分布

若连续型随机变量 X 的概率密度为

$$f(x) = \lambda e^{-\lambda x}, x > 0$$

其中$\lambda > 0$为常数，则称 X 服从参数为λ的指数分布，记为$X \sim E(\lambda)$。

6、高斯分布

高斯分布也称为正态分布，其概率密度函数曲线呈钟型，左右对称。正态分布是一种重要的连续分布，由其均值和协方差矩阵定义。

设列矩阵$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$，$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_n \end{bmatrix} = \begin{bmatrix} E(X_1) \\ E(X_2) \\ \cdots \\ E(X_n) \end{bmatrix}$，则 n 维正态随机变量$(X_1, X_2, \cdots, X_n)$的概率密度是

$$f(x_1, x_2, \cdots, x_n) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma_n)}} \exp\left(-\frac{1}{2}(X-\mu)^T {\Sigma_n}^{-1}(X-\mu)\right)$$

其中，Σ_n为随机变量$(X_1, X_2, \cdots, X_n)$的协方差矩阵

$$\Sigma_n = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix}$$

其中$c_{ij} = cov(X_i, X_j) = E\{[X_i - E(X_i)][X_j - E(X_j)]\},\ i,j = 1,2,\cdots,n$

当 n=1 时是常见的一维高斯分布，其概率密度函数为

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \qquad -\infty < x < \infty$$

一维高斯分布记为$X \sim N(\mu, \sigma^2)$，表示 X 是具有均值μ和方差σ^2的正态分布。

要谨慎使用高斯分布为数据建模，高斯分布的尾部下降非常快。如果一个随机变量实际上是具有拖尾较大的分布，使用高斯分布建模，将会使模型产生扭曲。

另一种具有较大拖尾的分布是高斯混合分布(Gaussian mixture distribution)。多个高斯分布的线性叠加能拟合非常复杂的概率密度函数。通过足够多的高斯分布叠加，并调节它们的均值、协方差矩阵以及线性组合的系数，可以精确地逼近任意连续分布。

设 K 个高斯分布进行线性叠加，得到的高斯混合分布的概率密度函数表示为：

$$P(x) = \sum_{k=1}^{K} \pi_k \, p(\mathbf{x}|\mu_k, \Sigma_k)$$

其中$p(\mathbf{x}|\mu_k, \Sigma_k)$表示参数为$\mu_k, \Sigma_k$的高斯分布的概率密度，是混合分布的一个子模型(Component)，其均值为μ_k，协方差矩阵为Σ_k。参数π_k是模型的混合系数(Mixing Coefficients)，满足$\sum_{k=1}^{K} \pi_k = 1$和$0 \le \pi_k \le 1$。图 4-*是一个简单示例，表明一个复杂的分布可以分解为多个高斯分布的组合。

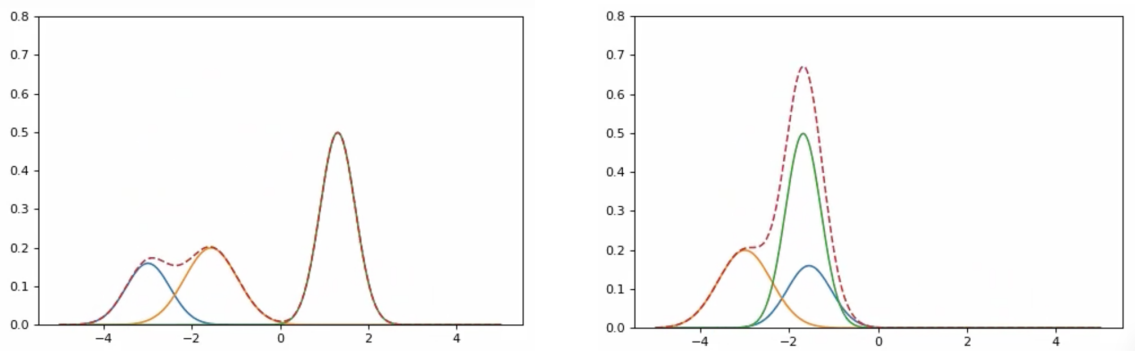
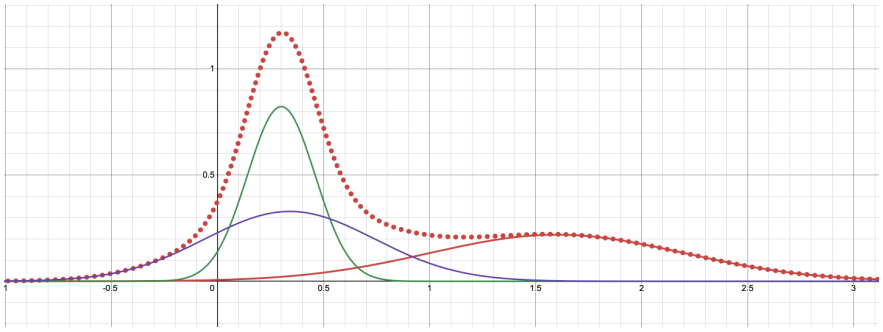


图 4 复杂的分布可以分解为多个高斯分布的组合，其中，点线表示的曲线可以分解为三个高斯分布之和。



$$Y = \frac{1}{3}N(1.6,0.6^2) + \frac{1}{3}N(0.3,0.16^2) + \frac{1}{3}N(0.34,0.4^2)$$

7、统计量及抽样分布

样本是进行统计推断的依据。在实际应用中，通常需要针对具体问题对样本值进行整理和加工，构造出适当的样本的函数(即统计量)，利用这些函数来进行统计推断，揭示总体的统计特性。

设 $X_1, X_2, \ldots, X_n$ 是来自总体 X 的一个样本，则样本均值和样本方差分别为：

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} X_i^2 - n\bar{X}^2\right)$$

χ^2分布：设 $X_1, X_2, \ldots, X_n$ 是来自总体 N(0,1)的样本，则称统计量

$$\boldsymbol{\chi^2 = X_1^2 + X_2^2 + \cdots + X_n^2}$$

服从自由的为 n 的$\boldsymbol{\chi^2}$分布，记为$\boldsymbol{\chi^2 \sim \chi^2(n)}$。

t 分布：设$X \sim N(0,1)$，$\boldsymbol{Y \sim \chi^2(n)}$，且 X 和 Y 相互独立，则称随机变量

$$\boldsymbol{t = \frac{X}{\sqrt{Y/n}}}$$

服从自由度为 n 的 t 分布，记为$\boldsymbol{t \sim t(n)}$。当 n 较小时，t 分布与 $N(0, 1)$分布相差较大；但 n 充分大时，t 分布可以用 $N(0, 1)$分布近似。

F 分布：设$\boldsymbol{U \sim \chi^2(n_1)}$，$\boldsymbol{V \sim \chi^2(n_2)}$，且 U 和 V 相互独立，则称随机变量

$$F = \frac{U/n_1}{V/n_2}$$

服从自由度为$\boldsymbol{(n_1, n_2)}$的 F 分布，记为$\boldsymbol{F \sim F(n_1, n_2)}$。

χ^2分布、t 分布、F 分布通常称为统计三大分布。

4.3 最大似然估计

最大似然估计在模型已知、参数未知的情况下，通过观测数据估计未知参数。

最大似然估计的求解思路是：给定一组样本值后，寻找未知参数$\theta_1, \theta_2, \cdots, \theta_n$的估计量，使得观察到样本数据的可能性最大。其求解步骤如下：

1、写出似然函数$L(\theta_1, \theta_2, \cdots, \theta_n)$

$$L(\theta_1, \theta_2, \cdots, \theta_n) = \begin{cases} \prod_{i=1}^{n} p(x_i; \theta_1, \theta_2, \cdots, \theta_n) \\ \prod_{i=1}^{n} f(x_i; \theta_1, \theta_2, \cdots, \theta_n) \end{cases}$$

2、对似然求极值

3、解出参数的估计量

例 4-* 设随机变量$X \sim N(\mu, \sigma^2)$，其中μ, σ^2未知，$x_1, x_2, \cdots, x_n$是一组观察得到的样本值，求参数μ, σ^2的最大似然估计量。

解：给定观察值$x_1, x_2, \cdots, x_n$，似然函数为：

$$\mathrm{L}(\mu, \sigma^2) = \prod_{i=1}^{n} f(x_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} exp\left\{-\frac{(x_i - \mu)^2}{2\sigma^2}\right\}$$

对似然函数取对数进行简化，

$$ln\, L(\mu, \sigma^2) = n\, ln \frac{1}{\sqrt{2\pi}} - \frac{n}{2} ln\, \sigma^2 - \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{2\sigma^2}$$

接下来求μ, σ^2取什么值的时候，函数$ln\, L(\mu, \sigma^2)$达到最大值。

令

$$\frac{\partial\, ln\, L\,(\mu, \sigma^2)}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{n}(x_i - \mu) = 0$$

$$\frac{\partial\, ln\, L\,(\mu, \sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{2\sigma^2} = 0$$

可得参数μ, σ^2的最大似然估计量为

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i = \overline{X}$$

$$\widehat{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})$$

例 4-* **（本地化差分隐私保护）**随机响应技术(randomized response)是本地化差分隐私保护的一种方法。每个用户首先对数据进行隐私化处理，再将处理后的数据发送给数据收集者。数据收集者对采集到的数据进行统计，得到有效的分析结果，同时不会泄漏个体的隐私信息。

假设有 n 个用户，希望统计艾滋病患者的。于是发起一个敏感的问卷调查：“你是否为艾滋病患者?”每个用户对此进行响应。用户出于隐私考虑，不会直接回答真实答

案，而是借助于一枚非均匀的硬币来给出答案。假设硬币正面向上的概率为 p，反面向上的概率为 1-p。用户秘密抛出该硬币，若正面向上，则回答真实答案，反面向上，则回答相反的答案。

利用上述扰动方法对个体的回答进行统计，可以得到艾滋病患者人数的真实统计值。

首先，进行扰动性统计。设第 i 个用户的答案X_i为“Yes”或“No”，统计结果中，回答“Yes”的人数为n_1，则回答“No”的人数为$n-n_1$。显然，按照上述统计，用户回答“Yes”和“No”的概率如下：

$$P(X_i = \text{"Yes"}) = \pi p + (1-\pi)(1-p)$$
$$P(X_i = \text{"No"}) = (1-\pi)p + \pi(1-p)$$

接着,利用收到的统计数据对比例π进行估计。采用最大似然估计方法，构建以下似然函数:

$$L(\pi) = [\pi p + (1-\pi)(1-p)]^{n_1}[(1-\pi)p + \pi(1-p)]^{n-n_1}$$

对上式求导并令导数为零，可得π的最大似然估计量为：

$$\hat{\pi} = \frac{p-1}{2p-1} + \frac{n_1}{(2p-1)n}$$

以下关于$\hat{\pi}$的数学期望保证了$\hat{\pi}$是真实分布π的无偏估计:

$$E(\hat{\pi}) = \frac{1}{2p-1}\left[p - 1 + \frac{1}{n}\sum_{i=1}^{n} X_i\right] = \frac{1}{2p-1}[p - 1 + \pi p + (1-\pi)(1-p)] = \pi$$

由此可以统计得到的艾滋病人数 N 的估计值为：

$$N = \hat{\pi} \times n = \frac{p-1}{2p-1} \times n + \frac{n_1}{2p-1}$$

综上所述,根据总人数 n、回答“yes”的人数n_1和扰动概率 p,即可得到真实患病人数的统计值，且这个值是真实值的无偏估计。

4.4 散列法与 bloom 过滤器

4.4.1 生日悖论

假设教室里有 30 个同学，有两位同学生日相同的概率大，还是没有两人生日相同的概率大？

由球和箱子模型可知，30 人全部生日不同的概率是

$$\frac{365}{365}\cdot\frac{364}{365}\cdots\frac{336}{365}=\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\cdots\left(1-\frac{29}{365}\right)\approx 0.2937$$

所以，有两个人生日相同的概率为

$$1-\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\cdots\left(1-\frac{29}{365}\right)\approx 0.7$$

类似可以知，如果教室里有 23 名同学，有两人生日相同的概率超过 0.5。

另外，在有 n 个学生的教室里，至少存在一个同学与 Alice 生日相同的概率是

$$P=1-\left(\frac{364}{365}\right)^{n}$$

当$n=84$时，$P\approx 0.5$。此时所需学生的人数显著多于任意两个同学生日相同的概率。这一反直觉的数学事实常称为生日悖论。

一般地，如果有 m 个人，有 n 个可能的生日，那么所有 m 人生日不同的概率为：

$$\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{m-1}{n}\right)=\prod_{i=1}^{m-1}\left(1-\frac{i}{n}\right)$$

当 k 与 n 相比较小时，即$k<<n$时，$1-\frac{k}{n}\approx e^{-\frac{k}{n}}$。因此，如果 m 相对 n 较小，上式的乘积则可以近似为如下：

$$\prod_{i=1}^{m-1}\left(1-\frac{i}{n}\right)\approx\prod_{i=1}^{m-1}e^{-\frac{i}{n}}=exp\left\{-\sum_{i=1}^{m-1}\frac{i}{n}\right\}=exp\left\{-\frac{m(m-1)}{2n}\right\}\approx exp\left\{-\frac{m^2}{2n}\right\}$$

因此，如果要使所以 m 个人有不同生日的概率为 1/2，则 m 的值可近似为：

$$m=\sqrt{2n\,ln\,2}$$

例如，当 n=365，m=22.49。

生日悖论常用于对密码算法的生日攻击中，如利用生日悖论求解离散对数问题。

例 4-*（生日攻击）给定单向函数$y=g^{x}\bmod p$ (p 为素数), 已知 y,g,p 求 x。利用穷举比较方法，需要最多 p-1 次运算能够找到。如果 p 很大，如$p=2^{100}$,则在计算上不可行。利用生日悖论，穷举空间可降为$\sqrt{p}=2^{50}$。

生日攻击的步骤如下：

Step 1: 敌手创建两个表 Tab_A 和 Tab_B；

Step 2: 随机生成一个 1～p-1 之间的整数 a，计算$A = g^a \bmod p$，并将 a 和 A 填入 Tab_A;

Step 3: 随机生成一个 1～p-1 之间的整数 b，计算$B = yg^b \bmod p$，并将 b 和 B 填入 Tab_B;

Step 4: 重复 Step 2 和 Step 3，直到$n = \lfloor\sqrt{p}\rfloor$次；

Step 5: 在表 Tab_A 和 Tab_B 中找重。如果找到 Tab_A 中的某个 A 与 Tab_B 中的某个 B 相等，则记下相对应的 a 和 b；如果没有找到，则回到 Step1；

Step 6: 计算$x = a - b \bmod (p-1)$，即为解。

根据生日悖论，随机$\lfloor\sqrt{p}\rfloor$个 a 和随机$\lfloor\sqrt{p}\rfloor$个 b，出现 A=B 的概率约为 50%，在表 Tab_A 和 Tab_B 中有一半的概率找到重项。再由下式可以计算得到 x：

$$g^a = y \cdot g^b (\bmod p) = g^x \cdot g^b (\bmod p) = g^{x+b} (\bmod p)$$

即有：

$$a = x + b\big(\bmod \Phi(p)\big) = x + b\big(\bmod (p-1)\big)$$

4.4.2 散列法

假设需要设计一个口令检查程序，保存一个不能接受的口令字典集合，存储容易破译的口令。当用户建立口令时，程序核查口令是否在此集合中。

方法 1 直接存储: 按字母顺序存储不能接受的口令，并对字典进行二元搜索，检查用户口令是否不能接受。对于 n 个单词，采用二分搜索，时间复杂度为$O(log_2 n)$。

方法 2 散列存储: 存储每个口令的散列值。

假设口令为 8 个 ASCII 字符，长度为 64 比特，用散列函数将每个口令映射到一个 32 比特的二进制串，称为信息指纹。然后将指纹保存在一个排序表中。检查口令时，首先计算口令的指纹，并在表中查找。如果指纹在表中，则判断该口令不能接受。

散列存储与直接存储相比，优势是降低了存储开销，但却可能误判，即将不属于口令字典中的口令误判为属于口令字典，称为假阳性。如图 1 所示，不合格的口令

88888888 与合格的口令 AB@18#1+映射到同一个散列值，两者的散列值产生碰撞，导致把合格的口令误判为不合格的口令。

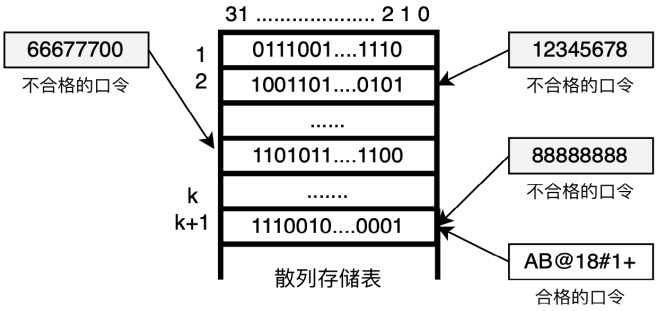


图 1 散列存储

给定不允许的口令字典集合$S = \{S_1, S_2, \cdots, S_n\}$，其中$S_i$是不允许的口令。散列函数将每个口令映射到一个 b 比特的二进制串。对于一个可以接受的合格口令，假阳性的概率为

$$1 - \left(1 - \frac{1}{2^b}\right)^n \geq 1 - e^{-\frac{n}{2^b}}$$

如果希望假阳性的概率小于常数 c，即要求

$$e^{-\frac{n}{2^b}} \geq 1 - c$$

即$b \geq log_2 \frac{n}{ln\left(\frac{1}{1-c}\right)}$。如果$b = 2\,log_2\, n$，则假阳性的概率下降到

$$1 - \left(1 - \frac{1}{n^2}\right)^n < \frac{1}{n}$$

假设口令字典集合有$2^{16} = 65536$个词，用 32 比特作为口令的散列长度，则假阳性的概率小于$1/65535$。

4.4.3 Bloom 过滤器

如果需要过滤垃圾邮件，全世界至少有几十亿个发垃圾邮件的邮箱地址，将这些地址都存储，需要占用大量存储空间。采用散列表，每存储一亿个垃圾邮件地址，需要 1.6GB 的内存（设一个 E-mail 地址对应一个 8 字节的信息指纹），十亿个地址，则需要 16 亿字节的存储空间。下面介绍另一种方法，只需散列表 1/8 到 1/4 的存储空间就能解决同样的问题，并且查询速度更快。

Bloom 过滤器是由伯顿·布隆（Burton Bloom）于 1970 年提出，主要用于检索一个元素是否在一个集合中，其空间效率和查询响应时间都远远超过一般的算法，虽然引入一定的误识率。

Bloom 过滤器由一个固定大小的二进制向量或者位数组和一系列映射函数组成。初始状态时，长度为 m 的位数组的所有位都被置为 0。当有元素加入集合时，通过 K 个映射函数将这个元素映射成位数组中的 K 个位，并置为 1。检索时则检查这些位的值，如果这些位有任何一位为 0，则被检索元素一定不在集合中；如果这些位都是 1，则认为被检索元素属于此集合。

假设存储 1 亿个垃圾邮件地址，先建一个 16 亿比特的全零位数组。对每个垃圾邮件地址，用 8 个不同的随机数产生器（F1,…,F8）产生 8 个信息指纹（f1,…,f8）。再用一个随机数产生器 G 把 8 个信息指纹映射到 1-16 亿中的 8 个自然数 g1,…,g8，把这 8 个位置的比特全部置为 1（图 2）。

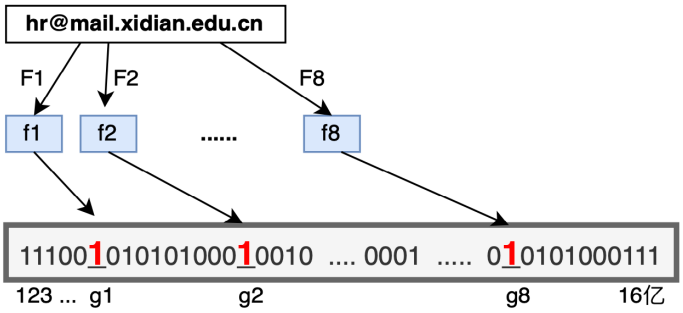


图 2、Bloom 过滤器

查询某个变量是否在集合中时，只需查看位数组相同位置的位是不是都是 1 就可以大概率知道集合中有没有它。如果这些位有任何一个 0，则被查询元素一定不在集合中；如果都是 1，则被查询元素很可能存在集合中，也存在一定的概率不在集合中。因为映射函数本身就是散列函数，散列函数可能会产生碰撞，位数组中相同的位可能被多次映射且置为 1。

假设 Bloom 过滤器的位数组长为 m 比特，用于存储 n 个元素的集合，每个元素对应 k 个信息指纹的散列函数。下面估计 Bloom 过滤器的误识率。

对于过滤器中的某一特定位置，如果第一个加入集合的元素的 k 个散列函数都没有把它置为 1 的概率是$\left(1-\frac{1}{m}\right)^{k}$。如果插入了 n 个元素，都没有把这个位置设置成 1，概率是$\left(1-\frac{1}{m}\right)^{nk}$。反过来，过滤器中的一个比特在插入 n 个元素后，被设置成 1 的概率是$1-\left(1-\frac{1}{m}\right)^{nk}$。

假设已经把 n 个元素存储到过滤器中，检验一个元素是否在集合中。一个不在集合中的元素被误认为在集合中，需要所有的散列函数对应的比特位均为 1，其误识率 P 为

$$P=\left[1-\left(1-\frac{1}{m}\right)^{nk}\right]^{k}\approx\left(1-e^{-\frac{kn}{m}}\right)^{k}$$

Bloom 过滤器的设计中，涉及到两个参数 m 和 k 的选择。Hash 函数的数目 k 的增加可以减少误识率 P，但随着 k 的继续增加，误识率反而会上升。误识率 P 是一个关于 k 的凸函数。一般而言，给定 m 和 n，可以确定最优的 Hash 函数数目 k 为

$$k=\frac{m}{n}ln\,2\approx 0.7\frac{m}{n}$$

此外，给定集合元素的数目 n 和误识率 P，Bloom 过滤器的比特数组的长度最短应设置为

$$m=-\frac{n\,ln\,P}{(ln\,2)^{2}}$$

表 1.1 Bloom 过滤器误识率

m/n	k=5	k=6	k=7	k=8
10	0.00943	0.00844	0.00819	0.00846
20	0.00053	0.000303	0.000196	0.00014
30	8.53e-05	3.55e-05	1.69e-05	9.01e-06

表 1.1 是 Bloom 过滤器选择不同参数时的误识率。通过合理选择参数，可以把误识率控制在一个很低的水平。

Bloom 过滤器优点:

1、相比于其它的数据结构，Bloom 过滤器在空间和时间方面都有巨大优势。Bloom 过滤器存储空间和插入/查询时间都是常数 O(K)。

2、散列函数相互之间没有关系，方便由硬件并行实现。

3、Bloom 过滤器不需要存储元素本身，在某些对保密要求非常严格的场合有优势。

利用 Bloom 过滤器可以快速地解决项目中一些比较棘手的问题。如网页 URL 去重、垃圾邮件识别、大集合中重复元素的判断等问题。

例 4-*（钓鱼网站过滤）浏览器通常使用 Bloom 过滤器识别恶意链接，警告用户访问的网站可能是钓鱼网站。请设计一个存储钓鱼网站的 Bloom 过滤器，存储已知的钓鱼网站，用户可以快速查询某个网址是否是已知的钓鱼网站。

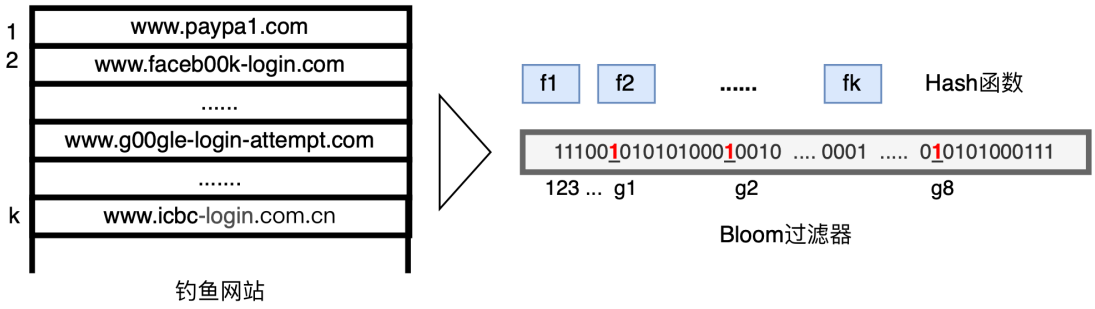


图 4-* 钓鱼网站过滤

假设已知的钓鱼网站有 n=100,000 个，把这些网站的 URL 存储在一个 Bloom 过滤器中。如果要求的误识率 P=0.01，则 Bloom 过滤器的比特数组的长度 m 应设置为

$$m = -\frac{n \ln P}{(\ln 2)^2} = -\frac{100000 \ln 0.01}{(\ln 2)^2} = 959{,}410\ bit$$

Hash 函数数目 k 为

$$k = 0.7\frac{m}{n} = 6.716$$

当误识率 P=0.001 时，

$$m = -\frac{n \ln P}{(\ln 2)^2} = -\frac{100000 \ln 0.001}{(\ln 2)^2} = 1{,}439{,}115\ bit$$

$$k = 0.7\frac{m}{n} = 10.07$$

例 4-*（数据库缓存穿透）在很多 Web 应用中，原始数据通常存储在后台的数据库 DB 中（如 MySQL、Hbase 等），但磁盘查找代价高昂，大大提高数据库查询操作的性能。为了避免不必要的磁盘查找，通常引入高速缓存（如 Redis）弥补 DB 的不足。当请求进来的时候，先从缓存中取数据，如果有则直接返回缓存中的数据；如果缓存中没数据，就去数据库中读取数据并写到缓存中，再返回结果。

数据库缓存穿透是指：用户查询数据库不存在的数据时，数据不存在 Redis 缓存中，也不存在数据库中，查询请求都会穿透到数据库，对数据库产生很大压力。

解决方案是采用 Bloom 过滤器：在数据写入数据库时，将这个数据标识同步到到布隆过滤器中。当请求的数据不存在 Bloom 过滤器中则说明该请求查询的数据一定没有在数据库中保存，则不需要去数据库查询。

假设数据库存储 100,000,000 条数据,要求数据库缓存穿透的概率小于 0.01，则 Bloom 过滤器的比特数组的长度 m 应设置为

$$m = -\frac{n \ln P}{(\ln 2)^2} = -\frac{10^8 \ \ln 0.01}{(\ln 2)^2} = 959{,}410{,}455\ bit = 0.8935Gb$$

如果 Bloom 过滤器判断数据不在数据库中，则查询的数据可定不在数据库中。如果过滤器判断数据在数据库中，则数据有 0.01 的概率不在数据库中，即数据库缓存穿透的概率不会超过 0.01。

4.5 贝叶斯推理

随着云计算、大数据等技术的发展，网络环境日益复杂，数据维度不断增加，网络空间安全复杂程度日益增长，传统分析方法不再适用目前的网络空间环境，问题主要包括：

1、依靠安全专家人工修复方法无法解决零日漏洞问题；

2、传统依靠规则匹配的网络入侵检测方法，面对不断增加的复杂网络行为，出现大量误判和判别时间过长；

3、依靠固定规则或黑白名单的垃圾邮件检测方法检测效率低下，规则更新不及时。

机器学习在系统安全、网络安全和应用安全方面有大量研究成果。在网络安全中，检测根据网络流量数据或主机数据来判断系统的行为正常或者异常。这可以抽象为分类问题，而分类问题可以用机器学习方法中的贝叶斯分类器很好解决。

贝叶斯分类器是基于贝叶斯定理而构造出来的，是一个统计分类器，能够预测类别所属的概率。对分类方法进行比较的有关研究结果表明：朴素贝叶斯分类器（称为基本贝叶斯分类器）在分类性能上与决策树和神经网络相当。在处理大规模数据集时，贝叶斯分类器表现出很高的分类准确性和性能。

4.5.1 朴素贝叶斯决策论

假设有 N 种可能的类别标记$Y = \{c_1, \cdots, c_N\}$，$x = \{x_1, \cdots, x_d\}$是类别未知的数据样本，其中 d 为属性值数量，x_i为x在第 i 个属性上的取值。对于分类问题，希望确定$P(c|x_1, \cdots, x_d)$，即给定观测数据样本$x = \{x_1, \cdots, x_d\}$，确定样本属于类别 c 的概率。贝叶斯定理给出了计算$P(c|x_1, \cdots, x_d)$简单有效的方法

$$P(c|x_1, \cdots, x_d) = \frac{P(x_1, \cdots, x_d, c)}{P(x_1, \cdots, x_d)} = \frac{P(x_1, \cdots, x_d|c)P(c)}{P(x_1, \cdots, x_d)}$$

其中，$P(c)$是先验概率，$P(x_1, \cdots, x_d|c)$是样本x相对于类别 c 的条件概率，$P(x_1, \cdots, x_d)$称为归一化因子。由此，估计$P(c|x_1, \cdots, x_d)$的问题转化为如何基于训练数据集来估计先验概率$P(c)$和条件概率$P(x_1, \cdots, x_d|c)$。

先验概率$P(c)$ 是样本空间中各类样本所占的比例。根据大数定律，当训练数据集包含大量独立同分布的样本时，$P(c)$可用各类样本出现的频率来估计。对于条件概率$P(x_1, \cdots, x_d|c)$，由于涉及到关于 x 所有属性的联合概率，直接根据样本出现的频率来估计将会遇到困难。例如，假设样本的 d 个属性都是二值的，则样本空间将有2^d种可能的取值，在现实应用中，这个值往往远大于训练样本数。并且，很多样本值在训练集中可能不会出现，直接使用频率来估计$P(x_1, \cdots, x_d|c)$不可行。为了避开这个障碍，朴素贝叶斯分类器采用“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。

基于属性独立性假设，条件概率$P(x_1, \cdots, x_d|c)$的计算可转化为以下公式：

$$\mathrm{P}(c|x_1,\cdots,x_d)=\frac{P(x_1,\cdots,x_d|c)P(c)}{P(x_1,\cdots,x_d)}=\frac{P(c)}{P(x_1,\cdots,x_d)}\cdot\prod_{i=1}^{d}\mathrm{P}(x_i|c)$$

因此，贝叶斯分类器的判定准则是

$$C_{NB}(x)=\arg\max_{c\in\mathrm{Y}}\quad P(c)\prod_{i=1}^{d}P(x_i|c)$$

朴素贝叶斯分类器的训练过程就是基于训练数据集D来估计先验概率$P(c)$和条件概率$P(x_i|c)$。

设D_c表示训练集D中由第c类样本组成的集合。如果有充足的独立同分布的样本，则可估计此类别出现的先验概率为

$$P(c)=\frac{|D_c|}{|D|}$$

其中，$|D_c|$和$|D|$分别表示集合D_c和D的基数。

对于离散型属性值而言，令D_{c,x_i}表示D_c中第 i 个属性上取值为x_i的样本组成的集合，则$P(x_i|c)$可采用以下方法估计

$$P(x_i|c)=\frac{|D_{c,x_i}|}{|D_c|}$$

对于连续型属性值，可以采用其概率密度函数来估计。假设$\mu_{c_i},\sigma_{c_i}^2$分别是第 c 类样本在第 i 个属性上取值的均值和方差，则设$P(x_i|c)\sim N\left(\mu_{c_i},\sigma_{c_i}^2\right)$，即$P(x_i|c)$用下面的公式表示

$$P(x_i|c)=\frac{1}{\sqrt{2\pi}\sigma_{c_i}}exp\left(-\frac{\left(x_i-\mu_{c_i}\right)^2}{2\sigma_{c_i}^2}\right)$$

如果某个属性值在训练集中没有出现在某个类别中，则会产生问题。例如，$P(x_i|c)=0$，则无论该样本中其他属性是什么，$\prod_{i=1}^{d}P(x_i|c)=0$。

为避免其他属性携带的信息被训练集中未出现的属性值“抹去”，在估计概率值时通常进行平滑处理。常用的平滑处理方法是拉普拉斯修正(Laplacian correction)。令 N 表示训练集 D 中可能的类别数，Ni 表示第 i 个属性可能的取值数，则估计概率值经过拉普拉斯修正以后的值为

$$P(c) = \frac{|D_c| + 1}{|D| + N}$$

$$P(x_i|c) = \frac{|D_{c,x_i}| + 1}{|D_c| + N_i}$$

4.5.2 朴素贝叶斯分类步骤

朴素贝叶斯分类由以下步骤组成：

1、准备数据：收集训练样本数据集，转化为计算机能理解的数据。

2、建立模型：统计各种类别出现的先验概率，以及在某个类别下某种属性出现的条件概率。

3、分类新数据：根据已建立的模型对新数据进行推导预测。

例 4-*、对水果进行分类。在准备数据阶段，首先提取不同种类水果的属性：形状、颜色、纹理、重量、握感、口感。表 1 是水果及其属性示例。

表 4-* 水果及其属性示例

水果	形状	颜色	纹理	重量	握感	口感
苹果	不规则圆	红	无	200 克	硬	酸甜
甜橙	圆形	橙	无	150 克	软	甜
西瓜	椭圆	绿	有	5000 克	硬	甜

将以上属性转化为数值。形状的不规则圆、圆形、椭圆分别对应为数值 1、2、3，颜色的红、橙、绿用数值 1、2、3 表示，重量小于 200 克为 1，200～500 克为 2，大于 500 克定义为 3。表 2 水果及其属性数值化示例。

表 4- 水果及其属性数值化示例

水果	形状	颜色	纹理	重量	握感	口感
苹果	1	1	1	2	1	1
甜橙	2	2	1	1	2	2
西瓜	3	3	2	3	1	2

假设在准备数据阶段收集了 10 个水果的属性数据作为数据集，并把属性转化为数值。下一步是用 10 个水果的属性数值建立模型，统计各种类别出现的先验概率，以及在某个类别下某种属性出现的条件概率。建立模型如表 3 所示。以苹果为例，10 个水果中有 3 个苹果，其中 2 个不规则圆、1 个圆形，则如果是苹果的条件下，这个水果是不规则圆（形状为 1）的条件概率是 0.67，是圆形（形状为 2）的条件概率是 0.33，是椭圆（形状为 3）的条件概率是 0。同理，其他条件概率作同样处理。

表 3 建立模型

水果	形状	颜色	纹理	重量	握感	口感
苹果（3 个）	**1**:0.67 **2**:0.33 **3**:0	1:0.67 2:0 3:0.33	1:1 2:0	1:0.67 2:0.33 3:0	1:0.67 2:0.33	1:1 2:0
甜橙（3 个）	1:0.33 2:0.67 3:0	1:0 2:1 3:0	1:1 2:0	1:0.33 2:0.67 3:0	1:0.33 2:0.67	1:0.33 2:0.67
西瓜（4 个）	1:0.25 2:0 3:0.75	1:0 2:0 3:1	1:0 2:1	1:0 2:0 3:1	1:0.75 2:0.25	1:0.25 2:0.75
总共	1:0.4 2:0.3 3:0.3	1:0.2 2:0.3 3:0.5	1:0.6 2:0.4	1:0.3 2:0.3 3:0.4	1:0.6 2:0.4	1:0.5 2:0.5

建立好统计模型以后，就可以根据已建立的模型对未分类的水果进行预测。现在假设有一个新的水果 X，它的形状是圆形($shape_2$)，口感是甜($taste_2$)，那么根据朴素贝叶斯推理，它属于苹果($apple$)的概率近似为：

$$P(apple|X) \propto P(apple)\prod_{i=1}^{d} P(x_i|apple) = P(apple)P(shape_2|apple)P(taste_2|apple)$$

其中，采用拉普拉斯修正可得

$$P(apple) = \frac{|D_{apple}| + 1}{|D| + N} = \frac{3+1}{10+3} = \frac{4}{13}$$

$$P(shape_2|apple) = \frac{|D_{apple,shape_2}| + 1}{|D_{apple}| + 3} = \frac{1+1}{3+3} = \frac{2}{3}$$

$$P(taste_2|apple) = \frac{|D_{apple,taste\_2}| + 1}{|D_{apple}| + 3} = \frac{0+1}{3+2} = \frac{1}{5}$$

所以，

$$P(apple|X) \propto P(apple)P(shape_2|apple)P(taste_2|apple) = 0.0205$$

同理，水果 X 属于甜橙$(orange)$和西瓜$(melon)$的概率分别近似为

$$P(orange|X) \propto P(orange)P(shape_2|orange)P(taste_2|orange)$$

$$= \frac{3+1}{10+3} \cdot \frac{3+1}{3+3} \cdot \frac{2+1}{3+2} = 0.1231$$

$$P(melon|X) \propto P(melon)P(shape_2|melon)P(taste_2|melon)$$

$$= \frac{4+1}{10+3} \cdot \frac{0+1}{4+3} \cdot \frac{3+1}{4+2} = 0.0366$$

比较以上三个后验概率值，可以得出结论：该形状是圆形$(shape_2)$、口感是甜的$(taste_2)$水果 X 是甜橙的可能性最大。

例 4-*（**入侵检测**）通过计算机的 5 个外在特征来判断计算机是否受到入侵。给定表 3 的入侵检测数据集，建立朴素贝叶斯模型，判断以下计算机的外部特征 **X** 和 **Y** 是否为入侵：

入侵/正常	时延	响应	流量	行为	内存
X	中	慢	异常	正常	不变
Y	高	快	正常	异常	不变

表 3 入侵检测数据集

入侵/正常	时延	响应	流量	行为	内存
正常	高	快	异常	正常	不变
正常	中	快	正常	异常	不变
正常	低	中	未知	正常	增大
正常	高	慢	正常	正常	减小
正常	中	中	未知	正常	增大
正常	低	快	正常	异常	不变
入侵	高	中	异常	正常	增大
入侵	中	慢	正常	异常	增大

入侵	中	慢	正常	异常	不变
入侵	高	慢	异常	正常	不变
入侵	低	中	未知	异常	增大

对于计算机的外部特征 **X**，根据朴素贝叶斯推理，它属于**正常**的概率为：

$$P(\text{正常}|X) \propto P(\text{正常})\prod_{i=1}^{d} P(x_i|\text{正常})$$

$$= P(\text{正常})P\left(\text{时延}_{\text{中}}\middle|\text{正常}\right)P\left(\text{响应}_{\text{慢}}\middle|\text{正常}\right)P\left(\text{流量}_{\text{异常}}\middle|\text{正常}\right)P\left(\text{行为}_{\text{异常}}\middle|\text{正常}\right)P\left(\text{内存}_{\text{不变}}\middle|\text{正常}\right)$$

$$= \frac{6+1}{11+2}\cdot\frac{2+1}{6+3}\cdot\frac{1+1}{6+3}\cdot\frac{1+1}{6+3}\cdot\frac{4+1}{6+2}\cdot\frac{3+1}{6+3}$$

$$= 0.002462$$

对于计算机的外部特征 **X**，属于**异常**的概率为：

$$P(\text{异常}|X) \propto P(\text{异常})\prod_{i=1}^{d} P(x_i|\text{异常})$$

$$= P(\text{异常})P\left(\text{时延}_{\text{中}}\middle|\text{异常}\right)P\left(\text{响应}_{\text{慢}}\middle|\text{异常}\right)P\left(\text{流量}_{\text{异常}}\middle|\text{异常}\right)P\left(\text{行为}_{\text{异常}}\middle|\text{异常}\right)P\left(\text{内存}_{\text{不变}}\middle|\text{异常}\right)$$

$$= \frac{5+1}{11+2}\cdot\frac{2+1}{5+3}\cdot\frac{3+1}{5+3}\cdot\frac{2+1}{5+3}\cdot\frac{2+1}{5+2}\cdot\frac{2+1}{5+3}$$

$$= 0.005650$$

比较以上两个后验概率值，可以得出结论：对于计算机的外部特征 **X**，它属于**异常**的可能性较大。

同理，计算机的外部特征 **Y**，根据朴素贝叶斯推理，它属于**正常**的概率为：

$$P(\text{正常}|Y) \propto P(\text{正常})\prod_{i=1}^{d} P(x_i|\text{正常})$$

$$= P(\text{正常})P\left(\text{时延}_{\text{高}}\middle|\text{正常}\right)P\left(\text{响应}_{\text{快}}\middle|\text{正常}\right)P\left(\text{流量}_{\text{正常}}\middle|\text{正常}\right)P\left(\text{行为}_{\text{异常}}\middle|\text{正常}\right)P\left(\text{内存}_{\text{不变}}\middle|\text{正常}\right)$$

$$= \frac{6+1}{11+2}\cdot\frac{2+1}{6+3}\cdot\frac{3+1}{6+3}\cdot\frac{3+1}{6+3}\cdot\frac{2+1}{6+2}\cdot\frac{3+1}{6+3}$$

$$= 0.0059$$

对于计算机的外部特征Y，属于**异常**的概率为：

$$P(\text{异常}|Y) \propto P(\text{异常})\prod_{i=1}^{d} P(x_i|\text{异常})$$

$$= P(\text{异常})P\left(\text{时延}_{\text{高}}\middle|\text{异常}\right)P\left(\text{响应}_{\text{快}}\middle|\text{异常}\right)P\left(\text{流量}_{\text{正常}}\middle|\text{异常}\right)P\left(\text{行为}_{\text{异常}}\middle|\text{异常}\right)P\left(\text{内存}_{\text{不变}}\middle|\text{异常}\right)$$

$$= \frac{5+1}{11+2}\cdot\frac{2+1}{5+3}\cdot\frac{0+1}{5+3}\cdot\frac{2+1}{5+3}\cdot\frac{3+1}{5+2}\cdot\frac{2+1}{5+3}$$

$= 0.00152$

比较以上两个后验概率值，可以得出结论：对于计算机的外部特征 **Y**，它属于**正常**的可能性较大。

采用 Python 的机器学习模块 Scikit-Learn 中的高斯朴素贝叶斯算法（Gaussian Naïve Bayes），可以进行快速贝叶斯推理和预测，代码如下：

```
# 贝叶斯推理
from sklearn.naive_bayes import GaussianNB
import numpy as np
# 网络时延分为:高、中、低 -> 1, 2, 3
# 响应速度分为:快、中、慢 -> 1, 2, 3
# 流量异常分为:异常、正常、未知 -> 1, 2，3
# 行为异常分为:异常、正常 -> 1, 2
# 存储增大分为:增大、减小、不变 -> 1, 2, 3
X = np.array([[1,1,1,2,3],
                    [2,1,2,1,3],
                    [3,2,3,2,1],
                    [1,3,2,2,2],
                    [2,2,3,2,1],
                    [3,1,2,1,3],
                    [1,2,1,2,1],
                    [2,3,2,1,1],
                    [2,3,2,1,3],
                    [2,3,1,2,3],
                    [3,2,3,1,1]]);
# 标签：1:正常， 2:入侵
Y = np.array([1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2])
clf = GaussianNB()
clf.fit(X, Y)
# 预测外部特征 X: 中，慢，异常，正常，不变
print(clf.predict([[2,3,1,2,3]]))

# 预测外部特征 Y: 高，快，正常，异常，不变
print(clf.predict([[1,1,2,1,3]]))
```

代码运行结果为：

[2]

[1]

表明 X 属于入侵，Y 属于正常。

4.6 信息熵与决策树

信息论是运用概率论与数理统计的方法研究信息熵、通信系统、密码学、数据压缩等问题的应用数学学科。信息论中包含的知识和概念在机器学习中也有应用，如决策树模型 ID3、C4.5 利用信息增益来构建决策树。

4.6.1 信息熵

信息的定义非常抽象。人们常说信息很多，或者信息很少，但却很难定量衡量信息，如一本五十万字的中文书到底有多少信息量。1948 年 C.E.Shannon（香农）从热力学中借用熵的概念，提出信息熵的定义，用数学语言阐明了概率与信息冗余度的关系，解决了信息的量化度量问题。

定义 4- 熵（Entropy）：随机变量 X 可能取的值为$\{x_1, x_2, \cdots, x_n\}$，其概率分布律为

$$P(X = x_i) = P(x_i) = p_i, i = 1,2, \cdots, n$$

则随机变量 X 的熵定义为$H(X)$，

$$H(X) = -\sum_{i=1}^{n} P(x_i)\, log\, P\,(x_i)$$

熵越大，表示随机变量的不确定度越大，其中蕴含的信息量越多。

例 4-*，假设随机变量 X 的分布律为

X	x_1	x_2	x_3	x_4
P(X)	1/2	1/4	1/8	1/8

则 X 的熵为

$$H(X) = -\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{8}\log\left(\frac{1}{8}\right) - \frac{1}{8}\log\left(\frac{1}{8}\right) = \frac{7}{4}$$

熵函数的性质：

1、$H(X) \geq 0$，当且仅当$P(X) = 1$时取等号。

2、如果$P(X)$是均匀的，则随机变量的熵最大。

定义 4- 联合熵（Joint Entropy）：度量一个多维联合分布的随机变量的不确定性。假设二维随机变量（X，Y）的联合分布为$P(X, Y)$，其联合熵定义为：

$$H(X,Y) = -\sum_{i=1}^{n}\sum_{j=1}^{n} P(x_i, y_j)\log P(x_i, y_j)$$

联合熵是观察一个多随机变量的随机系统获得的信息量，是对二维随机变量(X,Y)不确定性的度量。

定义 4- 条件熵（Conditional Entropy）:在随机变量 X 已知的条件下，另一个随机变量 Y 的熵，用$H(Y|X)$表示，记为

$$H(Y|X) = -\sum_{x\in X, y\in Y} P(x,y)\log P(y|x)$$

条件熵在得知某一确定信息的基础上获取另外一个信息时所获得的信息量，用来衡量在已知随机变量的 X 条件下，随机变量 Y 的不确定性。

一般地，$H(Y) \geq H(Y|X)$，表明已知随机变量 X 以后，另一个随机变量 Y 的不确定性会下降。$H(Y)$和$H(Y|X)$之间的差值，通常称为两个随机变量之间的互信息$I(X,Y)$，用于衡量两个随机事件的相关性，

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

定义 4- 相对熵（Kullback–Leibler divergence）：对于两个概率分布 P 和 Q，它们的相对熵定义为：

$$D(P||Q) = \sum_{i=1}^{n} P(x_i)\log\frac{P(x_i)}{Q(x_i)}$$

相对熵用来描述两个概率分布之间的差异，也称为 KL 散度，常用来表示当用概率分布 Q 来拟合真实的分布 P 时，产生的信息损耗。

相对熵具有不对称性，即$D(P||Q) \neq D(Q||P)$。对于两个完全相同的分布函数，它们的相对熵等于零。如果相对熵越大，两个分布的差异越大；反之，相对熵越小，两个分布的差异越小。

定义 4- 信息增益 G(X,Y)：已知观察数据 Y，对于减少信息 X 的不确定度的帮助有多大，即

$$G(X,Y) = H(X) - H(X|Y)$$

定义 4- 信息增益率 Gr(X, Y)：已知观察数据 Y，对于减少信息 X 的不确定度的帮助的比例，即

$$Gr(X,Y) = G(X,Y)/H(X)$$

信息增益常用于分类问题的特征选取中。在分类问题中，需要考虑选取哪种特征进行分类，衡量标准是特征能够为分类系统带来多少信息增益，带来的信息增益越多，该特征越重要。

例 4-*.假设收集到一组数据，每一条数据记录由三个特征（feature1、2、3）和一个标签（label）组成，如下表 3 数据集所示。

表 3、数据集

feature1	**feature2**	**feature3**	**label**
1	1	2	0
1	1	1	0
2	1	1	1
2	2	2	1

从数据集中可知，label={0,1}，且 P(label=0)=P(label=1)=0.5，因此，label 的熵为

$$H(label) = -P(label = 0) \cdot \log P(label = 0) - P(label = 1) \cdot \log P(label = 1) = 1$$

从数据集中可知，featrue1={1,2}, lable={0,1}，因此两者的联合取值范围为{(1,0), (1,1), (2,0), (2,1)}，且

P(feature1=1, label=0)=0.5

P(feature1=1, label=1)=0

P(feature1=2, label=0)=0

P(feature1=2, label=1)=0.5,

根据定义可以算出 featrue1 和 label 的联合熵为

H(feature1,label)=-(0.5*log0.5+0*log0+0.5*log0.5+0*log0)=-log0.5=1

如果要求已知 feature1 的情况下 label 的条件熵，则先需要求出以下条件概率，

P(label = 1 | feature1 = 1)=0，P(feature1 = 1, label = 1)=0

P(label = 0 | feature1 = 1)=1，P(feature1 = 1, label = 0)=0.5

P(label = 1 | feature1 = 2)=1，P(feature1 = 2, label = 1)=0.5

P(label = 0 | feature1 = 2)=0，P(feature1 = 2, label = 0)=0

根据定义可以算出

H(label | feature1) = -(0 * log 0 + 0.5 * log 1 + 0.5 * log 1 + 0 * log 0)

= - 1 * log 1=0

根据以上计算结果，可计算在已知 feature1 情况下，知道 label 的信息增益为：

G(label, feature1) = H(label) - H(label | feature1) = - log 0.5 - (-1 log 1) = - log 0.5=1

4.6.2 信息熵与决策树分类

决策树算法是一种典型的分类方法，利用归纳算法生成可读的规则和决策树，然后使用决策对新数据进行分析。本质上决策树是通过一系列规则对数据进行分类的过程。

决策树方法最早由 Ross Quinlan 提出，算法（ID3）的目的在于减少树的深度。根据信息论中的奥卡姆剃刀定律（Occam's Razor），即“简单有效原理”，为实现一个简短的问卷，每次选择问题时，选择信息增益最高的问题，使集合的熵值下降最快。

给定一个待分类的集合，可以采用熵来刻画集合的纯净度。如果一个集合里的元素来自不同的类别，则集合的熵可定义为

$$Entropy(P) = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i$$

其中，n 表示集合中类别的数量，P_i表示第 i 个类别的元素在集合中出现的概率。

例 4-*.假设一个容器里有 3 类水果，分别是 2 个苹果、3 根香蕉、3 个梨，则这个容器 P 的熵为

$$Entropy(P) = -\frac{2}{8}\log_2\frac{2}{8} - \frac{3}{8}\log_2\frac{3}{8} - \frac{3}{8}\log_2\frac{3}{8}$$

一个集合包含的类别越多，元素在这些类别中分布得越均匀，则集合的熵值越大，表示集合越混乱。

如果将一个集合划分为多个更小的集合之后，又该如何根据这些小集合来计算集合整体的熵值？一般可以使用以下方法来评估划分以后集合的加权熵值

$$\sum_{v \in T} \frac{|P_v|}{|P|} \cdot Entropy(P_v)$$

其中，T 表示一种划分，P_v表示划分后的某个子集，$Entropy(P_v)$是子集P_v的熵。

对一个集合 P 采用方法 T 进行划分以后，希望集合的整体熵值降低。将划分以后整体熵值的下降部分，称为对集合 P 按照方法 T 进行划分以后得到的信息增益，记为

$$Gain(P,T) = Entropy(P) - \sum_{v \in T} \frac{|P_v|}{|P|} \cdot Entropy(P_v)$$

$Gain(P,T)$越大，表示划分 T 带来的信息增益越多，所以，应该选择信息增益最大的特征来划分集合。

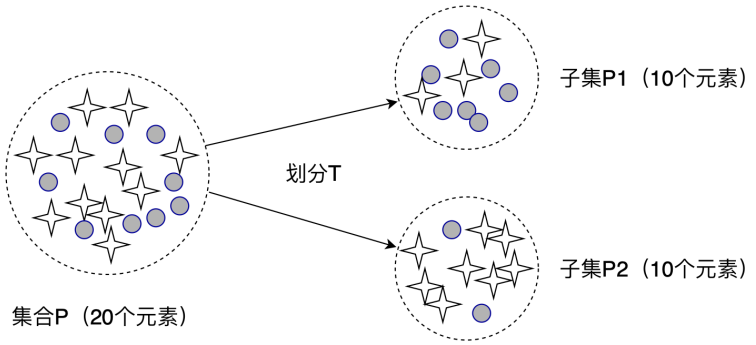


图 1 集合划分

例 4-*、图 1 是一个集合划分。划分前集合 P 由两类元素组成，包含 20 个元素。采用划分方法 T 以后，集合 P 被划分为两个子集 P1 和 P2。为了衡量划分方法 T 的优劣，计算划分 T 带来的信息增益。

首先计算集合 P 和两个子集 P1 和 P2 的熵值：

$$Entropy(P) = -\frac{9}{20}\log_2\frac{9}{20} - \frac{11}{20}\log_2\frac{11}{20} = 0.9928$$

$$Entropy(P1) = -\frac{7}{10}\log_2\frac{7}{10} - \frac{3}{10}\log_2\frac{3}{10} = 0.8813$$

$$Entropy(P2) = -\frac{2}{10}\log_2\frac{2}{10} - \frac{8}{10}\log_2\frac{8}{10} = 0.7219$$

由此可得划分 T 带来的信息增益为

$$Gain(P,T) = Entropy(P) - \left(\frac{|P1|}{|P|}Entropy(P1) + \frac{|P2|}{|P|}Entropy(P2)\right) = 0.1912$$

决策树的建立步骤如下：

Step 1: 根据集合中的样本分类，为每个集合计算信息熵。

Step 2: 根据信息增益，计算每个特征的区分能力。挑选带来信息增益最大的特征对集合进行划分。

Step 3: 对划分后的子集，重复 Step 1 和 Step 2，直到所有子集都被划分完毕（每个子集只包含同类元素）。

例 4-*．给定表 1 数据集 S，包含 14 条样本数据，代表 14 天的天气情况以及是否外出打网球。根据此数据集，构造一个决策树，根据天气情况来判断某天是否外出打网球。

表 1 数据集

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

首先计算四个特征 Outlook、Temperature、Humidity 和 Wind 的区分能力。对于以上数据样本 S，分为两种类型 Yes 和 No，样本可表示为 S:[9+,5-]，表示 9 个 Yes 和 5 个 No。样本集合 S 的信息熵为$Entropy(S) = 0.94$。

如果用特征 Outlook 对 S 进行划分，可以把 S 分为三个子集，分别是

1、子集 Sunny:　{D1,D2,D8,D9,D11}, [2+,3-]，子集的信息熵 E=0.9710

2、子集 Overcast: {D3,D7,D12,D13},　[4+,0-]，子集的信息熵 E=0

3、子集 Rain:　{D4,D5,D6,D10,D14},[3+,2-]，子集的信息熵 E=0.9710

由此可以计算出采用特征 Outlook 作为特征对 S 进行划分以后的信息增益为

$$Gain(S, Outlook) = 0.94 - \frac{5}{14} \times 0.971 - \frac{4}{14} \times 0 - \frac{5}{14} \times 0.971 = 0.2464$$

同理可以计算得到特征 Temperature、Humidity 和 Wind 的信息增益为

$Gain(S, Temperature) = 0.1647$

$Gain(S, Humidity) \quad = 0.151$

$Gain(S, Wind) \quad = 0.048$

图 3 是不同特性的划分结果和信息熵。对比四个特征的划分后的信息增益可知，采用特征 Outlook 对 S 进行划分能获得最大的信息增益，应该首先选择 Outlook 作为决策树的分类特征。

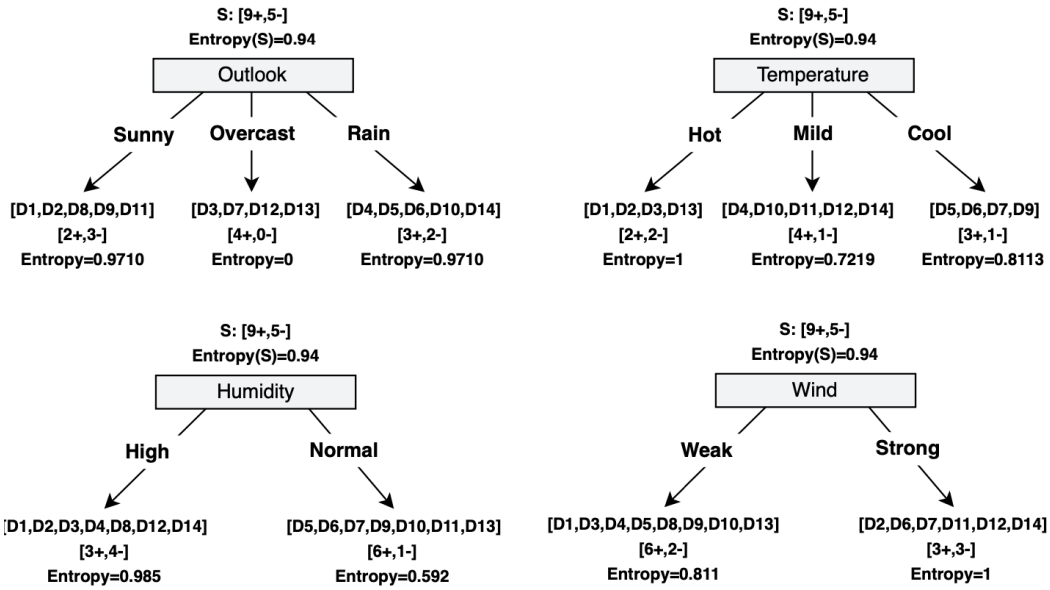


图 3 不同特性的划分结果和信息熵

接下来对基于 Outlook 划分后的子集进行进一步划分，直到划分后的子集只包含一种元素为止。以第一个子集$S_{Sunny} = \{D1, D2, D8, D9, D11\}$的划分为例，分别计算特征 Temperature、Humidity 和 Wind 的信息增益分别为

$Gain(S_{Sunny}, Temperature) = 0.971 - (2/5)0.0 - (2/5)1.0 - (1/5)0.0 = 0.57$

$Gain(S_{Sunny}, Humidity) = 0.9710 - (3/5)0.0 - (2/5)0.0 = 0.9710$

$Gain(S_{Sunny}, Wind) = 0.971 - (2/5)1.0 - (3/5)0.918 = 0.019$

因此，采用 Humidity 得到的信息增益最大。图 2 是对这个子集采用 Humidity 进行划分的结果。

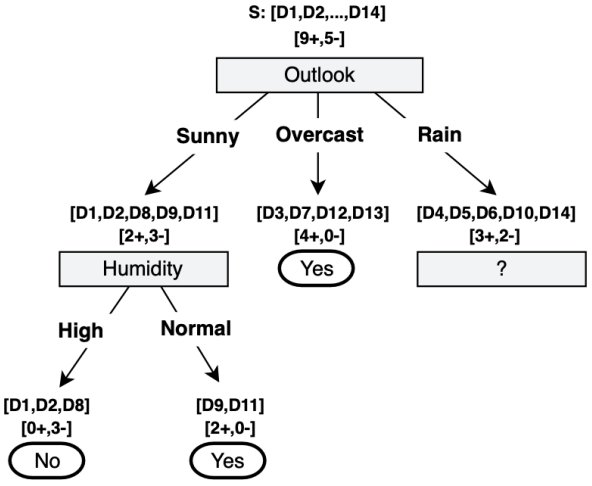


图 2、子集采用 Humidity 进行划分的结果

例 4-*.（入侵检测）根据表 3 的入侵检测数据集构造一个决策树，根据计算机的外部特征来判断计算机是否受到入侵。

表 3 入侵检测数据集

#	时延 (DELAY)	响应 (RESPONSE)	流量 (TRAFFIC)	行为 (BEHAVIOR)	内存 (MEMORY)	正常/入侵 （NORMAL/INTRUSION）
1	高	快	异常	正常	不变	**正常**
2	中	快	正常	异常	不变	**正常**
3	低	中	未知	正常	增大	**正常**
4	高	慢	正常	正常	减小	**正常**
5	中	中	未知	正常	增大	**正常**
6	低	快	正常	异常	不变	**正常**
7	高	中	异常	正常	增大	**入侵**
8	中	慢	正常	异常	增大	**入侵**
9	中	慢	正常	异常	不变	**入侵**
10	高	慢	异常	正常	不变	**入侵**
11	低	中	未知	异常	增大	**入侵**

首先计算五个特征的区分能力。对于以上数据样本 S，分为两种类型：**正常**和**入侵**，样本可表示为 S:[6+,5-]。样本集合 S 的信息熵为$Entropy(S) = 0.9940$。

如果用特征**时延**对 S 进行划分，可以把 S 分为三个子集，分别是

1、**时延**高:{1,4,7,10},[2+,2-]，子集的信息熵 E=1

2、**时延**中:{2,5,8,9},[2+,2-]，子集的信息熵 E=1

3、**时延**低:{3,6,9},[2+,1-]，子集的信息熵 E=0.9182

由此可以计算出采用特征 Outlook 作为特征对 S 进行划分以后的信息增益为

$$Gain(S, \textbf{时延}) = 0.994 - \frac{4}{11} \times 1 - \frac{4}{11} \times 1 - \frac{3}{11} \times 0.9182 = 0.0163$$

同理可以计算得到特征 Temperature、Humidity 和 Wind 的信息增益为

$Gain(S, \textbf{响应}) = 0.3353$

$Gain(S, \textbf{流量}) = 0.0518$

$Gain(S, \textbf{行为}) = 0.0518$

$Gain(S, \textbf{内存}) = 0.1353$

采用特征**响应**对 S 进行划分能获得最大的信息增益，应该首先选择**响应**作为决策树的分类特征。决策树向下的其他分支采用类似的方法可以同样得到。采用 Python 的机器学习模块 Scikit-Learn 中的决策树模块，非常容易就可以根据给定的数据集生产对应的决策树，代码如下：

```
# 决策树
import matplotlib.pyplot as plt
from sklearn import tree
import numpy as np
# 网络时延分为:高、中、低  -> 1, 2, 3
# 响应速度分为:快、中、慢  -> 1, 2, 3
# 流量异常分为:异常、正常、未知  -> 1, 2，3
# 行为异常分为:异常、正常  -> 1, 2
# 存储增大分为:增大、减小、不变  -> 1, 2, 3
X = np.array([[1,1,1,2,3],
              [2,1,2,1,3],
              [3,2,3,2,1],
              [1,3,2,2,2],
              [2,2,3,2,1],
              [3,1,2,1,3],
              [1,2,1,2,1],
              [2,3,2,1,1],
```

```
                    [2,3,2,1,3],
                    [2,3,1,2,3],
                    [3,2,3,1,1]]);
# 标签：1:正常，  2:入侵
Y = np.array([1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2])

clf = tree.DecisionTreeClassifier(criterion='entropy', splitter='best')
clf = clf.fit(X,Y)
tree.plot_tree(clf,
                        feature_names=['Delay', 'Response', 'Traffic', 'Behavior', 'Memory'],
                        class_names=['Normal', 'Intrusion'])
```

图 3 是以上代码生成的决策树。

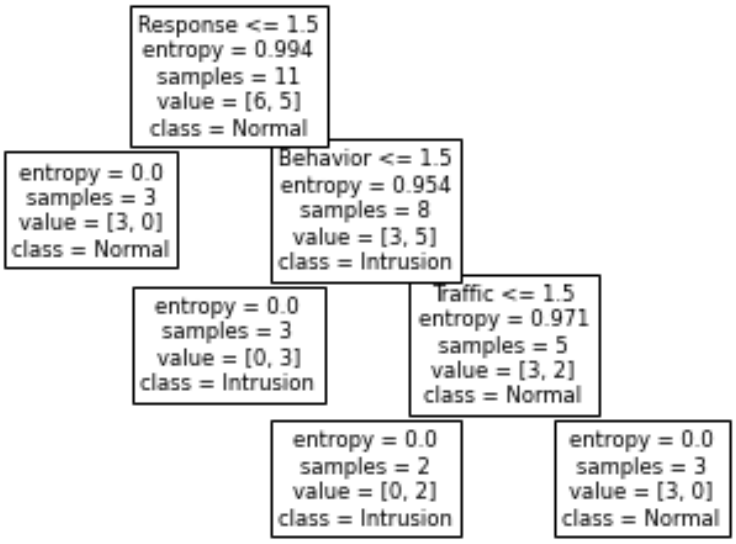


图 3、决策树

4.7 差分隐私基础

4.7.1 隐私保护

在大数据时代，利用数据可以为用户提供更好的服务，如基于位置的服务（LBS）。然而，数据的发布也会在一定程度上带来隐私保护问题。为了不泄漏用户的隐私，一般会采用以下保护方法：

1、 数据加扰：通过扰动原始数据来隐藏真实用户个体的数据，只呈现数据的统计特性。

2、 数据加密：通过加密实现数据的机密性，保护用户隐私，如多方安全计算。

3、 限制发布：有选择地发布原始数据，如泛化，对数据进行更概括、抽象的描述，*k*-匿名技术等。

以上方法的局限性在于：

1、 对敌手的能力有限制；

2、 没有严格的数学证明；

3、 数据可用性和隐私保护程度难以平衡。

差分隐私保护技术定义了一个严格的敌手模型，对隐私泄漏风险给出了定量化的表示和证明。

4.7.2 差分隐私（Differential Privacy）

例 4-*、假设某数据库记录了社区艾滋病人的信息。为了保护用户隐私，对数据库的查询作出限制：只能进行统计查询，且得到的信息条目数量必须大于规定的值 N，确保个人信息不会被直接查询而泄漏。如统计查询“20-30 岁之间艾滋病人的数量”，如果查询结果大于 N=10，则返回结果，如果查询结果小于 10，则不返回结果。即使采用以上的统计查询限制，敌手仍然可以通过差分攻击得到个体的具体信息：

步骤 1：查询整个数据库中的艾滋病人的数量 N1。

步骤 2：查询整个数据库中名字不叫“Alice”的艾滋病人的数量 N2。

步骤 3：若 N2 和 N1 相差为 1（N1-N2=1），则表明 Alice 有艾滋病，否则（N1=N2）没有艾滋病。

差分隐私的目的是使敌手无法区分用户是否在查询的数据集中。即使攻击者具有足够多的背景知识，也无法在输出中找出个体的某项属性。差分隐私常用于推荐系统、社交网络、基于位置的服务等系统中。

差分隐私通过在查询结果中引入一定的噪声来实现。假设原来两次的查询结果是确定的 4 和 5，加入特定的噪声之后，查询结果变成两个随机变量。如果 Alice 不在数据集 D 中，查询结果可能是 4.5，如果把 Alice 添加到数据集 D 中得到一个新的数据集

D’，对 D’的查询结果也可能是 4.5，或者两个数据集 D 和 D’的查询结果很接近，以至于敌手分不清查询结果来自哪一个数据集，从而保护了 Alice 的隐私信息。

差分隐私有两种处理方式：中心化差分隐私和本地化差分隐私。图 1 是中心化和本地化隐私保护，两种处理方法处于数据收集和处理流程中的不同位置。中心化差分隐私默认收集信息的第三方可信，由第三方保护数据查询结果的发布过程。本地化差分隐私则认为收集信息的第三方不可信，需要个体自己保护上传数据不会泄漏隐私。

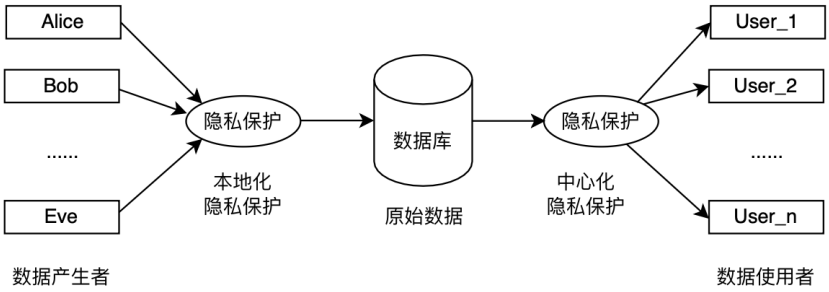


图 1 中心化和本地化隐私保护

定义 相邻数据集：D和D'具有相同的结构，且只相差一条数据记录。

假设算法$A(\cdot)$在数据集D上的操作得到结果 t，即$A(D)=t$，同样，在相邻数据集D'上的结果为$A(D')=t'$。如果没有对运算进行处理，$t-t'$可能会泄漏$D-D'$的隐私信息。差分隐私的主要思想就是对算法$A(\cdot)$的结果进行混淆，使得对于相邻数据集，得到的结果$t \approx t'$的概率保持在一定范围内。

差分隐私的定义有多种，下面给出最基本的ϵ-差分隐私（$\epsilon-DP$）的定义。

定义 $\epsilon-DP$：一个算法$A(\cdot)$满足$\epsilon-DP$（$\epsilon>0$），当且仅当对于任意相邻数据集D和D'，有：

$$\forall T \subseteq Range(A):\ \ \mathrm{P}[A(D)\in T] \le e^{\epsilon}\mathrm{P}[A(D')\in T]$$

其中，$Range(A)$表示算法$A(\cdot)$所有可能的输出。

以上的定义可以等价为下面的形式：

$$\forall t \in Range(A):\quad \frac{\mathrm{P}[A(D)=t]}{\mathrm{P}[A(D')=t]} \le e^{\epsilon}$$

这里定义$\frac{0}{0}=1$。

由以上定义可以看出，算法在相邻数据集上输出相同结果的概率相差不能太大，这个幅度由参数ϵ调控，$\epsilon>0$称为隐私预算。当$\epsilon=0$时，隐私保护能力最强，即$\mathrm{P}[A(D)=t]=\mathrm{P}[A(D')=t]$，但数据的可用性差。一般在实际中根据需要，提高隐私预算，既能保证数据的可用性，又能达到差分隐私保护的目的。

差分隐私的性质：

1、后加工（Posting-processing）不变性

若算法$A_1(*)$满足$\epsilon-DP$，则对于任意函数$A_2(*)$，$A_2\big(A_1(*)\big)$也满足$\epsilon-DP$。

证明：设D和D'是任意两个相邻数据集，$\mathbb{S}$是$A_1(*)$的值域$Range(A_1)$。

对于任意$t\in Range(A_2)$，有：

$$\begin{aligned}P\big[A_2\big(A_1(D)\big)=t\big] &= \sum_{s\in\mathbb{S}} P[A_1(D)=s]\cdot P[A_2(s)=t]\\ &\le \sum_{s\in\mathbb{S}} e^{\epsilon}P[A_1(D')=s]\cdot P[A_2(s)=t]\\ &= e^{\epsilon}P\big[A_2\big(A_1(D')\big)=t\big]\end{aligned}$$

所以，$A_2\big(A_1(*)\big)$满足$\epsilon-DP$。

2、**串行合成性**（Sequential Composition）

若算法$A_1(*)$满足ϵ_1-DP，$A_2(*)$满足ϵ_2-DP，则　算法$A(D)=A_2(A_1(D),D)$满足$(\epsilon_1+\epsilon_2)-DP$。

证明：设D和D'是任意两个相邻数据集，$\mathbb{S}$是$A_1(*)$的值域$Range(A_1)$。

对于任意$t\in Range(A_2)$，有：

$$\begin{aligned}P[A_2(A_1(D),D)=t] &= \sum_{s\in\mathbb{S}} P[A_1(D)=s]\cdot P[A_2(s,D)=t]\\ &\le \sum_{s\in\mathbb{S}} e^{\epsilon_1}P[A_1(D')=s]\cdot e^{\epsilon_2}P[A_2(s,D')=t]\\ &= e^{\epsilon_1+\epsilon_2}P[A_2(A_1(D'),D')=t]\end{aligned}$$

所以，算法$A(D)=A_2(A_1(D),D)$满足$(\epsilon_1+\epsilon_2)-DP$。

串行合成性可以扩展到多个算法。对于一系列满足$\epsilon_1 - DP$、$\epsilon_2 - DP$、…、$\epsilon_k - DP$的 k 个算法，若将这些算法顺序作用到数据集 D 上，则合成后的算法满足$(\epsilon_1 + \epsilon_2 + \cdots + \epsilon_k) - DP$。

3、**并行合成性**（Parallel Composition）

若将数据集 D 划分为 k 个不相交的子集$D_1, D_2, \cdots, D_k$，并对每个子集施加满足$\epsilon_1, \epsilon_2, \cdots, \epsilon_k$的差分隐私算法，则合成后的算法满足$\max\{\epsilon_1, \epsilon_2, \cdots, \epsilon_k\} - DP$。

证明：给定任意两个相邻数据集D和D'，假设D比D'多一个数据元素。设D和D'划分为以下不相交的子集：

$$\begin{cases} D\text{:}\ \ D_1,\ D_2, \cdots, D_k \\ D'\text{:}\ \ D'_1, D'_2, \cdots, D'_k \end{cases}$$

其中，存在一个 j，D_j比D'_j多包含一个数据元素，且对于任意$i \neq j$，$D_i = D'_i$。

设$A(D)$表示$A_1(D_1), A_2(D_2), \cdots, A_k(D_k)$的组合。由于 k 个算法独立运行在互不相交的集合子集D_i上，对于算法输出的任意序列$t = (t_1, t_2, \cdots, t_k)$，$t_i \in Range(A_i)$，有：

$$\begin{aligned} P[A(D) = t] &= P[(A_1(D_1) = t_1) \wedge (A_2(D_2) = t_2) \wedge \cdots \wedge (A_k(D_k) = t_k)] \\ &= P\left[A_j\left(D_j\right) = t_j\right] \prod_{i \neq j} P[A_i(D_i) = t_i] \\ &\leq e^{\epsilon_j} \cdot P\left[A_j\left(D'_j\right) = t_j\right] \prod_{i \neq j} P[A_i(D'_i) = t_i] \quad (D_i = D'_i \text{ for } i \neq j) \\ &\leq e^{max\{\epsilon_1, \cdots, \epsilon_k\}} P[A(D') = t] \end{aligned}$$

4.7.3 差分隐私的实现机制

全局敏感度（Global Sensitivity）：对于任意查询$f: D \to \mathbb{R}$，全局敏感度定义为：

$$\Delta f_{GS} = \max_{\mathrm{D,D'}} \| f(D) - f(D') \|_1$$

其中，$\|\cdot\|_1$是 1-范式，表示$f(D) - f(D')$绝对值的最大值。

从全局敏感度的定义可知，全局敏感度是在所有可能的D及其相邻数据集D'上找到$f(D)$和$f(D')$之间距离的最大值。

全局敏感度只和查询函数 f 相关，反映查询函数 f 在一对相邻数据集查询时所有可能的结果最大的变化程度。如常见的计数查询 count，其全局敏感度就是 1，因为无论

怎样选择数据集，相差一个元素的相邻数据集对计数查询最大的变化量就是 1。但像 mean(平均数)、排序、聚类等函数，其全局敏感度可能会非常大，导致添加的噪声也非常大。

为了达到$\epsilon - DP$，可以给输出结果加上一个噪声来进行掩盖和混淆，即输出为：

$$f'(D) = f(D) + X$$

其中 X 是需要添加的噪声。

首先，添加后的结果应满足：

$$\frac{P[f'(D) = t]}{P[f'(D') = t]} = \frac{P[f(D) + X = t]}{P[f(D') + X = t]} = \frac{P[X = t - f(D)]}{P[X = t - f(D')]} \leq e^{\epsilon}$$

令$d = f(D) - f(D')$，$x = t - f(D)$，则有：

$$\frac{P[X = x]}{P[X = x + d]} \leq e^{\epsilon}$$

下面引入 Laplace 分布和 Laplace 函数。如果随机变量服从 Laplace 分布，其概率密度函数 PDF 为：

$$f(x|\mu, \beta) = \frac{1}{2\beta}\exp\left[-\frac{|x - \mu|}{\beta}\right] = \frac{1}{2\beta}\begin{cases}\exp\left[-\dfrac{\mu - x}{\beta}\right], & x < \mu \\ \exp\left[-\dfrac{x - \mu}{\beta}\right], & x \geq \mu\end{cases}$$

如果$\mu = 0$，在称随机变量满足$Lap(\beta)$的 Laplace 分布，随机变量取某个值的概率定义为以下 Laplace 函数：

$$P[Lap(\beta) = x] = \frac{1}{2\beta}\exp\left(-\frac{|x|}{\beta}\right), \quad (-\infty < x < \infty)$$

在上面的推导中，$f'(D) = f(D) + X$，如果添加的噪声 X 满足 Laplace 函数，$X \sim Lap(\beta)$，则有

$$\frac{P[X = x]}{P[X = x + d]} = \frac{P[Lap(\beta) = x]}{P[Lap(\beta) = x + d]} = \frac{\frac{1}{2\beta}\exp\left(-\frac{|x|}{\beta}\right)}{\frac{1}{2\beta}\exp\left(-\frac{|x + d|}{\beta}\right)} = \exp\left(-\frac{|x + d| - |x|}{\beta}\right)$$

由于$|a| - |b| \leq |a - b|$，所以必有$|x + d| - |x| \leq |d|$成立，并且

$$|d| = |f(D) - f(D')| \leq \max_{D,D'} \| f(D) - f(D') \|_1 = \Delta f_{GS}$$

因此，上式可以写成以下形式：

$$\frac{P[X=x]}{P[X=x+d]} = \exp\left(-\frac{|x+d|-|x|}{\beta}\right) \leq \exp\left(\frac{|d|}{\beta}\right) \leq \exp\left(\frac{\Delta f_{GS}}{\beta}\right)$$

要使上式小于e^{ϵ}，只需令$\frac{\Delta f_{GS}}{\beta} = \epsilon$，即$\beta = \frac{\Delta f_{GS}}{\epsilon}$。

因此，如果加入参数为$\beta = \frac{\Delta f_{GS}}{\epsilon}$的 Laplace 噪声$X = Lap(\beta) = Lap\left(\frac{\Delta f_{GS}}{\epsilon}\right)$，则算法$f'(D) = f(D) + X$满足$\epsilon - DP$。

除了加入 Laplace 噪声以外，还可以添加其他类型的噪声，如在更复杂的差分隐私保护方案中采用的高斯机制或指数机制。

4.7.4 本地化差分隐私

用户对原始数据进行满足ε-本地化差分隐私的扰动，然后将其传输给第三方数据收集者，数据收集者收到扰动后的数据再进行一系列的查询和求精处理，以得到有效的统计结果。随机响应技术(randomized response)是一种常见的本地化差分隐私实现方法，包括两个步骤：扰动性统计和校正。

例 4-是二值离散数据随机响应的一个示例。此方法仅对包含两种取值的离散型数据进行响应，而对于具有超过两种取值的数据并不适用。因此，对离散型数据进行扰动需要对变量的不同取值进行编码和转化，使其满足二值变量的要求。连续型数据的随机响应需要对连续型数据进行转换，将连续型数据离散化，然后利用离散型数据下的随机响应方法，对数据进行扰动。

通过离散化并扰动后的值得到统计量，如变量的平均值，出于数据可用性的考虑，需保证统计结果与真实结果的无偏性。

4.7.5 差分隐私数据发布机制

差分隐私数据发布的目的是为了向公众提供数据查询服务，同时又不泄漏任何个体的记录。对于一个数据集 D，收到一系列查询 f1,f2,…,fm，要求在差分隐私的限制下回答每个查询。查询根据提交的顺序分为两种：

- 交互式查询：用户提出问题，系统进行回答，用户再提出下一个问题，系统再回答，以此类推。

- 非交互式查询：用户同时提出多个问题，系统同时回答全部问题。

例 4-*、假设一个存储用户医疗信息的数据库，每一行记录是一条个体信息，包括用户标识、年龄、性别、是否艾滋病患者等字段。现在收到用户提交的两个查询：

f1：20-40 岁之间艾滋病患者的数量是多少？

f2：20-50 岁之间艾滋病患者的数量是多少？

如果是交互式查询，系统首先回答 f1，由于问题的全局敏感度$\Delta f_{GS}=1$，如果采用 Laplace 机制，系统需要添加的 Laplace 噪声为$Lap(1/\epsilon)$。回答 f2 时，由于 D 中一条数据发生变化，f1 和 f2 的结果都有可能发生变化，因此，$\Delta f_{GS}=2$，此时需要添加的 Laplace 噪声为$Lap(2/\epsilon)$。

如果是非交互式查询，系统需要同时回答两个查询 f1 和 f2。任意一条数据的变化都可能导致两个查询结果都发生变化，此时$\Delta f_{GS}=2$。所以，不管是回答 f1 还是 f2，都需要添加噪声$Lap(2/\epsilon)$。

任何需要保护隐私的算法里都可以使用差分隐私。只要算法每一个步骤都满足差分隐私的要求，那么可以保证算法的最终输出结果满足差分隐私。即使攻击者具有足够多的背景知识，也无法在最终的输出中找出个体的某项属性。

差分隐私作为一个非常漂亮的数学工具，为隐私研究指明了一个发展方向。早期人们很难证明采用的方法保护了隐私，更无法证明究竟保护了多少隐私。现在差分隐私用严格的数学证明告诉人们，只要按照差分隐私的方法设计算法，就能保证隐私不会泄露。

习题

1、设总体$X \sim U(a,b)$,a,b 未知，$x_1, x_2, \cdots, x_k$是来自 X 的样本值，求 a,b 的最大似然估计量。

2、证明差分隐私具有后处理不变性。

3、把随机响应技术的掷硬币方式改为：如果是反面，请如实回应。如果是正面，那么再掷第二枚均匀的硬币，如果正面回答“是”，如果是反面则回答“否”。重新统计艾滋病患者的比例。

4、设计一个 Bloom 过滤器，要求采用 8 个散列函数，存储 10000 个元素，假阳性的概率小于 0.99，则此 Bloom 过滤器的长度最少是多少？

参考文献：

1、盛骤，谢式千，潘承毅，概率论与数理统计，高等教育出版社，2020.

2、Thomas M. Cover, Joy A. Thomas.信息论基础[M]. 机械工业出版社, 2008.

3、周志华，机器学习，清华大学出版社，2016

4、DWORK C,ROTH A. The algorithmic foundations of differential privacy[J].Foundations and Trends in Theoretical Computer Science,2014,9(3-4):211-407.