统计学习方法参考解答

2 感知机

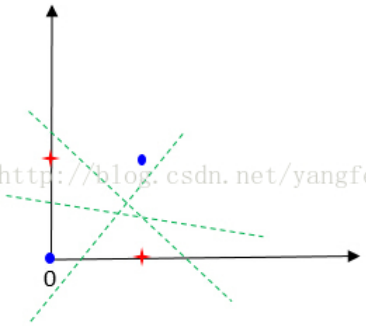
2.1 Minsky 和 Papert 指出：感知机因为是线性模型，所以不能表示复杂的函数，如异或（XOR）。验证感知机为什么不能表示异或。

异或（XOR）是一个数学运算符。它应用于逻辑运算。异或的数学符号为“⊗”，计算机符号为“XOR”。其运算法则为：

$$a \otimes b = (\neg a \wedge b) \vee (a \wedge \neg b)$$

即如果 a, b 两个值不相同，则异或结果为 1。如果 a, b 两个值相同，异或结果为 0。

输入		输出
x_{i1}	x_{i2}	y_i
0	0	0
0	1	1
1	0	1
1	1	0



其中，蓝色的点表示负实例点（y=0），红色的十字表示正实例点（y=1）。从图形上看，我们找不到一条直线将红色的十字与蓝色的点完全分开。

下面我们给出严格的证明，即证明对于任意的 $w \in \mathbb{R}^2, b \in \mathbb{R}$ ，感知机模型 $f(x) = sign(w \cdot x + b)$ 都存在误分类点。

用反证法，假设存在 $w = (w_1, w_2)^T \in \mathbb{R}^2, b \in \mathbb{R}$，使得所有的正实例点和负实例点被正确分类，我们有

$$\begin{cases} w \cdot (0,0)^T + b < 0 \\ w \cdot (0,1)^T + b > 0 \\ w \cdot (1,0)^T + b > 0 \\ w \cdot (1,1)^T + b < 0 \end{cases} \tag{1}$$

上述方程组无解，因此对于任意的 $w \in \mathbb{R}^2, b \in \mathbb{R}$ ，感知机模型 $f(x) = sign(w \cdot x + b)$ 都存在误分类点，即感知机不能表示异或。

2.3 证明以下定理：样本集线性可分的充分必要条件是正实例点集所构成的凸壳与负实例点集所构成的凸壳互不相交。

证 凸壳：设集合 $S \subset \mathbb{R}^n$ 是由 $\mathbb{R}^n$ 中 k 个点所组成的集合，即 $S = x_1, x_2, \ldots, x_k$。定义 S 的凸壳为：

$$conv(S) = \{x = \sum_{i=1}^{k} \lambda_i x_i | \sum_{i=1}^{k} \lambda_i = 1, \lambda_i \geq 0, i = 1, 2, \ldots, k\}$$

.

设 S_+ 为正实例点集，S_- 为正实例点集，则定理可写成 $conv(S_+) \cap conv(S_-) = \emptyset \iff S_+$ 和 S_- 线性可分

充分性：$conv(S_+) \cap conv(S_-) = \emptyset \Rightarrow S_+$ 和 S_- 线性可分

定义 x_1 和 x_2 之间的距离为

$$dist(x_1, x_2) = ||x_1 - x_2||_2 = \sqrt{(x_1 - x_2) \cdot (x_1 - x_2)}$$

定义 $conv(S_+)$ 和 $conv(S_-)$ 之间的距离为

$$dist(conv(S_+), conv(S_-)) = \min ||s_+, s_-||, s_+ \in conv(S_+), s_- \in conv(S_-)$$

取 $x_+ \in conv(S_+), x_- \in conv(S_-)$ 使得 $dist(x_+, x_-) = dist(conv(S_+), conv(S_-))$，由于 $conv(S_+) \cap conv(S_-) = \emptyset$，则

对任意 $x \in conv(S_+)$，有 $dist(x, x_-) > dist(x, x_+)$，

对任意 $x \in conv(S_-)$，有 $dist(x, x_+) > dist(x, x_-)$.

定义超平面 $w^T x + b = 0$，其中

$$w = x_+ - x_-, b = -\frac{x + \cdot x_+ - x_- \cdot x_-}{2}$$

对任意 $x \in conv(S_+)$，有

$$\begin{aligned} w^T x + b &= (x_+ - x_-) \cdot x + -\frac{x + \cdot x_+ - x_- \cdot x_-}{2} \\ &= \frac{||x_- - x||_2^2 - ||x_+ - x||_2^2}{2} \\ &= \frac{dist(x, x_-)^2 - dist(x, x_+)^2}{2} > 0 \end{aligned} \tag{2}$$

类似的，对任意 $x \in conv(S_-)$，$w^T x + b < 0$.

由于 $S_+ \subset conv(S_+), S_- \subset conv(S_-)$，所以 S_+ 和 S_- 线性可分。

必要性：S_+ 和 S_- 线性可分 $\Rightarrow conv(S_+) \cap conv(S_-) = \emptyset$

若 S_+ 和 S_- 线性可分，则可以找到一个超平面 $w^T x + b = 0$ 将 S_+ 和 S_- 分开，则对任意 $x_i \in S_+$，我们有

$$w^T x_i + b = \varepsilon_i > 0, i = 1, 2, \ldots, p$$

由凸壳定义，对任意 $s_+ \in conv(S_+)$，有

$$\begin{aligned} w^T s_+ &= w^T \sum_{i=1}^{p} \lambda_i x_i = \sum_{i=1}^{p} \lambda_i w^T x_i \\ &= \sum_{i=1}^{p} \lambda_i (\varepsilon_i - b) = \sum_{i=1}^{p} \lambda_i \varepsilon_i - b \sum_{i=1}^{p} \lambda_i \\ &= \sum_{i=1}^{p} \lambda_i \varepsilon_i - b \end{aligned} \tag{3}$$

从而有 $w^T s_+ + b = \sum_{i=1}^{p} \lambda_i \varepsilon_i > 0$

类似的，对任意 $s_- \in conv(S_-)$，有 $w^T s_- + b = \sum_{i=1}^{p} \lambda_i \varepsilon_i < 0$

假设 $conv(S_+) \cap conv(S_-) \neq \emptyset$，则存在 $s \in conv(S_+) \cap conv(S_-)$，

由上述，s 必须满足 $w^T s + b = \sum_{i=1}^{p} \lambda_i \varepsilon_i > 0$ 和 $w^T s + b = \sum_{i=1}^{p} \lambda_i \varepsilon_i < 0$，这与 S_+ 和 S_- 线性可分矛盾。

因此，$conv(S_+) \cap conv(S_-) = \emptyset$，必要性得证。

线性可分：（详见书中定义 2.2）X 和 Y 是 n 维欧氏空间中的两个点集，如果存在 n 维向量 w 和实数 b，使得所有属于 X 的点 x_i 都有 $wx_i + b > 0$，而对于所有属于 Y 的点 y_i 则有 $wy_i + b < 0$。则我们称 X 和 Y 线性可分。

注意到此处是严格分离。

凸集分离定理：（超平面分离定理）设 $S_1, S_2 \subseteq \mathbb{R}^n$ 为两个非空集合，如果存在非零向量 $p \in \mathbb{R}^n$ 及 $\alpha \in \mathbb{R}$ 使得

$$S_1 \subseteq H^- = \{x \in \mathbb{R} | p^T x \leq \alpha\} S_2 \subseteq H^+ = \{x \in \mathbb{R} | p^T x \geq \alpha\}$$

则称超平面 $H = \{x \in \mathbb{R} | p^T x = \alpha\}$ 分离了集合 S_1 与 S_2。

注意点此处不是严格分离。

不相交的闭凸集不一定存在严格将他们分离的超平面： 如集合 $X = \{(x, y) | xy \geq 1, x, y > 0\}$ 和集合 $Y = \{(x, y) | x \leq 0\}$

3 k 近邻法

3.2 利用例题 3.2 的构造的 kd 树求点 $x = (3, 4.5)^T$ 的最近邻点。

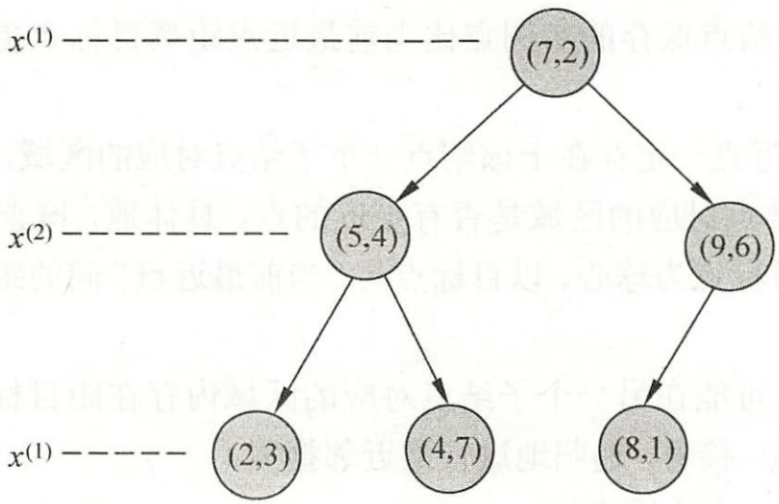


图 3.4 kd 树示例

1. 二叉树搜索： 先从 $(7, 2)$ 点开始进行二叉查找，然后到达 $(5, 4)$，最后到达 $(4, 7)$，此时搜索路径中的结点为 $< (7, 2), (5, 4), (4, 7) >$，首先以 $(4, 7)$ 作为当前最近邻点，计算其到查询点 $(3, 4.5)$ 的距离为 2.69。

注意： 若目标点 x 当前维的坐标小于切分点的坐标，则移动到左子结点，否则移动到右子结点。此处的 x 当前维与切分结点的深度有关。

所以从 $(7, 2)$ 点开始进行查找时，因为 $x^{(1)} = 3 < 7$，所以移动到左子结点 $(5, 4)$，又 $x^{(2)} = 4.5 > 4$，所以移动到右子结点 $(4, 7)$。

2. 回溯查找：（1）在得到 $(4,7)$ 为查询点的最近点后，回溯到其父结点 $(5,4)$，计算其到查询点 $(3,4.5)$ 的距离为 2.06，小于 2.69，则父结点 $(5,4)$ 比 $(4,7)$ 距离查询点更近；

（2）更新 $(5,4)$ 为当前最近邻点，并判断在该父结点的其他子结点空间中是否有距离查询点更近的数据点。以 $(3,4.5)$ 为圆心，以 2.06 为半径画圆，发现该圆和超平面 $y=4$ 交割，因此进入 $(5,4)$ 结点的左子空间中去搜索；

（3）计算 $(2,3)$ 到查询点 $(3,4.5)$ 的距离为 1.80，小于 2.06，则点 $(2,3)$ 比 $(5,4)$ 距离查询点更近，更新 $(2,3)$ 为当前最近点。

3. 最后，再回溯到 $(7,2)$，以 $(3,4.5)$ 为圆心，以 1.80 为半径的圆不与 $x=7$ 超平面交割，因此不用进入 $(7,2)$ 右子空间进行查找。

至此，搜索路径中的结点已经全部回溯完，结束整个搜索，返回最近邻点 $(2,3)$，最近距离为 1.80。

3.3 参照算法 3.3，写出输出为 x 的 k 近邻的算法。

算法（用 kd 树的 k 近邻搜索）

输入：已构造的 kd 树，目标点 x；

输出：x 的 k 近邻。

1. 在 kd 树中找出包含目标点 x 的叶结点：从根结点出发，递归地向下访问 kd 树。若目标点 x 当前维的坐标小于切分点的坐标，则移动到左子结点，否则移动到右子结点，直到子结点为叶结点为止。

2. 构建“当前 k 近邻点集”，将该叶结点插入“当前 k 近邻点集”，并计算该结点到目标点 x 的距离。

3. 递归地向上回退，在每个结点进行以下操作：

(a) 如果“当前 k 近邻点集”的元素数量小于 k，则将该结点插入“当前 k 近邻点集”，并计算该结点到目标点 x 的距离；

(b) 如果“当前 k 近邻点集”的元素数量等于 k，但该结点到目标点 x 的距离小于“当前 k 近邻点集”中最远点到目标点 x 的距离，则将该结点插入“当前 k 近邻点集”，并删除原先的最远点。

(c) 检查另一子结点对应的区域是否与以目标点 x 为球心、以目标点 x 与“当前 k 近邻点集”中最远点的距离为半径的超球体相交。

如果相交，可能在另一个子结点对应的区域内存在距离目标点更近的点，移动到另一个子结点。接着，递归地进行 k 近邻搜索；

如果不相交，向上回退。

4. 当回退到根结点时，搜索结束（若此时“当前 k 近邻点集”中的元素不足 k 个，则需要访问另一半树的结点）。最后的“当前 k 近邻点集”中的 k 个点即为 x 的 k 近邻点。

4 朴素贝叶斯法

4.1 用极大似然估计法推出朴素贝叶斯法中的概率估计公式 (4.8) 及公式 (4.9)。

1. 公式 (4.8)——$\mathbf{P(Y=c_k)=\frac{\sum_{i=1}^{N} I(y_i=c_k)}{N},\quad k=1,2,\dots,K}$

令 $p=P(Y=c_k)$，c_k 在随机变量 Y 中出现的次数为 $m=\sum_{i=1}^{N} I(y_i=c_k)$，则似然函数为

$$L(p)=C_N^m p^m(1-p)^{N-m}$$

将上式两端取对数并关于 p 求导令其为 0，即得似然方程

$$\frac{\partial \ln L(p)}{\partial p}=m\ln p-(N-m)\ln(1-p)=0$$

解之即得 p 的最大似然估计，为

$$\hat{p} = \frac{m}{N}$$

即先验概率 $P(Y = c_k)$ 的极大似然估计是

$$P(Y = c_k) = \hat{p} = \frac{\sum_{i=1}^{N} I(y_i = c_k)}{N}$$

2. 公式 (4.9)——$\mathbf{P(X^{(j)} = a_{jl}|Y = c_k)} = \frac{\sum_{i=1}^{N} \mathbf{I(x_i^{(j)} = a_{jl}, y_i = c_k)}}{\sum_{i=1}^{N} \mathbf{I(y_i = c_k)}}$，$\mathbf{j = 1, 2, \ldots, n; l = 1, 2, \ldots, S_j; k = 1, 2, \ldots, K}$

在条件 $Y = c_k$ 下，随机变量满足条件独立性。

令 $p = P(X^{(j)} = a_{jl}|Y = c_k)$，$c_k$ 在随机变量 Y 中出现的次数为 $m = \sum_{i=1}^{N} I(y_i = c_k)$，$Y = c_k$ 和 $X^{(j)} = a_{jl}$ 同时发生的次数为 $q = \sum_{i=1}^{N} I(x_i^{(j)} = a_{jl}, y_i = c_k)$, 则似然函数为

$$L(p) = C_m^q p^q (1 - p)^{m-q}$$

将上式两端取对数并关于 p 求导令其为 0，即得似然方程

$$\frac{\partial \ln L(p)}{\partial p} = q \ln p - (m - q) \ln(1 - p) = 0$$

解之即得 p 的最大似然估计，为

$$\hat{p} = \frac{q}{m}$$

即条件概率 $P(X^{(j)} = a_{jl}|Y = c_k)$ 的极大似然估计是

$$P(X^{(j)} = a_{jl}|Y = c_k) = \hat{p} = \frac{\sum_{i=1}^{N} I(x_i^{(j)} = a_{jl}, y_i = c_k)}{\sum_{i=1}^{N} I(y_i = c_k)}$$

5 决策树

5.1 根据表 5.1 所给的训练数据集，利用信息增益比（C4.5 算法）生成决策树。

表 5.1 贷款申请样本数据表

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

分别以 A_1, A_2, A_3, A_4 表示年龄、有工作、有自己的房子和信贷情况 4 个特征，计算各特征对数据集 D 的信息增益比，有

(1)

$$g_R(D, A_1) = \frac{g(D, A_1)}{H_{A_1}(D)} = \frac{0.083}{3(-\frac{5}{15}\log_2\frac{5}{15})} = 0.052$$

(2)

$$g_R(D, A_2) = \frac{g(D, A_2)}{H_{A_2}(D)} = \frac{0.324}{-\frac{5}{15}\log_2\frac{5}{15} - \frac{10}{15}\log_2\frac{10}{15}} = 0.353$$

(3)

$$g_R(D, A_3) = \frac{g(D, A_3)}{H_{A_3}(D)} = \frac{0.420}{-\frac{6}{15}\log_2\frac{6}{15} - \frac{9}{15}\log_2\frac{9}{15}} = 0.433$$

(4)

$$g_R(D, A_4) = \frac{g(D, A_4)}{H_{A_4}(D)} = \frac{0.363}{-\frac{5}{15}\log_2\frac{5}{15} - \frac{6}{15}\log_2\frac{6}{15} - \frac{4}{15}\log_2\frac{4}{15}} = 0.232$$

由于特征 A_3（有自己的房子）的信息增益比最大，所以选择特征 A_3 作为根结点的特征。它将训练集 D 划分为两个子集 D_1（A_3 取值为“是”）和 D_2（A_3 取值为“否”）。由于 D_1 只有同一类的样本点，所以它成为一个叶结点，结点的类标记为“是”。

对于 D_2 则需从特征 A_1（年龄），A_2（有工作），A_4（信贷情况）中选择新的特征。计算各个特征的信息增益比：

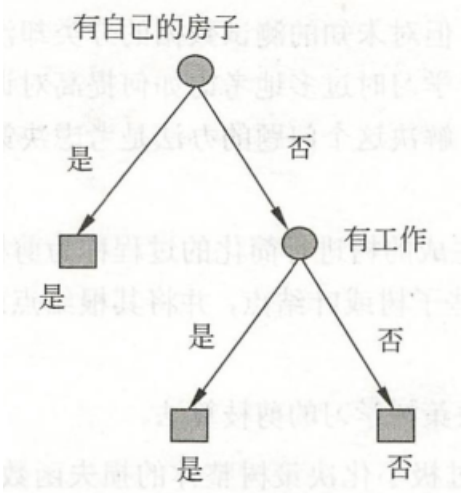
$$g_R(D_2, A_1) = \frac{g(D_2, A_1)}{H_{A_1}(D_2)} = \frac{0.251}{-\frac{4}{9}\log_2\frac{4}{9} - \frac{2}{9}\log_2\frac{2}{9} - \frac{3}{9}\log_2\frac{3}{9}} = 0.164$$

$$g_R(D_2, A_2) = \frac{g(D_2, A_2)}{H_{A_2}(D_2)} = \frac{0.918}{-\frac{3}{9}\log_2\frac{3}{9} - \frac{6}{9}\log_2\frac{6}{9}} = 1.000$$

$$g_R(D_2, A_4) = \frac{g(D_2, A_4)}{H_{A_4}(D_2)} = \frac{0.474}{-\frac{4}{9}\log_2\frac{4}{9} - \frac{4}{9}\log_2\frac{4}{9} - \frac{1}{9}\log_2\frac{1}{9}} = 0.340$$

选择信息增益比最大的特征 A_2（有工作）作为结点的特征，从这一结点引出两个子结点：一个是对应“是”（有工作）的子结点，包含 3 个样本，它们属于同一类，所以这是一个叶结点，类标记为“是”；另一个是对应“否”（无工作）的子结点，包含 6 个样本，它们也属于同一类，所以这也是一个叶结点，类标记为“否”。

这样生成一个如下图所示的决策树。该决策树只用了两个特征（有两个内部节点）。



5.2 已知如表 5.2 所示的训练数据，使用平方误差损失准则生成一个二叉回归树。

表 5.2 训练数据表

x_i	1	2	3	4	5	6	7	8	9	10
y_i	4.50	4.75	4.91	5.34	5.80	7.05	7.90	8.23	8.70	9.00

(1) 由数据表可知，j 只能取 1，选择最优切分点 s，遍历求解

$s=1$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=22.648$$

$s=2$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=17.702$$

$s=3$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=12.193$$

$s=4$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=7.379$$

$s=5$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=3.359$$

$s=6$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=5.074$$

$s=7$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=10.052$$

$s=8$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=15.178$$

$s=9$ 时，

$$\sum_{x_i\in R_1(1,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(1,s)}(y_i-\hat{c_2})^2=21.328$$

$s=10$ 时，所有的点都在 R_1 内，则输入空间不变。

(2) 则当 $j=1, s=5$ 时取得最小值，将输入空间划分为两个区域 $x\leq 5$ 和 $x>5$。

(3) 接着，对两个子区域按上述过程递归计算，设定阈值为 0.2，即

$$\sum_{x_i\in R_1(j,s)}(y_i-\hat{c_1})^2+\sum_{x_i\in R_2(j,s)}(y_i-\hat{c_2})^2<0.2$$

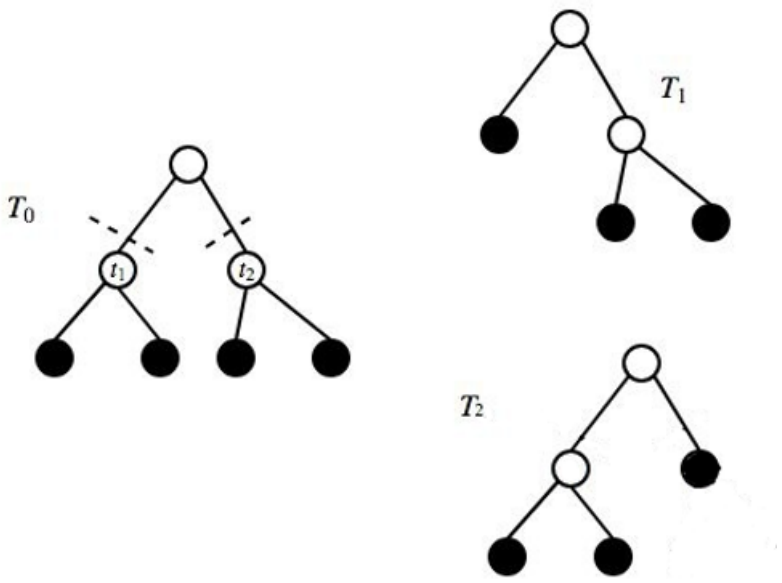
时不再划分，最终得到二叉回归树如下

$$f(x)=\begin{cases}4.72, & x\leq 3\\ 5.57, & 3<x\leq 5\\ 7.05, & 5<x\leq 6\\ 7.90, & 6<x\leq 7\\ 8.23, & 7<x\leq 8\\ 8.85, & x>8\end{cases}$$

5.3 证明 CART 剪枝算法中，当 α 确定时，存在唯一的最小树 T_α 使损失函数 $C_\alpha(T)$ 最小。

存在性：子树的损失函数的公式为 $C_\alpha(T)=C(T)+|\alpha(T)|$，对于固定的 α，每一棵子树 T_i 都对应某一个 $C_\alpha(T_i)$ 损失函数。由于子树的数量有限，一定存在一个最小的损失函数 $C_\alpha(T)$.

唯一性：反证法：假设存在两个或以上的最小树使损失函数最小，不妨从中任取两个最小树 T_1,T_2 使损失函数 $C_\alpha(T)$ 最小，有 $C_\alpha(T_1)=C_\alpha(T_2)$。



如图所示，树 T_0 有两个子树 T_1,T_2，其剪枝位置分别为 t_1,t_2，则有

$$C_\alpha(t_1)<C_\alpha(T_1)$$

$$C_\alpha(t_2)<C_\alpha(T_2)$$

由假设 $C_\alpha(T_1)=C_\alpha(T_2)$，进而有

$$C_\alpha(t_1)<C_\alpha(T_2)$$

$$C_\alpha(t_2)<C_\alpha(T_1)$$

这表明对于子树 T_1,T_2，总存在进一步的剪枝使损失函数变小（在 T_1 中剪枝 t_2，在 T_2 中剪枝 t_1），即 T_1,T_2 均不能使损失函数最小，矛盾。

因此，当 α 确定时，存在唯一的最小树 T_α 使损失函数 $C_\alpha(T)$ 最小。

6 逻辑斯谛回归与最大熵模型

6.2 写出逻辑斯谛回归模型学习的梯度下降算法。

对于给定的训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\dots,(x_N,y_N)\}$，其中 $x_i\in\mathbb{R}^n$，$y_i\in\{0,1\}$，考虑对应的逻辑斯谛回归模型。

为了方便,将权值向量和输入向量加以扩充,仍记作 w,x,即 $w=(w^{(1)},w^{(2)},\cdots,w^{(n)},b)^T,x=(x^{(1)},x^{(2)},\cdots,x^{(n)},1)$。这时，逻辑斯谛回归模型如下:

$$P(Y=1|x)=\frac{\exp(w\cdot x)}{1+\exp(w\cdot x)}$$
$$P(Y=0|x)=\frac{1}{1+\exp(w\cdot x)}$$

设

$$P(Y=1|x)=\pi(x),P(Y=1|x)=1-\pi(x)$$

似然函数为

$$\prod_{i=1}^{N}[\pi(x_i)]^{y_i}[1-\pi(x_i)]^{1-y_i}$$

对数似然函数为

$$\begin{aligned}L(w)&=\sum_{i=1}^{N}[y_i\log\pi(x_i)+(1-y_i)\log(1-\pi(x_i))]\\&=\sum_{i=1}^{N}[y_i\log\frac{\pi(x_i)}{1-\pi(x_i)}+\log(1-\pi(x_i))]\\&=\sum_{i=1}^{N}[y_i(w\cdot x_i)-\log(1+\exp(w\cdot x_i))]\end{aligned}$$

算法——逻辑斯谛回归模型学习的梯度下降法

输入：目标函数 $f(w)=-L(w)$，梯度函数 $g(w)=\nabla L(w)$，计算精度 ε；

输出：最优参数值 w^*；最优模型 $P(Y=1|x)=\frac{\exp(w^*\cdot x)}{1+\exp(w^*\cdot x)},P(Y=0|x)=\frac{1}{1+\exp(w^*\cdot x)}$。

(1) 取初值 $w^{(0)}\in\mathbb{R}^{n+1}$，置 $k=0$。

(2) 计算 $f(w^{(k)})$。

(3) 计算梯度 $g_k=g(w^{(k)})$，当 $||g_k||<\varepsilon$ 时，停止迭代，令 $w^*=w^{(k)}$；否则，令 $p_k=-g(w^{(k)})$，求 λ_k，使

$$f(w^{(k)}+\lambda_k p_k)=\min_{\lambda\geq 0}f(w^{(k)}+\lambda p_k)$$

(4) 置 $w^{(k+1)}=w^{(k)}+\lambda_k p_k$，计算 $f(w^{(k+1)})$

当 $||f(w^{(k+1)})-f(w^{(k)})||<\varepsilon$ 或 $||w^{(k+1)}-w^{(k)}||<\varepsilon$ 时，停止迭代，令 $w^*=w^{(k+1)}$。

(5) 否则，置 $k=k+1$，转 (3)。

注意：梯度下降法是用于求解极小值点，应用到逻辑斯谛回归模型中，需要对目标函数略作调整。

6.3 写出最大熵模型的 DFP 算法。

对于最大熵模型而言,

$$P_w(y|x)=\frac{\exp\left(\sum_{i=1}^{n}w_if_i(x,y)\right)}{\sum_y\exp\left(\sum_{i=1}^{n}w_if_i(x,y)\right)}$$

目标函数：

$$\min_{w\in\mathbb{R}^n} \quad f(w)=\sum_x \tilde{P}(x)\log\sum_y \exp\left(\sum_{i=1}^n w_i f_i(x,y)\right)-\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^n w_i f_i(x,y))$$

梯度：

$$g(w)=\left(\frac{\partial f(w)}{\partial w_1},\frac{\partial f(w)}{\partial w_2},\ldots,\frac{\partial f(w)}{\partial w_n}\right)^T$$

其中，

$$\frac{\partial f(w)}{\partial w_i}=\sum_{x,y}\tilde{P}(x)P_w(y|x)f_i(x,y)-E_{\tilde{P}}(f_i),\quad i=1,2,\ldots,n$$

算法——最大熵模型学习的 DFP 算法

输入：特征函数 $f_1,f_2,\cdots,f_n$；经验分布 $\tilde{P}(x,y)$，目标函数 $f(w)$，梯度 $g(w)=\Delta L(w)$，精度要求 ε；

输出：最优参数值 w^*；最优模型 $P_{w^*}(y|x)$。

(1) 选定初始点 $w^{(0)}$，取 G_0 为正定对称矩阵，置 $k=0$。

(2) 计算 $g_k=g(w^{(k)})$，若 $||g_k||<\varepsilon$，则停止计算，得近似解 $w^*=w^{(k)}$；否则转 (3)。

(3) 置 $p_k=-G_k g_k$。

(4) 一维搜索：求 λ_k 使得

$$f(w^{(k)}+\lambda_k p_k)=\min_{\lambda\geq 0} f(w^{(k)}+\lambda p_k)$$

(5) 置 $w^{(k+1)}=w^{(k)}+\lambda_k p_k$。

(6) 计算 $g_{k+1}=g(w^{(k+1)})$，若 $||g_{k+1}||<\varepsilon$，则停止计算，得近似解 $w^*=w^{(k+1)}$；否则按下式求出 G_{k+1}：

$$G_{k+1}=G_k+\frac{\delta_k\delta_k^T}{\delta_k^T y_k}-\frac{G_k y_k y_k^T G_k}{y_k^T G_k y_k}$$

其中，

$$y_k=g_{k+1}-g_k,\delta_k=w^{(k+1)}-w^k$$

(7) 否则，置 $k=k+1$，转 (3)。

7 支持向量机

7.2 已知正例点 $x_1=(1,2)^T,x_2=(2,3)^T,x_3=(3,3)^T$，负例点 $x_4=(2,1)^T,x_5=(3,2)^T$，试求最大间隔分离超平面和分类决策函数，并在图上画出分离超平面、间隔边界及支持向量。

方法一：根据已知数据构造约束最优化问题

$$\begin{aligned}\min_{w,b,} \quad & \frac{1}{2}(w_1^2+w_2^2)\\ \text{s.t.} \quad & w_1+2w_2+b\geq 1\\ & 2w_1+3w_2+b\geq 1\\ & 3w_1+3w_2+b\geq 1\\ & -2w_1-w_2-b\geq 1\\ & -3w_1-2w_2-b\geq 1\end{aligned}$$

求得此最优化问题的解 $w_1=-1,w_2=2,b=-2$。于是最大间隔分离超平面为

$$-x^{(1)}+2x^{(2)}-2=0$$

分类决策函数为

$$f(x) = \text{sign}\left(-x^{(1)} + 2x^{(2)} - 2\right)$$

其中，$x_1 = (1,2)^T, x_3 = (3,3)^T, x_5 = (3,2)^T$ 是支持向量。

方法二：根据已知数据，对偶问题是

$$\begin{aligned}
\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{5}\sum_{j=1}^{5}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{5}\alpha_i \\
& = \frac{1}{2}(5\alpha_1^2 + 13\alpha_2^2 + 18\alpha_3^2 + 5\alpha_4^2 + 13\alpha_5^2 + 16\alpha_1\alpha_2 + 18\alpha_1\alpha_3 - 8\alpha_1\alpha_4 - 14\alpha_1\alpha_5 + 30\alpha_2\alpha_3 \\
& \quad - 14\alpha_2\alpha_4 - 24\alpha_2\alpha_5 - 18\alpha_3\alpha_4 - 30\alpha_3\alpha_5 + 16\alpha_4\alpha_5) - \alpha_1 - \alpha_2 - \alpha_3 - \alpha_4 - \alpha_5 \\
\text{s.t.} \quad & \alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 - \alpha_5 = 0 \\
& \alpha_i \geq 0, \quad i = 1,2,3,4,5
\end{aligned}$$

利用 MATLAB 求解得到，$s(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$ 在 $\alpha_1 = \frac{1}{2}, \alpha_2 = 0, \alpha_3 = 2, \alpha_4 = 0, \alpha_5 = \frac{5}{2}$ 达到最小。

这样，$\alpha_1^* = \frac{1}{2}, \alpha_3^* = 2, \alpha_5^* = \frac{5}{2}$ 对应的实例点 $x_1 = (1,2)^T, x_3 = (3,3)^T, x_5 = (3,2)^T$ 是支持向量。进一步，求得

$$w^* = \sum_{i=1}^{n}\alpha_i^* y_i x_i = 2$$

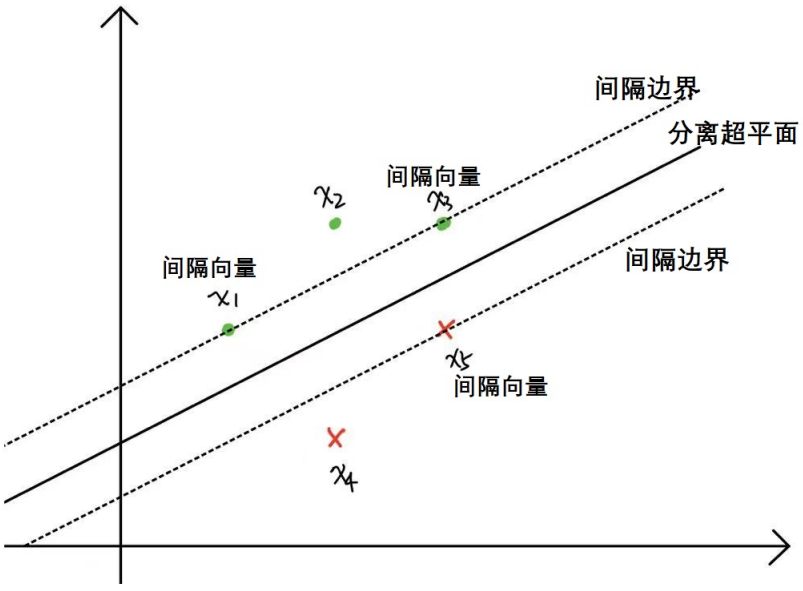
$$b^* = y_j - \sum_{i=1}^{n}\alpha_i^* y_i (x_i \cdot x_j) = -2$$

最大间隔分离超平面为

$$-x^{(1)} + 2x^{(2)} - 2 = 0$$

分类决策函数为

$$f(x) = \text{sign}\left(-x^{(1)} + 2x^{(2)} - 2\right)$$



编写目标函数文件 fun1.m

```
function f = fun1(x)
    f = 1/2*(5*x(1)^2 + 13*x(2)^2 +18*x(3)^2+5*x(4)^2+13*x(5)^2 + 16*x(1)*x(2)+...
    18*x(1)*x(3)-8*x(1)*x(4)-14*x(1)*x(5)+ 30*x(2)*x(3)-14*x(2)*x(4)-24*x(2)*x(5)-...
    18*x(3)*x(4)-30*x(3)*x(5)+16*x(4)*x(5))-x(1)-x(2)-x(3)-x(4)-x(5);
end
```

编写约束条件文件 fun2.m

```
function [g,h]= fun2(x)
    g= -x;
    h=x(1)+x(2)+x(3)-x(4)-x(5);
end
```

利用 fmincon() 函数进行计算

```
[x, y] = fmincon('fun1', rand(5,1), [], [], [], [], zeros(5,1), [], 'fun2')
```

7.3 线性支持向量机还可以定义为以下形式：

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i^2 \\ \text{s.t.} \quad & y_i(w\cdot x_i + b) \geq 1-\xi_i, \quad i=1,2,\cdots,N \\ & \xi_i \geq 0, \quad i=1,2,\cdots,N \end{aligned}$$

试求其对偶形式。

首先构建拉格朗日函数。对每一个不等式约束引进拉格朗日乘子 $\alpha_i, \mu_i \geq 0, i=1,2,\cdots,N$，定义拉格朗日函数：

$$L(w,b,\xi,\alpha,\mu) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i y_i(w\cdot x_i+b) + \sum_{i=1}^{N}(1-\xi_i)\alpha_i - \sum_{i=1}^{N}\mu_i\xi_i$$

对偶问题是拉格朗日函数的极大极小问题。首先求 $L(w,b,\xi,\alpha,\mu)$ 对 w,b,ξ 的极小，由

$$\nabla_w L(w,b,\xi,\alpha,\mu) = w - \sum_{i=1}^{N}\alpha_i y_i x_i = 0$$

$$\nabla_b L(w,b,\xi,\alpha,\mu) = -\sum_{i=1}^{N}\alpha_i y_i = 0$$

$$\nabla_{\xi_i} L(w,b,\xi,\alpha,\mu) = 2C\xi_i - \alpha_i - \mu_i = 0$$

得

$$w = \sum_{i=1}^{N}\alpha_i y_i x_i$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0$$

$$2C\xi_i - \alpha_i - \mu_i = 0$$

将上三式带入拉格朗日函数，得

$$\min_{w,b,\xi} L(w,b,\xi,\alpha,\mu) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{N}\alpha_i - \sum_{j=1}^{N}\frac{(\alpha_i+\mu_i)^2}{4C}$$

再对 $\min_{w,b,\xi} L(w,b,\xi,\alpha,\mu)$ 求 α 和 μ 的极大，即得对偶问题:

$$\begin{aligned}
\max_{\alpha,\mu} \quad & -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{N}\alpha_i - \sum_{j=1}^{N}\frac{(\alpha_i+\mu_i)^2}{4C} \\
\text{s.t.} \quad & \textstyle\sum_{i=1}^{N}\alpha_i y_i = 0 \\
& \alpha_i \geq 0 \\
& \mu_i \geq 0, \quad i = 1, 2, \cdots, N
\end{aligned}$$

再将对目标函数求极大转换为求极小，于是得到对偶问题:

$$\begin{aligned}
\min_{\alpha,\mu} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i + \sum_{j=1}^{N}\frac{(\alpha_i+\mu_i)^2}{4C} \\
\text{s.t.} \quad & \textstyle\sum_{i=1}^{N}\alpha_i y_i = 0 \\
& \alpha_i \geq 0 \\
& \mu_i \geq 0, \quad i = 1, 2, \cdots, N
\end{aligned}$$

7.4 证明内积的正整数幂函数:

$$K(x,z) = (x \cdot z)^p$$

是正定核函数，这里 p 是正整数， $x, z \in \mathbb{R}^n$。

注意: 通常所说的核函数就是正定核函数。

方法一:

当 $p = 1$ 时，$K(x,z) = x \cdot z = \phi_1(x) \cdot \phi_1(z)$，其中 $\phi_1(x) = x$，则 $K(x,z)$ 是正定核函数。

假设当 $p = t(t > 1)$ 时 $K(x,z)$ 是正定核函数，则存在一个从输入空间 $\mathbb{R}^n$ 到特征空间 $\mathcal{H}$ 的映射 $\phi_t(x)\colon \mathbb{R}^n \to \mathcal{H}$，使得对所有的 $x, z \in \mathbb{R}^n$，都有 $K(x,z) = \phi_t(x) \cdot \phi_t(z)$。

不妨设 $\phi_t(x) = (f_1(x), f_2(x), \cdots, f_m(x))^T$，其中 $x = (x^{(1)}, x^{(2)}, \cdots, x^{(n)})$。

则当 $p=t+1$ 时，有

$$\begin{aligned}
K(x,z) &= (x\cdot z)^{t+1} \\
&= (x\cdot z)^k(x\cdot z) \\
&= (\phi_t(x)\cdot\phi_t(z))(x\cdot z) \\
&= (f_1(x)f_1(z)+f_2(x)f_2(z)+\cdots+f_m(x)f_m(z))(x^{(1)}z^{(1)}+x^{(2)}z^{(2)}+\cdots+x^{(n)}z^{(n)}) \\
&= (f_1(x)x^{(1)})(f_1(z)z^{(1)})+\cdots+(f_1(x)x^{(n)})(f_1(z)z^{(n)})+(f_2(x)x^{(1)})(f_2(z)z^{(1)})+\cdots \\
&\quad +(f_2(x)x^{(n)})(f_2(z)z^{(n)})+\cdots+(f_m(x)x^{(1)})(f_m(z)z^{(1)})+\cdots+(f_m(x)x^{(n)})(f_m(z)z^{(n)}) \\
&= \phi_{t+1}(x)\cdot\phi_{t+1}(z)
\end{aligned}$$

其中 $\phi_{t+1}(x)=(f_1(x)x^{(1)},\cdots,f_1(x)x^{(n)},f_2(x)x^{(1)},\cdots,f_2(x)x^{(n)},\cdots,f_m(x)x^{(1)},\cdots,f_m(x)x^{(n)})^T$。

所以当 $p=t+1$ 时，$K(x,z)$ 也是正定核函数。

由数学归纳法，可知 $K(x,z)=(x\cdot z)^p \quad (p\in\mathbb{N}_+,x,z\in\mathbb{R}^n)$ 是正定核函数。

方法二：

（也可以利用实对称矩阵半正定充要条件证明）

显然 $K(x,z)=(x\cdot z)^p$ 是对称函数，由正定核的充要条件，要证 $K(x,z)$ 是正定核函数，只要证 $K(x,z)$ 对应的 Gram 矩阵 $K=[K(x_i,x_j)]_{n\times n}$ 是半正定的。

对任意 $\mathbf{c}=(c_1,c_2,\cdots,c_n)^T,c_1,c_2,\cdots,c_n\in\mathbb{R}$，当 $p=1$ 时，记 $K(x,z)$ 对应的 Gram 矩阵为 K_1，有

$$\begin{aligned}
\mathbf{c}^TK_1\mathbf{c} &= \sum_{i,j=1}^n c_ic_jK(x_i,x_j) \\
&= \sum_{i,j=1}^n c_ic_j(x_i\cdot x_j) \\
&= \left(\sum_i^n c_ix_i\right)\cdot\left(\sum_j^n c_jx_j\right) \\
&= \left|\left|\sum_i^n c_ix_i\right|\right|^2 \geq 0
\end{aligned}$$

表明 $p=1$ 时，$K(x,z)$ 对应的 Gram 矩阵 K_1 是半正定的。

假设 $p=t(t>1)$ 时 $K(x,z)$ 对应的 Gram 矩阵 K_t 是半正定的，则对任意向量 $\mathbf{d}=(d_1,d_2,\cdots,d_n)^T$，有 $\mathbf{d}^TK_t\mathbf{d}\geq 0$。

考虑 $p=t+1$ 时的情形，记 $\mathbf{d}=(d_1,d_2,\cdots,d_n)^T, d_i=\sum_i^n c_i x_i$，有

$$\begin{aligned}\mathbf{c}^T K_{t+1}\mathbf{c} &= \sum_{i,j=1}^{n} c_i c_j K(x_i,x_j)\\ &= \sum_{i,j=1}^{n} c_i c_j (x_i\cdot x_j)^{t+1}\\ &= \sum_{i,j=1}^{n} c_i c_j (x_i\cdot x_j)(x_i\cdot x_j)^{t}\\ &= \sum_{i}^{n} c_i x_i \sum_{j}^{n} c_j x_j (x_i\cdot x_j)^{t}\\ &= \mathbf{d}^T K_t \mathbf{d}\geq 0\end{aligned}$$

即 $p=t+1$ 时矩阵 K_{t+1} 也是半正定的。

由数学归纳法，可知 $K(x,z)$ 对应的 Gram 矩阵 $K=[K(x_i,x_j)]_{n\times n}$ 是半正定的，从而 $K(x,z)$ 是正定核函数。

8 提升方法

8.2 比较支持向量机、AdaBoost、逻辑斯谛回归模型的学习策略和算法

方法	学习策略	学习算法
支持向量机	极小化正则化合页损失，软间隔最大化	序列最小最优化算法（SMO）
AdaBoost	极小化加法模型的指数损失	前向分步加法算法
逻辑斯谛回归模型	极大似然估计，正则化的极大似然估计	改进的迭代尺度算法，梯度下降，拟牛顿法

学习策略

在二类分类的监督学习中，支持向量机、AdaBoost、逻辑斯谛回归模型各自使用合页损失函数、逻辑斯谛损失函数、指数损失函数，分别如下：

$$[1-yf(x)]_+$$

$$log[1+\exp(-yf(x))]$$

$$\exp(-yf(x))$$

这三种损失函数都是 0-1 损失函数的上界，可以认为支持向量机、AdaBoost、逻辑斯谛回归模型使用不同的代理损失函数表示分类的损失，定义经验风险或结构经验风险，实现二类分类学习任务。学习的策略是优化以下结构风险函数：

$$\min_{f\in\mathcal{H}} \frac{1}{N}\sum_{i=1}^{N} L(y_i,f(x_i)) + \lambda J(f)$$

其中第一项为经验风险，第二项为正则化项，$L(y,f(X))$ 为损失函数，$J(f)$ 为模型的复杂度，$\lambda\geq 0$ 为系数。

支持向量机用 L_2 范数表示模型的复杂度；AdaBoost 没有显式的正则化项，通常通过早停止的方法达到正则化的效果；原始的逻辑斯谛回归模型没有正则化项，可以给它加上 L_2 范数正则化项。

学习算法

支持向量机学习，可以解凸二次规划的对偶问题，有序列最小最优化算法等方法。

AdaBoost 利用学习的模型是加法模型、损失函数是指数损失函数的特点，启发式地从前向后逐步学习模型，以达到逼近优化目标函数的目的。

逻辑斯谛回归模型的学习利用梯度下降算法、拟牛顿法等。这些都是一般的无约束最优化问题的解法。

支持向量机和逻辑斯谛回归模型是凸优化问题，全局最优解保证存在；而 AdaBoost 则不是凸优化问题。

补充题　已知错分误差 $\mathcal{R}(f)$ 的定义如下，$Y=\{-1,1\}$，求 $\mathcal{R}(f)$ 的极小值点。

Let ρ be a probability distribution on $Z:=X\times Y$. The *misclassificaiton error* $\mathcal{R}(f)$ for a classifier $f\colon X\to Y$ is defined to be the probability of a wrong prediction, i.e., the measure of the event $\{f(x)\neq y\}$,

$$\mathcal{R}(f):=\mathrm{Prob}_{z\in Z}\{f(x)\neq y\}=\int_X \mathrm{Prob}_{y\in Y}(y\neq f(x)|x)\mathrm{d}\rho_X$$

方法一：

由 $Y=\{-1,+1\}$ 可知，对任意 $x\in X$，$y\in\{-1,+1\}$，$f(x)\in\{-1,+1\}$，从而有

$$\begin{aligned}
\mathcal{R}(f)&=\int_X \mathrm{Prob}_{y\in Y}(y\neq f(x)|x)\mathrm{d}\rho_X\\
&=\int_X \mathrm{Prob}_{y\in Y}(y=1,f(x)=-1|x)+\mathrm{Prob}_{y\in Y}(y=-1,f(x)=1|x)\mathrm{d}\rho_X\\
&=\int_X \mathrm{Prob}_{y\in Y}(y=1|x)(1-\frac{1}{2}(f(x)+1))+\mathrm{Prob}_{y\in Y}(y=-1|x)(\frac{1}{2}(f(x)+1))\mathrm{d}\rho_X\\
&=\frac{1}{2}\int_X(\mathrm{Prob}_{y\in Y}(y=1|x)+\mathrm{Prob}_{y\in Y}(y=-1|x))+(\mathrm{Prob}_{y\in Y}(y=-1|x)-\mathrm{Prob}_{y\in Y}(y=1|x))f(x)\mathrm{d}\rho_X\\
&=\frac{1}{2}\int_X 1+(\mathrm{Prob}_{y\in Y}(y=-1|x)-\mathrm{Prob}_{y\in Y}(y=1|x))f(x)\mathrm{d}\rho_X
\end{aligned}$$

由 $f(x)\in\{-1,+1\}$，令

$$f_c(x)=\begin{cases}1, & \text{if}\quad \mathrm{Prob}_{y\in Y}(y=1|x)\geq\mathrm{Prob}_{y\in Y}(y=-1|x)\\ -1, & \text{if}\quad \mathrm{Prob}_{y\in Y}(y=1|x)<\mathrm{Prob}_{y\in Y}(y=-1|x)\end{cases}$$

则对任意的 f，总有 $\mathcal{R}(f)\geq\mathcal{R}(f_c)$。

因此，$\mathcal{R}(f)$ 的极小值点为 f_c。

方法二：

令 ρ 的回归函数 $f_\rho\colon X\to Y$ 为

$$f_\rho(x)=\int_Y y\mathrm{d}\rho(y|x)$$

由 $Y=\{1,-1\}$ 可知，$f_\rho(x)=\mathrm{Prob}_{y\in Y}(y=1|x)-\mathrm{Prob}_{y\in Y}(y=-1|x)$。

考虑 $f_c(x):=sgn(f_\rho)(x)$，有

$$f_c(x)=sgn(f_\rho)(x)=\begin{cases}1, & \text{if}\quad \mathrm{Prob}_{y\in Y}(y=1|x)\geq\mathrm{Prob}_{y\in Y}(y=-1|x)\\ -1, & \text{if}\quad \mathrm{Prob}_{y\in Y}(y=1|x)<\mathrm{Prob}_{y\in Y}(y=-1|x)\end{cases}$$

注意到对 $x \in X$，有 $\text{Prob}_{y \in Y}(y = f_c(x)|x) \geq \text{Prob}_{y \in Y}(y \neq f_c(x)|x)$。

那么对任意的 f，总有 $f(x) = f_c(x)$ 或 $\text{Prob}_{y \in Y}(y \neq f(x)|x) = \text{Prob}_{y \in Y}(y = f_c(x)|x) \geq \text{Prob}_{y \in Y}(y \neq f_c(x)|x)$ 成立，进而由 $\mathcal{R}(f)$ 定义可知 $\mathcal{R}(f) \geq \mathcal{R}(f_c)$。

因此，$\mathcal{R}(f)$ 的极小值点为 f_c。

8.1 某公司招聘职员考查身体、业务能力、发展潜力这 3 项。身体分为合格 1、不合格 0 两级，业务能力和发展潜力分为上 1、中 2、下 3 三级。分类分为合格 1、不合格-1 两类。已知 10 个人的数据，如表 8.5 所示。假设弱分类器为决策树桩。试用 AdaBoost 算法学习一个强分类器。

表 8.5 应聘人员情况数据表

	1	2	3	4	5	6	7	8	9	10
身体	0	0	1	1	1	0	1	1	1	0
业务能力	1	3	2	1	2	1	1	1	3	2
发展潜力	3	1	2	3	3	2	2	1	1	1
分类	−1	−1	−1	−1	−1	−1	1	1	−1	−1

决策树桩（decision stump，也称单层决策树）是一种简单的决策树，它仅基于单个特征来做决策。由于这棵树只有一次分裂过程，因此它实际上就是一个树桩。

初始化数据权值分布

$$D_1 = (w_{11}, w_{12}, \cdots, w_{110})$$

$$w_{1i} = 0.1, \quad i = 1, 2, \cdots, 10$$

记 $x = (x^{(1)}, x^{(2)}, x^{(3)})$，其中 $x^{(1)}, x^{(2)}, x^{(3)}$ 分别对应身体、业务能力和发展潜力。考虑弱分类器为决策树桩，由 $x^{(k)} < v$ 或 $x^{(k)} > v$ $(k = 1, 2, 3)$ 产生，其阈值 v 使该分类器在训练数据集上分类误差率最低。

对于“身体”，取值有 0、1，则阈值 v 的可能取值有 $\{-0.5, 0.5, 1.5\}$；

对于“业务能力”，取值有 1、2、3，则阈值 v 的可能取值有 $\{0.5, 1.5, 2.5, 3.5\}$；

对于“发展潜力”，取值有 1、2、3，则阈值 v 的可能取值有 $\{0.5, 1.5, 2.5, 3.5\}$。

对$m = 1$，

(a) 在权值分布为 D_1 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_1(x) = \begin{cases} 1, & x^{(1)} > 1.5 \\ -1, & x^{(1)} < 1.5 \end{cases}$$

(b) $G_1(x)$ 在训练数据集上的误差率 $e_1 = \sum_{i=1}^{10} P(G_1(x_i) \neq y_i) = 0.2$。

(c) 计算 $G_1(x)$ 的系数：$\alpha_1 = \frac{1}{2} \log \frac{1-e_1}{e_1} = 0.6931$。

(d) 更新训练数据的权值分布：

$$D_2 = (w_{21}, w_{22}, \cdots, w_{210})$$
$$w_{2i} = \frac{w_{1i}}{Z_1}\exp(-\alpha_1 y_i G_1(x_i)), \quad i = 1, 2, \cdots, 10$$
$$D_2 = (0.0625, 0.0625, 0.0625, 0.0625, 0.0625, 0.0625, 0.2500, 0.2500, 0.0625, 0.0625)$$
$$f_1(x) = 0.6931 G_1(x)$$

分类器 $\text{sign}[f_1(x)]$ 在训练数据集上有 2 个误分类点。

对$\underline{m = 2}$,

(a) 在权值分布为 D_2 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_2(x) = \begin{cases} 1, & x^{(2)} < 1.5 \\ -1, & x^{(2)} > 1.5 \end{cases}$$

(b) $G_2(x)$ 在训练数据集上的误差率 $e_2 = 0.1875$。

(c) 计算 $G_2(x)$ 的系数：$\alpha_2 = 0.7331$。

(d) 更新训练数据的权值分布：

$$D_3 = (0.1667, 0.0385, 0.0385, 0.1667, 0.0385, 0.1667, 0.1538, 0.1538, 0.0385, 0.0385)$$
$$f_2(x) = 0.6931 G_1(x) + 0.7331 G_2(x)$$

分类器 $\text{sign}[f_2(x)]$ 在训练数据集上有 3 个误分类点。

对$\underline{m = 3}$,

(a) 在权值分布为 D_3 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_3(x) = \begin{cases} 1, & x^{(3)} < 1.5 \\ -1, & x^{(3)} > 1.5 \end{cases}$$

(b) $G_3(x)$ 在训练数据集上的误差率 $e_3 = 0.2692$。

(c) 计算 $G_3(x)$ 的系数：$\alpha_3 = 0.4993$。

(d) 更新训练数据的权值分布：

$$D_4 = (0.1140, 0.0714, 0.0263, 0.1140, 0.0263, 0.1140, 0.2857, 0.1053, 0.0714, 0.0714)$$
$$f_3(x) = 0.6931 G_1(x) + 0.7331 G_2(x) + 0.4993 G_3(x)$$

分类器 $\text{sign}[f_3(x)]$ 在训练数据集上有 1 个误分类点。

对$\underline{m = 4}$,

(a) 在权值分布为 D_4 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_4(x) = \begin{cases} 1, & x^{(1)} > 0.5 \\ -1, & x^{(1)} < 0.5 \end{cases}$$

(b) $G_4(x)$ 在训练数据集上的误差率 $e_4 = 0.2381$。

(c) 计算 $G_4(x)$ 的系数：$\alpha_4 = 0.5816$。

(d) 更新训练数据的权值分布：

$$D_5 = (0.0748, 0.0469, 0.0553, 0.2395, 0.0553, 0.0748, 0.1875, 0.0691, 0.150, , 0.0469)$$

$$f_4(x) = 0.6931G_1(x) + 0.7331G_2(x) + 0.4993G_3(x) + 0.5816G_4(x)$$

分类器 $\text{sign}[f_4(x)]$ 在训练数据集上有 1 个误分类点。

对$m = 5$,

(a) 在权值分布为 D_5 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_5(x) = \begin{cases} 1, & x^{(1)} > 1.5 \\ -1, & x^{(1)} < 1.5 \end{cases}$$

(b) $G_5(x)$ 在训练数据集上的误差率 $e_5 = 0.2566$。

(c) 计算 $G_5(x)$ 的系数：$\alpha_5 = 0.5319$。

(d) 更新训练数据的权值分布：

$$D_6 = (0.0503, 0.0315, 0.0372, 0.1611, 0.0372, 0.0503, 0.3654, 0.1346, 0.1009, 0.0315)$$

$$f_5(x) = 0.6931G_1(x) + 0.7331G_2(x) + 0.4993G_3(x) + 0.5816G_4(x) + 0.5319G_5(x)$$

分类器 $\text{sign}[f_5(x)]$ 在训练数据集上有 1 个误分类点。

对$m = 6$,

(a) 在权值分布为 D_6 的训练数据上，遍历所有可能的阈值，得到基本分类器为

$$G_6(x) = \begin{cases} 1, & x^{(3)} < 2.5 \\ -1, & x^{(3)} > 2.5 \end{cases}$$

(b) $G_6(x)$ 在训练数据集上的误差率 $e_6 = 0.2514$。

(c) 计算 $G_6(x)$ 的系数：$\alpha_6 = 0.5455$。

(d) 于是得到：

$$f_6(x) = 0.6931G_1(x) + 0.7331G_2(x) + 0.4993G_3(x) + 0.5816G_4(x) + 0.5319G_5(x) + 0.5455G_6(x)$$

分类器 $\text{sign}[f_6(x)]$ 在训练数据集上误分类点个数为 0。

于是最终分类器为

$$G(x) = \text{sign}[f_6(x)] = \text{sign}[0.6931G_1(x)+0.7331G_2(x)+0.4993G_3(x)+0.5816G_4(x)+0.5319G_5(x)+0.5455G_6(x)]$$

代码如下：

```
import numpy as np
from math import *

#特征，对身体特征做调整，统一标准，均为数值越小越好，后续分类器中还原
```

```
data = np.array([[1, 1, 0, 0, 0, 1, 0, 0, 0, 1],
                 [1, 3, 2, 1, 2, 1, 1, 1, 3, 2],
                 [3, 1, 2, 3, 3, 2, 2, 1, 1, 1]])
#标签，将-1改记为0便于后续计算
label = np.array([0, 0, 0, 0, 0, 0, 1, 1, 0, 0])
#权值分布
D = list()
D.append(np.array([1/len(label)] * len(label)))
#基本分类器
G = list()

#遍历切分点得到基本分类器
def find_min(weight):
    #所有切分点
    cut = [[-0.5, 0.5, 1.5],
           [0.5, 1.5, 2.5, 3.5],
           [0.5, 1.5, 2.5, 3.5]]
    #遍历记录各切分点的分类误差率及切分点标记
    k = 0
    p = 0
    E = np.array([0, 0.0] * 11)
    F = np.array([0.0] * 11)
    #遍历计算各切分点的分类误差率
    for i in range(len(cut)):
        for j in cut[i]:
            #分类误差率
            err = np.sum(weight * ((data[i] < j) != label))
            #记录切分点标记
            E[k] = i
            E[k+1] = j
            k = k + 2
            #记录各切分点的分类误差率
            F[p] = err
            p = p + 1
    #获取第一个达到最小分类误差率的切分点
    t = np.where(F == np.min(F))[0][0]
    err = F[t]
```

```
    subscript = (int(E[2*t]),E[2*t+1])
    return subscript

#设置最大迭代次数
M = 20
#开始迭代计算
for m in range(M):
    #记录基本分类器
    g = find_min(D[m])
    #预测结果初始化
    predict = np.array([0] * 10)
    #利用分类器得到结果
    predict = (data[g[0]] < g[1]) + 0
    #计算分类误差率
    e = np.sum(D[m] * (predict != label))
    #计算Gm(x)的系数
    a = np.log((1 - e) / e) / 2
    #更新权值分布，计算时把标签中的0变回-1
    D.append(D[m] * np.exp(-a * (label * 2 - 1) * (predict * 2 - 1)) /
             np.sum(D[m] * np.exp(-a * (label * 2 - 1)* (predict * 2 - 1))))
    #输出更新的权值分布
    print(np.around(D[m],decimals=4))
    #输出系数、基本分类器，针对关于身体特征的分类器时，还原原数据情况
    if g[0] == 0:
        print(f"G{m + 1}在训练数据集上的误差率为{np.round(e,decimals=4)},"
              f"G{m + 1}的系数为{np.round(a,decimals=4)},基本分类器为"
              f"第{g[0] + 1}个变量{'>'}{1-g[1]}时为合格1,", end="")
    else:
        print(f"G{m + 1}在训练数据集上的误差率为{np.round(e, decimals=4)},"
              f"G{m + 1}的系数为{np.round(a, decimals=4)},基本分类器为"
              f"第{g[0] + 1}个变量{'<'}{g[1]}时为合格1,", end="")

    #记录基本分类器及系数
    G.append((a, g))
    #初始化最终分类结果
    ff = np.array([0.0] * 10)
    #构建基本分类器的线性组合
```

```
    for a, g in G:
        predict = np.array([0] * 10)
        predict = data[g[0]] < g[1]
        ff += a * (2 * predict - 1)
    #得到最终分类器并将分类结果中的 -1调整为0
    ff = (np.sign(ff)+1)/2

    print(f"分类器在训练数据集上有{np.sum(ff != label)}个误分类点。")
    #如果不存在误分类点，则提前停止迭代
    if sum(ff != label) == 0:
        break
```

14 聚类方法

14.1 试写出分裂聚类算法，从上而下地对数据进行聚类，并给出其算法复杂度。

分裂聚类算法从考虑将分为一个类的所有样本划分为两个非空子集开始，这相当于有 $2^{n-1}-1$ 种可能性，其数量会以指数形式快速增长，即使是中等规模的数据集，这种完全列举的方法在计算上也是难以实现的。下述为不考虑所有划分情况的分裂聚类算法：

算法（分裂聚类算法）

输入：n 个样本组成的样本集合及样本之间的距离；

输出：对样本集合的一个层次化聚类。

(1) 计算 n 个样本两两之间的欧氏距离 $\{d_{ij}\}$，记作矩阵 $D=[d_{ij}]_{n\times n}$。

(2) 将所有样本分到一个类 G_1 中。

(3) 选择当前类 G_t $(t=1,2,\ldots,n)$ 中样本间距离 d_{ij} 最大的两个样本 x_i,x_j，构建两个新类，使得 x_i,x_j 分别分到其中一类。对当前类 G_t 中任一样本 x_k，若 $d_{ik}<d_{jk}$，则将 x_k 分到 x_i 所在的类，反之则分到 x_j 所在的类。

(4) 若类的个数为 n，终止计算，否则对新产生的类重复进行步 (3)。

可以看出上述分裂聚类算法的复杂度是 $O(n^3m)$，其中 m 是样本的维数，n 是样本个数。

14.3 证明式 (14.21) 成立，即 k 均值的可能解的个数是指数级的。（课堂上已对该式做出更正）

$$S(n,k)=\frac{1}{k!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^n \tag{14.21}$$

当 $k=1$，此时

$$1=S(n,1)=\frac{1}{1!}\sum_{l=0}^{1}(-1)^l\binom{1}{l}(1-l)^n$$

下面假设 $1 < k \le n$。

考虑样本为 $n+1$，类别仍为 k 的情形，若前 n 个样本已经分成了 k 个类，则第 $n+1$ 个样本只需任意插入一类即可，有 k 种可能；若前 n 个样本仅分成了 $k-1$ 个类，则第 $n+1$ 个样本自动成为第 k 类。于是有

$$S(n+1,k) = k \cdot S(n,k) + S(n,k-1)$$

假设样本数为 n 时式 (14.21) 成立，即

$$S(n,k) = \frac{1}{k!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^n, \quad 1 < k \le n$$

则当样本数为 $n+1$ 时，对任意 $1 < k < n$ 有

$$\begin{aligned}
S(n+1,k) &= k \cdot S(n,k) + S(n,k-1) \\
&= k \cdot \frac{1}{k!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^n + \frac{1}{(k-1)!}\sum_{l=0}^{k-1}(-1)^l\binom{k-1}{l}(k-1-l)^n \\
&= \frac{1}{(k-1)!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^n + \frac{1}{(k-1)!}\sum_{l=1}^{k}(-1)^{l-1}\binom{k-1}{l-1}(k-l)^n \\
&= \frac{k^n}{(k-1)!} + \frac{1}{(k-1)!}\sum_{l=1}^{k}\binom{k}{l}(k-l)^n(1-\frac{l}{k}) \\
&= \frac{k^{n+1}}{k!} + \frac{1}{k!}\sum_{l=1}^{k}\binom{k}{l}(k-l)^{n+1} \\
&= \frac{1}{k!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^{n+1}
\end{aligned}$$

对于 $k = n+1$，有 $1 = S(n+1,n+1)$。要证

$$1 = S(n+1,n+1) = \frac{1}{(n+1)!}\sum_{l=0}^{n+1}(-1)^l\binom{n+1}{l}(n+1-l)^{n+1}$$

只要证

$$(n+1)! = \sum_{l=0}^{n+1}(-1)^l\binom{n+1}{l}(n+1-l)^{n+1}$$

不失一般性，下面证明

$$n! = \sum_{l=0}^{n}(-1)^l\binom{n}{l}(n-l)^n$$

做变量替换，令 $t = n-l$，上式变成

$$n! = \sum_{t=0}^{n}(-1)^{n-t}\binom{n}{t}t^n$$

考虑二项式展开

$$(x-1)^n = \sum_{t=0}^{n} \binom{n}{t} (-1)^{n-t} x^t$$

等式两边关于 x 求导并乘以 x，有

$$n(x-1)^{n-1}x = \sum_{t=0}^{n} \binom{n}{t} (-1)^{n-t} t x^t$$

再关于 x 求导并乘以 x，有

$$n(n-1)(x-1)^{n-2}x^2 + n(x-1)^{n-1}x = \sum_{t=0}^{n} \binom{n}{t} (-1)^{n-t} t^2 x^t$$

重复上述操作到第 n 次，有

$$n!x^n + \frac{n!}{n}(x-1)x^{n-1} + \cdots + n(x-1)^{n-1}x = \sum_{t=0}^{n} \binom{n}{t} (-1)^{n-t} t^n x^t$$

取 $x=1$，有

$$n! = \sum_{t=0}^{n} (-1)^{n-t} \binom{n}{t} t^n$$

从而样本数量为 $n+1$ 时，对任意 $1 < k \leq n$ 有

$$S(n+1,k) = \frac{1}{k!} \sum_{l=0}^{k} (-1)^l \binom{k}{l} (k-l)^{n+1}$$

因此由数学归纳法可知，式 (14.21) 成立。

15 奇异值分解

15.1 试求矩阵

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 0 & 2 \end{bmatrix}$$

的奇异值分解。

(1) 求矩阵 A^TA 的特征值和特征向量

求对称矩阵 A^TA

$$A^TA = \begin{bmatrix} 1 & 2 \\ 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 & 0 \\ 2 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 4 \\ 2 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}$$

求解 $|\lambda I - A^TA| = 0$，得矩阵 A^TA 的特征值 $\lambda_1 = 9$，$\lambda_2 = 4$ 和 $\lambda_3 = 0$。

再将三个特征值分别带入特征方程 $(A^TA-\lambda I)x=0$，得到对应的单位特征向量分别为

$$v_1=\begin{bmatrix}\frac{5}{3\sqrt{5}}\\ \frac{2}{3\sqrt{5}}\\ \frac{4}{3\sqrt{5}}\end{bmatrix},\quad v_2=\begin{bmatrix}0\\ \frac{2}{\sqrt{5}}\\ -\frac{1}{\sqrt{5}}\end{bmatrix},\quad v_3=\begin{bmatrix}-\frac{2}{3}\\ \frac{1}{3}\\ \frac{2}{3}\end{bmatrix}$$

(2) 求正交矩阵 V

构造正交矩阵 V

$$V=\begin{bmatrix}\frac{5}{3\sqrt{5}} & 0 & -\frac{2}{3}\\ \frac{2}{3\sqrt{5}} & \frac{2}{\sqrt{5}} & \frac{1}{3}\\ \frac{4}{3\sqrt{5}} & -\frac{1}{\sqrt{5}} & \frac{2}{3}\end{bmatrix}$$

(3) 求对角矩阵 Σ

奇异值为 $\sigma_1=\sqrt{\lambda_1}=3$，$\sigma_2=\sqrt{\lambda_2}=2$，$\sigma_3=\sqrt{\lambda_3}=0$。构造对角矩阵

$$\Sigma=\begin{bmatrix}3 & 0 & 0\\ 0 & 2 & 0\end{bmatrix}$$

(4) 求正交矩阵 U

基于 A 的正奇异值计算得到列向量 u_1，u_2

$$u_1=\frac{1}{\sigma_1}Av_1=\frac{1}{3}\begin{bmatrix}1 & 2 & 0\\ 2 & 0 & 2\end{bmatrix}\begin{bmatrix}\frac{5}{3\sqrt{5}}\\ \frac{2}{3\sqrt{5}}\\ \frac{4}{3\sqrt{5}}\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{5}}\\ \frac{2}{\sqrt{5}}\end{bmatrix}$$

$$u_2=\frac{1}{\sigma_2}Av_2=\frac{1}{2}\begin{bmatrix}1 & 2 & 0\\ 2 & 0 & 2\end{bmatrix}\begin{bmatrix}0\\ \frac{2}{\sqrt{5}}\\ -\frac{1}{\sqrt{5}}\end{bmatrix}=\begin{bmatrix}\frac{2}{\sqrt{5}}\\ -\frac{1}{\sqrt{5}}\end{bmatrix}$$

构造正交矩阵 U

$$U=\begin{bmatrix}\frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}}\\ \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}}\end{bmatrix}$$

(5) 矩阵 A 的奇异值分解

$$A=U\Sigma V^T=\begin{bmatrix}\frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}}\\ \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}}\end{bmatrix}\begin{bmatrix}3 & 0 & 0\\ 0 & 2 & 0\end{bmatrix}\begin{bmatrix}\frac{5}{3\sqrt{5}} & \frac{2}{3\sqrt{5}} & \frac{4}{3\sqrt{5}}\\ 0 & \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}}\\ -\frac{2}{3} & \frac{1}{3} & \frac{2}{3}\end{bmatrix}$$

15.3 比较矩阵的奇异值分解与对称矩阵的对角化的异同。

相同点：

1. 都可以将目标矩阵表示成几个矩阵的乘积的形式；
2. 都需要进行矩阵的特征值和特征向量的计算。

不同点：

1. 矩阵 A 的奇异值分解计算的是矩阵 A^TA 的特征值和特征向量，而对称矩阵对角化计算的是矩阵自身的特征值和特征向量；

2. 奇异值分解中矩阵的奇异值是对特征值开方，都是非负的，而对称矩阵对角化中矩阵的特征值可以是负的；

3. 奇异值分解中只需将特征向量单位化即可构成正交矩阵，而对称矩阵对角化中还需要将同一特征值的不同特征向量作正交化处理；

4. 奇异值分解的矩阵可以是任意矩阵，对称矩阵对角化需要是对称矩阵；

5. 如果奇异值分解的矩阵行列数不相等，则分解式中的两个正交矩阵至少有一个不是满秩的，而对称矩阵对角化中的正交矩阵一定是满秩的。

15.5 搜索中的点击数据记录用户搜索时提交的查询语句，点击的网页 URL，以及点击的次数，构成一个二部图，其中一个结点集合 $\{q_i\}$ 表示查询，另一个结点集合 $\{u_i\}$ 表示 URL，边表示点击关系，边上的权重表示点击次数。图 15.2 是一个简化的点击数据例。点击数据可以由矩阵表示，试对该矩阵进行奇异值分解，并解释得到的三个矩阵所表示的内容。

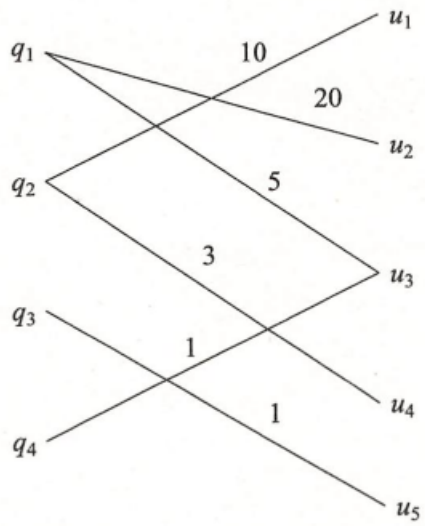


图 15.2　搜索点击数据例

由题意可知，点击数据对应的矩阵为

$$A=\begin{bmatrix}0 & 20 & 5 & 0 & 0\\ 10 & 0 & 0 & 3 & 0\\ 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0 & 0\end{bmatrix}$$

求对称矩阵 A^TA

$$A^TA = \begin{bmatrix} 100 & 0 & 0 & 30 & 0 \\ 0 & 400 & 100 & 0 & 0 \\ 0 & 100 & 26 & 0 & 0 \\ 30 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

求解 $|\lambda I - A^TA| = 0$，得到特征值分别为

$$\lambda_1 = 213 + \sqrt{44969},\ \lambda_2 = 109,\ \lambda_3 = 1,\ \lambda_4 = 213 - \sqrt{44969},\ \lambda_5 = 0$$

发现有两个特征值较为复杂，下用 python 求解得到

正交矩阵 V 为

$$V = \begin{bmatrix} 0 & 0.9578 & 0 & 0 & 0.2873 \\ 0.9700 & 0 & 0 & -0.2431 & 0 \\ 0.2431 & 0 & 0 & 0.9700 & 0 \\ 0 & 0.2873 & 0 & 0 & -0.9578 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

对角矩阵 Σ 为

$$\Sigma = \begin{bmatrix} 20.6170 & 0 & 0 & 0 & 0 \\ 0 & 10.4403 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.9701 & 0 \end{bmatrix}$$

正交矩阵 U 为

$$U = \begin{bmatrix} 0.9999 & 0 & 0 & -0.0118 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0.0118 & 0 & 0 & 0.9999 \end{bmatrix}$$

于是该矩阵奇异值分解为

$$A = \begin{bmatrix} 0.9999 & 0 & 0 & -0.0118 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0.0118 & 0 & 0 & 0.9999 \end{bmatrix} \begin{bmatrix} 20.6170 & 0 & 0 & 0 & 0 \\ 0 & 10.4403 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.9701 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0.9700 & 0.2431 & 0 & 0 \\ 0.9578 & 0 & 0 & 0.2873 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & -0.2431 & 0.9700 & 0 & 0 \\ 0.2873 & 0 & 0 & -0.9578 & 0 \end{bmatrix}$$

考虑矩阵 A 的外积展开式，有 $A = \sum_{i=1}^{4} \sigma_i u_i v_i^T$，其中 $u_i (i = 1, 2, 3, 4)$ 为 U 的列向量，$v_i (i = 1, 2, 3, 4)$ 为 V 的列向量。A 中的行表示查询，列表示 URL。

将 V 的每一列看作是一个 URL，因为第五个奇异值为 0，根据外积展开式，去掉第五列。每列的各元素表示该维度的特征对当前 URL 的重要性，如第一列的表示的 URL1 的第二个维度的特征较为显著，第二列表示的 URL2 的第一个维度的特征比较显著。

将 U 的每一列看作是一个查询，每列的各元素值表示该维度的特征对当前查询的重要性，如第一个查询倾向于第一个维度的特征比较显著的 URL，结合 V 矩阵可以知道，URL2 的第一个维度的特征比较显著，则第一个查询倾向于 URL2。

Σ 的每个对角元表示对应的每个维度特征的重要性。

```
import numpy as np
from numpy import *

A = np.matrix([[0,20,5,0,0],
               [10,0,0,3,0],
               [0,0,0,0,1],
               [0,0,1,0,0]])
U,S,VT = np.linalg.svd(A)
np.set_printoptions(suppress=True, precision=4)
print('U=',U)
print('S=',S)
print('VT=',VT)
```

15.2 试求矩阵

$$A=\begin{bmatrix}2 & 4\\ 1 & 3\\ 0 & 0\\ 0 & 0\end{bmatrix}$$

的奇异值分解并写出其外积展开式。

（1）求矩阵 A^TA 的特征值和特征向量

求对称矩阵 A^TA

$$A^TA=\begin{bmatrix}2 & 1 & 0 & 0\\ 4 & 3 & 0 & 0\end{bmatrix}\begin{bmatrix}2 & 4\\ 1 & 3\\ 0 & 0\\ 0 & 0\end{bmatrix}=\begin{bmatrix}5 & 11\\ 11 & 25\end{bmatrix}$$

求解 $|\lambda I-A^TA|=0$，得矩阵 A^TA 的特征值 $\lambda_1=15+\sqrt{221}\approx 29.8661$，$\lambda_2=15-\sqrt{221}\approx 0.1339$。

再将两个特征值分别带入特征方程 $(A^TA-\lambda I)x=0$，得到对应的单位特征向量分别为

$$v_1=\begin{bmatrix}0.4046\\ 0.9145\end{bmatrix},\quad v_2=\begin{bmatrix}-0.9145\\ 0.4046\end{bmatrix},$$

（2）求正交矩阵 V

构造正交矩阵 V

$$V = \begin{bmatrix} 0.4046 & -0.9145 \\ 0.9145 & 0.4046 \end{bmatrix}$$

(3) 求对角矩阵 Σ

奇异值为 $\sigma_1 = \sqrt{\lambda_1} \approx 5.4650$，$\sigma_2 = \sqrt{\lambda_2} \approx 0.3660$。构造对角矩阵

$$\Sigma = \begin{bmatrix} 5.4650 & 0 \\ 0 & 0.3660 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

(4) 求正交矩阵 U

基于 A 的正奇异值计算得到列向量 u_1，u_2

$$u_1 = \frac{1}{\sigma_1} A v_1 = \begin{bmatrix} 0.8174 \\ 0.5760 \\ 0 \\ 0 \end{bmatrix}$$

$$u_2 = \frac{1}{\sigma_2} A v_2 = \begin{bmatrix} -0.5760 \\ 0.8174 \\ 0 \\ 0 \end{bmatrix}$$

列向量 u_3，u_4 是 A^T 的零空间 $N(A^T)$ 的一组标准正交基。为此，求解以下线性方程组

$$A^T x = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 4 & 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

得到 $N(A^T)$ 的一组标准正交基是

$$u_3 = (0, 0, 1, 0)^T, \quad u_4 = (0, 0, 0, 1)^T$$

构造正交矩阵 U

$$U = \begin{bmatrix} 0.8174 & -0.5760 & 0 & 0 \\ 0.5760 & 0.8174 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(5) 矩阵 A 的奇异值分解

$$A = U\Sigma V^T = \begin{bmatrix} 0.8174 & -0.5760 & 0 & 0 \\ 0.5760 & 0.8174 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5.4650 & 0 \\ 0 & 0.3660 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0.4046 & 0.9145 \\ -0.9145 & 0.4046 \end{bmatrix}$$

矩阵 A 的外积展开式为

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T = 5.4650 \begin{bmatrix} 0.8174 \\ 0.5760 \\ 0 \\ 0 \end{bmatrix} (0.4046, 0.9145) + 0.3660 \begin{bmatrix} -0.5760 \\ 0.8174 \\ 0 \\ 0 \end{bmatrix} (-0.9145, 0.4046) = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

15.4 证明任何一个秩为 1 的矩阵可以写成两个向量的外积形式，并给出实例。

设 $m \times n$ 矩阵 A 的秩为 1，即 $r(A) = 1$，则矩阵 $A^T A$ 的秩 $r(A^T A) = 1$。

于是，矩阵 $A^T A$ 只有一个不为 0（大于 0）的特征值，记为 λ_1，其他特征值均为 0，即

$$\lambda_2 = \lambda_3 = \cdots = \lambda_n = 0$$

记对应的单位特征向量分别为 $v_1, v_2, \cdots, v_n$。

进一步求解 A 的奇异值 $\sigma_1 = \sqrt{\lambda_1} > 0$，$\sigma_i = \sqrt{\lambda_i} = 0$，$i = 2, 3, \cdots, n$。

相应的再求解正交矩阵 $U = [u_1, u_2, \cdots, u_m]$，则 A 的外积展开式为

$$A = \sigma_1 u_1 v_1^T$$

即任何一个秩为 1 的矩阵可以写成两个向量的外积形式。

实例 (例 15.5)

取矩阵 A 为

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 0 & 0 \end{bmatrix}$$

显然 $r(A) = 1$，其外积展开式为

$$A = \sigma_1 u_1 v_1^T = \sqrt{10} \begin{bmatrix} \frac{1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \\ 0 \end{bmatrix} (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}) = \begin{bmatrix} \frac{1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \\ 0 \end{bmatrix} (\sqrt{5}, \sqrt{5})$$

16 主成分分析

16.1 对以下样本数据进行主成分分析：

$$X = \begin{bmatrix} 2 & 3 & 3 & 4 & 5 & 7 \\ 2 & 4 & 5 & 5 & 6 & 8 \end{bmatrix}$$

（1）计算样本均值向量 $\bar{x}$ 为

$$\bar{x} = \frac{1}{6}\sum_{j=1}^{6} \boldsymbol{x}_j = \begin{bmatrix} 4 \\ 5 \end{bmatrix}$$

（2）计算样本协方差矩阵 S 为

$$S = \begin{bmatrix} 3.2 & 3.4 \\ 3.4 & 4.0 \end{bmatrix}$$

（3）计算样本相关矩阵 R

对样本数据规范化

$$x_{ij}^* = \frac{x_{ij} - \bar{x}_i}{\sqrt{s_{ii}}}, \quad i = 1, 2;\ j = 1, 2, 3, 4, 5, 6$$

矩阵改写为

$$X = \begin{bmatrix} -\frac{\sqrt{5}}{2} & -\frac{\sqrt{5}}{4} & -\frac{\sqrt{5}}{4} & 0 & \frac{\sqrt{5}}{4} & \frac{3\sqrt{5}}{4} \\ -\frac{3}{2} & -\frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{3}{2} \end{bmatrix}$$

样本相关矩阵为

$$R = \frac{1}{n-1}XX^T = \begin{bmatrix} 1 & \frac{17\sqrt{5}}{40} \\ \frac{17\sqrt{5}}{40} & 1 \end{bmatrix}$$

（4）求样本相关矩阵 R 的 k 个特征值和对应的 k 个单位特征向量。

求解 R 的特征方程

$$|R - \lambda I| = 0$$

得到 R 的 2 个特征值

$$\lambda_1 = 1 + \frac{17\sqrt{5}}{40}, \quad \lambda_2 = 1 - \frac{17\sqrt{5}}{40}$$

求对应的单位特征向量

$$a_1 = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})^T, \quad a_2 = (-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})^T$$

假设要求主成分的累计方差贡献率大于 90%，那么只需取第一个主成分即可，即 $k = 1$，因为

$$\frac{\lambda_1}{\lambda_1 + \lambda_2} = 0.9752$$

(5) 求样本主成分以单位特征向量为系数进行线性变换，求出主成分

$$y_1 = a_1^T \boldsymbol{x} = \frac{\sqrt{2}}{2}x_1 + \frac{\sqrt{2}}{2}x_2$$

(6) 由特征值和特征向量求出主成分的因子负荷量和对变量 x_i 的贡献率。

因子负荷量:（此处仍用 s_{ii} 表示规范化后的 s_{ii}^*）

$$\rho(y_1, x_1) = \frac{\sqrt{\lambda_1} a_{11}}{\sqrt{s_{11}}} = 0.9875$$

$$\rho(y_1, x_2) = \frac{\sqrt{\lambda_1} a_{21}}{\sqrt{s_{22}}} = 0.9875$$

贡献率:

$$\nu_1 = \rho^2(x_1, y1) = 0.9752$$

$$\nu_2 = \rho^2(x_2, y1) = 0.9752$$

（7）求得 $1 \times n$ 样本主成分矩阵为

$$\begin{aligned} Y = a_1^T X &= (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}) \begin{bmatrix} -\frac{\sqrt{5}}{2} & -\frac{\sqrt{5}}{4} & -\frac{\sqrt{5}}{4} & 0 & \frac{\sqrt{5}}{4} & \frac{3\sqrt{5}}{4} \\ -\frac{3}{2} & -\frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{3}{2} \end{bmatrix} \\ &= (-\frac{\sqrt{10}+3\sqrt{2}}{4}, -\frac{\sqrt{10}+2\sqrt{2}}{8}, -\frac{\sqrt{10}}{8}, 0, \frac{\sqrt{10}+2\sqrt{2}}{8}, \frac{3\sqrt{10}+6\sqrt{2}}{8}) \end{aligned}$$

16.2　证明样本协方差矩阵 S 是总体协方差矩阵 Σ 的无偏估计。

设 $x_1, x_2, \cdots, x_n$ 是从总体 X 中得到的随机样本，记 X 的期望为

$$E(X) = E(x_i) = \mu, \quad i = 1, 2, \cdots, n$$

协方差矩阵为

$$\Sigma = Cov(X, X) = E[(X-\mu)(X-\mu)^T] = E[(x_i-\mu)(x_i-\mu)^T] = Cov(x_i, x_i), \quad i = 1, 2, \cdots, n$$

样本 $x_1, x_2, \cdots, x_n$ 的均值为

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

考虑样本独立性，有 $Cov(x_i, x_j) = 0,\ i \neq j,\ i, j = 1, 2, \cdots, n$，则样本均值向量的期望与协方差矩阵如下:

$$E(\bar{x}) = \frac{1}{n} E(\sum_{i=1}^{n} x_i) = \frac{n\mu}{n} = \mu$$

$$\begin{aligned} Cov(\bar{x}, \bar{x}) &= Cov(\frac{1}{n} \sum_{i=1}^{n} x_i, \frac{1}{n} \sum_{j=1}^{n} x_j) \\ &= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} x_j Cov(x_i, x_j) \\ &= \frac{1}{n^2} \sum_{i=1}^{n} Cov(x_i, x_i) \\ &= \frac{1}{n} \Sigma \end{aligned}$$

于是有

$$
\begin{aligned}
E(S) &= E[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})(x_i-\bar{x})^T] \\
&= E[\frac{1}{n-1}\sum_{i=1}^{n}((x_i-\mu)-(\bar{x}-\mu))((x_i-\mu)-(\bar{x}-\mu))^T] \\
&= \frac{1}{n-1}\sum_{i=1}^{n}E[(x_i-\mu)(x_i-\mu)^T] - \frac{1}{n-1}E[n\bar{x}\bar{x}^T - n\bar{x}\mu^T - n\mu\bar{x}^T + n\mu\mu^T] \\
&= \frac{1}{n-1}\sum_{i=1}^{n}Cov(x_i,x_i) - \frac{n}{n-1}E[(\bar{x}-\mu)(\bar{x}-\mu)^T] \\
&= \frac{n}{n-1}\Sigma - \frac{n}{n-1}Cov(\bar{x},\bar{x}) \\
&= \frac{n}{n-1}\Sigma - \frac{1}{n-1}\Sigma \\
&= \Sigma
\end{aligned}
$$

所以样本协方差矩阵 S 是总体协方差矩阵 Σ 的无偏估计。

16.3 设 X 为数据规范化样本矩阵，则主成分等价于求解以下最优化问题：

$$
\begin{aligned}
\min_{L} \quad & ||X-L||_F \\
\text{s.t.} \quad & rank(L) \le k
\end{aligned}
$$

这里 F 是弗罗贝尼乌斯范数，k 是主成分个数。试问为什么？

设 X 为 $m \times n$ 矩阵, 由弗罗贝尼乌斯范数定义可知，求解最优化问题：

$$
\begin{aligned}
\min_{L} \quad & ||X-L||_F \\
\text{s.t.} \quad & rank(L) \le k
\end{aligned}
$$

等价于求解下述最优化问题：

$$
\begin{aligned}
\min_{L} \quad & ||\frac{X^T}{\sqrt{n-1}} - \frac{L^T}{\sqrt{n-1}}||_F \\
\text{s.t.} \quad & rank(L) \le k
\end{aligned}
$$

为方便讨论，下述仍将 $||\frac{X^T}{\sqrt{n-1}} - \frac{L^T}{\sqrt{n-1}}||_F$ 记作 $||X'-L'||_F$。

（1）考虑利用奇异值分解算法进行主成分分析（主成分个数为 k）：

1）构造新的 $n \times m$ 矩阵 $X' = \frac{1}{\sqrt{n-1}}X^T$。

2）对矩阵 X' 进行截断奇异值分解，得到 $X' \approx U\Sigma V^T$，有 k 个奇异值、奇异向量，那么矩阵 V 的列向量就是协方差矩阵 $S_X = (X')^T X'$ 的单位特征向量。

3）因此，V 的前 k 列对应 X 的 k 个样本主成分。

4）求解 $Y = V^T X$ 即为样本主成分矩阵。

（2）再考虑求解最优化问题：

对 X' 进行奇异值分解，记作 $X' = U\Sigma V^T$，有

$$||X' - L'||_F = ||U\Sigma V^T - L'||_F = ||U\Sigma V^T - UU^T L'VV^T||_F = ||\Sigma - U^T L'V||_F$$

由于 Σ 为 $m \times n$ 矩形对角矩阵，记对角元为 $\sigma_1, \sigma_2, \cdots, \sigma_n$，且 $rank(L) \leq k$，则当 $U^T L'V$ 是秩为 k、对角元为 $\sigma_1, \sigma_2, \cdots, \sigma_k$ 的 $m \times n$ 矩形对角矩阵时，$||X' - L'||_F = ||\Sigma - U^T L'V||_F$ 取得最小值。

记

$$S = \begin{bmatrix} S_1 & 0 \\ 0 & 0 \end{bmatrix}$$

其中 S_1 为 k 阶对角阵 $\mathrm{diag}\{\sigma_1, \sigma_2, \cdots, \sigma_k\}$。

则 $L = (\sqrt{n-1}L')^T = \sqrt{n-1}VS^T U^T$ 即为该最优化问题的解。

一方面，从上述求解过程中可以看出，两个问题均需要求解 X' 的奇异值分解，只需得到的分解结果矩阵做一定的调整即可分别得到主成分和最优化问题的解；

另一方面，结合定理 15.2、定理 15.3，可以看出题中的最优化问题是要求矩阵 X 在弗罗贝尼乌斯范数意义下的最优近似（即奇异值分解），$rank(L) \leq k$ 表示秩为 k 的截断奇异值分接得到的矩阵 A_k 能够达到这个最优值，而求 X 的主成分的奇异值分解算法则主要是对矩阵 X' 进行截断奇异值分解。

19 马尔可夫链蒙特卡罗法

19.1 用蒙特卡罗积分法求

$$\int_{-\infty}^{\infty} x^2 \exp(-\frac{x^2}{2})\mathrm{d}x$$

令 $f(x) = \sqrt{2\pi}x^2$

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2})$$

$p(x)$ 是标准正态分布的密度函数。

使用蒙特卡罗积分法，按照标准正态分布在区间 $(-\infty, \infty)$ 抽样 $x_1, x_2, \cdots, x_n$(取 $n = 1000000$)，取其平均值，用程序求解得

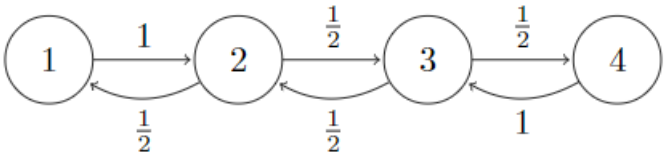
$$\int_{-\infty}^{\infty} x^2 \exp(-\frac{x^2}{2})\mathrm{d}x = \int_{-\infty}^{\infty} \sqrt{2\pi}x^2 \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2})\mathrm{d}x \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i) = 2.5084$$

```python
import numpy as np
import math
#按照标准正态分布抽样
X = np.random.normal(loc=0, scale=1, size=1000000)
#计算平均值
I = math.sqrt(2*np.pi)*np.dot(X,X)/1000000
print('I=',round(I,4))
```

19.4 验证具有以下转移概率矩阵的马尔可夫链是不可约的，但是周期性的。

$$P=\begin{bmatrix}0 & 1/2 & 0 & 0\\ 1 & 0 & 1/2 & 0\\ 0 & 1/2 & 0 & 1\\ 0 & 0 & 1/2 & 0\end{bmatrix}$$

由转移概率矩阵可知，该马尔可夫链如下图所示



直观上来看，任意两个状态 i,j 之间都能互相到达，从而该马尔可夫链是不可约的。下面通过计算验证，即计算对于任意 $i,j\in\mathcal{S}$，一定存在时刻 $t(t>0)$ 满足

$$P(X_t=i|X_0=j)>0$$

计算 P^2,P^3

$$P^2=\begin{bmatrix}1/2 & 0 & 1/4 & 0\\ 0 & 3/4 & 0 & 1/2\\ 1/2 & 0 & 3/4 & 0\\ 0 & 1/4 & 0 & 1/2\end{bmatrix},\quad P^3=\begin{bmatrix}0 & 3/8 & 0 & 1/4\\ 3/4 & 0 & 5/8 & 0\\ 0 & 5/8 & 0 & 3/4\\ 1/4 & 0 & 3/8 & 0\end{bmatrix}$$

从而有

$$P+P^2+P^3=\begin{bmatrix}1/2 & 7/8 & 1/4 & 1/4\\ 7/4 & 3/4 & 9/8 & 1/2\\ 1/2 & 9/8 & 3/4 & 7/4\\ 1/4 & 1/4 & 7/8 & 1/2\end{bmatrix}$$

可以看出矩阵 $P+P^2+P^3$ 中所有元素都大于 0，则对于任意 $i,j\in\mathcal{S}$，一定存在时刻 $t(t>0)$ 满足 $P(X_t=i|X_0=j)>0$。

下面证明周期性。

由于 P 的对角元均为 0，则从任意状态出发，至少需要跳转 2 次才能返回该状态；又 P^2 的对角元均大于 0，则从任意状态出发，经过偶数次跳转可以返回该状态。

从而对任意状态 $i \in \mathcal{S}$，时刻 0 从状态 i 出发，t 时刻返回状态的所有时长的最大公约数为 2，因此该马尔可夫链是周期性的。

19.7 假设进行伯努利试验，后验概率为 $P(\theta|y)$，其中变量 $y \in \{0,1\}$ 表示实验可能的结果，变量 θ 表示结果为 1 的概率。再假设先验概率 $P(\theta)$ 遵循 Beta 分布 $B(\alpha,\beta)$，其中 $\alpha=1,\beta=1$；似然函数 $P(y|\theta)$ 遵循二项分布 $\mathrm{Bin}(n,k,\theta)$，其中 $n=10,k=4$，即实验进行 10 次其中结果为 1 的次数为 4。试用 Metropolis-Hastings 算法求后验概率分布 $P(\theta|y) \propto P(\theta)P(y|\theta)$ 的均值和方差。（提示：可采用 Metropolis 选择，即假设建议分布是对称的。）

由已知,先验概率 $P(\theta)$ 遵循 $B(1,1)$,即均匀分布 $U(0,1)$,似然函数 $P(y|\theta)$ 遵循二项分布 $Bin(10,4,\theta)$,则对应的密度函数为

$$p(\theta)=1,\quad p(y|\theta)=\binom{10}{4}\theta^4(1-\theta)^6=210\theta^4(1-\theta)^6$$

于是由后验概率分布 $P(y|\theta)\propto P(\theta)P(y|\theta)$，可以取后验概率分布的密度函数为 $p(\theta)=\theta^4(1-\theta)^6$。

假设建议分布是对称的，即对任意的 θ 和 θ' 有

$$q(\theta,\theta')=q(\theta',\theta)$$

不妨取建议分布为 (0,1) 上的均匀分布。

这时，接收分布 $\alpha(\theta,\theta')$ 简化为

$$\alpha(\theta,\theta')=\min\left\{1,\frac{p(\theta')}{p(\theta)}\right\}=\min\left\{1,\frac{\theta'^4(1-\theta')^6}{\theta^4(1-\theta)^6}\right\}$$

利用 Metropolis-Hastings 算法求后验概率分布 $P(\theta|y)\propto P(\theta)P(y|\theta)$ 的均值为 0.420427，方差为 0.019162。

```
import numpy as np
#定义后验概率分布的密度函数
def p(x):
    return np.power(x, 4) * np.power(1-x, 6)

#Metropolis-Hastings算法
def Metropolis(m, n):
    X = np.ones(n) / 2 #取初始值，避免除数为0的情况
    for i in range(n):
        x = X[i-1]
        x1 = np.random.uniform(0, 1) #按建议分布随机抽取一个候选状态x1
        alpha = min(1, p(x1) / p(x)) #计算接受概率
        u = np.random.uniform(0, 1)
```

```
            if u <= alpha:
                X[i] = x1
            else:
                X[i] = x
        return np.mean(X[m:n]), np.var(X[m:n])

    #输出后验概率分布的均值和方差
    mean, var = Metropolis(5000,10000)
    print("后验概率分布的均值为", round(mean,6), '\n'
          "后验概率分布的方差为", round(var,6))
```