人工智能实验报告

林展辉

计算机 94 班 2194411249

1 主观贝叶斯折线图

1.1 实验内容

主观贝叶斯画图，在证据不确定的情况下，根据充分性量度 LS、必要性量度 LN、E 的先验概率 $P(E)$ 和 H 的先验概率 $P(H)$ 作为前提条件，分析 $P(H/S)$ 和 P(E/S) 的关系。

1.2 实验原理

1.2.1 证据不确定性的表示

1. 在主观 Bayes 方法中，证据的不确定性用概率表示。对于证据 E，由用户根据观察 S 给出 $P(E/S)$ ，即动态强度。用 $P(E/S)$ 描述证据的不确定性（证据 E 不是可以直接观测的）。

2. 证据肯定存在时，$P(E/S) = 1$

3. 证据肯定不存在时，$P(E/S) = 0$

4. 证据具有不确定性时，$0 < P(E/S) < 1$

1.2.2 LN 和 LS 的意义

1. 当证据 E 愈是支持 H 为真时，则应是使相应的 LS 值愈大。若证据 E 对 H 愈是必要，则相应 LN 的值愈小。

2. 不能出现 LS>1 且 LN>1 的取值因为：LS>1：表明证据 E 是对 H 有利的证据。LN>1：表明证据 E 是对 H 有利的证据。

3. 不能出现 LS<1 且 LN<1 的取值因为：LS<1: 表明证据 E 是对 H 不利的证据。LN<1：表明证据?E 是对 H 不利的证据。

4. 一般情况下，取 LS>1，LN<1。

1.2.3 证据不确定的情况

在现实中，证据肯定存在和肯定不存在的极端情况是不多的，更多的是介于二者之间的不确定情况。对初始证据来说，由于用户对客观事物或现象的观察不是很精确，因而所提供的证据是不确定的；另外，一条知识的证据往往来源于另一条知识推出的结论，一般也具有某种程度的不确定性。所以我们要在 S 对 E 的观察的先验概率 $0 < P(E/S) < 1$ 的情况下确定 H 的后验概率 $P(H/S)$。在证据确定的情况下，我们因该用杜达等人 1976 年证明了的公式来进一步讨论：

$$P(H/S) = P(H/E) \times P(E/S) + P(H/\neg E) \times P(\neg E/S) \tag{1}$$

分四种情况讨论这个公式：

1. $P(E/S) = 1$ 当 $P(E/S) = 1$ 时，$P(\neg E/S) = 0$ 。此时公式变成：

$$P(H/S) = P(H/E) = \frac{LS \times P(H)}{(LS-1) \times P(H) + 1} \tag{2}$$

2. $P(E/S) = 0$ 当 $P(E/S) = 0$ 时，$P(\neg E/S) = 1$ 此时公式变成：

$$P(H/S) = P(H/\neg E) = \frac{LN \times P(H)}{(LN-1) \times P(H) + 1} \tag{3}$$

3. $P(E/S) = P(E)$ 当 $P(E/S) = P(E)$ 时，表示 E 与 S 无关。利用全概率公式就将公式变为：

$$P(H/S) = P(H/E) \times P(E) + P(H/\neg E) \times P(\neg E) = P(H) \tag{4}$$

4. 当 P(E/S) 为其它值时，通过分段线性插值就可得到计算 P(H/S) 的公式：

$$P(H/S) = \begin{cases} P(H/\neg E) + \frac{P(H)-P(H/\neg E)}{P(E)} \times P(E/S) & 0 \le P(E/S) < P(E) \\ P(H) + \frac{P(H/E)-P(H)}{1-P(E)} \times [P(E/S) - P(E)] & P(E) \le P(E/S) \le 1 \end{cases} \tag{5}$$

1.3 实验结果

输入:

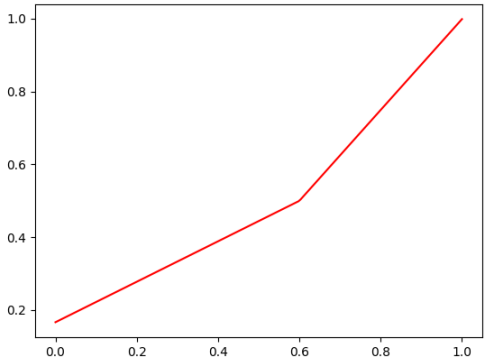
pH= 0.5

pE= 0.3

LN= 0.2

LS= 1000

输出:



2 九宫问题

2.1 实验内容

在一个 3x3 的方格棋盘上放置 8 个标有 1、2、3、4、5、6、7、8 数字的将牌，留下一个空格（用　表示)。规定: 与空格相邻的将牌可以移入空格。要求: 利用非启发式算法和 A* 算法，寻找一条从初始状态到目标状态的将牌移动路线。

2.2 实验原理

2.2.1 搜索

搜索的一般描述如算法 1 所述。

算法 1 搜索算法

输入: 初始节点 S_0，目标节点条件

输出: 搜索结果

1: 把初始节点 S_0 放入 OPEN 表，并建立目前只包含 S_0 的图，记为 G;
2: 检查 OPEN 表是否为空，若为空则问题无解，退出;
3: 把 OPEN 表的第一个节点取出放入 CLOSE 表，并计该节点为 n;
4: 考察节点 n 是否为目标节点。若是，则求得了问题的解，退出;
5: 扩展节点 n, 生成一组子节点。把其中不是节点 n 先辈的那些子节点记做集合 M，并把这些子节点作为节点 n 的子节点加入 G 中;
6: 针对 M 中子节点的不同情况，分别进行如下处理:
7: 对于那些未曾在 G 中出现过的 M 成员设置一个指向父节点（即节点 n 的指针，并把它们放入 OPEN 表;(不在 OPEN 表)
8: 对于那些先前已经在 G 中出现过的 M 成员，确定是否需要修改它指向父节点的指针;(在 OPEN 表中)
9: 对于那些先前已在 G 中出现并且已经扩展了的 M 成员，确定是否需要修改其后继节点指向父节点的指针;(在 CLOSE 表中)
10: 按某种搜索策略对 OPEN 表中的节点进行排序;
11: 转第 2 步。

2.2.2 广度搜索

广度优先搜索的基本思想: 从初始节点 S_0 开始，逐层地对节点进行扩展，并考察它是否为目标节点。在第 n 层的节点没有全部扩展并考察之前，不对第 n + 1 层的节点进行扩展。OPEN 表中节点总是按进入的先后顺序排列，先进入的节点排在前面，后进入的排在后面。

算法如算法 2 所述。

算法 2 广度搜索

输入: 初始节点 S_0，目标节点条件

输出: 搜索结果

1: 把初始节点 S_0 放入 OPEN 表。
2: 检查 OPEN 表是否为空，若为空则问题无解，退出;
3: 把 OPEN 表的第一个节点取出放入 CLOSE 表，并计该节点为 n;
4: 考察节点 n 是否为目标节点。若是，则求得了问题的解，退出;
5: 若节点 n 不可扩展，则转第 2 步。
6: 扩展节点 n，将其子节点放入 OPEN 表的尾部，并为每一个子节点都配置指向父节点的指针，然后转第 2 步。

算法 3 A* 算法

输入: 初始节点 S_0，目标节点条件，估值函数 $h(x)$

输出: 搜索结果

1: 把初始节点 S_0 放入 OPEN 表，并建立目前只包含 S_0 的图，记为 G;
2: 检查 OPEN 表是否为空，若为空则问题无解，退出;
3: 把 OPEN 表的第一个节点取出放入 CLOSE 表，并计该节点为 n;
4: 考察节点 n 是否为目标节点。若是，则求得了问题的解，退出;
5: 扩展节点 n, 生成一组子节点。把其中不是节点 n 先辈的那些子节点记做集合 M，并把这些子节点作为节点 n 的子节点加入 G 中;
6: 针对 M 中子节点的不同情况，分别进行如下处理:
7: 对于那些未曾在 G 中出现过的 M 成员设置一个指向父节点（即节点 n 的指针，并把它们放入 OPEN 表;(不在 OPEN 表)
8: 对于那些先前已经在 G 中出现过的 M 成员，确定是否需要修改它指向父节点的指针;(在 OPEN 表中, 对 g(x) 进行更新)
9: 对于那些先前已在 G 中出现并且已经扩展了的 M 成员，确定是否需要修改其后继节点指向父节点的指针;(在 CLOSE 表中, 对节点 n 子节点的子节点更新 g(x)
10: 对 OPEN 表中的节点按估价函数进行排序;
11: 转第 2 步。

2.2.3 A* 算法

如果一般搜索过程满足如下限制，则它就称为 A* 算法:

(1) 把 OPEN 表中的节点按估价函数:$f(x) = g(x) + h(x)$ 的值从小至大进行排序（一般搜索过程的第 7 步)。

(2) g(x) 是从初始节点 SO 到节点 x 的路径的代价，g(x) 是对 g*(x) 的估计，g(x)>0。(3) h(x) 是 h*(x) 的下界，即对所有的 x 均有: h(x) h*(x)。其中，g*(x) 是从初始节点 S_0 到节点 x 的最小代价;h*(x) 是从节点 x 到目标节点的最小代价。

算法如算法 3 所述。

2.3 实验结果

输入:

开始状态: [1,2,3,5,0,6,4,7,8]

结束状态: [1,2,3,4,5,6,7,8,0]

输出:

BFS: ['L', 'D', 'R', 'R']

space: 51

time: 0.001584768295288086

A*: ['L', 'D', 'R', 'R']

space: 13

time: 0.0008876323699951172

2.4 实验结论

广度优先搜索算法比较可靠，只要问题有解，这种算法总可以得到解，而且得到的是最优解。但同时也有很大的缺点，例如它的盲目性较大，当目标节点距初始节点较远时将会产生许多无用节点，搜索效率低。相比之下，使用到代价函数的 A* 搜索算法的效率较高。