

# 人工智能原理与实践

张鑫航

国防科技大学

版本：1.0

更新：2024年1月19日



各位同学，关于考试题型及内容范围，简要介绍如下：

1. 考试题型：单项选择题 15 题，每题 2 分，共计 30 分；多项选择题 5 题，每题 4 分，共计 20 分；简答计算题 4 题，分数构成为  $10+10+15+15$ ，共计 50 分。
2. 考试范围：80% 以上都为课件上知识点，或者能从课件上找到参考。重点考察知识的运用能力。

简答计算题范围包括：搜索策略（能给出图搜索或树搜索框架下不同搜索策略的扩展节点序列、解序列和解代价）、博弈搜索算法、基于谓词逻辑的归结、贝叶斯推理、有/无监督学习、深度学习、强化学习等。

考算法也不是写代码，是算法原理相关知识点的运用。

# 目录

<b>1 基本概念</b>	<b>3</b>
1.1 三大学派 . . . . .	3
1.2 Agent . . . . .	3
<b>2 搜索</b>	<b>7</b>
2.1 基本搜索算法 . . . . .	7
2.1.1 树搜索算法和图搜索算法 . . . . .	7
2.2 盲目搜索策略 . . . . .	7
2.3 最佳优先搜索 . . . . .	8
2.3.1 评价函数 . . . . .	9
2.3.2 一致代价搜索 . . . . .	9
2.3.3 贪婪搜索 . . . . .	9
2.3.4 A <sup>*</sup> 搜索 . . . . .	9
2.4 博弈树搜索 . . . . .	13
2.4.1 博弈问题 . . . . .	13
2.4.2 minimax 搜索算法 . . . . .	14
2.4.3 $\alpha - \beta$ 剪枝 . . . . .	15
<b>3 知识表示与推理</b>	<b>15</b>
3.1 逻辑 Agent . . . . .	15
3.1.1 基于知识的 Agent . . . . .	16
3.1.2 命题逻辑的推理 . . . . .	16
3.2 逻辑谓词 . . . . .	20
3.2.1 谓词逻辑的基本概念 . . . . .	20
3.2.2 逻辑谓词的归结 . . . . .	21

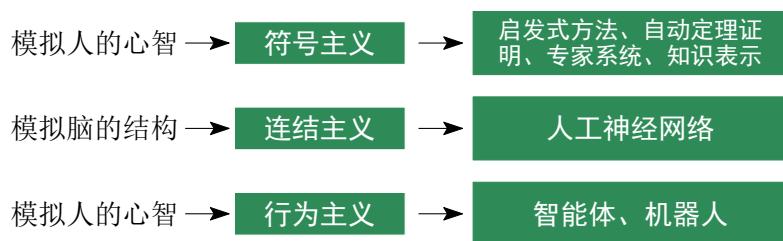
3.2.3 定理证明和求解	25
3.3 不确定知识与推理	28
3.3.1 不确定性描述	28
3.3.2 不确定领域的知识表示	28
3.3.3 贝叶斯网络	29
3.3.4 基于贝叶斯网络的推理	30
<b>4 机器学习</b>	<b>31</b>
4.1 机器学习概述	31
4.1.1 基本概念	31
4.2 有监督学习基本方法	34
4.2.1 回归问题	36
4.2.2 分类问题	36
4.3 无监督学习基本方法	42
4.3.1 相似性的度量	42
4.3.2 划分法	43
4.3.3 层次法	46
4.4 人工神经网络	48
4.4.1 生物神经元与神经网络	48
4.4.2 人工神经网络三大要素	49
4.4.3 感知器	55
4.4.4 BP 神经网络	57
<b>5 深度学习与强化学习</b>	<b>60</b>
5.1 深度学习	61
5.1.1 深度学习的基本概念	61
5.1.2 深度学习的训练过程	62

5.1.3	卷积神经网络 . . . . .	64
5.2	强化学习 . . . . .	67
5.2.1	强化学习基本概念 . . . . .	67
5.2.2	马尔科夫决策过程 . . . . .	68
5.2.3	典型强化学习算法 . . . . .	71
5.3	深度强化学习 . . . . .	76
5.3.1	深度 Q 学习 . . . . .	76

# 1 基本概念

## 1.1 三大学派

人工智能研究的三大学派：



- 符号主义学派的核心是符号演算与机器推理
- 连接主义学派的核心是神经网络与深度学习
- 行为主义学派推崇控制、自适应与进化计算

## 1.2 Agent

Agent 是能够通过传感器感知环境，并且通过执行器对环境产生影响的任何东西。



- Agent 函数：将感知序列映射为行动：

$$f : P^* \rightarrow A$$

- Agent 程序：实现 Agent 函数，并在物理平台上运行

$$\text{Agent} = \text{物理平台} + \text{Agent 程序}$$

Agent 三个要素：

- 感知环境能力
- 作用环境能力
- 感知信息与动作决策之间的映射机制

感知序列：

- 有记忆
- 自身状态
- 之前的行为对当前决策与动作有影响

**定义 1.1 (理性 Agent)** 对每一个可能的感知信息，根据已知的感知序列提供的证据和 Agent 具有的先验知识，理性 Agent 应该选择能使其性能测度最大化的行动。

**定义 1.2 (性能测度)** 性能测度是对行动序列导致期望的环境变化的度量。

性能测度对 Agent 的行为模型具有重要影响。

性能测度	Performance Measurement
环境	Environment
执行器	Actuator
传感器	Sensors

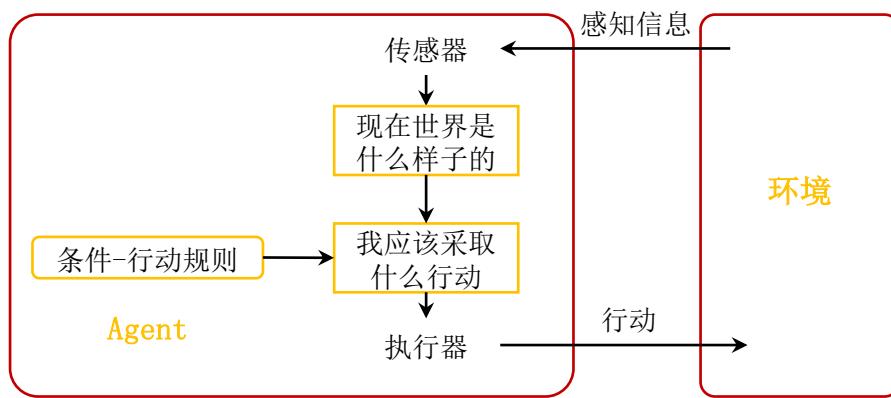
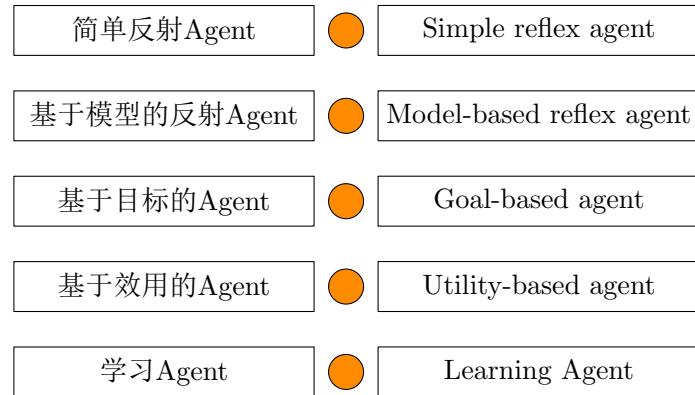
### 环境属性

- 完全可观 vs 部分可观 (Fully observable vs. partially observable)
- 确定的 vs 随机的 (Deterministic vs. stochastic )
- 片段式 vs 延续式 (Episodic vs. sequential)
- 静态 vs 动态 (Dynamic vs. static )
- 离散 vs 连续 ( Discrete vs. continuous )
- 单 Agent vs 多 Agent ( Single agent vs. multi-agent )
- 已知 vs 未知 (Known vs. unknown)

任务环境	魔方 Agnet	吸尘器 Agent	无人侦察机
完全可观	完全可观	部分可观	部分可观
确定的	确定的	随机的	随机的
片段式	延续式	延续式	延续式
静态	静态	动态	动态
离散	离散	连续	连续
单 Agent	单	单	多
已知	已知	未知	未知

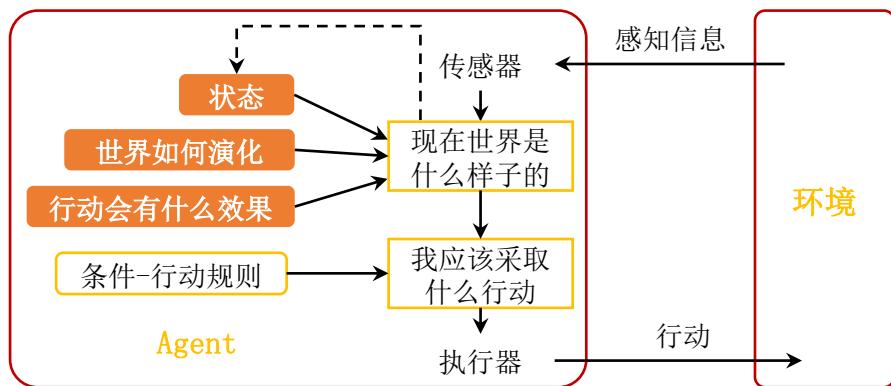
真实的环境是：部分可观察、随机、延续、动态、连续、多 Agent 和未知。

Agent 类型：

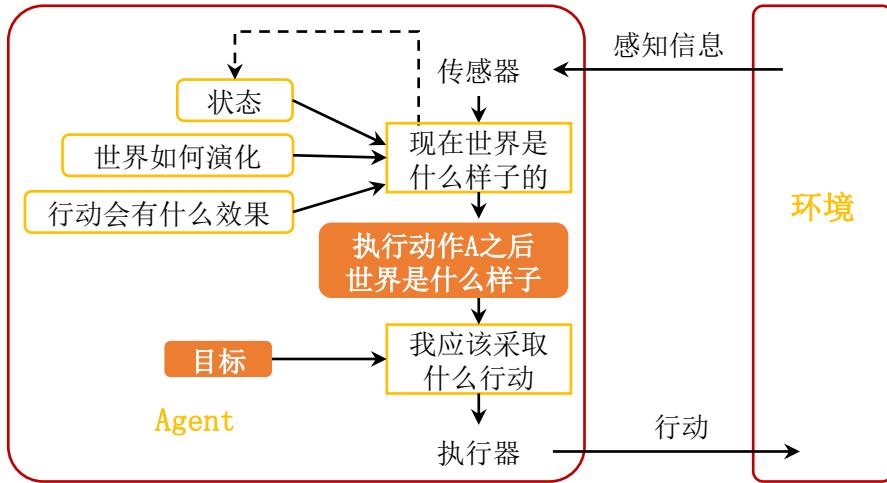


**定义 1.3 (简单反射 Agent)** 简单反射 Agent 仅仅在当前感知的基础上动作忽略其余的感知历史。

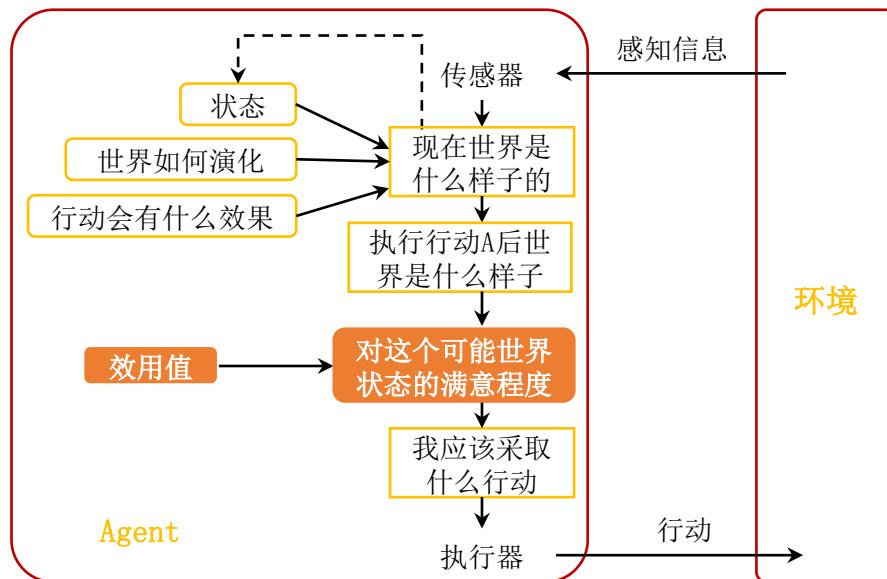
**定义 1.4 (基于模型的反射 Agent)** 基于模型的反射 Agent 可以处理部分可观的环境。其当前状态存储在 Agent 内部，它描述不可见的环境的一部分。



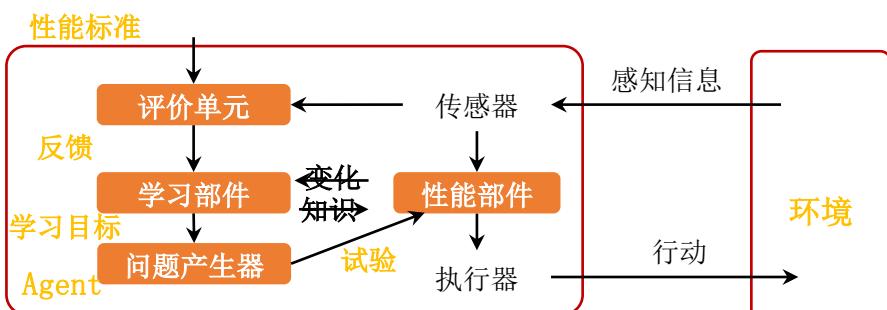
**定义 1.5 (基于目标的 Agent)** 利用“目标”信息，基于目标的 Agent 进一步扩展了基于模型的反射 Agent 的功能。



**定义 1.6 (基于效用的 Agent)** 使用效用函数得到对 Agent 的行动进行性能度量 Agent 选择最大效用的行动。



**定义 1.7 (学习 Agent)** 学习允许 Agent 在最初未知的环境中运行。与只具有最初知识相比，越来越胜任要执行的任务。



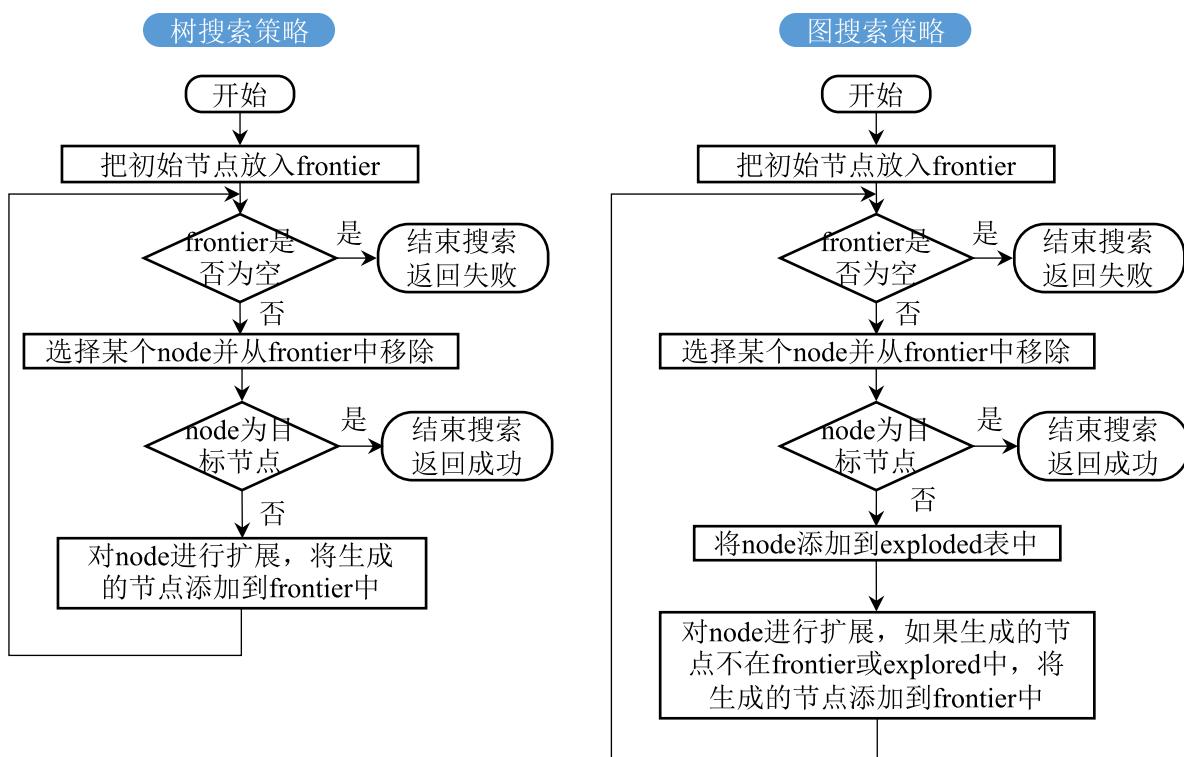
表征环境状态的方式

- 原子化  
每种状态都是内部结构未知的黑盒
- 要素化  
每种状态由固定的属性和值组成
- 结构化  
每个状态都包括对象，每个状态都具有与其他对象的属性和关系

## 2 搜索

### 2.1 基本搜索算法

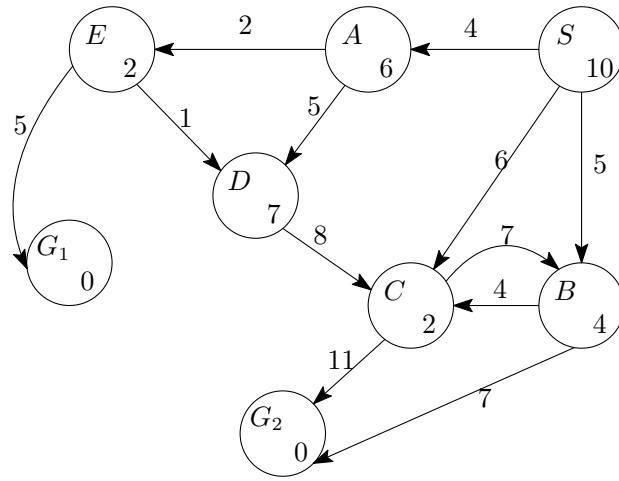
#### 2.1.1 树搜索算法和图搜索算法



### 2.2 盲目搜索策略

**例 2.1** 如图， $S$  为初始状态， $G_1$  和  $G_2$  为目标状态。忽略各状态之间的代价，注意连线是有方向的。在其他条件都相同的情况下，按照字母先后顺序对节点进行扩展。

- 应用宽度优先搜索策略，指出树搜索算法将会到达的目标状态，并按顺序写出搜索过程中所扩展出的节点。



2. 应用宽度优先搜索策略，指出图搜索算法将会到达的目标状态，并按顺序写出搜索过程中所扩展出的节点。

**树搜索：**达到目标状态为  $G_2$ ，解序列： $S \rightarrow B \rightarrow G_2$

Step	frontier
Step 1	$A, B, C$
Step 2	$B, C, D, E$
Step 3	$C, D, E, C, G_2$
Step 4	$D, E, C, G_2, B, G_2$
Step 5	$E, C, G_2, B, G_2, C$
Step 6	$C, G_2, B, G_2, C, D, G_1$
Step 7	$G_2, B, G_2, C, D, G_1, B, G_2$

**图搜索：**达到目标状态为  $G_2$ ，解序列： $S \rightarrow B \rightarrow G_2$

Step	frontier	explored
Step 1	$A, B, C$	$S$
Step 2	$B, C, D, E$	$S, A$
Step 3	$C, D, E, G_2$	$S, A, B$
Step 4	$D, E, G_2$	$S, A, B, C$
Step 5	$E, G_2$	$S, A, B, C, D$
Step 6	$G_2, G_1$	$S, A, B, C, D, E$

## 2.3 最佳优先搜索

### 2.3.1 评价函数

**定义 2.1 (评价函数)** 用来估算经过节点  $n$  的路径代价的函数叫做评价函数，也称评估函数，用  $f(n)$  表示。

**注** 路径的代价包括两部分：

- 一部分是确定性的代价。从起点到  $n$  的路径已经探索出来了，所耗费的代价可以精确计算，记为  $g(n)$
- 一部分是不确定的代价。从  $n$  到终点的可能路径还没有探索出来，其代价只能进行预估，记为  $h(n)$

$$f(n) = g(n) + h(n)$$

### 2.3.2 一致代价搜索

**定义 2.2 (一致代价搜索)** 从初始状态到节点  $n$  已经产生的代价，无信息搜索策略

$$f(n) = g(n)$$

### 2.3.3 贪婪搜索

**定义 2.3 (贪婪搜索)** 贪婪搜索

$$f(n) = h(n)$$

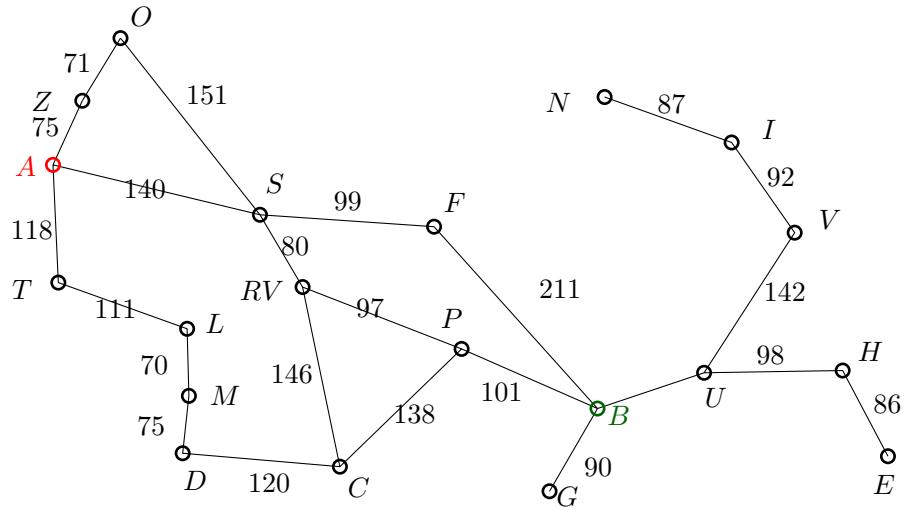
### 2.3.4 A\* 搜索

**定义 2.4 (A\* 搜索)** A\* 搜索

$$f(n) = g(n) + h(n)$$

搜索策略	一致代价	贪婪最佳优先	A*
评估函数	$g(n)$	$h(n)$	$g(n) + h(n)$
完备性	Yes*: 每步代价都 $> \varepsilon > 0$ 即无零代价步 (图、树)	树搜索：否 有限状态图搜索：是	$h(n)$ 若满足特定条件： A* 即完备且最优
最优化	Yes*: 比最优解代价小 的节点数目优先 (图、树)	否	
时间复杂度	$b^{1+[C^*/\varepsilon]}$	$b^m,$ $m$ 搜索空间最大深度	扩展节点以解路径的长度呈指数增长。对于每 步骤代价为常量的问题时间复杂度是最优解 所在深度 $d$ 的函数。
空间复杂度	$b^{1+[C^*/\varepsilon]}$	$b^m,$ 保存所有节点在内存中	

例 2.2 试用以下算法进行求解进行求解。



### 1. 一致代价图搜索

解序列:  $A \rightarrow S \rightarrow RV \rightarrow P \rightarrow B$

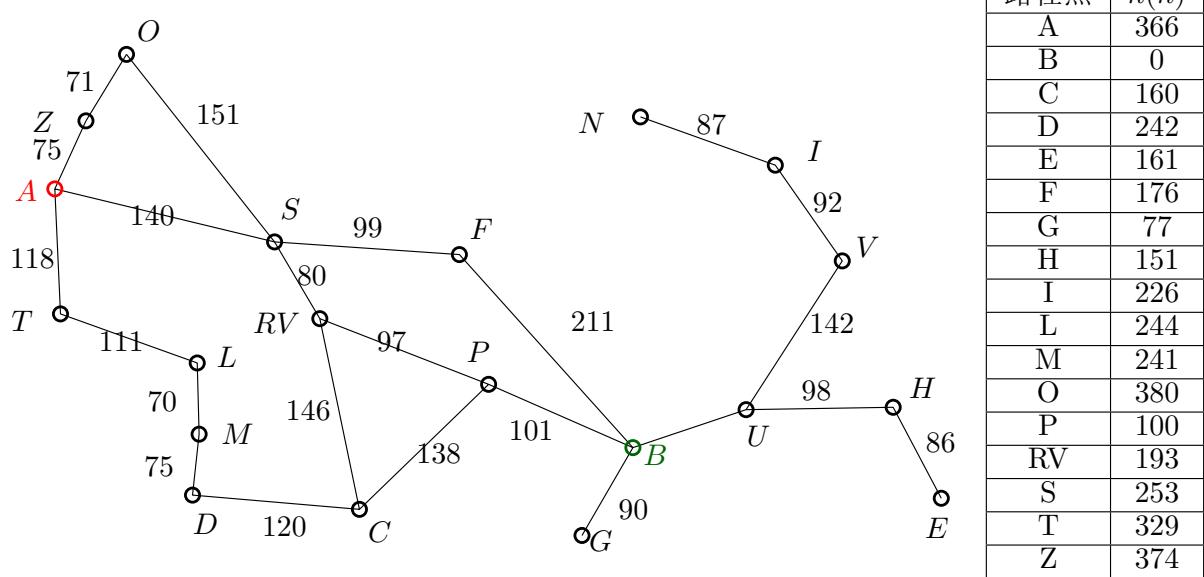
解代价: 408

Step	frontier
Step 1	$A(0)$
Step 2	$Z(75), T(118), S(140)$
Step 3	$T(118), S(140), O(146)$
Step 4	$S(140), O(146), L(229)$
Step 5	$O(146), RV(220), L(229), F(239)$
Step 6	$RV(220), L(229), F(239)$
Step 7	$L(229), F(239), P(317), C(366)$
Step 8	$F(239), M(299), P(317), C(366)$
Step 9	$M(299), P(317), C(366), B(450)$
Step 10	$P(317), C(366), D(374), B(450)$
Step 11	$C(366), D(374), B(408)$
Step 12	$D(374), B(408)$
Step 13	$B(408)$

### 2. 贪婪搜索

解序列:  $A \rightarrow S \rightarrow F \rightarrow B$

解代价: 450



Step	frontier	explored
Step 1	$A(366)$	
Step 2	$S(253), T(329), Z(374)$	$A$
Step 3	$F(176), RV(193), T(329), Z(374), O(380)$	$A, S$
Step 4	$B(0), RV(193), T(329), Z(374), O(380)$	$A, S, F$

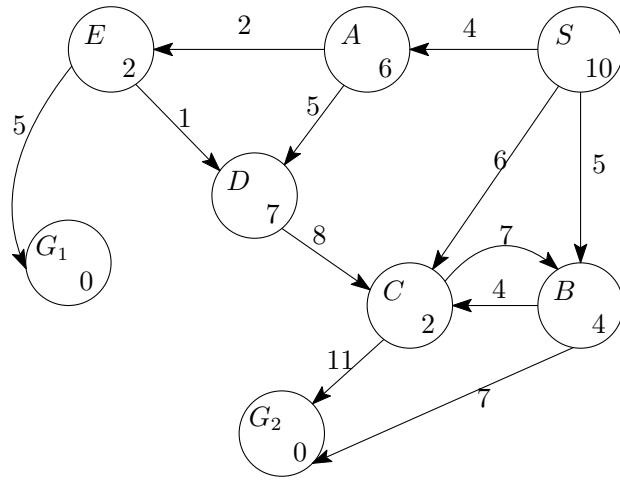
### 3. A\* 搜索

解序列:  $A \rightarrow S \rightarrow RV \rightarrow P \rightarrow B$

解代价: 408

Step	frontier	explored
Step 1	$A(366)$	
Step 2	$S(393), T(447), Z(449)$	$A$
Step 3	$RV(413), F(415), T(447), Z(449), O(671)$	$A, S$
Step 4	$F(415), P(417), T(447), Z(449), C(526), O(671)$	$A, S, RV$
Step 5	$P(417), T(447), Z(449), B(450), C(526), O(671)$	$A, S, RV, F$
Step 6	$B(408), T(447), Z(449), C(526), O(671)$	$A, S, RV, F, P$

**例 2.3**  $S$  为初始状态,  $G_1$  和  $G_2$  为目标状态。各状态之间的代价已标注在连接两个状态的连线上 (注意连线是有方向的), 每个状态到达目标的估计代价标注在其内部。分别应用一致代价、贪婪、 $A^*$  搜索策略, 指出图搜索算法将会到达的目标状态 ( $G_1$  或  $G_2$ ), 并按顺序写出搜索过程中所考察的节点。在其他条件都相同的情况下, 按照字母先后顺序对节点进行扩展。



### 1. 一致代价

解序列:  $S \rightarrow A \rightarrow E \rightarrow G_1$

解代价: 11

Step	frontier	explored
Step 1	$S(0)$	
Step 2	$A(4), B(5), C(6)$	$S$
Step 3	$B(5), C(6), E(6), D(9)$	$S, A$
Step 4	$C(6), E(6), D(9), G_2(12)$	$S, A, B$
Step 5	$E(6), D(9), G_2(12)$	$S, A, B, C$
Step 6	$D(7), G_1(11), G_2(12)$	$S, A, B, C, E$
Step 7	$G_1(11), G_2(12)$	$S, A, B, C, E, D$

### 2. 贪婪

解序列:  $S \rightarrow C \rightarrow G_2$

解代价: 17

Step	frontier	explored
Step 1	$S(0)$	
Step 2	$C(2), B(4), A(6)$	$S$
Step 4	$G_2(0), B(4), A(6)$	$S, C$

### 3. A\*

解序列:  $S \rightarrow A \rightarrow E \rightarrow G_1$

解代价: 11

Step	frontier	explored
Step 1	$S(10)$	
Step 2	$C(8), B(9), A(10)$	$S$
Step 3	$B(9), A(10), G_2(12)$	$S, C$
Step 4	$A(10), G_2(12)$	$S, C, B$
Step 5	$E(8), G_2(12), D(16)$	$S, C, B, A$
Step 6	$G_1(11), G_2(12), D(14)$	$S, C, B, A, E$

## 2.4 博弈树搜索

### 2.4.1 博弈问题

**定义 2.5 (博弈)** 在一定条件下，遵守一定的规则，一个或几个拥有绝对理性思维的人或团队，从各自允许选择的行为或策略中选择并加以实施，并从中各自取得相应结果或收益的过程。

**定义 2.6 (双方信息完备的零和博弈)** 包括以下内容：

- 对抗的双方  
双方参与者轮流采取行动，选取对自己最有利而对对方最不利的对策
- 双方信息完备的零和博弈  
任何一方都了解当前的格局及过去的历史
- 零和  
参与者的利益严格对立，双方得失之和为零

**注 信息完备：** 已知对方走过的棋步以及当前的局面。

**信息不完备：** 参与人并不完全清楚对手的情况，只能进行估计。

**例 2.4** 下面哪些属于双方信息完备的零和博弈：

- A. 五子棋
- B. 桥牌
- C. 中国象棋
- D. 足球
- E. 围棋
- F. 中美经济博弈

博弈问题分析的关键要素：

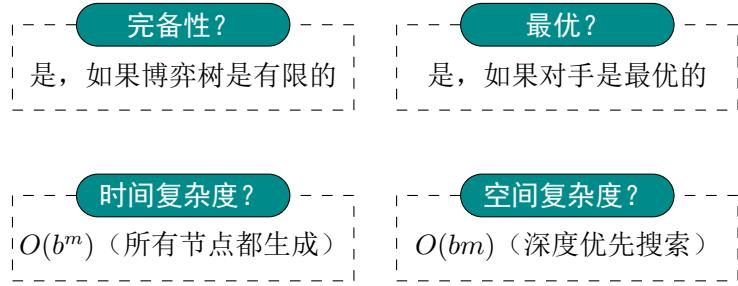
1. 状态: 棋盘上棋子位置布局
2. 动作: 各类棋子合法走步
3. 开局/终局: 游戏开始/结束的状态
4. 终局效用值: 终局下某个棋手的效用值
5. 状态空间: 合法动作作用于当前状态生成的博弈树
6. 某个玩家的解为一个策略: 状态 → 动作

### 2.4.2 minimax 搜索算法

极小极大搜索算法（三个阶段）

1. 生成规定深度的全部博弈树，计算所有最底层节点的静态估计函数值
2. 自底向上逐层计算非终结节点的倒推估计值
  - 对于 MAX 层节点，取其所有子节点的最大值
  - 对于 MIN 层节点，取其所有子节点的最小值
3. 标记最佳走步:
  - 对于 MAX 选手，选择使其  $f$  值最大的走步
  - 对于 MIN 选手，选择使其  $f$  值最小的走步

算法性质分析：



- $b$ : 搜索树的最大分支因子
- $m$ : 状态空间的最大深度（可能 $\infty$ ）

**注** 博弈树的规模一般都很大

- 国际象棋：走 10 个回合后，总节点数量级  $10^{31}$
- 中国象棋：走 10 个回合后，总节点数量级  $10^{32}$
- 围棋：走 10 个回合后，总节点数量级  $10^{51}$

极大极小搜索过程存在的问题由于要生成置顶深度以内的所有节点，其节点数将随着搜索深度的增加成指数增长。

### 2.4.3 $\alpha - \beta$ 剪枝

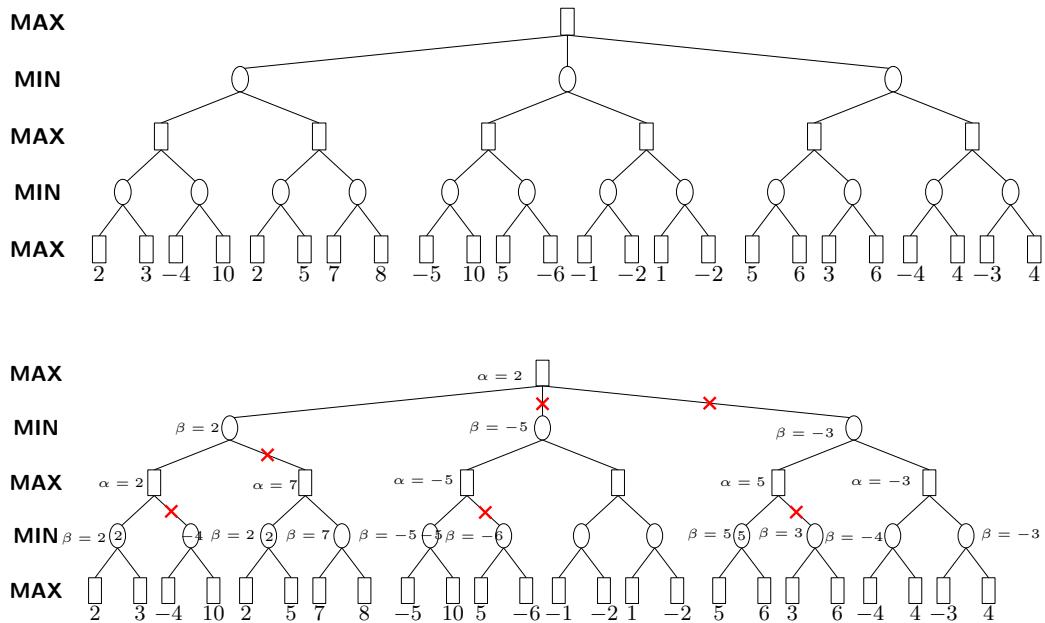
**注** 进行  $\alpha - \beta$  剪枝时，应注意的问题

- 应相邻层比较（MAX 节点和 MIN 节点间），不能进行同层比较
- 至少一个子节点的推导值固定以后，才能向父节点传递
- 在实际搜索时，不能按宽度优先建立搜索树，再进行剪枝，而是按照有界深度优先搜索的方式生成节点，边生成边剪枝。

**注** 特性分析

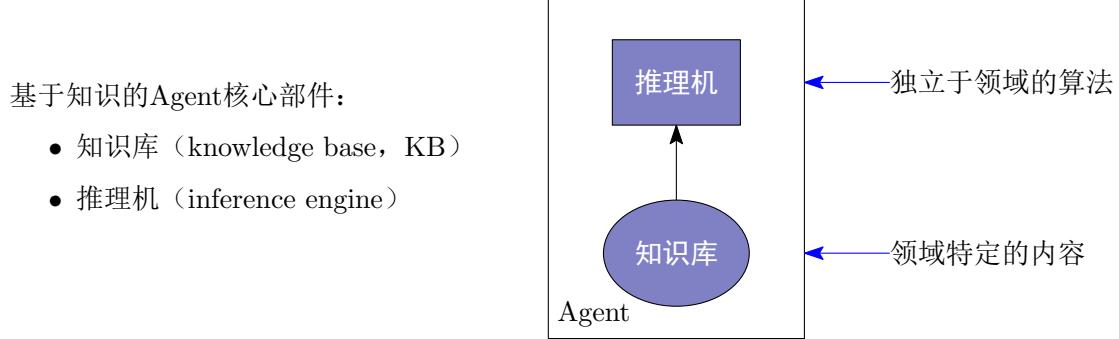
- $\alpha - \beta$  剪枝对计算根节点的 MiniMax 值没有影响
- 剪枝效率最坏情况  $O(b^m)$ （其中  $b$  是分支数， $m$  是解的深度）
- 剪枝效率最好情况  $O(b^{m/2})$ 
  - 不管是 MAX 选手还是 MIN 选手，最好选择都是博弈树的最左分枝，存在较多的剪枝。

**例 2.5** 对下面的博弈树按从左到右的顺序进行  $\alpha - \beta$  剪枝搜索，试标明各生成节点的倒推值，何处发生剪枝及应选择的走步。



## 3 知识表示与推理

### 3.1 逻辑 Agent



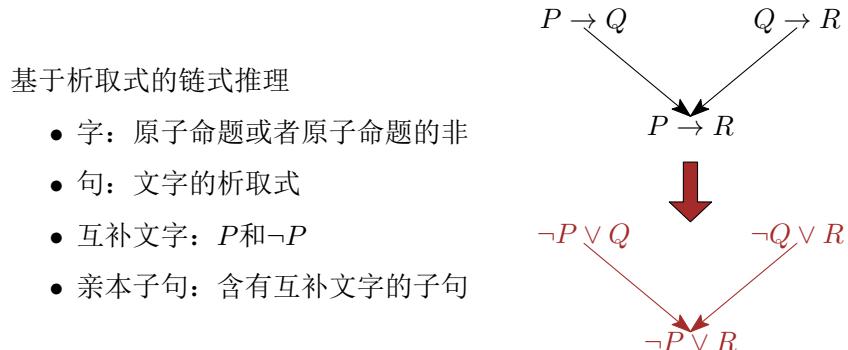
### 3.1.1 基于知识的 Agent

**定义 3.1 (蕴含 (Entailment))** 知识库 KB (一组对已知世界和规则描述的逻辑语句) 蕴含语句  $\alpha$  (某个待确定的逻辑结论), 当且仅当在 KB 为真的所有世界中  $\alpha$  也为真。蕴含表示为

$$KB \models \alpha$$

即, 若  $KB \models \alpha$ , 当且仅当  $M(KB) \subseteq M(\alpha)$

### 3.1.2 命题逻辑的推理



**注 归结 (Resolution):**

链式推理和假言推理都可以转化成相同的推理规则: 两个含有互补文字的亲本子句可以进行推理, 推理得到的结论是去除互补文字后两个子句的析取式。

**定义 3.2 (析取范式)** 设  $A$  为如下形式的命题公式  $B_1 \vee B_2 \vee \dots \vee B_n$ , 其中  $B_i$ , ( $i = 1, 2, \dots, n$ ) 为原子命题或其否定, 则  $A$  称为析取范式或析取式。

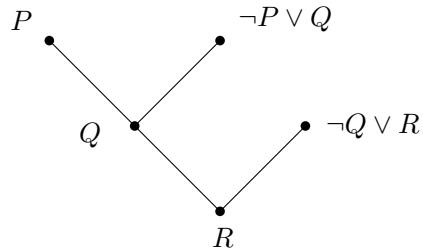
**注 归结中需要注意的问题 1:**

- 进行归结之前, 需要把已知的事实、定理等化成析取范式或者是单个文字的形式, 单个文字是原子命题, 也可以看成一个析取范式。
- 析取范式之间的逻辑关系是合取, 也就是“与”因为子句表示的是已知事实或者定理, 这些子句必须同时成立才能推导出结论。

子句间是合取关系，子句内是析取形式。

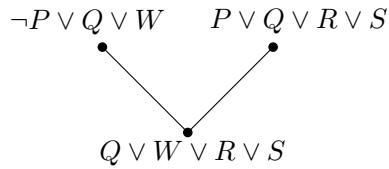
**例 3.1** 选取两个含有互补文字的子句进行归结，产生新的子句，如果新子句和原始子句含有互补文字，那么继续归结，直到再没有亲本子句。

$$P \wedge (\neg P \vee Q) \wedge (\neg Q \vee R)$$

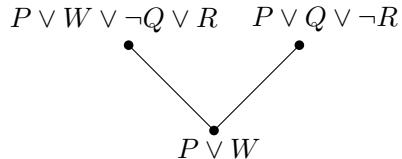


**例 3.2** 将以下问题进行归结：

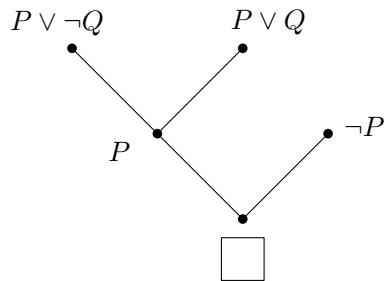
- 将  $\neg P \vee Q \vee W$  与  $P \vee Q \vee R \vee S$  归结。



- 将  $P \vee W \vee \neg Q \vee R$  与  $P \vee Q \vee \neg R$  归结的结果是什么？



- 将  $P \vee \neg Q$  与  $P \vee Q$  以及  $\neg P$  归结



**定义 3.3 (空子句)** 不含任何文字的子句称为空子句。

由于空子句不含有任何文字，也就不能被任何解释所满足因此空子句是永假的，不可满足的。

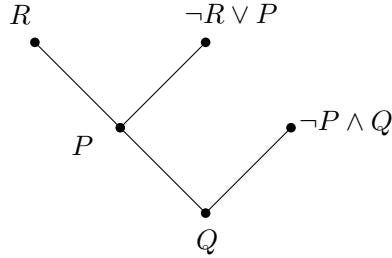
**注** 归结中需要注意的问题 2

- 归结结果为空子句意味着推导出的结果为永假式，说明前面给出的推导条件存在着矛盾，这在用归结原理进行定理证明时会用到。
- 合式公式可直接归结的条件：子句内为析取，子句间为合取
- 合式公式归结步骤：先化为子句的合取式形式，然后对子句进行归结

**例 3.3** 将  $[(R \wedge \neg P) \vee (R \wedge Q)] \wedge (R \rightarrow P)$  化为标准子句。

$$\begin{aligned} & R \wedge (R \vee Q) \wedge (\neg P \vee R) \wedge (\neg P \vee Q) \wedge (\neg R \vee P) \\ & = R \wedge (\neg P \vee Q) \wedge (\neg R \vee P) \end{aligned}$$

化为标准子句后，可以进行归结



**注** 定理可拆分为包含条件和结论的蕴含式。

- 如果满足“条件 1”，“条件 2”，…，“”，那么有“结论”
- “条件 1 式”  $\wedge$  “条件 2 式”  $\wedge \dots \wedge \dots \rightarrow$  “结论式”

定理证明就是要证明这个蕴含式是“永真式”

**定理 3.1 (基于归结的定理证明)** 证明以下：

- 证明该式永真：条件 1 式  $\wedge$  条件 2 式  $\rightarrow$  结论式
- 证明该式永假： $\neg$  (条件 1 式  $\wedge$  条件 2 式  $\rightarrow$  结论式)
- 证明该式永假：条件 1 式  $\wedge$  条件 2 式  $\wedge \neg$  结论式

**注** 归结为空意味着归结的子句间存在矛盾，推导出了永假的结果。即要证明这个式子永假，只要把这个式子归结出空就可以了。

**例 3.4** 机器人搬箱子，求证箱子超重。

- 机器人不能直接感知箱子是否超重
- 如果机器人电池有电并且箱子不超重，则机器人能够移动该箱子
- 已知电池有电且箱子搬不动

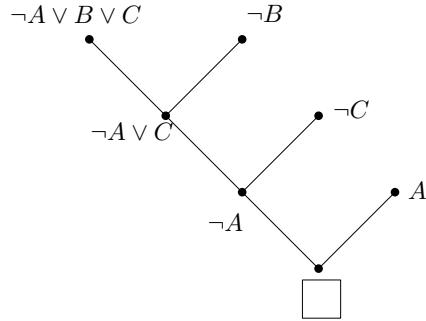
**证明.** 有如下：设  $A$ : 电池好、 $B$ : 箱子超重、 $C$ : 移动

- 事实：  $A, \neg C$
- 定理：  $A \wedge \neg B \rightarrow C$

- 待证结论:  $B$

把所有条件和结论的非写成合式公式，转化成标准子句:

$$\begin{aligned} & A \wedge \neg C (A \wedge \neg B \rightarrow C) \wedge \neg B \\ &= A \wedge \neg C (\neg A \vee B \vee C) \wedge \neg B \end{aligned}$$



证毕！

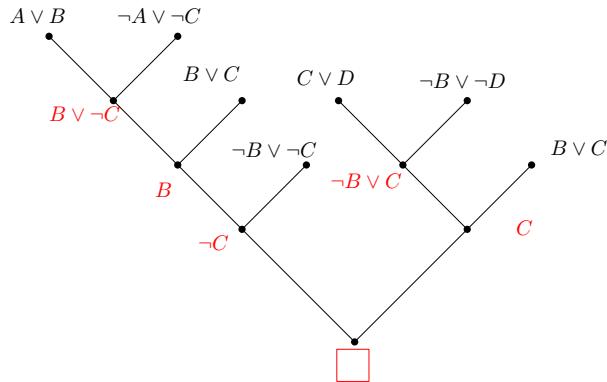
### 例 3.5 超级大间谍，谁是间谍？

某个机密文件被泄露，保密局派出 5 个侦查员去调查

- 侦察员 1 说：“A 与 B 中至少有一人作案”
- 侦察员 2 说：“B 与 C 中至少有一人作案”；
- 侦察员 3 说：“C 与 D 中至少有一人作案”；
- 侦察员 4 说：“A 与 C 中至少有一人与此案无关”；
- 侦察员 5 说：“B 与 D 中至少有一人与此案无关”。

证明：如果 5 个侦察员的话都可信，那么 B 和 C 都是间谍。

**证明.** 化为字句集合: C1:  $A \vee B$ 、C2:  $B \vee C$ 、C3:  $C \vee D$ 、C4:  $\neg A \vee \neg C$ 、C5:  $\neg B \vee \neg D$ 、T1:  $\neg B \vee \neg C$ 。把所有条件 C1-C5 和结论的非 T1 写成合式公式



证毕！

采用归结原理进行命题逻辑定理证明的基本思想是

- A 归纳法
- B 演绎法
- C 反证法
- D 枚举法

## 3.2 逻辑谓词

**注** 命题逻辑的局限性：

- 不能表示事务的共同属性  
“亚里士多德是人，牛顿是人”需要用两个独立的命题来表示。这两个命题都是不可拆分的原子命题，两个命题共同的部分没法表示。
- 不能表达原子命题内部结构  
在这个原子命题中无法“张三和李四是同学”体现张三和李四的关系。

### 3.2.1 谓词逻辑的基本概念

谓词逻辑由个体词、谓词、联结词及量词组成。

**定义 3.4 (个体词 (项))** 指谓词描述的对象，能够独立存在的客体。

- 常量：张三、A、…，通常是对象的名字
- 变量：习惯上用小写字母表示，如 x、z 等。Human(x)
- 函数：习惯上用小写字母或字母串表示，如 f, g 等。比如：father(小李)、LeftLeg(小李)、sum(x,y)

**定义 3.5 (谓词)** 用来刻画个体词的属性或个体词之间的关系

- 表示属性
- 表示关系

谓词不能单独使用，它必须与个体词结合才能构成谓词公式。

**定义 3.6 (联结词)** 表示谓词之间的关系

- 合取  $\wedge$
- 析取  $\vee$
- 蕴含  $\rightarrow$
- 非  $\neg$

**定义 3.7 (量词)** 说明个体变量的取值范围。

- 全称量词，表示“所有”， $\forall$
- 存在量词，表示“存在”， $\exists$

**注** 使用量词应注意

- 一般情况下， $\rightarrow$ 是与 $\forall$ 一起的主要连接词
- 一般情况下， $\wedge$ 是与 $\exists$ 一起的主要连接词

**定义 3.8 (量词的辖域)** 量词的辖域是邻接量词之后的最小子公式，因此除非辖域是个原子公式，否则应该在该子公式两端有括号

- 例： $\forall x P(x) \rightarrow Q(x)$   
 $\forall x$  的辖域是  $P(x)$
- 例： $\exists x (P(x) \rightarrow Q(x)) \vee P(y, z)$   
 $\exists x$  的辖域是  $P(x) \rightarrow Q(x)$

**定义 3.9 (约束变元和自由变元)** 在一个量词的辖域中与该量词的指导变元相同的变元称为约束变元，其他变元称为自由变元。

例如： $\forall x (P(x) \vee Q(y)) \wedge (\exists z) (R(z))$

**注** 多个量词并用时

$\forall x \exists y P(x, y)$  与  $\exists y \forall x P(x, y)$  不等价

$x, y$  分别表示一个自然数； $P(x, y) \therefore y > x$

多个全称量词和存在量词混用时，顺序不可以颠倒。

### 3.2.2 逻辑谓词的归结

**注** 置换注意事项：

- 只有变量才能被置换，常量和函数不能被置换。
- 置换要整体进行，即不同位置的同一变量要一起置换。
- 置换的变量不能出现在置换项中。即：对于置换  $s = \{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$ ，任一变量  $x_i$ ，不能出现在某个置换项  $t_i$  中。
- 同一个变量不能被多个置换项同时置换。
- 置换不符合交换律： $s_1 s_2 \neq s_2 s_1$

**注** 置换与合一

- 原始知识：

所有人都会死

亚里士多德是人

- 谓词公式:

$$\begin{aligned} \forall x \ (Human(x) \rightarrow WillDie(x)) \\ Human(Aristotle) \end{aligned}$$

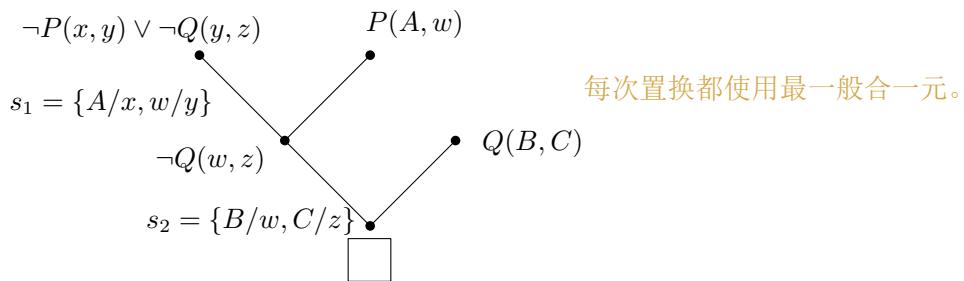
- 化成子句:

$$\begin{aligned} \neg Human(x) \vee WillDie(x) \\ Human(Aristotle) \end{aligned}$$

这两个子句不能归结，因为形式上并不完全相同

- 用亚里士多德替换  $x$ , 可以进行归结
- 合一: 寻找项对变量的置换, 以使两个合式公式表达一致。若存在一个置换  $s$  使得表达式集合  $\{E_i\}$  中每个元素经置换后有:  $E_1s = E_2s = E_3s = \dots$ , 则称表达式集  $\{E_i\}$  是可合一的, 这个置换  $s$  称为合一元。
- 最一般合一元  $\sigma$  (The most general unifier, mgu),  $\{w_i\}$  的  $\sigma$  有如下特性  
如果  $\theta$  是  $\{w_i\}$  任意合一元, 那么存在一个置换  $\lambda$  使得  $\{w_i\}\theta = \{w_i\}\sigma\lambda$ , 即  $\sigma$  是能够使合一的最小置换。
- 为何最需要最一般合一元

求证:  $\{\neg P(x, y) \vee \neg Q(y, z), P(A, w)\} \Rightarrow \neg Q(B, C)$



**定理 3.2 (归结原理)** 定理的证明过程:

1. 将结论的非与所有条件式进行合取, 得到待证明的合式公式;
2. 将上述合式公式化成可以归结的子句集形式; (子句间为合取关系, 子句内为析取形式)
3. 对上述式子进行归结, 如果归结为空, 则定理得证。

**注** 命题演算中公式标准化过程:

- 消除蕴含符号

使用蕴含表达式  $P \rightarrow Q \equiv \neg P \wedge Q$

- 否定深入

减少否定符号的范围, 否定符号  $\neg$  最多只用到一个命题符号上

- 转换成子句的合区范式  
字句内析取、子句间合区
- 形成子句合集

**注** 谓词公式化为标准子句：

目的：将谓词公式化为机器可以接收、处理的标准形式，这种标准形式是不含量词的子句集 相比命题公式标准化的难点：

- 全称量词和存在量词处理
- 各种变量名称如何处理

标准化步骤：

1. 消去蕴含符号
2. 否定深入（减少否定符号的范围）
3. 变量标准化（每个量词有唯一的变量符号）

例如：

- 量词辖域不重叠

$(\forall x) P(x) \vee (\forall x) Q(x)$  转化为  $(\forall x) P(x) \vee (\forall y) Q(y)$

- 量词辖域重叠

$(\forall x) (P(x) \rightarrow (\exists x) Q(x))$  转化为  $(\forall x) (P(x) \rightarrow (\exists y) Q(y))$

通过改变变量名称，使每个量词修饰的变量有自己唯一的变量名。（不同量词修饰的变量符号不同）

#### 4. 消去存在量词

- 如果存在量词不受全称量词修饰：直接实例化即可  
 $\exists x P(x)$  实例化为  $P(A)$ ，要求  $A$  必须是新的常量符号

- 如果存在量词受全称量词修饰：使用 Skolem 函数

$(\forall x) ((\exists y) Cap(x, y))$ ，用  $g(x)$  替换  $y$ ，消去存在量词得到  $(\forall x) Cap(x, g(x))$ ，Skolem 函数必须是新的

消去存在量词，存在量词的约束变元使用关于全称量词的指导变元的函数来代替  
如果存在量词处在多个全称量词的辖域内，则同时受多个全称量词共同影响，Skolem 函数为关于多个全称量词修饰变量的函数。

#### 5. 消去全称量词

- 直接将全称量词的符号去除。
- 经过前面几步，已经不存在量词修饰的变量，剩下的变量可以任意取值，所以没必要保留全称量词符号。
- 6. 把公式化为合取范式（子句间为合取关系，子句内为析取形式）写成子句形式（消除合取符号）
- 7. 更换变量名称（一个变量符号不用于多个子句）

更换变量符号的名称，使一个变量符号不出现在一个以上的子句中。

对于子句集

$$\begin{aligned} & \{\neg P(x_1) \vee \neg P(y) \vee P(f(x_1, y)), \\ & \neg P(x_2) \vee Q(x_2, g(x_2)) \\ & \neg P(x_3) \vee \neg P(g(x_3))\} \end{aligned}$$

**例 3.6** 将下列谓词公式化为标准子式

- $\forall x ((\forall y P(x, y)) \rightarrow \neg(\forall y (Q(x, y) \rightarrow R(x, y))))$

1. 消去蕴含符号

$$\forall x (\neg(\forall y P(x, y)) \vee \neg(\forall y (\neg Q(x, y) \vee R(x, y))))$$

2. 否定深入

$$\forall x (\exists y \neg P(x, y) \vee (\exists y (Q(x, y) \wedge \neg R(x, y))))$$

3. 变量标准化

$$\forall x (\exists y \neg P(x, y) \vee (\exists z (Q(x, z) \wedge \neg R(x, z))))$$

4. 消去存在量词

$$\forall x (\neg P(x, f(x)) \vee (Q(x, g(x)) \wedge \neg R(x, g(x))))$$

5. 消去全称量词

$$\neg P(x, f(x)) \vee (Q(x, g(x)) \wedge \neg R(x, g(x)))$$

6. 化为合取范式

$$(\neg P(x, f(x)) \vee Q(x, g(x))) \wedge (\neg P(x, f(x)) \neg R(x, g(x)))$$

7. 化为子句集形式

$$\begin{aligned} & \{\neg P(x, f(x)) \vee Q(x, g(x)), \\ & \neg P(x, f(x)) \neg R(x, g(x))\} \end{aligned}$$

8. 更换变量名称

$$\begin{aligned} & \{\neg P(x, f(x)) \vee Q(x, g(x)), \\ & \neg P(y, f(y)) \neg R(y, g(y))\} \end{aligned}$$

- $((\exists x)P(x) \vee (\exists x)Q(x)) \rightarrow ((\exists x)(P(x) \vee Q(x)))$

1. 消去蕴含符号

$$\neg((\exists x)P(x) \vee (\exists x)Q(x)) \vee ((\exists x)(P(x) \vee Q(x)))$$

2. 否定深入

$$((\forall x) \neg P(x) \wedge (\forall x) \neg Q(x)) \vee ((\exists x)(P(x) \vee Q(x)))$$

### 3. 变量标准化

$$((\forall x)\neg P(x) \wedge (\forall y)\neg Q(y)) \vee ((\exists z)(P(z) \vee Q(z)))$$

### 4. 消去存在量词

$$((\forall x)\neg P(x) \wedge (\forall y)\neg Q(y)) \vee (P(A) \vee Q(A))$$

### 5. 消去全称量词

$$(\neg P(x) \wedge \neg Q(y)) \vee (P(A) \vee Q(A))$$

### 6. 化为合取范式

$$(\neg P(x) \vee P(A) \vee Q(A)) \wedge (\neg Q(y) \vee P(A) \vee Q(A))$$

### 7. 化为子句集形式

$$\begin{aligned} & \{\neg P(x) \vee P(A) \vee Q(A), \\ & \neg Q(y) \vee P(A) \vee Q(A)\} \end{aligned}$$

### 8. 更换变量名称

$$\begin{aligned} & \{\neg P(x) \vee P(A) \vee Q(A), \\ & \neg Q(y) \vee P(A) \vee Q(A)\} \end{aligned}$$

## 3.2.3 定理证明和求解

**注** 谓词定理自动归结证明过程

- 将合式公式化为子句集的过程不同

因为谓词逻辑引入了个体词和量词，所以化为子句时和命题化为子句的步骤是不一样的

- 归结过程不同

因为个体词有变量和常量，所以谓词归结的过程和命题归结相比需要置换和合一。

### 例 3.7 机器人搬箱子的归结证明

已知：27号房间的所有箱子都比29号房间的箱子小。现在机器人知道箱子A在27号或29号房间中，箱子B在27号房间中，且B不比A小。试证明箱子A在27号房间中

**证明.** 有以下谓词公式

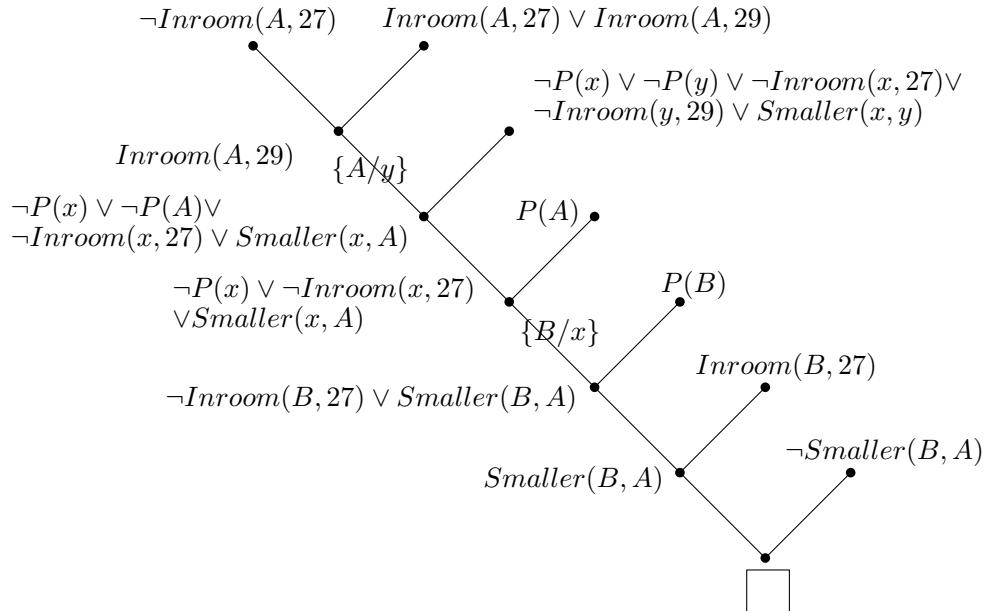
- F1:  $\forall x \forall y (P(x) \wedge P(y) \wedge Inroom(x, 27) \wedge Inroom(y, 29) \rightarrow Smaller(x, y))$
- F2:  $Inroom(A, 27) \vee Inroom(A, 29)$
- F3:  $Inroom(B, 27) \wedge \neg Smaller(B, A)$

- F4:  $P(A)$
- F5:  $P(B)$
- T:  $\neg Inroom(A, 27)$

化为子句集形式

- C1:  $\neg P(x) \vee \neg P(y) \vee \neg Inroom(x, 27) \vee \neg Inroom(y, 29) \vee Smaller(x, y)$
- C2:  $Inroom(A, 27) \vee Inroom(A, 29)$
- C3:  $Inroom(B, 27)$
- C4:  $\neg Smaller(B, A)$
- C5:  $P(A)$
- C6:  $P(B)$
- C7:  $\neg Inroom(A, 27)$

归结定理证明



### 例 3.8 机器人搬箱子的问题求解

已知：27 号房间的所有箱子都比 29 号房间的箱子小。现在机器人知道箱子 A 在 27 号或 29 号房间中，箱子 B 在 27 号房间中，且 B 不比 A 小。问箱子 A 在哪个房间？

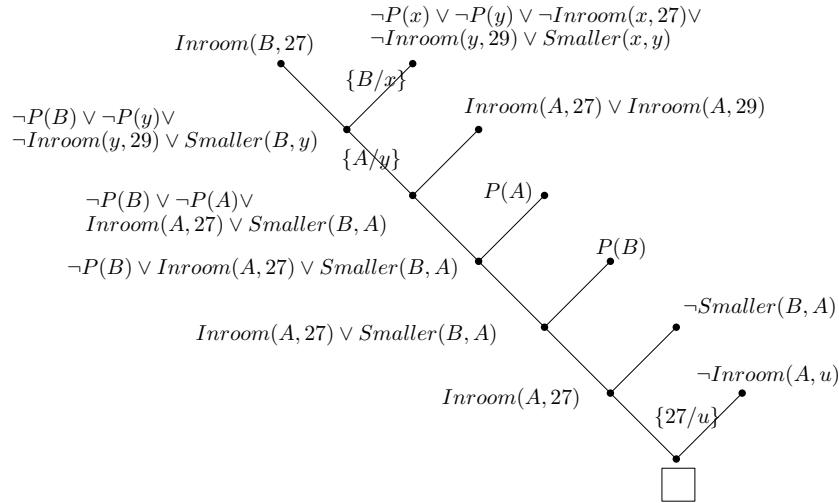
解：有以下谓词公式

- F1:  $\forall x \forall y (P(x) \wedge P(y) \wedge Inroom(x, 27) \wedge Inroom(y, 27) \rightarrow Smaller(x, y))$
- F2:  $Inroom(A, 27) \vee Inroom(A, 29)$
- F3:  $Inroom(B, 27) \wedge \neg Smaller(B, A)$
- F4:  $P(A)$

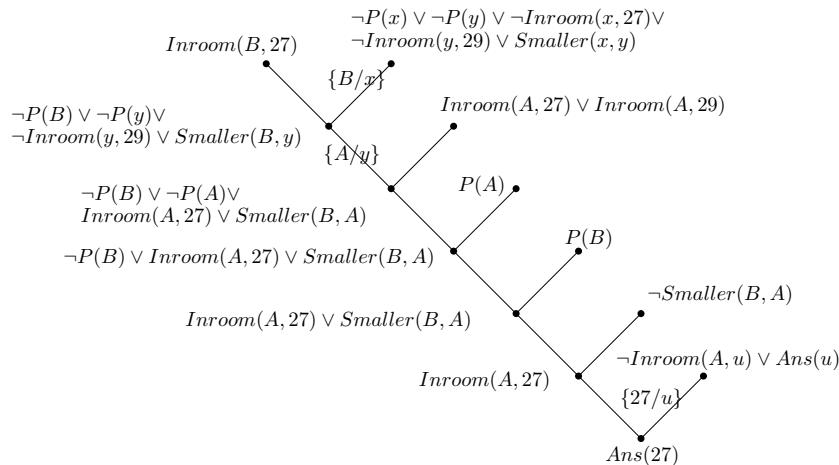
- F5:  $P(B)$
- T:  $(\exists u) \text{ Inroom}(A, u)$

化为子句集形式

- C1:  $\neg P(x) \vee \neg P(y) \vee \neg \text{Inroom}(x, 27) \vee \neg \text{Inroom}(y, 27) \vee \text{Smaller}(x, y)$
- C2:  $\text{Inroom}(A, 27) \vee \text{Inroom}(A, 29)$
- C3:  $\text{Inroom}(B, 27)$
- C4:  $\neg \text{Smaller}(B, A)$
- C5:  $P(A)$
- C6:  $P(B)$
- C7:  $\neg \text{Inroom}(A, u)$

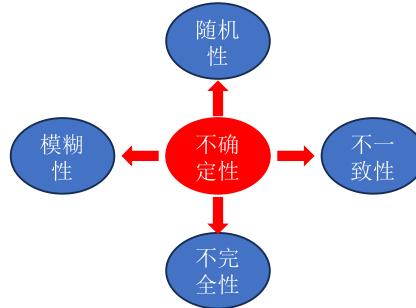


用归结来回答由合式公式表示的关于领域知识的问题，而不只是证明一个定理。例如，不仅要证明形如  $(\exists u)W(u)$  的定理，而且想得到已经证明存在的  $u$ 。为此，将文字  $Ans(u)$  加到要证明定理的否定子句中，执行归结直到只剩下一个回答文字。**加入子句**  $\neg W(u) \vee Ans(u)$



## 3.3 不确定知识与推理

### 3.3.1 不确定性描述



- 随机性

如果乌云密布并且电闪雷鸣，则很可能要下暴雨。

如果头痛发烧，则大概是患了感冒。

- 模糊性

姚明是个高个子。

张三和李四是好朋友。

- 不完全性

战场上，交战双方对对手的信息掌握往往是不完全的。

警方破案过程中，所掌握的关于罪犯的有关信息，往往就是不完全的。

- 不一致牲

牛顿运动定律对于宏观世界是正确的，但对于微观世界和宇观世界却是不适合的。

### 3.3.2 不确定领域的知识表示

**例 3.9** 例医疗诊断中，医生知道脑膜炎会引起病人脖子僵硬（如 70% 的概率）；此外，医生还了解一些无条件事实：病人患脑膜炎先验概率 1/50000，而任何一个病人脖子僵硬的先验概率为 1%。目前有一病人脖子僵硬，判断是否为脑膜炎。

令  $s$  表示“病人脖子僵硬”， $m$  表示“病人患有脑膜炎”，则有

$$p(s|m) = 0.7$$

$$p(m) = 1/50000$$

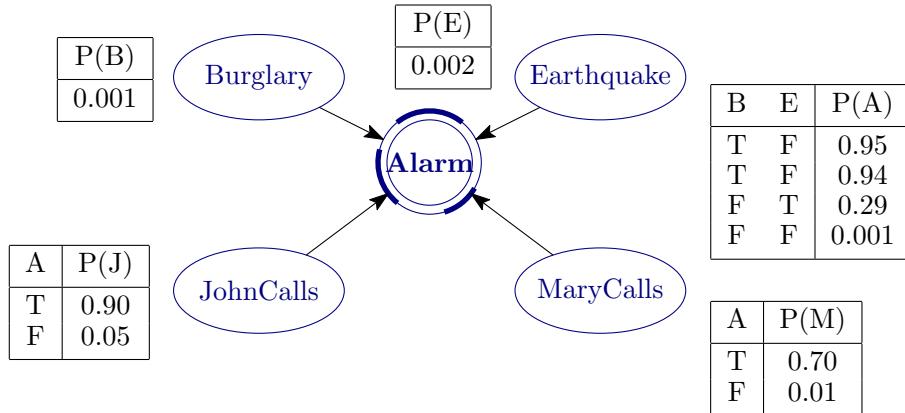
$$p(s) = 0.01$$

$$p(m|s) = \frac{p(s|m)p(m)}{p(s)} = \frac{0.7 \times 1/50000}{0.01} = 0.0014$$

### 3.3.3 贝叶斯网络

**定义 3.10 (条件概率表(CPT))** 对于一个变量  $Z$ , CPT 定义了一个条件分布  $P(Z|parent(Z))$ ; 其中,  $parent(Z)$  表示  $Z$  的父节点。

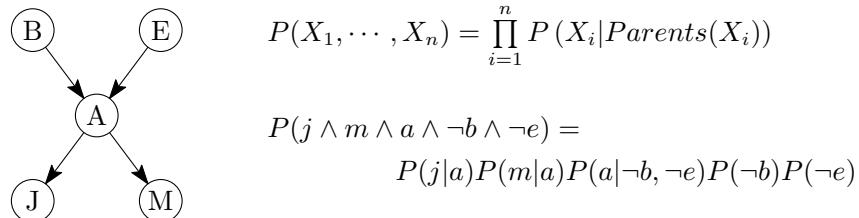
- 如果结点  $X$  没有父结点, 则表中只包含先验概率  $P(X)$ ;
- 如果结点  $X$  只有一个父结点  $Y$ , 则表中包含条件概率  $P(X|Y)$ ;
- 如果结点  $X$  有多个父结点  $\{Y_1, Y_2, \dots, Y_k\}$ , 则表中包含条件概率  $P(X|Y_1, Y_2, \dots, Y_k)$ 。



在非因果方向上, 确定条件独立是困难的, 且网络的压缩效果也不理想。

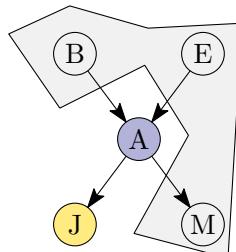
**注** 贝叶斯网络 (Bayesian Networks, BN) 语法语义

- 全局语义: 全联合概率分布与局部条件概率分布之间的关系



- 局部语义: 充分利用条件独立性, 分析哪些变量之间是独立的

贝叶斯网络的一个结点, 如果它的父结点已知, 则该结点条件独立于它的所有非后代结点

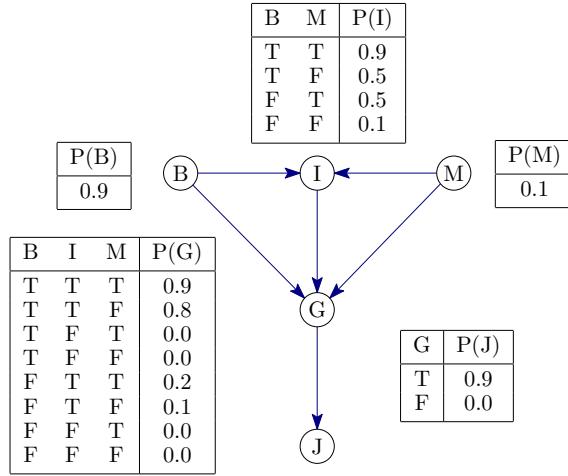


**例 3.10** 构造贝叶斯网络的步骤, 主要包括:

- A 确定变量集和变量域
- B 确定网络结构
- C 确定节点条件概率表
- D 确定网络的规模

### 3.3.4 基于贝叶斯网络的推理

**例 3.11** 考虑下图所示的贝叶斯网络，解答下列问题：



1. 网路结构能否断言下列语句？若不能，请写出正确表达式

(a).  $P(B, I, M) = P(B)P(I)P(M)$

不能，正确表达式为  $P(B, I, M) = P(I|B, M)P(B)P(M)$

(b).  $P(J|G) = P(J|G, I)$

能

(c).  $P(M|G, B, I) = P(M|G, B, I, J)$

能

2. 计算  $P(b, i, \neg m, G, \neg j)$

$$\begin{aligned}
 P(b, i, \neg m, g, \neg j) &= P(\neg j|g)P(g|b, i, \neg m)P(i|b, \neg m)P(b)P(\neg m) \\
 &= (1 - 0.9) \times 0.8 \times 0.5 \times 0.9 \times (1 - 0.1) \\
 &= 0.0324
 \end{aligned}$$

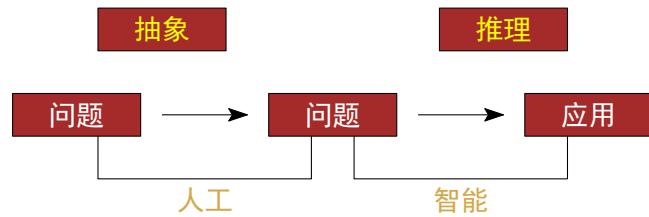
$$\begin{aligned}
 P(b, i, \neg m, \neg g, \neg j) &= P(\neg j|\neg g)P(\neg g|b, i, \neg m)P(i|b, \neg m)P(b)P(\neg m) \\
 &= (1 - 0) \times (1 - 0.8) \times 0.5 \times 0.9 \times (1 - 0.1) \\
 &= 0.081
 \end{aligned}$$

另一种写法

$$P(b, i, \neg m, G, \neg j) = \langle 0.0324, 0.081 \rangle$$

# 4 机器学习

符号主义的困境

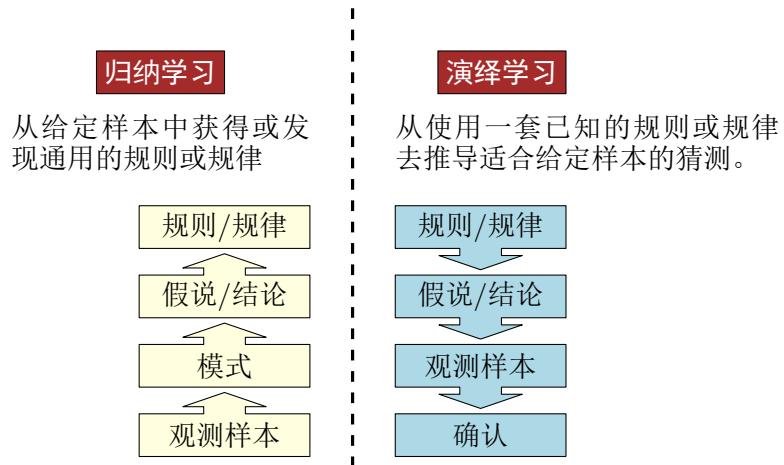


- 将问题抽象为模型的过程（知识表示）需要人工完成。把知识总结出来再交给计算机相当困难，存在知识获取瓶颈。
- 很多涉及大量数据和多变量的复杂问题，没有现成的推理规则和处理方式。

## 4.1 机器学习概述

### 4.1.1 基本概念

两种通用的学习类型

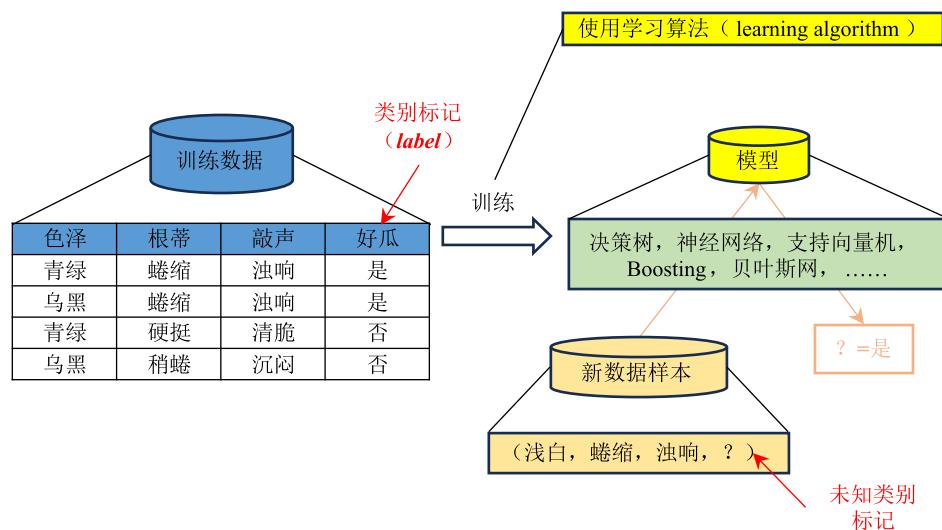


机器学习思路

- 给定数据（样本、实例）和一定的学习规则，赋予机器自动从数据中获取知识的能力。
- 能够直接从样本（数据）中学习知识，不依赖于给定的判断规则。
- 机器学习的对象：计算机及互联网上的各种数字、文字、图像、视频、音频数据以及它们的组合。

数据的基本假设是同类数据具有一定的统计规律性。

- 用于对数据（特别是未知数据）进行预测和分析。



#### 例 4.1 下列说法正确的是？

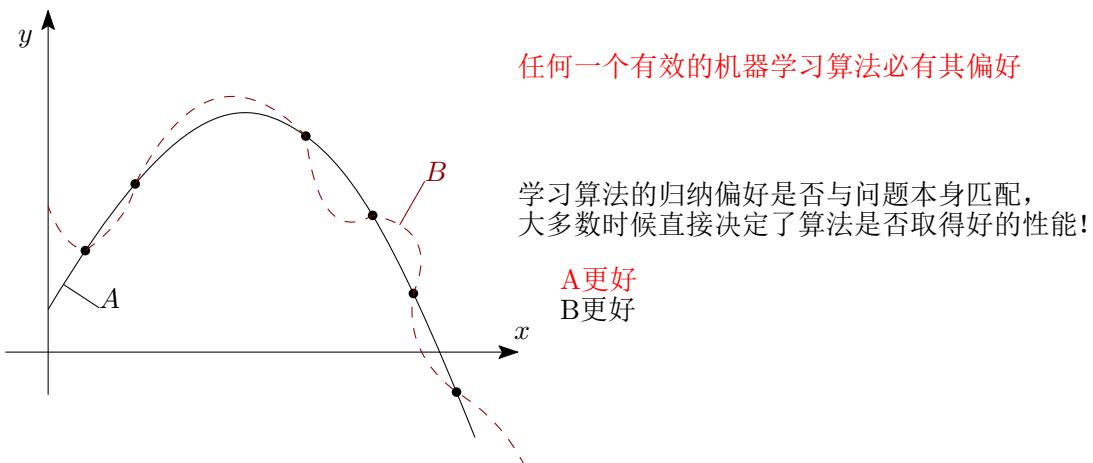
- A. 机器学习从数据中自动分析并获得规律
- B. 机器学习中所采用的判断规则是事先给定的
- C. 利用已有公式可以计算的问题需要使用机器学习
- D. 机器学习适于解决数据丰富且规则不明确的问题

#### 机器学习的数据

- 样本：关于一个对象或事件描述的一条记录
- 特征（属性）：反映对象或事件在某方面的表现或性质
- 标签：样本标注的类别信息（离散）或者程度信息（连续），也可能没有。

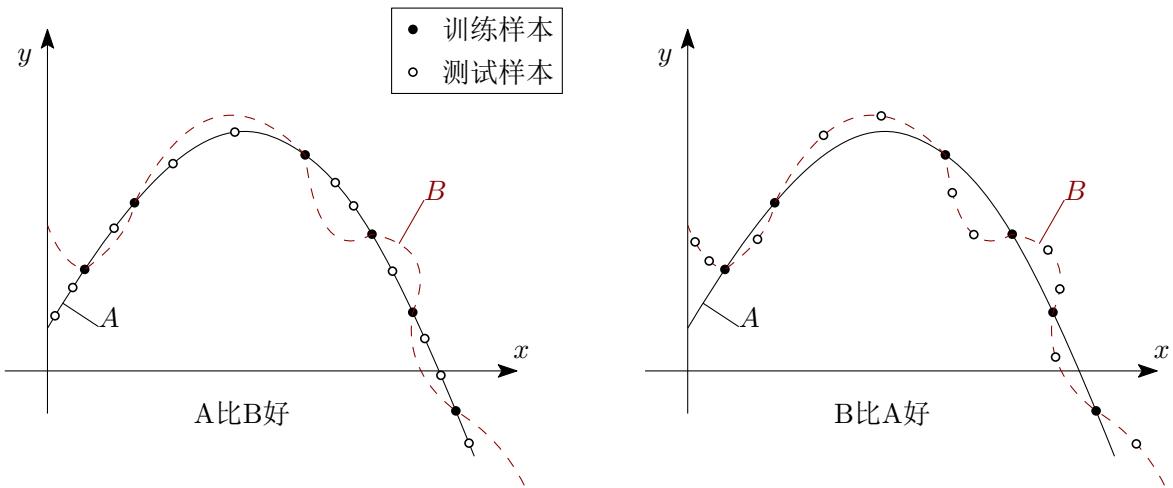
#### 归纳偏好

机器学习算法在学习过程中对某种类型假设的偏好。

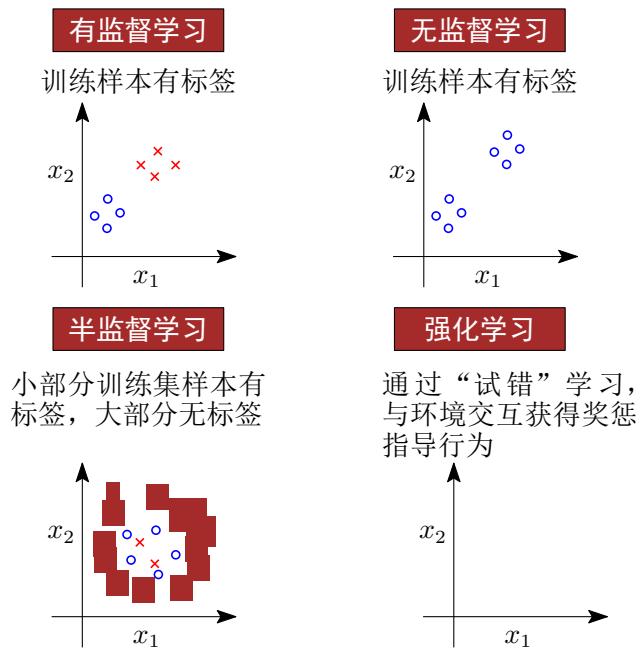


#### NFL (No Free Lunch) 定理

**定理 4.1 (NFL 定理)** 一个算法 A 若在某些问题上比另一个算法 B 好，必存在另一些问题 B 比 A 好。



**定义 4.1 (泛化能力)** 机器学习的目标是使得学到的模型能很好的适用于“新样本”而不仅仅是训练集合，模型适用于新样本的能力被称为泛化（Generalization）能力。



### 强化学习 vs. 有监督学习

- 相似点：

1. 强化学习中“状态”对应监督学习中的“示例”、“动作”对应“标签”，“策略”相当于“分类器”或“回归器”
2. 强化学习中没有（监督学习中的那种）有标签样本（即“示例-标签”），换言之，没有人告诉机器在什么状态下应该做什么，而是通过最终结果“反思”之前的动作是否正确来进行学习

3. 因此，强化学习在某种意义上可以看成具有“延迟标签信息”的监督学习问题
- 不同点

1. 有监督学习的训练样本是有标签的，强化学习的训练是没有标签的，只有环境给出的奖惩
2. 有监督学习的学习过程是静态的，不与环境进行交互，给什么样本就学什么。强化学习的学习过程是动态的，与环境进行交互，通过环境给出奖惩来学习。
3. 有监督学习主要感觉感知问题，强化学习主要解决决策问题。

**例 4.2** 下面关于有监督学习说法正确的是？

- A 通过已知的训练样本来训练
- B 训练样本是带有标签的
- C 对于训练集之外的样本，学习到的模型不能够预测结果
- D 可以应用于目标识别中

## 4.2 有监督学习基本方法

回归和分类取决于训练用的标签是连续还是离散的，它们预测获得标签也对应的是连续和离散的。

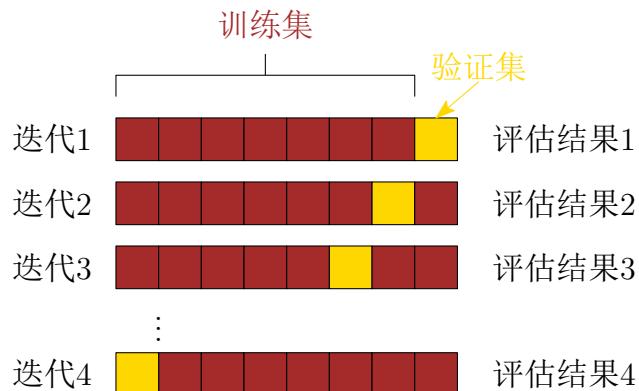
### 有监督学习的一般过程

- 数据准备
  - 如果问题是全新的，采集或爬取带标签的数据；如果数据仓库有相应的数据，将数据提取出来。
  - 数据准备好之后，需要对数据进行预处理，主要包括重复数据检测、数据标准化、数据编码、缺失值处理、异常值处理等。
- 特征选择或抽取
  - 特征选择：选取原始特征集合的有效子集，保留有用特征，移除冗余或无关的特征。
  - 特征抽取：构造一个新的特征空间，将原始特征投影到新的空间中表示。
- 训练——从数据中学习获得模型的过程，包括选择模型和确定参数。
  - 超参数，如深度学习模型中各层的神经元数量；
  - 模型内部参数，如深度学习模型中神经元间的连接权值。
- 验证/测试——选取一部分已知类别的样本评估模型分类效果
  - 误差：模型输出与真值的偏离程度
  - 训练（经验）误差：模型在训练集上的误差
  - 测试误差：模型在测试集上的误差
  - 泛化误差：在除训练集外所有样本上的误差

### 注 验证/测试的注意:

- 假设测试集是从样本真实分布中独立采样获得，将测试集上的“测试误差”作为泛化误差的近似，所以测试集要和训练集中的样本互斥。
- 独立测试集：选择与训练样本相独立的数据集进行验证，评估模型的分类效果。
- 交叉验证：将原始数据分组，大部分作为训练集，进行训练模型；另一部分作为验证集，测试训练得到的模型的正确率，用来调整模型参数。k 折交叉验证是将原始数据分为 k 组，每次留 1 组作为验证集。特例：留 1 法。

### • K 折交叉验证 (K-fold cross-validation)



- 初始采样分割成 K 个子样本，一个单独的子样本被保留作为验证模型的数据，其他 K-1 个样本用来训练。
- 交叉验证重复 K 次，每个子样本验证一次，最终得到一个单一估测。
- 这个方法的优势在于，同时重复运用随机产生的子样本进行训练和验证，每次的结果验证一次，10 折交叉验证是最常用的。

### 例 4.3 思考题：

1. 训练数据、验证数据可以有交叠吗？

不能有交叠。

2. 为什么要进行交叉验证，随机选取其中一部分只做一次测试好不好？

多次交叉验证的效果更好，可以充分利用已知数据计算出多个预测正确率，获得最终预测正确率的平均值和方差，更准确地评估模型的预测效果。

### 例 4.4 影响有监督学习性能的因素有哪些？

- A 可用数据的样本规模
- B 数据预处理
- C 特征抽取与选择
- D 机器学习算法设计

### 4.2.1 回归问题

#### 最小二乘法

回归方程:  $\hat{y} = f(x) = \mathbf{x}^T \mathbf{w} + b$

$$\begin{aligned} (\mathbf{w}^*, b^*) &= \arg \min_{(\mathbf{w}, b)} \sum_{i=1}^m (f(x_i) - y_i)^2 \\ &= \arg \min_{(\mathbf{w}, b)} \sum_{i=1}^m (y_i - \mathbf{x}_i^T \mathbf{w} - b)^2 \\ Q &= \sum_{i=1}^m (y_i - \mathbf{x}_i^T \mathbf{w} - b)^2 = (\mathbf{y} - \mathbf{X} \hat{\mathbf{w}})^T (\mathbf{y} - \mathbf{X} \hat{\mathbf{w}}) \end{aligned}$$

其中,  $\mathbf{X}$  和  $\hat{\mathbf{w}}$  为

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T & 1 \\ \vdots & \vdots \\ \mathbf{x}_m^T & 1 \end{bmatrix}, \quad \hat{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

求其偏导数并令其为 0, 即:

$$\begin{aligned} \frac{\partial Q}{\partial \hat{\mathbf{w}}} &= 2\mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{X}^T \mathbf{y} = 0 \\ \hat{\mathbf{w}} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \end{aligned}$$

**例 4.5** 改进线性回归问题性能的方法包括

- A 引入新的特征
- B 提高多项式模型的阶次
- C 选择合适的曲线回归模型
- D 抽取更有效的特征

#### 非线性回归

**定义 4.2 (非线性回归)** 两种回归通过在目标函数中引入正则化项来处理多重共线性问题, 并达到防止过拟合的目的。岭回归采用的是 L2 正则化, 而 Lasso 回归采用的是 L1 正则化。

#### 岭回归

#### Lasso 回归

$$J = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^p \beta_j^2 \quad J = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^p |\beta_j|$$

### 4.2.2 分类问题

**定义 4.3 (分类问题)** 分类也是有监督学习的一种。它解决的是样本的标签是离散值的情况, 其试图给出的预测值也是离散的。

- **二分类问题**: 非此即彼的分类问题, 输出值, 0 或 1, 0 代表不合格, 1 代表合格
- **多分类问题**: 如分四等次, 对应预测离散输出值为 0、1、2、3。0 代表不及格、1 表示及格、2 表示良好、3 表示优秀。

**例 4.6** 能否将分类问题的标签作为输出值直接利用回归算法求解?

A 能

B 不能

### 分类问题性能度量

衡量模型泛化能力的评价标准, 反映了任务需求, 使用不同的性能度量往往会导致不同的评判结果。

- 在对于分类任务, 错误率和准确率是最常用的两种性能度量:

- 分类错误率: 分错样本占样本总数的比例

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m I_{f(x_i) \neq y_i}$$

- 分类准确率: 分对样本占样本总数的比例

$$acc(f; D) = \frac{1}{m} \sum_{i=1}^m I_{f(x_i) = y_i} = 1 - E(f; D)$$

- 诊断、检测等应用场景中, 查准率和查全率比分类错误率和分类精度更适合。

分类结果混淆矩阵

真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

查准率 (准确率, precision)

被预测为整理的样本中真正例的比率

$$P = \frac{TP}{TP + FP}$$

查全率, (召回率, recall)

所有真正例中被预测为正例的比率

$$P = \frac{TP}{TP + FN}$$

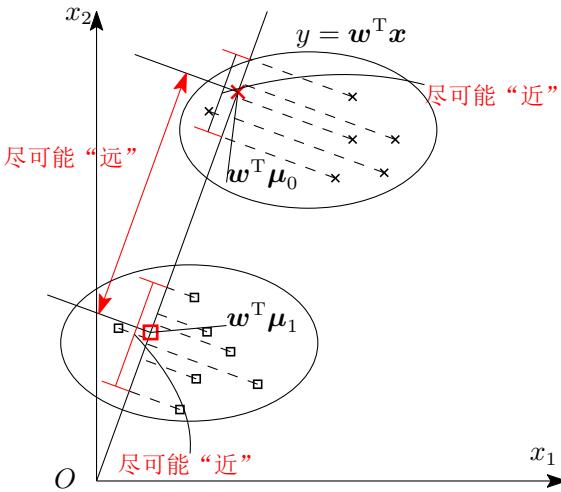
查准率和查全率是一对矛盾的度量。

### 线性判别分析

$$\begin{aligned} J &= \frac{\|\mathbf{w}^T \boldsymbol{\mu}_0 - \mathbf{w}^T \boldsymbol{\mu}_1\|_2^2}{\mathbf{w}^T \boldsymbol{\Sigma}_0 \mathbf{w}^T + \mathbf{w}^T \boldsymbol{\Sigma}_1 \mathbf{w}^T} \\ &= \frac{\mathbf{w}^T (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T \mathbf{w}}{\mathbf{w}^T (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1) \mathbf{w}} \end{aligned}$$

记  $\mathbf{S}_b \triangleq (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T$ ,  $\mathbf{S}_w \triangleq \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1$  那么

$$J = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$



给定训练样本集，设法让样本投影到一条直线上，使得同类样本的投影点尽可能接近，不同类样本的投影点尽可能远离。

寻找一个函数，**最大化类间距离，最小化类内距离**

$$\max \frac{\text{类间距离}}{\text{类内距离}}$$

$$J = \frac{\|w^T \mu_0 - w^T \mu_1\|_2^2}{w^T \Sigma_0 w + w^T \Sigma_1 w}$$

令  $w^T S_w w = 1$ ，最大化广义瑞利商等价形式为

$$\min_w -w^T S_b w$$

$$\text{s. t. } w^T S_w w = 1$$

由拉格朗日乘子法可得

$$S_b w = \lambda S_w w \rightarrow w = S_w^{-1} (\mu_0 - \mu_1)$$

两类数据同先验、满足高斯分布且协方差相等时，线性判别分析达到最优分类。

**例 4.7** 现有两类样本，请计算其线性判别分类器。

• 第 1 类：

$$X_1 = \{(4, 2), (2, 4), (2, 3), (3, 6), (4, 4)\}$$

• 第 2 类：

$$X_2 = \{(9, 10), (6, 8), (9, 5), (8, 7), (10, 8)\}$$

均值向量：

$$\mu_1 = [3 \ 3.8]^T, \mu_2 = [8.4 \ 7.6]^T$$

散度矩阵：

$$\Sigma_1 = \begin{bmatrix} 1 & -0.25 \\ -0.25 & 2.2 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 2.3 & -0.05 \\ -0.05 & 3.3 \end{bmatrix}$$

类内散度：

$$S_w = \Sigma_1 + \Sigma_2 = \begin{bmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{bmatrix}$$

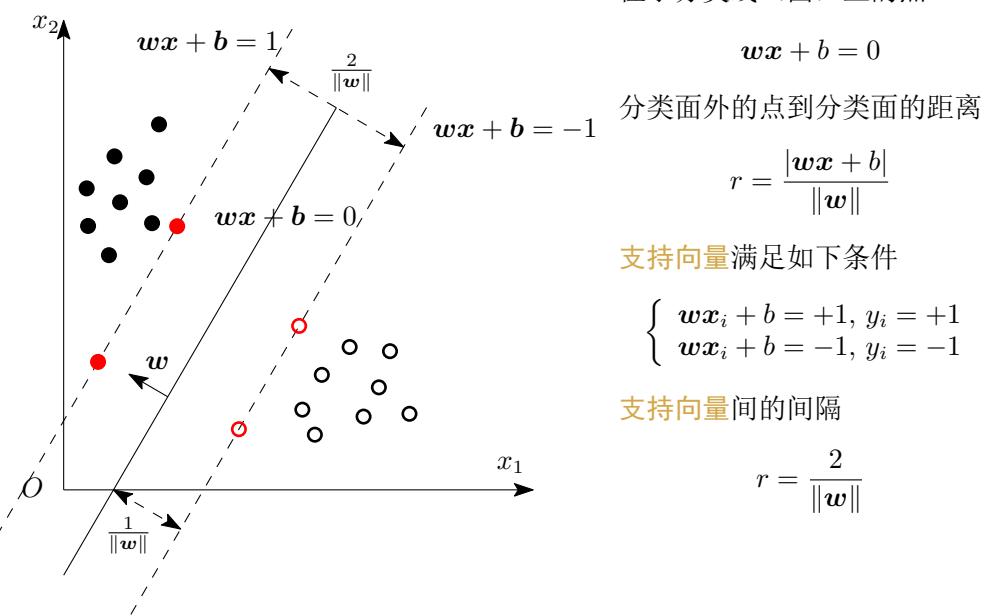
$S_w$  的逆

$$S_w^{-1} = \begin{bmatrix} 0.30454 & 0.016611 \\ 0.016611 & 0.182724 \end{bmatrix}$$

故而  $w$  为

$$w = S_w^{-1} (\mu_0 - \mu_1) = [-1.70764 \ -0.78405]^T$$

## 支持向量机



**最大间隔原则:**

选择参数对  $(w, b)$  使得训练集对于线性函数  $wx + b = 0$  的几何间隔取最大值，构造决策函数

$$\begin{aligned} & \min_{w,b} \frac{1}{2} \|w\|^2 \\ & \text{s. t. } y_i(wx_i + b) \geq 1 \end{aligned}$$

为求解上述问题，使用 Lagrange 乘子法将其转化为对偶问题。于是引入 Lagrange 函数：

$$L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i (y_i(wx_i + b) - 1)$$

首先求 Lagrange 函数关于  $w, b$  的极小值。由极值条件有：

$$\nabla_b L(w, b, \alpha) = 0, \nabla_w L(w, b, \alpha) = 0$$

得到

$$\sum_{i=1}^n y_i \alpha_i = 0, w = \sum_{i=1}^n y_i \alpha_i x_i$$

**例 4.8** 现有三个样本能被支持向量机正确分类且远离决策超平面。如果把这三个样本加入到训练集，支持向量机的决策超平面是否会受其影响？为什么？

A 会

B 不会

K 近邻

建立疾病判别模型：甲的基因A值为11.5，B 值为7.5，分别取 $K = 1, 3$ ，判断甲是否患病（0 未患病， 1 患病）？

编号	基因A	基因B	类标号
1	10	8	1
2	8	5	1
3	5	4	1
4	8	1	1
5	12	6	1
6	3	8	1
7	9	3	1
8	7	2	1

编号	基因A	基因B	类标号
9	11	7	0
10	15	8	0
11	20	4	0
12	21	9	0
13	22	16	0
14	24	32	0
15	16	17	0
16	18	14	0

编号	1	2	3	4	5	6	7	8
$D^2$	2.5	18.5	54.5	54.5	2.5	72.5	26.5	50.5
编号	9	10	11	12	13	14	15	16
$D^2$	0.5	12.5	84.5	92.5	182.5	756.5	110.5	84.5

- $K = 1$ ，未患病。
- $K = 3$ ，患病。

## K 近邻法的优点

- 简单，易于理解，易于实现，无需参数估计，无需训练；
- 对异常值不敏感，个别噪音数据对结果的影响不是很大；
- 适合对稀有事件进行分类；
- 适合于多分类问题，K 近邻要比支持向量机表现要好。

## K 近邻法的缺点

- 对测试样本分类时的计算量大，内存开销大，需要计算新样本到全体已知样本的距离，才能求得它的 K 个最近邻点；
- 当样本不平衡时，可能导致新样本的 K 个邻居中大容量类的样本占多数，出现系统分类偏差；
- K 近邻是一种消极学习方法、懒惰算法。

## 决策树

**注** 决策树学习的关键是：选择最优划分属性

- 什么样的划分属性是最优的？

希望决策树的分支结点所包含的样本尽可能属于同一类别，即结点的“纯度”越来越高，可以高效地从根结点到达叶结点，得到决策结果。

- 三种度量结点“纯度”的指标

- 信息增益：ID3

- 增益率：C4.5
- 基尼指数：CART

**注** 决策树的优缺点：

- 优点
  - 计算复杂度不高；
  - 可解释性强；
  - 能处理具有许多属性的数据集；
  - 在相对短的时间内能够对大数据集做出可行且效果良好的分类结果。
- 缺点
  - 可能存在过拟合问题

**注** 剪枝是解决决策树过拟合的一种方法：——通过主动去掉一些分支来降低过拟合的风险

- 预剪枝：
 

在决策树生成过程中，对每个结点在划分前先进行估计，若当前结点的划分不能带来决策树泛化性能提升，则停止划分并将当前结点标记为叶结点。
- 后剪枝：
 

先从训练集生成一棵完整的决策树，然后自底向上地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点。

**注** 在医疗诊断等应用场景中，**灵敏度、特异度、阳性预测值、阴性预测值**等比单纯的错误率准确率更重要。

真实情况	预测结果	
	正例	反例
正例	TP（真正例）	FN（假反例）
反例	FP（假正例）	TN（真反例）

$$\text{灵敏度} \quad SS = \frac{TP}{TP + FN} \quad \text{阳性预测值} \quad PPV = \frac{TP}{TP + FP}$$

$$\text{特异度} \quad SC = \frac{TN}{TN + FP} \quad \text{阴性预测值} \quad NPV = \frac{TN}{TN + FN}$$

**例 4.9** 已知某核酸检测分类器对 COVID 19 阳性和阴性样本预测的混淆矩阵如下表所示，该分类方法的灵敏度为  $\frac{3}{3+7} = 30\%$ 、特异度为  $\frac{9}{1+9} = 90\%$ 、准确率为  $\frac{3+9}{20} = 30\%$ 。

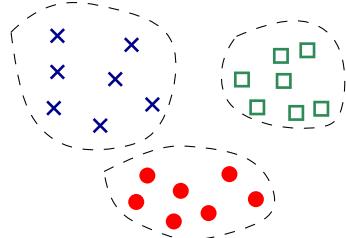
真实情况	预测结果		合计
	阳性	阴性	
阳性	3	7	10
阴性	1	9	10
合计	4	16	20

## 4.3 无监督学习基本方法

无监督学习（聚类）的一般过程

**定义 4.4 (无监督学习)** 根据数据的相似性将数据分为多类的过程。

**目标：**将数据集中的样本按照相似性划分为若干个通常不相交的子集“簇”。



**样本间聚类：**

将特征相似的样本分在同一个簇，特征差异较大的样本分在不同的簇。

### 4.3.1 相似性的度量

闵可夫斯基距离

**定义 4.5 (闵可夫斯基距离)** 设  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$  和  $\mathbf{x}_j = (x_{j1}, \dots, x_{jp})$  是  $i$  和  $j$  号样本的特征向量，则二者之间的距离为

$$d_{ij} = \left( \sum_{k=1}^p |x_{ik} - x_{jk}|^g \right)^{1/g}$$

- 当  $g = 1$  时（绝对值距离）， $d_{ij} = \sum_{k=1}^p |x_{ik} - x_{jk}|$
- 当  $g = 2$  时（欧式距离）， $d_{ij} = \sqrt{\sum_{k=1}^p (x_{ik} - x_{jk})^2}$

**注** 闵式距离的缺陷

- 容易受变量的量纲影响
- 没有考虑变量间的相关性

**注** 两种改进措施

- 变量标准化处理 
$$\begin{cases} \text{极差标准化} & y_i = \frac{x_i - \min x}{\max x - \min x} \\ \text{Z-score 标准化} & z_i = \frac{x_i - \text{mean}(x)}{\sigma} \end{cases}$$

- 马氏距离  $D_M(x) = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{S}^{-1} (\mathbf{x} - \boldsymbol{\mu})}$

方向余弦度量相似度

**定义 4.6 (方向余弦)**  $n$  维向量  $x$  和  $y$  的夹角记作  $\theta$ , 根据余弦定理, 其余弦值为

$$\cos(\theta) = \frac{\mathbf{x}^T \mathbf{y}}{|\mathbf{x}| |\mathbf{y}|} = \frac{\sum_{k=1}^p x_{ik} x_{jk}}{\sqrt{\sum_{k=1}^p x_{ik}^2 \sum_{k=1}^p x_{jk}^2}}$$

相关系数

**定义 4.7 (相关系数)** 设  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$  和  $\mathbf{x}_j = (x_{j1}, \dots, x_{jp})$  是  $i$  和  $j$  号样本的特征向量, 则二者之间的相关系数为

$$r_{ij} = \frac{\sum_{k=1}^p (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)}{\sqrt{\left[ \sum_{k=1}^p (x_{ik} - \bar{x}_i)^2 \right] \left[ \sum_{k=1}^p (x_{jk} - \bar{x}_j)^2 \right]}}$$

**注** 相关系数和方向余弦关系

$$r_{ij} = \frac{\sum_{k=1}^p (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)}{\sqrt{\left[ \sum_{k=1}^p (x_{ik} - \bar{x}_i)^2 \right] \left[ \sum_{k=1}^p (x_{jk} - \bar{x}_j)^2 \right]}} \quad \text{向量标准化} \quad \cos(\theta) = \frac{\sum_{k=1}^p x_{ik} x_{jk}}{\sqrt{\sum_{k=1}^p x_{ik}^2 \sum_{k=1}^p x_{jk}^2}}$$

**注** 常用聚类方法:

- 划分法: 需要输入类别数, 适用于样本数较多的情况
- 层次法: 不需要输入类别数, 适用于样本数较少的情况

### 4.3.2 划分法

**定义 4.8 (划分法)** 给定一个有  $N$  个样本的数据集, 划分法将构造  $K$  个分组, 每一个分组就代表一个聚类,  $K < N$ 。而且这  $K$  个聚类满足两个条件:

- 每一个聚类至少包含一个样本;
- 每一个样本属于且仅属于一个聚类。

**标准:** 同一聚类中的样本相似性高, 不同聚类中的样本相似性低

**典型算法:** K 均值聚类算法、K MEDOIDS 算法、CLARANS 算法

## K 均值聚类

### 注 基本思想

将每一个样本分配到最近中心（样本均值）所属的簇

$$\text{目标函数: } C^* = \arg \min_C \sum_{l=1}^K \sum_{C(i)=l} \|\mathbf{x}_i - \bar{\mathbf{x}}_l\|^2$$

**优化方法:** 采用贪心策略，通过迭代优化来近似求解，不能保证收敛到全局最优

算法主要包括以下步骤：

1. 指定聚类数，确定初始簇的中心

用户指定或系统指定。

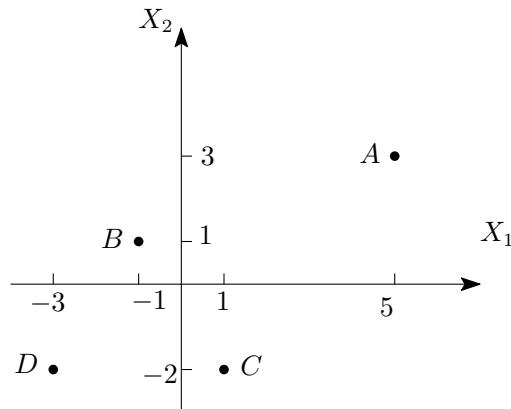
2. 根据距离最近原则进行分类

计算每个样本到各簇中心点的距离，并按距离最近原则对所有样本进行分类，计算新的聚类中心。

3. 迭代，直至收敛

**例 4.10** 对 A、B、C、D 样本分别测量特征  $X_1$  和  $X_2$ ，得到如下结果。试用 K 均值

样本	特征	
	$X_1$	$X_2$
A	5	3
B	-1	1
C	1	-2
D	-3	-2



方法将这四个样本聚成两簇。

1. 按要求取  $K = 2$ ，将样本随意分成两簇  $(A, B)$  和  $(C, D)$ ，计算两个聚类的中心坐标。计算样本到各簇中心的欧氏距离，将样本分配给最近的一簇。对于样本有变动的簇，重新计算它们的中心坐标。

聚类	中心坐标		$D^2(i, C_l)$			
	$X_1$	$X_2$	A	B	C	D
$(A, B)$	2	2	10	10	17	41
$(C, D)$	-1	-2	61	9	4	4

B 被分配到  $(C, D)$

2. 再次检查每个样本，以决定是否需要重新分簇。计算样本到各中心的距离。

聚类	中心坐标		$D^2(i, C_l)$			
	$X_1$	$X_2$	A	B	C	D
(A)	5	3	0	40	41	89
(B, C, D)	-1	-1	52	4	5	5

由于每个样本都已经分配到距离中心最近的簇，因此聚类过程到此结束。最终得到  $K = 2$  的聚类结果是 A 独自成一簇，B, C, D 聚成一簇。

**例 4.11** 请判断以下说法是否正确：当距离度量和聚类数目选定后， $K$  均值聚类的结果是唯一的。

- A 正确
- B 错误

**注**  $K$  均值聚类：优缺点

- 优点
  - 简单快速，可以用于多种数据类型
  - 有效，可处理大数据集
- 缺点
  - 结果受初始聚类中心影响较大
  - 不适合处理非球形簇、不同尺度和不同密度的簇
  - 结果受聚类数影响较大
  - 对噪声和孤立点数据敏感

**注** 均值聚类的改进

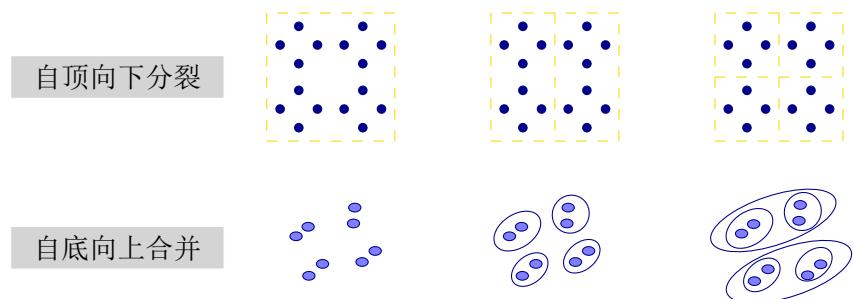
- 初始聚类中心的选择
  - 选择彼此距离尽可能远的  $K$  个点
  - 选择密度最大的  $K$  个点
- 聚类数的选择
  - 观察法
  - 肘部法
    - 随着聚类数  $K$  增大，样本划分更加精细，每个簇的聚合程度逐渐提高，误差平方和变小；
    - 当  $K$  小于真实聚类数时， $K$  增大会大幅增加每个簇的聚合程度，误差平方和的下降幅度很大；
    - 当  $K$  到达真实聚类数时，误差平方和的下降幅度骤减，然后随着  $K$  值的继续增大而趋于平缓；

- 误差平方和与  $K$  的关系图是一个手肘的形状，而肘部对应的  $K$  值就是数据的真实聚类数。
- 离群点的处理
  - 离群点检测和删除

### 4.3.3 层次法

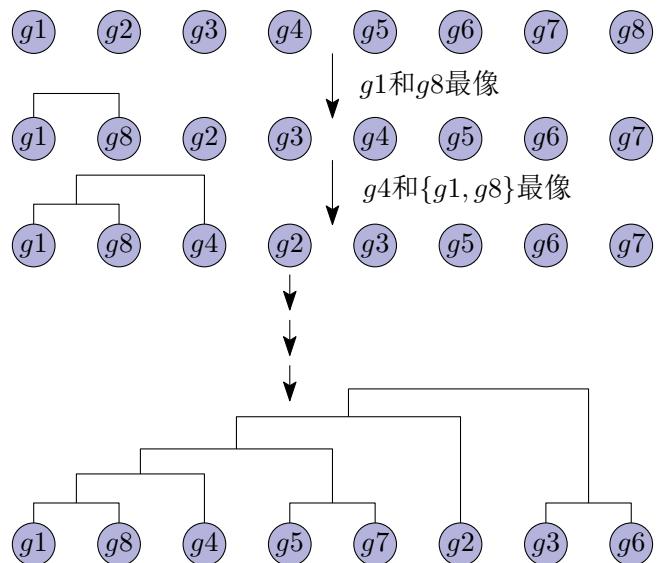
**定义 4.9 (层次聚类方法)** 通过计算数据集中样本间的相似性来创建一棵有层次的嵌套聚类树。

创建聚类树的策略



自底向上的层次聚类过程

- 先将每个样本各归为一族
- 然后每次将最相似的两个簇合并成一个新的簇；
- 直至将所有的样本归为一族或者达到预定结束条件。



**注** 三个要素：距离度量、合并规则和结束条件

**注** 思考：

给定两两样本之间的距离度量函数，对于由两个或者多个样本合并之后得到的新簇，要如何确定它们与其他簇之间的距离呢？

## 注 簇间的距离

给定据类簇  $C_i$  和  $C_j$

- 最小距离:  $d_{\min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(\mathbf{x}, \mathbf{z})$
- 最大距离:  $d_{\max}(C_i, C_j) = \max_{x \in C_i, z \in C_j} \text{dist}(\mathbf{x}, \mathbf{z})$
- 平均距离:  $d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{z \in C_j} \text{dist}(\mathbf{x}, \mathbf{z})$

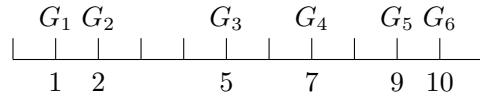
最小距离由两个簇的最近样本决定，最大距离由两个簇的最远样本决定，而平均距离由两个簇的所有样本共同决定。

**例 4.12** 层次聚类的结果是不是唯一的？

A 唯一

B 不唯一

**例 4.13** 设有 6 个样本，对每个样本测量一个特征，分别是 1、2、5、7、9、10，用基于最小距离的层次法将它们分类。

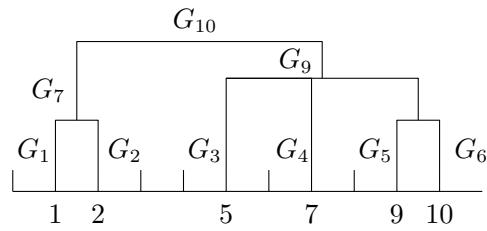


1. 选择绝对距离作为距离度量，形成距离矩阵  $D_{(0)}$

	$G_1$	$G_2$	$G_3$	$G_4$	$G_5$	$G_6$
$G_1$	0					
$G_2$	1	0				
$G_3$	4	3	0			
$G_4$	6	5	2	0		
$G_5$	8	7	4	2	0	
$G_6$	9	8	5	3	1	0

2. 矩阵  $D_{(0)}$  中最小元素是  $D_{12} = D_{56} = 1$ ，将  $G_1$  和  $G_2$  合并为  $G_7$ ， $G_5$  和  $G_6$  合并为  $G_8$ ，并计算新簇与其他簇的距离  $D_{(1)}$

	$G_7$	$G_3$	$G_4$	$G_8$
$G_7$	0			
$G_3$	3	0		
$G_4$	5	2	0	
$G_8$	7	4	2	0



3. 距离矩阵  $D_{(1)}$  中最小值是  $D_{34} = D_{48} = 2$ , 将  $G_4$  与  $G_3$  合并, 再与  $G_8$  合并, 因此  $G_3, G_4, G_8$  合并成为一个新簇  $G_9$ , 计算它与其他簇的距离  $D_{(2)}$

	$G_7$	$G_9$
$G_7$	0	
$G_9$	3	0

4. 最后将  $G_7$  和  $G_9$  合并成  $G_{10}$ , 这时所有的样本聚为一类, 聚类过程结束

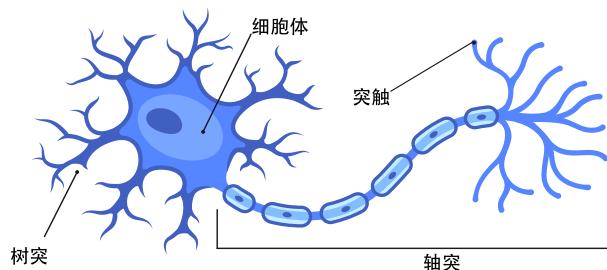
**注** 层次聚类的优缺点:

- 优点
  - 不需要预先设定聚类数
  - 可以发现簇的层次关系
  - 可以聚类成除球形簇之外的其它形状
- 缺点
  - 计算复杂度高
  - 对离群点敏感
  - 可能聚成链状

## 4.4 人工神经网络

### 4.4.1 生物神经元与神经网络

神经元是神经系统结构与功能的基本单元。



- 树突: 神经纤维接收器, 将电信号传送到细胞体

- 细胞体：对输入信号进行整合和阈值处理
- 轴突：把细胞体的输出信号导向其他神经元
- 突触：一个神经细胞的轴突和另一个神经细胞树突的结合点

**注** 生物神经元的信息处理过程：

- 信息输入
- 空间整合：同一时刻产生的刺激所引起的膜电位变化，大致等于各单独刺激引起的膜电位变化的代数和。
- 时间整合：各输入脉冲抵达神经元的时间先后不一样。总的突触后膜电位为一段时间内的累积。
- 信息产生
- 信息传导
- 兴奋或抑制

小结：树突从细胞体伸向其它神经元，神经元之间通过突触连接。通过突触输入的信号起着兴奋抑制作用。当细胞体接受的累加兴奋作用超过某阈值时，细胞进入兴奋状态，产生冲动，并由轴突输出。

**注** 生物神经元的基本特点：

- 神经元传递信息过程为多输入、单输出；
- 输入信号可以起兴奋作用，也可以起抑制作用；
- 神经元之间的连接强度可以随训练而改变，并决定信号传递的强弱；
- 每个神经元有一个“阈值”，一个神经元接受信号的累积效果决定该神经元的状态。

**例 4.14** 生物神经元基本特点不包括：

- A 多输入单输出
- B 时空整合功能
- C 兴奋性
- D 阈值特性

**例 4.15** 生物神经网络是如何实现信息的记忆和分布存储的？

- A 大脑里有类似于硬盘的存储脑区
- B 通过突触连接强度的改变进行存储

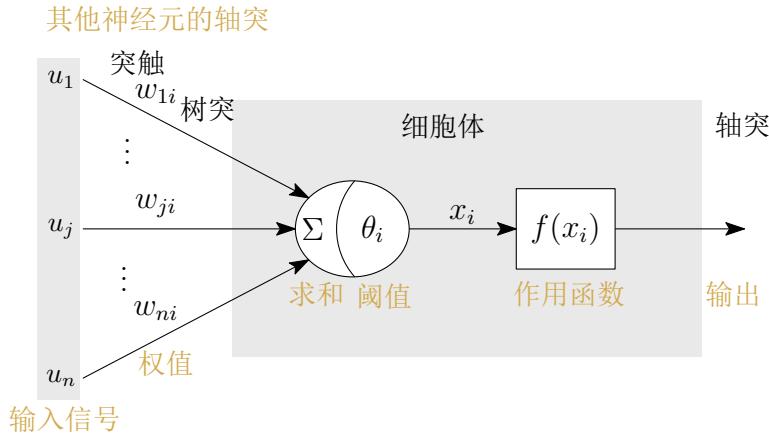
#### 4.4.2 人工神经网络三大要素

人工神经网络三大要素

- 神经元特性

- 网络结构
- 学习规则/学习算法

## 一、人工神经元数理模型

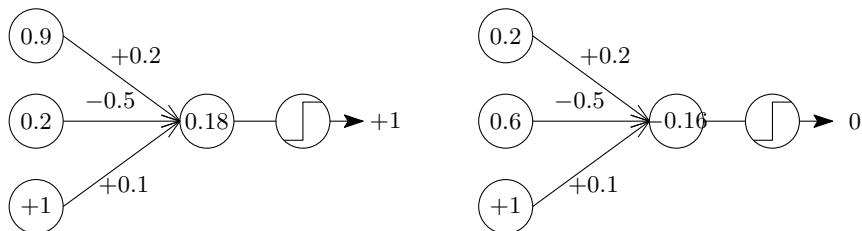


- 求和操作  $x_i = \sum_{j=1}^n w_{ji} u_j - \theta_i$
- 作用函数  $y_i = f(x_i) = f\left(\sum_{j=1}^n w_{ji} u_j - \theta_i\right)$ 
  - 作用函数控制输入对输出的激活作用
  - 作用函数对输入、输出进行函数转换
  - 作用函数将无限域的输入变成有限范围内的输出
- MP 神经元模型中的作用函数为单位阶跃函数：

$$f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

当神经元  $i$  的输入信号加权和超过阈值时，输出为“1”，即“兴奋”状态；反之输出为“0”，是“抑制”状态。

**例 4.16** 给定神经元的结构和权值，采用单位阶跃函数作为作用函数，对于不同的输入，分别计算神经元的输出。



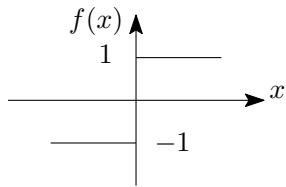
### 常见的神经元作用函数

神经元的作用函数反映了神经元输出与其激活状态之间的关系。神经元的各种数学模型的主要区别在于采用了不同的作用函数，从而使神经元具有不同的信息处理特性。

### • 阈值型

采用阶跃作用函数的神经元，称为阈值逻辑单元

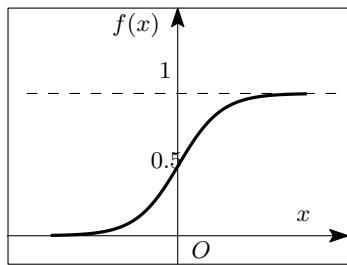
$$f(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$



### • 连续非线性作用函数

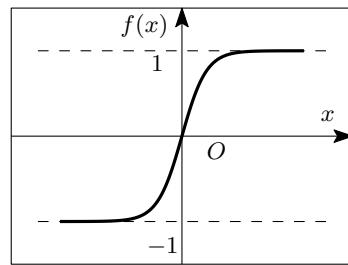
Sigmoid函数

$$f(x) = \frac{1}{1 + e^{-x}}$$



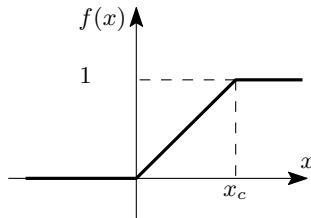
tanh函数

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



### • 分段线性作用函数

$$f(x) = \begin{cases} 0 & x < 0 \\ cx & 0 < x \leq x_c \\ 1 & x_c < x \end{cases}$$



## 人工神经元特性与生物神经元特性的比较

生物神经元	人工神经元
多输入单输出	$[u_1, u_2, \dots, u_n]$ 到 $y_i$ 的映射
信号分兴奋和抑制	输出值的正负对应兴奋和抑制
连接强度决定信号传递强弱	$w_{ji}$ 值为输入 $u_j$ 提供权值
连接强度可变	$w_{ji}$ 值可随训练改变
每个神经元有一个阈值	阈值 $\theta_i$ 表现阈值特性
信号累加效应决定神经元状态	净输入 $x_i$ 表现空间整合特性

例 4.17 关于作用函数的基本作用不正确的是：

- A 控制输入对输出的激活作用

- B 决定信号传递强弱
- C 对输入、输出进行函数转换
- D 将可能无限域的输入变换成指定的有限范围内的输出

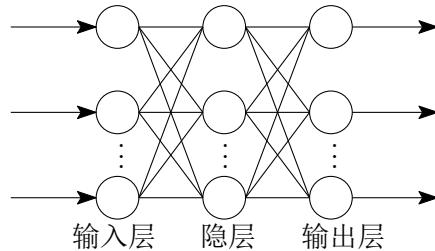
## 二、人工神经网络结构

**注** 人工神经网络结构的分类：

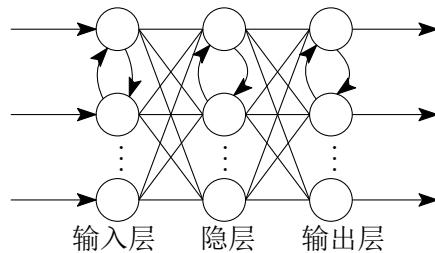
- 神经网络强大的计算功能是通过神经元的互连而达到的。根据网络拓扑结构形式不同，人工神经网络分为**层次型神经网络**和**互连型神经网络**。
- 如果从神经网络内部信息传递方向来看，人工神经网络可分为**前馈型神经网络**和**反馈型神经网络**。

**注** 层次型神经网络

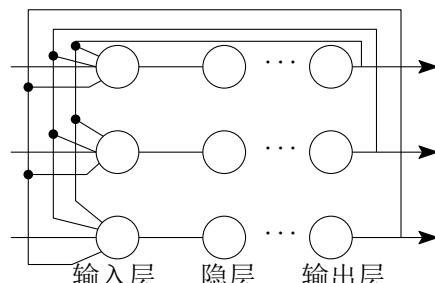
- 单纯型（前向神经网络）：神经元分层排列，顺序连接，神经元之间不存在反馈。如感知器、BP 神经网络等。



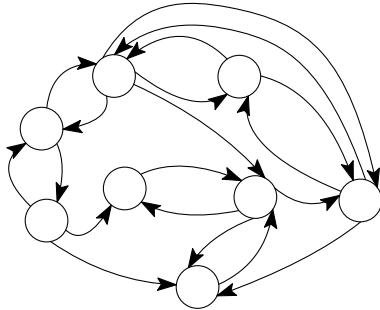
- 层内有互连型：层内神经元相互连接，实现同层内神经元间的横向抑制或兴奋机制，以实现各层神经元的自组织。



- 有反馈型：输出由当前输入和先前输出共同决定，类似于人类短期记忆。



**定义 4.10 (互连型神经网络)** 任意两个神经元之间都可能有相互连接。有的神经元之间的连接是双向的，有的是单向的。如 Hopfield 网络、Boltzman 机网络。



**注** 无反馈的前向网络和互连型网络：

- 在无反馈的前向网络中，信号一旦通过某个神经元，过程就结束了。
- 在互连型网络中，信号在神经元之间反复往返传递，网络处在不断改变状态的动态中。从初始状态开始，经过若干次变化，到达某平衡状态或进入周期振荡。

**注** 人工神经网络结构的比较：

- 前馈型神经网络

网络中各神经元接受前一层的输入，并输出到下一层，没有反馈。可实现信号从输入空间到输出空间的变换，信息处理能力来自于简单非线性函数的多次整合。网络结构简单，易于实现。适合于解决一般的函数逼近和预测问题。

- 反馈型神经网络

网络中神经元间有反馈，信息处理体现为状态的变换，可以用动力学系统描述。系统的稳定性与联想记忆功能有密切关系。适合于解决动态序列分析或联想记忆等问题。

### 三、学习算法

**注** 神经网络的运行阶段

- 学习阶段：也称自适应期或设计期，通过学习训练样本或其他方法调整权值矩阵；
- 工作阶段：各连接权值不再改变，用于求解实际问题。
- 学习的目的是从训练数据中提取隐含的知识和规律，并存储于网络中供工作阶段使用。
- 学习是改变各神经元连接权值的有效方法，也是体现人工神经网络智能特性最主要的标志。离开了学习，神经网络就失去了诱人的自适应、自组织能力。

**注** 神经网路的学习方式

- 有监督学习

根据实际输出与期望输出的偏差，按照一定的准则调整各神经元连接权值。期望输出是评价学习的标准，又称为有导师学习。

- 无监督学习

神经网络仅根据其输入调整权值，网络的学习评价标准隐含于内部。

**注** 神经网络的学习规则：Delta( $\delta$ ) 学习规则

- 对于输入  $u$ ，期望输出  $d$ ，实际输出与期望输出之间存在着误差  $e$

$$e = d - y$$

- 调整权值，使误差  $e$  减小到一定范围，设置目标函数

$$E = \frac{1}{2}e^2$$

该学习过程称为纠错学习，或 Delta 学习规则。

- 平方误差

$$E = \frac{1}{2}(d - y(t))^2 = \frac{1}{2}(d - f(\mathbf{W}^T(t)\mathbf{U}))^2$$

- 沿着负梯度方向调整权值

$$\begin{aligned}\Delta\mathbf{W}(t) &= -\eta \nabla E \\ &= \eta(d - y(t))f'(x(t))\mathbf{U}\end{aligned}$$

**例 4.18** 对于四输入单输出神经元设作用函数为双极性 Sigmod 函数，学习率  $\eta = 0.1$ ，输出为  $\mathbf{U}_1 = \begin{bmatrix} -1 & 1 & -2 & 0 \end{bmatrix}^T$ ,  $\mathbf{U}_2 = \begin{bmatrix} -1 & 0 & 1.5 & -0.5 \end{bmatrix}^T$ ,  $\mathbf{U}_3 = \begin{bmatrix} -1 & -1 & 1 & 0.5 \end{bmatrix}^T$ , 权值初始值为  $\mathbf{W}(0) = \begin{bmatrix} 0.5 & 1 & -1 & 0 \end{bmatrix}^T$ ，期望输出为  $d_1 = -1, d_2 = -1, d_3 = 1$ ，试按照  $\delta$  规则进行网络学习。

解：已知作用函数为  $f(x) = \frac{1 - e^{-x}}{1 + e^{-x}}$ ，则

$$f'(x) = \frac{2e^{-x}}{(1 + e^{-x})^2}, f'(x) = \frac{1}{2}[1 - f(x)^2]$$

1. 输入  $\mathbf{U}_1$ ，计算净输入  $x_1$  及权向量  $\mathbf{W}(1)$

$$\begin{aligned}x_1 &= \mathbf{W}^T(0)\mathbf{U}_1 = 2.5, y_1 = f(x_1) = 0.848 \\ f'(x_1) &= \frac{1}{2}(1 - y_1^2) = 0.140 \\ \mathbf{W}(1) &= \begin{bmatrix} 0.526 & 0.974 & -0.948 & 0 \end{bmatrix}^T\end{aligned}$$

2. 输入  $\mathbf{U}_2$ ，计算净输入  $x_2$  及权向量  $\mathbf{W}(2)$

$$\begin{aligned}x_2 &= \mathbf{W}^T(1)\mathbf{U}_2 = -1.948, y_2 = f(x_2) = -0.75 \\ f'(x_2) &= \frac{1}{2}(1 - y_2^2) = 0.218 \\ \mathbf{W}(2) &= \begin{bmatrix} 0.531 & 0.974 & -0.956 & 0.002 \end{bmatrix}^T\end{aligned}$$

### 3. 输入 $\mathbf{U}_3$ , 计算净输入 $x_3$ 及权向量 $\mathbf{W}(3)$

$$x_3 = \mathbf{W}^T(2)\mathbf{U}_3 = -2.46, y_3 = f(x_3) = -0.842$$

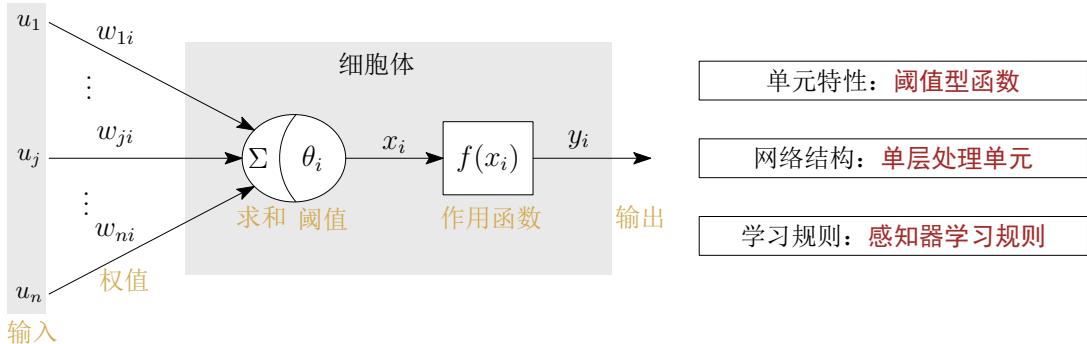
$$f'(x_3) = \frac{1}{2}(1 - y_3^2) = 0.145$$

$$\mathbf{W}(3) = \begin{bmatrix} 0.505 & 0.947 & -0.929 & 0.016 \end{bmatrix}^T$$

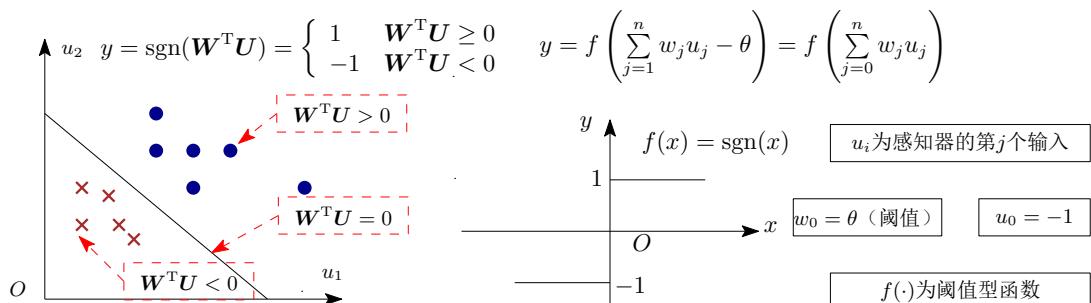
注 Delta( $\delta$ ) 学习规则的特点

- $\delta$  规则要求作用函数可导, 适用于有监督学习的连续作用函数, 如 Sigmoid 函数。可推广到多层前向网络, 权值可初始化为任意值。

#### 4.4.3 感知器

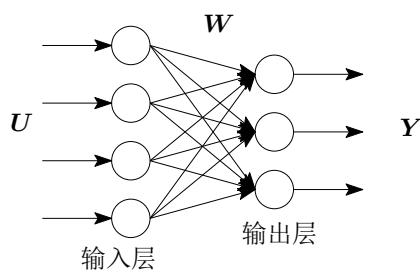


注 神经元单层特性: 感知器是一种线性分类模型



例 4.19 若感知器输入为  $n$  维, 输出为  $m$  维, 那么可以实现将样本分为  $2^m$  类

注 网络结构: 单层处理单元



例 4.20 感知器学习规则是有监督的还是无监督的?

- A 有监督
- B 无监督

例 4.21 感知器学习得到的分类面是否唯一?

- A 是
- B 否

注 感知器学习规则:

初始化权值向量  $w_j (j = 0, 1, \dots, n)$ , 每分错一个样本, 就用它来更新权值。权值调整规则为

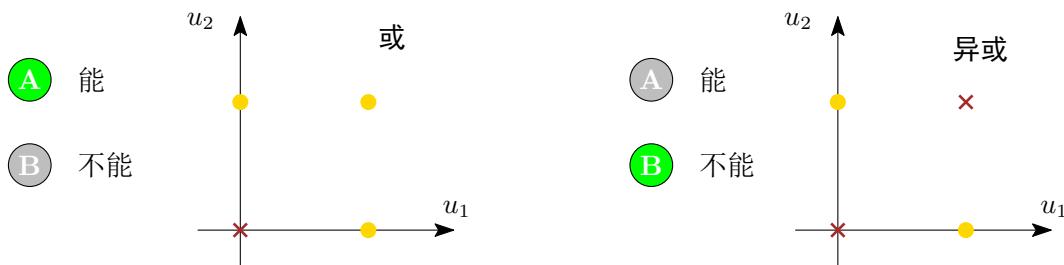
$$w_j(t+1) = w_j(t) + \eta (d_p - y_p(t)) u_{jp}, \eta > 0$$

$\eta$	学习率
$d_p - y_p(t)$	学习误差: 输出信号
$u_{jp}$	输入量

感知器学习规则是  $\delta$  学习规则的一种特殊情况, 它不需要对作用函数求导数。不仅学习速度较快, 而且具有较高的精度。权值可以初始化为任意值。

例 4.22 单层感知器-或、异或

1. 单层感知器能否学习实现逻辑函数“或”运算? 为什么?
2. 单层感知器能否学习实现逻辑函数“异或”运算? 为什么?

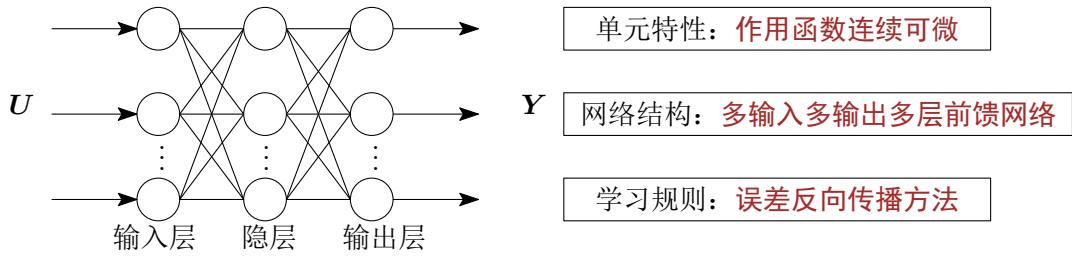


注 单层感知器的局限性:

- 若输入的两类模式是线性可分集合 (指存在一个超平面能将其分开), 则算法一定收敛。
- 若输入模式为线性不可分集合, 网络的学习算法不收敛, 无法进行正确分类。

#### 4.4.4 BP 神经网络

BP 神经网络是一种特殊的多层感知器模型。

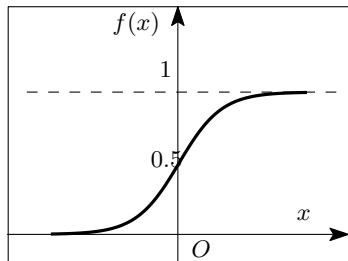


**例 4.23** 以下哪些神经元作用函数是连续可微的?

- A 单极型 Sigmoid 函数
- B 双极型 Sigmoid 函数
- C 阈值型函数
- D 分段线性函数

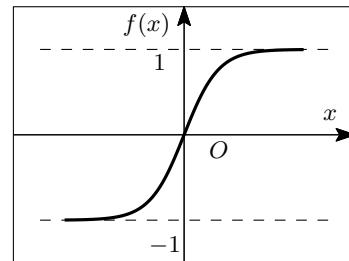
单极型Sigmoid函数

$$f(x) = \frac{1}{1 + e^{-x}} \quad f'(x) = f(x)(1 - f(x))$$



双极型Sigmoid函数

$$f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \quad f'(x) = \frac{2}{1 + e^{-x}} \left(1 - \frac{1}{1 + e^{-x}}\right)$$



**注** BP 网络结构

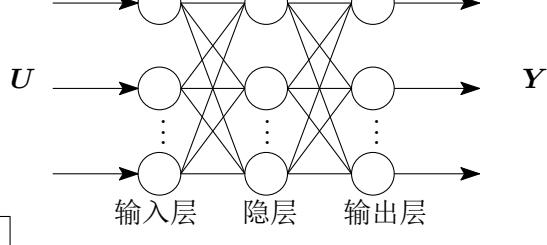
设网络的层数为  $L$ , 第  $l$  层 ( $0 \leq l \leq L$ )

$l = 0$  表示输入层

$l = L$  表示输出层

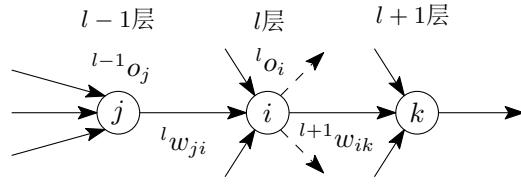
$0 \leq l \leq L$  表示隐层

第  $l$  层的权系数记为  ${}^l W$



**注** BP 网络相关记号

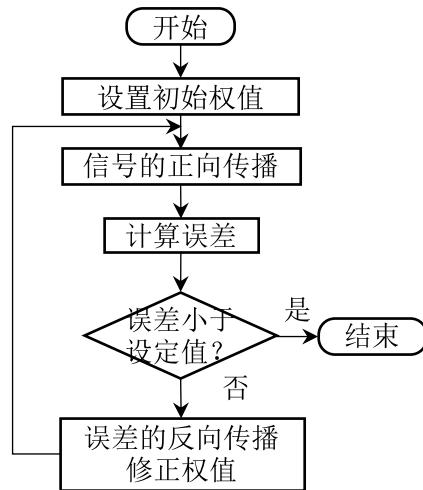
- ${}^l o_i = f({}^l x_i)$  是神经元  $i$  的输出,  ${}^l x_i$  为净输入,  $f(\cdot)$  为作用函数
- ${}^0 O = u$  为输入信号,  ${}^L O = y$  为输出信号



- $l w_{ji}$  是  $l - 1$  层第  $j$  个神经元与第  $l$  层第  $i$  个神经元的连接权值
- 对于 BP 网络，设输入向量  $u$  是  $n$  维的，输出向量是  $m$  维的。已获得  $P$  个输入/输出样本对，记第  $p$  个样本为  $\{u_p, d_p\}$

**注** BP 学习的基本思想

在外界输入样本的刺激下，不断改变网络的连接权值，使得网络的实际输出不断地接近期望的输出。



**例 4.24** BP 神经网络的学习方式是：

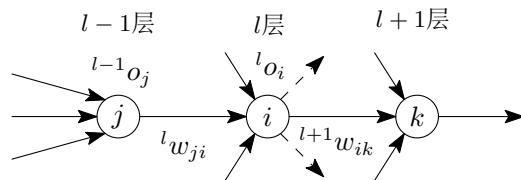
A 有监督学习

B 无监督学习

**注** BP 学习的数学原理

### 1. 利用误差梯度修正权系数

$$l w_{ji}(t + 1) = l w_{ji}(t) - \eta \frac{\partial E_p}{\partial l w_{ji}}, \eta > 0$$



$$\frac{\partial E_p}{\partial l w_{ji}} = \frac{\partial E_p}{\partial l x_{ip}} \cdot \frac{\partial l x_{ip}}{\partial l w_{ji}} = l \delta \cdot l^{-1} o_{jp}$$

## 2. 输出误差

$$E_p = \frac{1}{2} \sum_{i=1}^m (d_{ip} - y_{ip})^2 = \frac{1}{2} \sum_{i=1}^m (d_{ip} - f(\mathbf{x}_i)^T \mathbf{w})^2$$

## 3. 输出层灵敏度

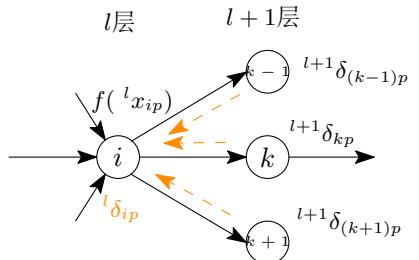
$${}^L\delta_{ip} = \frac{\partial E_p}{\partial {}^Lx_{ip}} = -(d_{ip} - y_{ip}) \cdot f'({}^Lx_{ip})$$

## 4. 对于非输出层（利用向量的链式法则）

$${}^l\delta_{ip} = \frac{\partial E_p}{\partial {}^lx_{ip}} = \left[ \frac{\partial {}^{l+1}\mathbf{x}_p}{\partial {}^lx_{ip}} \right]^T \cdot \frac{\partial E_p}{\partial {}^{l+1}\mathbf{x}_p}$$

$$\begin{aligned} {}^l\delta_{ip} &= \frac{\partial E_p}{\partial {}^lx_{ip}} = \left[ \frac{\partial {}^{l+1}\mathbf{x}_p}{\partial {}^lx_{ip}} \right]^T \cdot \frac{\partial E_p}{\partial {}^{l+1}\mathbf{x}_p} \\ &= \frac{\partial {}^{l+1}x_{kp}}{\partial {}^lx_{ip}} = \frac{\partial E_p}{\partial {}^{l+1}x_{kp}} \\ &= \frac{\partial \sum_i {}^{l+1}w_{ik} {}^lO_{ip}}{\partial \sum_j {}^lx_{ip}} \\ &= {}^{l+1}w_{ik} \cdot f'({}^lx_{ip}) \end{aligned}$$

## 5. $\delta$ 信号的 BP 过程



$${}^l\delta_{ip} = \frac{\partial E_p}{\partial {}^lx_{ip}} = \left( \sum_k {}^{l+1}\delta_{kp} \cdot {}^{l+1}w_{ik} \right) \cdot f'({}^lx_{ip})$$

## 6. $\delta$ 信号的反向传播

- 误差梯度

$$\frac{\partial E_p}{\partial {}^lw_{ji}} = {}^l\delta \cdot {}^{l-1}o_{jp}$$

- 输出层信号

$${}^L\delta_{ip} = \frac{\partial E_p}{\partial {}^Lx_{ip}} = -(d_{ip} - y_{ip}) \cdot f'({}^Lx_{ip})$$

- 灵敏度

$${}^l\delta_{ip} = \frac{\partial E_p}{\partial {}^lx_{ip}}$$

- 其他层 ( $\delta$  信号反向传播)

$${}^l\delta_{ip} = \frac{\partial E_p}{\partial {}^l x_{ip}} = \left( \sum_k {}^{l+1}\delta_{kp} \cdot {}^{l+1}w_{ik} \right) \cdot f'({}^l x_{ip})$$

- 权值调整公式

$$\Delta \mathbf{W}(t) = \eta (d - y(t)) f'(x(t)) \mathbf{U}$$

**例 4.25** BP 算法中的灵敏度  $\delta$  跟 Delta( $\delta$ ) 学习规则中的  $\delta$  的关联是：

- A 两者没什么关系
- B 灵敏度  $\delta$  对应于 Delta( $\delta$ ) 学习规则中的  $\delta$

**例 4.26** 基本 BP 算法步骤

1. 设置初始权值  $\mathbf{W}(0)$  为较小的随机非 0 值
2. 给定输入/输出样本集合  $\{\mathbf{u}_p, \mathbf{d}_p\}$ , 重复以下过程直至满足收敛条件  $E_{\text{总}} \leq \varepsilon$ 
  - 对于任意一个样本  $p$ , 计算

正向过程  $\mathbf{u}_p, \dots, {}^{l-1}\mathbf{o}_p, {}^l\mathbf{x}_p, \dots, \mathbf{y}_p$

反向过程

$$\begin{cases} {}^L\delta_{ip} = -(d_{ip} - y_{ip}) \cdot f'({}^L x_{ip}) \\ {}^l\delta_{ip} = \left( \sum_k {}^{l+1}\delta_{kp} \cdot {}^{l+1}w_{ik} \right) \cdot f'({}^l x_{ip}) \\ \frac{\partial E_p}{\partial {}^l w_{ji}} = {}^l\delta_{ip} \cdot {}^{l-1}O_{jp} \end{cases}$$

$$• \text{权值修正 } {}^l w_{ji}(t+1) = {}^l w_{ji}(t) - \eta \frac{\partial E_p}{\partial {}^l w_{ji}}$$

**注** BP 网络的训练方式

- 串行方式  
每获得一个样本, 就计算一次误差并更新权值
- 批量方式  
在所有样本输入后, 根据总误差计算各层的误差信号并调整权值

**例 4.27** 下列说法正确的是：

- A 串行方式需要的存储空间较少
- B 串行方式需要的计算量较少
- C 批量方式比串行方式更容易实现并行化
- D 批量方式的学习速度往往优于串行方式

## 5 深度学习与强化学习

## 5.1 深度学习

### 5.1.1 深度学习的基本概念

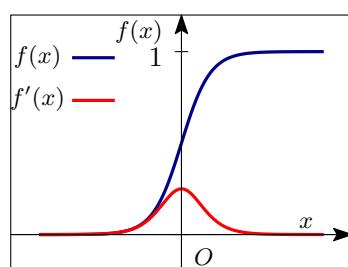
**注** 深度学习提出的背景

1. 非结构化大数据的涌现
2. 特征工程的瓶颈
  - 良好的**特征表达**，对最终算法的性能起了关键作用，而且系统主要的计算和测试工作都耗在这一部分。但是实际中一般是靠**人工提取特征**。
  - 机器学习中，获得好的特征是识别成功的关键。
  - 人工选取特征费时费力，需要启发式专业知识，很大程度上靠经验和运气。
3. 自动特征提取：生物视觉机理的启发
  - 机器可以自动地学习特征
  - 人工神经网络是分层的；输入层输入原始图像，第一个隐层的神经元从中提取低层特征，后面隐层的神经元在此基础上获得高层特征；输出层用来分类。
4. 传统神经网络的局限

BP 算法无法胜任深层网络的训练

- 反馈调整时，梯度越来越稀疏，误差校正信号越来越小，容易出现梯度弥散或梯度消失现象；
- 容易收敛到局部最优，由于采用随机值初始化，当初值远离最优区域时易导致这一情况；
- 待训练的参数较多，需要大量有标签数据来训练，容易出现过拟合，无法用于无标签数据。

$$f(x) = \frac{1}{1 + e^{-x}} \quad f'(x) = f(x)(1 - f(x))$$



**定义 5.1 (深度学习的概念)** 深度学习是一种基于多层神经网络的机器学习方法，主要基于人工神经网络（Artificial Neural Network Network, ANN）实现，是人工神经网络的“深度”版本，具有更强的数据表达能力。

**注** 深度学习概念的分析

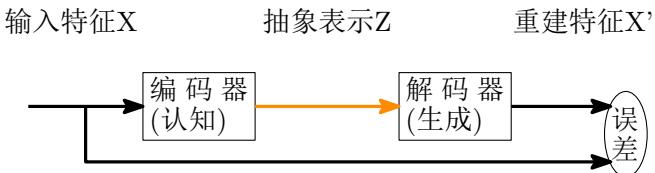
- 通过构建多隐层模型和海量训练数据（可为无标签数据），来学习更有用的特征，从而提升分类或预测的准确性。“深度模型”是手段，“特征学习”是目的。
- 与浅层学习的区别  
强调了模型结构的深度，通常有5~10个或更多的隐层。突出了特征学习的重要性，与人工提取特征的方法相比，利用大数据来学习特征，能够刻画数据的丰富内在信息。
- 深度学习网络的三要素
  - 单元特性：多种函数可选
  - 网络结构：多隐层的前向层次网络，主要为全连接
  - 学习规则：逐层训练机制

### 5.1.2 深度学习的训练过程

**注** 深度学习的逐层训练方式

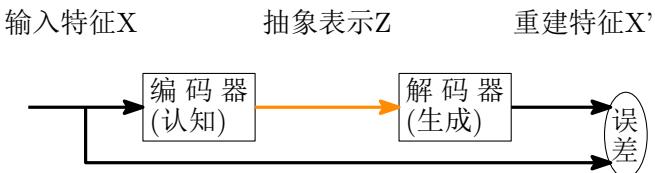
#### 1. 第一步：采用自下而上的无监督学习

- 每次仅调整一层，每层采用wake sleep算法进行调优。
- 这可以看作是一个特征学习的过程，是与传统神经网络区别最大的部分。
  - wake阶段



通过下层的输入特征 $X$ 和编码器上行的认知权重产生每层的抽象表示 $Z$ ，再通过当前（解码器）的生成权重产生重建信息 $X'$ ，计算输入特征 $X$ 和重建信息 $X'$ 的残差，使用梯度下降修改层间（解码器）下行的生成（权重）。

#### 2. sleep阶段

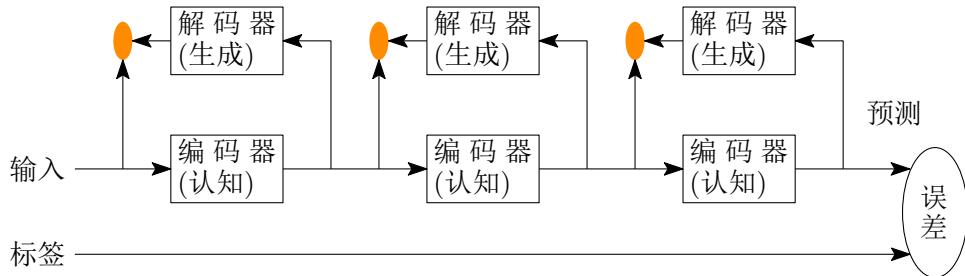


通过抽象表示 $Z$ 和（解码器）下行的生成权重，生成下层的特征 $X'$ ，再利用（编码器）上行的认知权重产生抽象表示 $Z'$ 。利用初始抽象表示 $Z$ 和新建的抽象表示 $Z'$ 的残差，利用梯度下降修改层间（编码器）上行的认知（权重）。

#### 2. 第二步：自顶向下的有监督学习

在学习获得各层参数的基础上，在最顶的编码层添加一个分类器，通过带标签数据的有监督学习，利用梯度下降法去微调整整个网络参数。

**注** 逐层训练方式的特点



深度学习的第一步实质上是一个网络参数初始化过程。区别于传统神经网络的随机初始化，深度学习模型是通过无监督学习输入数据的结构得到的，因而这个初值更接近全局最优，能够取得更好的效果。

**定理 5.1 (通用逼近定理)** 通用逼近定理声明，包含至少一个隐层的多层前馈网络，只要其隐层神经元数量足够，能以任意精度逼近任意的连续函数。

**注** 深度学习的优势：通过学习深层非线性网络结构，实现复杂函数逼近，对输入数据进行分布式表示。

- 学习更加有效：深层网络含隐层数目较多而隐层的单元数量相对较少，如果浅层网络想达到同样的计算结果，需要**指数级增长的单元数量**。使用更深层的模型可以减少表示函数所需的单元数，并减少泛化误差。
- 能够逐层提取特征：具有单隐层的前馈网络足以表示任何函数，但是该层**可能过大而无法正确地学习和提取特征**。逐层加工处理是深度学习成功的关键因素。
- 通过学习一种深层非线性网络结构，实现复杂函数逼近，表征输入数据分布式表示；
- 能够自动提取特征，对于各层学习到的特征有更好的解释性；
- 能够更充分地利用无标签数据，不容易陷入局部最优；
- 结合大数据，不容易出现过拟合现象。
- 不需要人为特征提取，通过算法直接获取特征。在学习过程中不进行分模块或分阶段训练，直接优化任务的总体目标。中间过程不需要人为干预，使用起来更加方便。

**注** 深度学习的局限性

- 主要体现在网络结构难设计、重复性差、结果不具可解释性、易受欺骗等，以及在小数据集上难以使用、黑箱模型、理论分析困难等。
- 典型有监督学习模型
  - 卷积神经网络

- 循环神经网络
- 深度前馈神经网络
- 胶囊网络
- 典型无监督学习模型
  - 限制玻尔兹曼机
  - 深度信念网络
  - 生成对抗网络
  - 自编码器

### 5.1.3 卷积神经网络

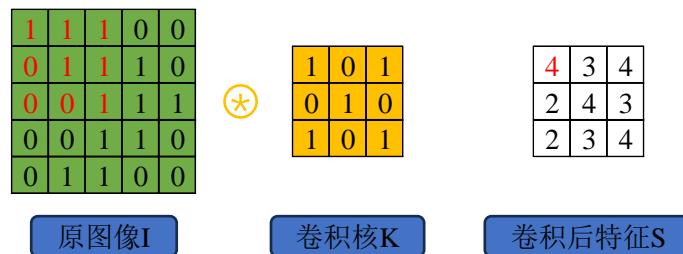
CNN 是一种带有卷积结构的深度神经网络，通常至少有两个可训练的卷积层（包括非线性作用函数）、两个池化层和一个全连接层，一共至少 5 个隐含层。

**注** 卷积神经网络主要执行四个操作：

- 卷积
- 非线性 (ReLU)
- 池化或下采样
- 分类 (全连接层)

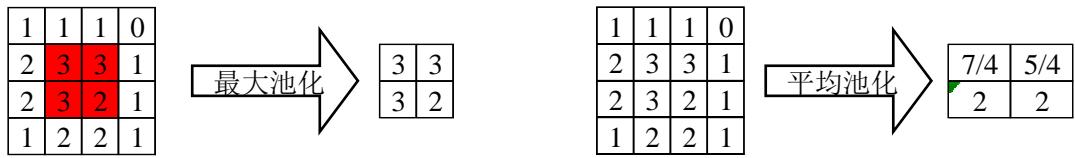
**注** 卷积层

$$S(m, n) = (\mathbf{I} * \mathbf{K})(m, n) = \sum \sum \mathbf{I}(m + i, n + j) \mathbf{K}(i, j)$$



**注** 池化层

- 池化层通常紧接着在卷积层之后使用，简化从卷积层输出的信息。
- 使用“压缩”方法，通过一个下采样的过程，来减小图像的规模。
- 由于卷积得到的特征图中含有对于识别物体不必要的冗余信息，而通过下采样可以去除这些冗余信息，所以通常不会影响识别结果。
- 池化降低了每个特征映射的维度，但是保留了最重要的信息。



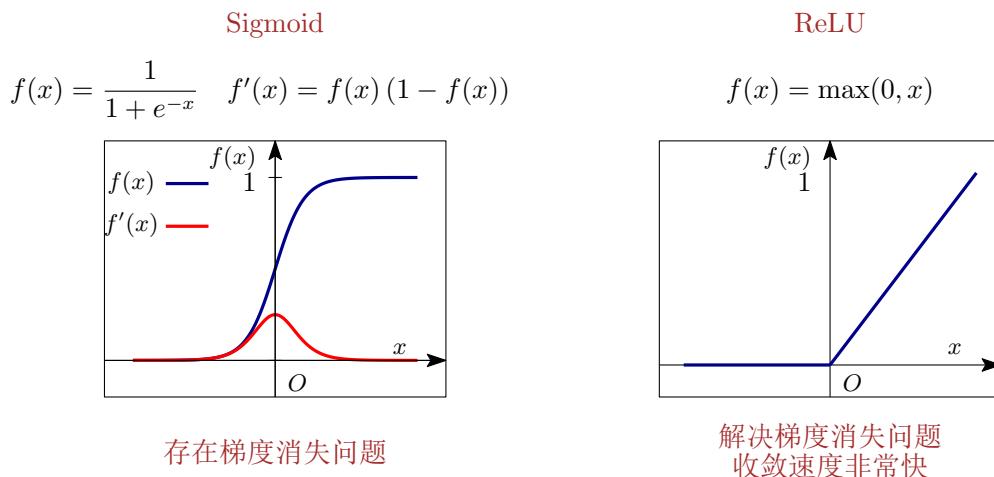
- 池化有多种形式：最大、平均、求和等。最大池化效果最好。

### 池化层的功能

- 减少网络中待计算的参数数量，从而遏制过拟合
- 增强网络对输入图像中小变形、扭曲、平移的鲁棒性
- 帮助获得不因尺寸而改变的等效图片特征。

### 注 非线性激励层

- 可用于卷积层和池化层之后，也可用于两层之间。
- 它是深度网络非线性的主要来源。



### 注 全连接层和输出层

- 上层和下层的每个神经元之间都是相互连接的。
- 卷积层和池化层的输出代表了输入图像的高级特征，全连接层的目的是利用这些特征进行分类。
- 计算样本属于每类的概率作为输出。
- 输出层使用单极性 Sigmoid 函数或归一化指数函数（Softmax 函数）计算样本属于每类的概率，输出分类概率或标签。

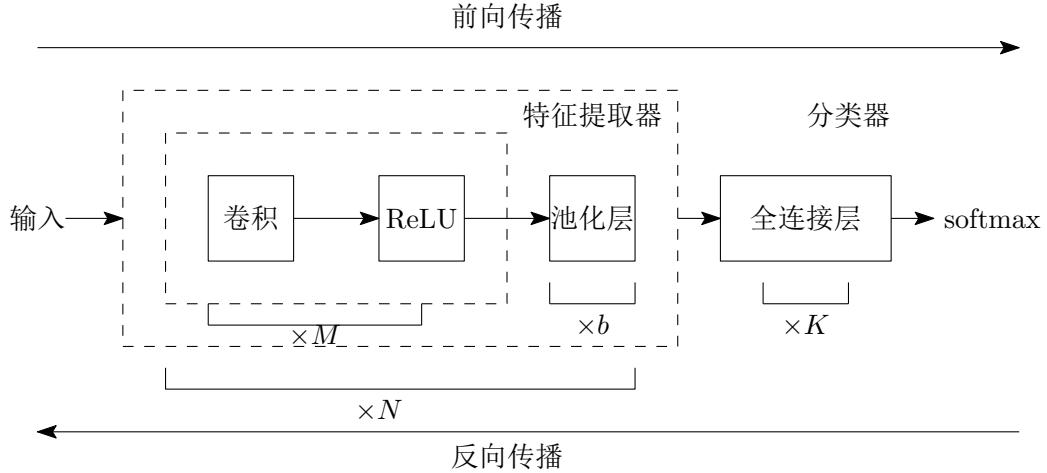
**例 5.1** 相对于全连接网络，卷积神经网络需要训练的参数规模大大减小，得益于下列哪些操作？

- A 池化
- B 池化

C 非线性作用函数

D 全连接

**注** 常用的卷积网络整体结构



- 一个卷积块包括  $M$  个卷积层和  $b$  个池化层 ( $M$  通常设置为  $2 \sim 5$ ,  $b$  为 0 或 1)
- 一个卷积网络可以堆叠  $N$  个连续的卷积块, 然后在后面接着  $K$  个全连接层 ( $N$  的取值区间比较大, 如  $1 \sim 100$  或更大,  $K$  一般为  $0 \sim 2$ )

**例 5.2** 卷积神经网络中需要学习的参数有:

A 卷积核内的参数

B 卷积核的数目

C 池化大小

D 全连接权值

**注** 现代深度学习的训练方法

- 避免欠拟合的方法

- 增加训练次数

- 改变网络结构, 如增加网络的深度和每一个隐藏层的神经元个数

- 调整学习率

- 避免过拟合的方法

- 数据增强 (Data Augmentation)

- 旋转、缩放、翻转、加噪声

- 使用合适的模型: 减少网络的层数、神经元个数

- 正则化

通过约束权重的 L1 范数或者 L2 范数, 对模型的复杂度进行惩罚, 减小模型在训练数据集上的过拟合。

$$E_1 = E + \frac{\lambda}{2} \sum \sum (\ell w_{ij})^2$$

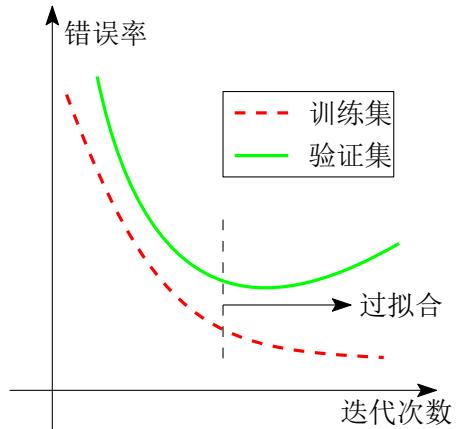
$$\delta \ell w_{ij} = -\eta \left( \frac{\partial E}{\partial \ell w_{ij}} + \lambda \ell w_{ij} \right)$$

- Dropout

在深度网络学习的训练过程中，对于神经网络单元，按照一定的概率将其暂时从网络中丢弃。（**在网络测试时不丢弃神经元！**）

- 限制训练时间；或者通过评估测试，提前终止训练

在每次迭代时，把新学习到的模型在验证集上进行测试，并计算错误率。验证集上的错误率通常会先下降后上升，拐点处预示着开始过拟合，就停止迭代。

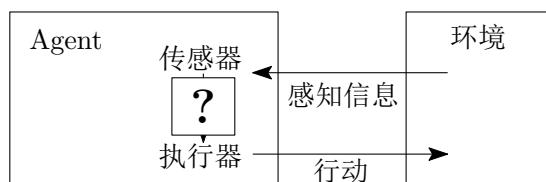


## 5.2 强化学习

### 5.2.1 强化学习基本概念

**定义 5.2 (强化学习)** 强化学习是一种从环境状态到动作映射的学习，目标是使动作从环境中获得的累积奖赏值最大。

强化学习的生理基础：强化学习的最重要的反馈信号是多巴胺 基本思想



- Agent 以奖励的形式得到环境的反馈；
- Agent 学习如何更好行动以极大化累积奖励；
- 基于观察环境反馈的结果进行学习。

**注** 强化学习中的常用术语

- Agent: 在环境中执行动作以获得奖励的某个实体
- 环境 ( $e$ ): Agent 面对的场景
- 奖励 ( $r$ ): Agent 执行特定动作或任务时获得的收益
- 状态 ( $s$ ): 状态是指环境返回的当前状态
- 策略 ( $\pi$ ): Agent 根据当前状态决定下一步采取什么动作
- 价值 ( $V$ ): 将来累积总收益的期望值。
- 价值函数  $V(s)$ : 用价值函数来对当前状态进行估价, 进入现在这个状态, 可以对 Agent 后面的收益带来多大的影响。当这个价值函数大的时候, 说明 Agent 进入这个状态越有利。
- 环境模型: 模型表示了 Agent 对这个环境的状态进行了理解。
- Q 值或行动值 ( $Q(s, a)$ ): Q 值与价值非常相似, 对 (状态-动作) 对进行评价。

**注** 强化学习方法的分类:

- 基于使用策略与使用值函数, 强化学习算法分为三类:
  - 基于值函数的强化学习算法
  - 基于策略的强化学习算法
  - Actor Critic 算法
- 基于是否使用模型, 强化学习算法分为两类:
  - 无模型算法 (Model Free)
  - 基于模型的强化学习算法 (Model Based)

“模型”特指环境, 即环境的动力学模型, 与深度学习中的模型不同。

**例 5.3** 强化学习的主要特点有那哪些?

- A Agent 需要通过探索环境来获取对环境的理解
- B Agent 执行动作从环境里面获得奖励
- C 在训练过程中, 时间非常重要。因为 Agent 得到的数据都是有时间关联的 (sequential data), 而不是独立同分布的
- D Agent 的行为会影响它随后得到的数据

### 5.2.2 马尔科夫决策过程

**注** 5 个关键要素:

- $S$  为有限的状态集
- $A$  为有限的动作集
- $P$  为状态转移概率

$$P_{ss'}^a = P(S_{t+1} = s' | S_t = s, A_t = a)$$

- $R$  为奖励函数

$$R_{ss'}^a = E(R_{t+1}|S_t = s, A_t = a, S_{t+1} = s')$$

Agent 的目标是最大化回报值: ( $\gamma \in [0, 1]$  为折扣因子)

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$

### 注 马尔科夫决策过程数学模型

- 在随机过程中, 假设状态历史为  $h_t = \{s_0, \dots, s_t\}$ , 马尔科夫性是指当前状态的下一个状态只取决于当前状态,

$$\begin{aligned} P(s_{t+1}|s_t) &= P(s_{t+1}|h_t) \\ P(s_{t+1}|s_t, a_t) &= P(s_{t+1}|h_t, a_t) \end{aligned}$$

- 策略函数:

随机性策略:

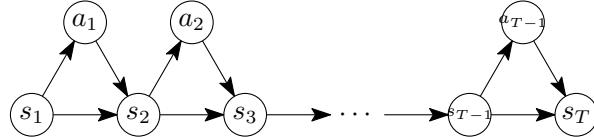
$$\pi(a|s) = P(A_t = t|S_t = s)$$

确定性策略:

$$a^* = \arg \max_a (a|s)$$

- 马尔可夫决策过程是一个轨迹

$$\tau = s_0, a_0, s_1, r_1, a_1, \dots, s_{T-1}, a_{T-1}, s_T, r_T$$



$$\begin{aligned} p(\tau) &= p(s_0, a_0, s_1, a_1, \dots) \\ &= p(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t) p(s_{t+1}|s_t, a_t) \end{aligned}$$

- 给定策略  $\pi(a|s)$ , 智能体和环境一次交互过程中的轨迹  $\tau$  所收到的累计奖励作为总回报

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$

$\gamma \in [0, 1]$  是折扣率。当  $\gamma$  接近于 0 时, 智能体更在意短期回报; 而当  $\gamma$  接近于 1 时, 长期回报变得更重要。

- 状态值函数:

状态值函数也简称为值函数, 是确定 Agent 在策略  $\pi$  下处于某一特定状态  $s$  的最佳程度, 即在策略  $\pi$  下从  $s$  开始获得的期望回报, 表示为

$$V_\pi(s) = E_\pi(G_t|s_t = s)$$

根据  $G_t$  的表达式，可以得到

$$V_\pi(s) = E_\pi \left( \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | s_t = s \right)$$

- 状态-动作值函数 (Q 函数)

状态-动作值函数，也称作 Q 函数，用于表明遵循策略  $\pi$  再某一状态执行特定动作的最佳程度，也就是遵循策略  $\pi$  在状态  $s$  执行动作  $a$  开始获得的期望汇报

$$Q^\pi(s, a) = E_\pi(R_t | s_t = s, a_t = a)$$

根据  $G_t$  的表达式

$$Q^\pi(s, a) = E_\pi \left( \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | s_t = s, a_t = a \right)$$

**例 5.4** 强化学习只能通过马尔科夫决策过程进行建模？

A 是

B 否

**注** 最优策略与贝尔曼方程

- 强化学习的最终目的是找到一个最优策略；

最优策略对应状态值  $V$  函数和  $Q$  函数的最大值  $V^*(s)$  和  $Q^*(s, a)$

$$V^*(s) = \max_\pi V^\pi(s)$$

$$Q^*(s, a) = \max_\pi Q^\pi(s, a)$$

- 递推计算最优状态值的贝尔曼方程：

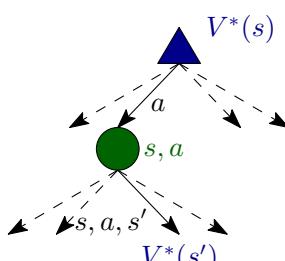
根据：

$$V^* = \max_a Q^*(s, a)$$

$$Q^*(s, a) = \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s'))$$

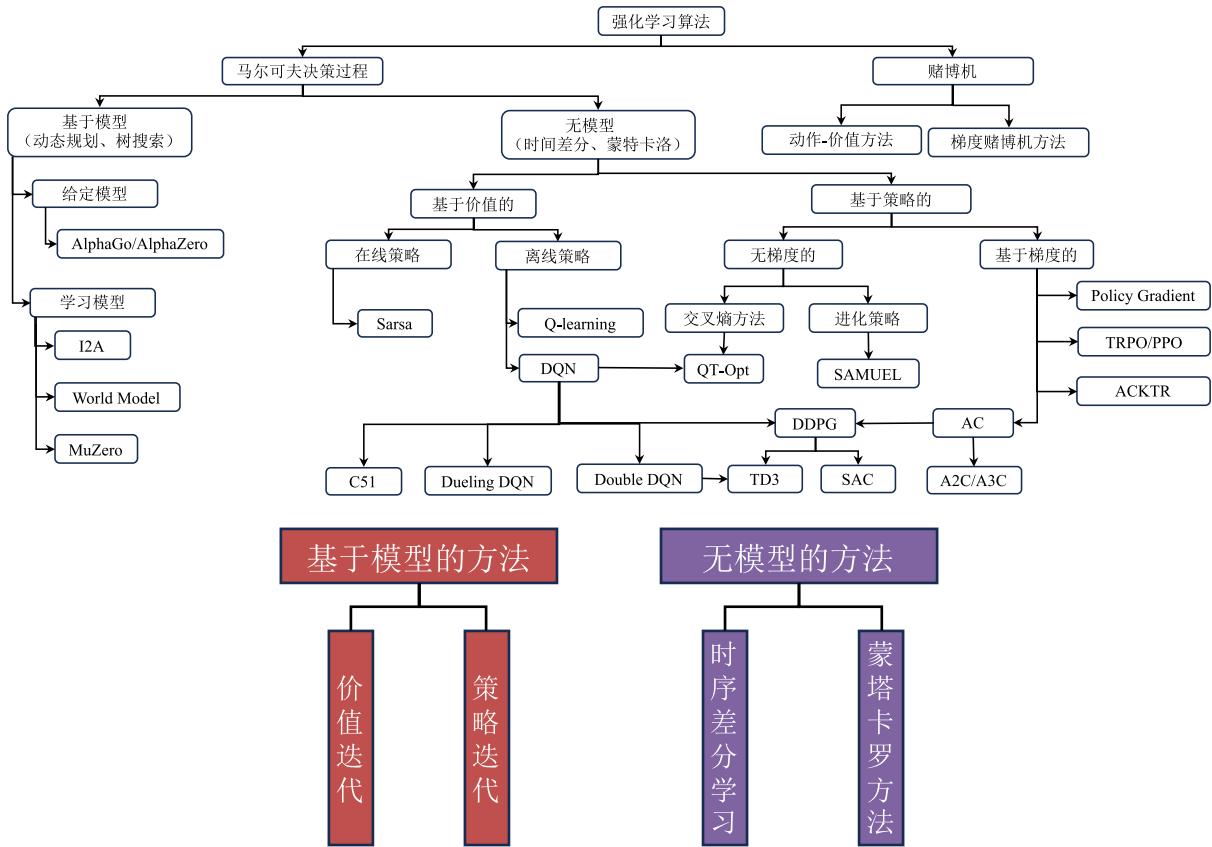
得到相邻状态最优值关系：

$$V^*(s) = \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s'))$$



- 可以得到对应的最优策略

$$\pi^*(s) = \arg \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s'))$$



### 5.2.3 典型强化学习算法

“模型”特指环境，即环境的动力学模型，与深度学习中的模型不同。

**注** 基于模型的方法

- 基本思想：
  - 学习得到一个近似环境模型；
  - 基于该近似模型进行求解。
- 步骤一：学习训练得到模型
  - 统计每个  $(s, a)$  情况下的结果  $s'$
  - 标准化后得到的估计值
  - 发现  $(s, a, s')$  对应的奖励值
- 步骤二：基于学得的模型进行求解
  - 和之前一样，可以使用**价值迭代**、**策略迭代**等算法
- 基于模型的方法：价值迭代和策略迭代均需要**环境模型**（即状态转移函数  $P$  和回报函数  $R$ ）：
  - 状态值迭代

$$V_{k+1}(s) = \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')) , \forall s$$

- Q 值迭代

$$Q_{k+1}(s) = \arg \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')), \forall s, a$$

- 策略提炼

$$\pi_V(s) = \arg \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')), \forall s$$

- 策略评估

$$V_{k+1}^{\pi_i}(s) = \sum_{s'} P_{ss'}^{\pi_i(s)} \left( R_{ss'}^{\pi_i(s)} + \gamma V_k^{\pi_i}(s') \right), \forall s$$

- 策略改进

$$\pi_{i+1}(s) = \arg \max_a \sum_{s'} P_{ss'} (R_{ss'}^a + \gamma V^{\pi_i}(s')), \forall s$$

**例 5.5** 如何得到智能体的环境模型（即 P 函数和 R 函数）？

规则：从 A 出去减 10 分，从 D 出去加 10 分

B,east,C,-1 C,east,D,-1 D,exit,x,+10	B,east,C,-1 C,east,D,-1 D,exit,x,+10	P(B,east,C) = 1.00 P(C,east,D) = 0.75 P(C,east,A) = 0.25
E,north,C,-1 C,east,D,-1 D,exit,x,+10	E,north,C,-1 C,east,A,-1 A,exit,x,-10	R(B,east,C) = -1 R(C,east,D) = -1 R(D,exit,x) = +10

**例 5.6** 基于模型的强化学习方法需要知道环境模型，其主要的求解方法有以下哪些？

- A 价值迭代方法
- B 时序差分方法
- C 策略迭代方法
- D 蒙特卡罗方法

**注** 无模型的方法解决 P 函数和 R 函数未知时的决策问题：

- 状态值迭代

$$V_{k+1}(s) = \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')), \forall s$$

- Q 值迭代

$$Q_{k+1}(s) = \arg \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')), \forall s, a$$

## • 策略提炼

$$\pi_V(s) = \arg \max_a \sum_{s'} P'_{ss'} (\cancel{R'_{ss'}} + \gamma V_k(s')) , \forall s$$

## • 策略评估

$$V_{k+1}^{\pi_i}(s) = \sum_{s'} P_{ss'}^{\pi_i(s)} \left( R_{ss'}^{\pi_i(s)} + \gamma V_k^{\pi_i}(s') \right), \forall s$$

### • 策略改进

$$\pi_{i+1}(s) = \arg \max_a \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V^{\pi_i}(s')) , \forall s$$

## • 基本思想

- 不需要得到环境模型 ( $P$  函数和  $R$  函数)
  - 直接根据环境的反馈进行学习

### • 典型算法:

- 时序差分学习：时序差分值学习、Q 学习、Sarsa 等
  - 蒙特卡洛方法

注 时序差分值学习

## • 主要思想

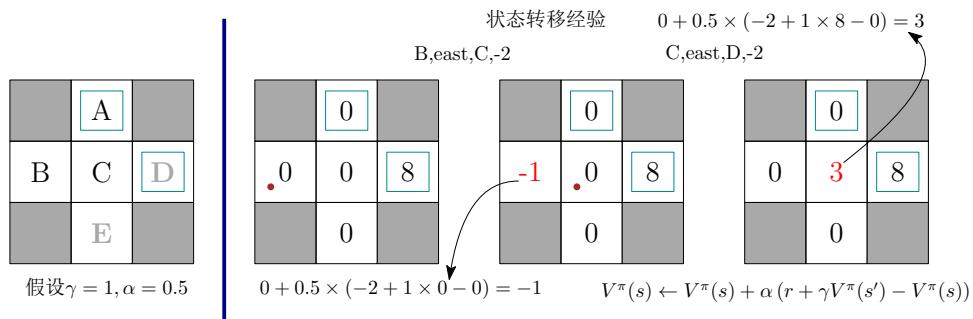
- 基于状态转移经验  $(s, a, s', r)$  更新  $V(s)$

$$V^\pi(s) \leftarrow V^\pi(s) + \alpha (r + \gamma V^\pi(s') - V^\pi(s))$$

更新后的状态值  $\leftarrow$  当前状态值 + 学习率  $(\text{奖励} + \text{折扣因子} \times \text{下一状态值} - \frac{\text{当前状态值}}{\text{实际值} - \text{预测值}})$

- 实际值与预测值之差成为时序差分误差
  - 学习率  $\alpha$  越大，表示采用新结果比例越大
  - 存在问题
    - 只能估计值函数，无法对策略进行优化

**例 5.7 举例：**



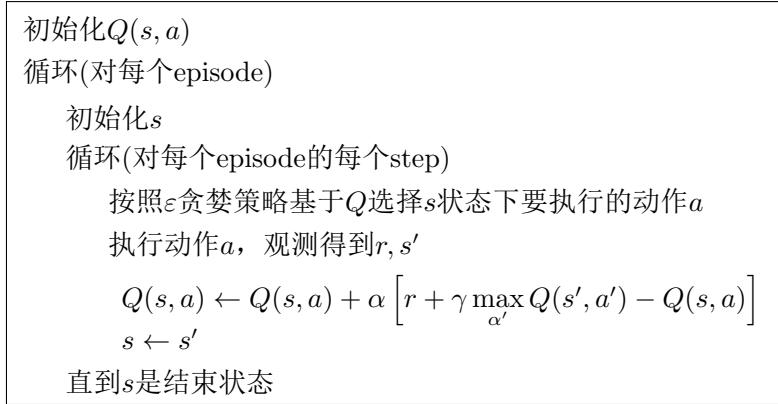
## 注 Q 学习

- 主要思想

- 基于状态转移经验  $(s, a, s', r)$  更新  $Q(s, a)$

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

- 算法流程 当满足以下两个条件是，Q 学习算法能在时间趋于无穷时得到最优策略

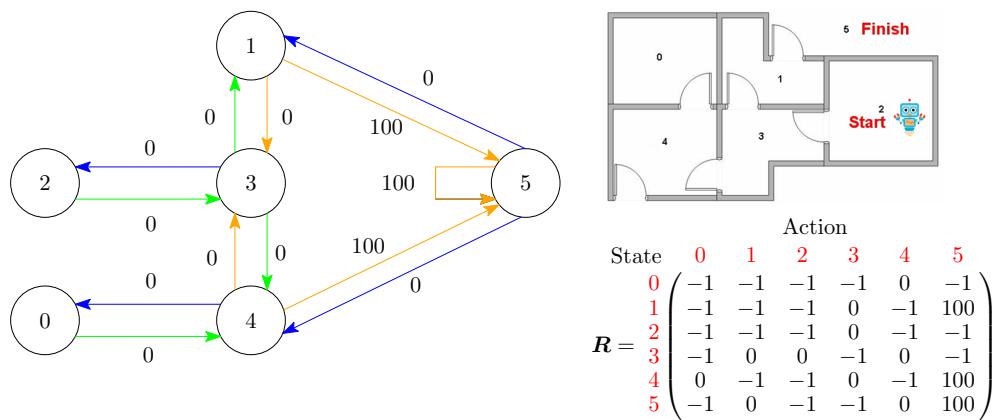


(1)

$$\sum_{t=0}^{\infty} \alpha_t^2 < \infty, \sum_{t=0}^{\infty} \alpha_t < \infty$$

(2) 所有的状态和动作都能够被无限次遍历。

**例 5.8** Q 学习：Q 学习将 State 与 Action 构建成一张 Q-table 来存储 Q 值，然后根据 Q 值来选取能够获得最大的收益的动作。



- -1 表示状态之间是不互通
- 0 表示状态之间互通，奖励为 0
- 100 表示状态之间互通，奖励为 100

$$Q = \begin{matrix} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left( \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \right) \end{matrix}$$

设  $\gamma = 0.8, \alpha = 1$

- 第一局：随机性初始状态 1

- 第一步：在 R-table 的状态 1 下一跳能到达状态 3、5
- 第二步：随机选择下一跳，假设为状态 5；
- 计算 Q-table，更新，因为 5 是目标状态，所以完成了一轮

$$Q(1, 5) = 0 + 1 \times (R(1, 5) + 0.8 \times \max\{Q(5, 1), Q(5, 4), Q(5, 5)\} - 0) = 100$$

$$Q = \begin{matrix} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left( \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \right) \end{matrix}$$

第二局：随机选择初始状态 3

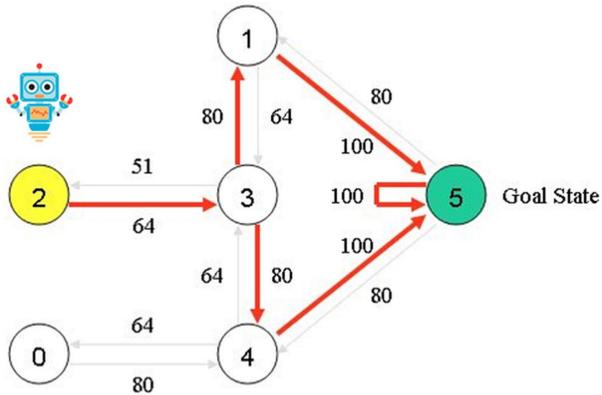
- 第一步：在 R-table 的状态 3 下一跳能到达状态 1、2、4；
- 第二步：随机选择下一跳，假设为状态 1；
- 在 R-table 的状态 1 下一跳能到达状态 3、5；
- 第四步：计算 Q-table，更新

$$Q(3, 1) = 0 + 1 \times (R(3, 1) + 0.8 \times \max\{Q(1, 3), Q(1, 5)\} - 0) = 80$$

经过多次循环迭代，最终得到 Q-table：

$$Q = \begin{matrix} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left( \begin{matrix} 0 & 0 & 0 & 0 & 80 & 0 \\ 0 & 0 & 0 & 64 & 0 & 100 \\ 0 & 0 & 0 & 64 & 0 & 0 \\ 0 & 80 & 51 & 0 & 80 & 0 \\ 64 & 0 & 0 & 64 & 0 & 100 \\ 0 & 80 & 0 & 0 & 80 & 100 \end{matrix} \right) \end{matrix}$$

从状态 2 开始，智能体使用矩阵 Q 计算动作



**注** 大多实际问题中状态集是庞大的无法采用矩阵形式表达，比如王者荣耀和围棋的状态空间，无法得到 P 和 R 函数。

**定义 5.3 (蒙特卡罗方法)** 蒙特卡洛方法是一种无模型的方法，不需要事先知道马尔可夫决策过程的状态转移概率以及奖励，通过随机采样找到近似解，采样越多、累积奖励的平均值越接近真实的值函数。

通过与环境交互，从所采集的样本中学习，获得关于决策过程的状态、动作和奖赏的大量数据（经验），最后计算出累积奖励的平均值。

**注** 从初始状态开始，采用某种策略进行采样，执行该策略 T 步，获得一条轨迹：

$$\langle s_0, a_0, r_1, s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T, s_T \rangle$$

对轨迹中每一个状态-动作对，记录其的累积奖励，作为一次奖励的采样值。多次执行策略得到多条轨迹后，将每个状态-动作对的累积奖励值进行平均，得到状态-动作值函数的估计。

## 5.3 深度强化学习

### 5.3.1 深度 Q 学习

**注** Atari 游戏

- 维度灾难

计算机玩 Atari 游戏的输入为 210x160 像素的原始图像数据，从理论上看，图像中每一个像素都有 RGB 三个通道，每个通道有 256 种值，总的状态空间为：

$$(3 \times 256)^{210 \times 160}$$

- 灾难性遗忘

在玩视频游戏中，人工神经网络会遭遇灾难性遗忘问题。当神经网络学会玩一种新的游戏时，原来学会的游戏就会忘掉。

**定义 5.4 (DQN)** DQN 思想：Q 表被深度神经网络（Q 网络）取代，后者可以把环境状态映射为智能体动作（非线性逼近）

**注** 深度强化学习需要解决的 2 个问题：

- 深度学习需要大量有标签的数据样本；而强化学习是智能体主动获取样本，样本量稀疏且奖励有延迟。
- 深度学习要求每个样本相互之间是独立同分布的；而强化学习获取的相邻样本相互关联性较大，并不是相互独立的。

**注** 解决问题的关键技术：

- 样本池：将采集到的样本先放入样本池，然后从样本池中随机选出一条样本用于对网络的训练。这种处理打破了样本间的关联，使样本间相互独立。
- 固定目标值网络：计算网络目标值需用到现有的 Q 值，用一个更新较慢的网络专门提供此 Q 值，提高了训练的稳定性和收敛性。

**注** DQN 优缺点：

- 优点

- 算法普适性较强，相同的网络可以学习不同的 Atari 游戏；
- 采用端到端的训练方式，无需人工提取特征，采用 CNN 进行特征提取；
- 通过不断的训练学习，使用奖励构造标签，生成大量样本用于监督学习；
- 通过经验回放方法来解决样本相关性及非静态分布问题，即建立一个经验池，把每次的经验都存起来，训练时随机采样一批样本训练。

- 缺点

- 只适用于处理短时记忆任务，无法处理需要长时间经验的任务
- 网络输出是有限离散 Q 值，对应离散的动作，不能处理连续值