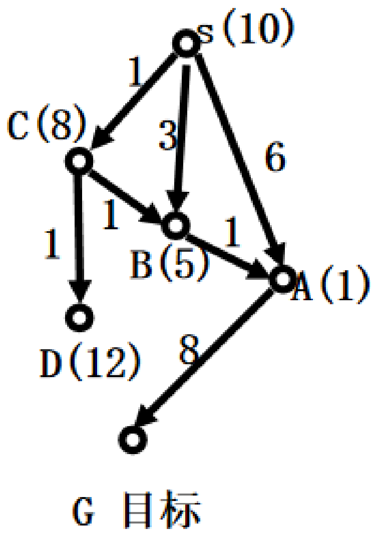
1 Ch01

- A 算法
  - OPEN=(s), CLOSED=()
  - while OPEN 不空，取其中第一个结点 n，如果是目标就返回，不然把 n 移动到 CLOSED 表，接着考虑其所有的子节点，计算 $f(n, m_i) = g(n, m_i) + h(m_i)$，然后：
    - 如果是 m_j 类型的结点，那么标记 m_j 到 n 的指针，加入 OPEN 表
    - 如果是 m_k 类型的结点且当前值更小，那么更新 $f(m_k)$ 与相应指针
    - 如果是 m_l 类型结点且当前值更小，那么更新 $f(m_l)$ 与相应指针，然后加入 OPEN 表
- A* 算法：满足条件 $h(n) \le h^\star(n)$ 的 A 算法
  - 若存在从初始节点s到目标节点t有路径，则A*必能找到最佳解结束。
  - 如果h2(n) > h1(n) (目标节点除外)，则A1扩展的节点数⩾A2扩展的节点数
- A* 算法的改进
  - 对 h 加以限制（称 h 是单调的）
    - $h(n_i) - h(n_j) \le c(n_i, n_j)$, $h(t) = 0$
    - 结论是当扩展到结点 n 时就有 $g(n) = g^\star(n)$
    - 满足单调条件的一定满足 A* 条件
  - 对算法加以改进
    - 记 f_m 为到目前为止已扩展结点的最大 f 值，以 f_m 为分界
    - 在每个迭代步的最开始，构造 NEST 表为 OPEN 表中所有满足 $f(n_i) < f_m$ 的 n_i
    - 如果 NEST 不空，取 NEST 中 g 最小的结点；不然取 OPEN 中 f 最小的结点

例题：A*算法，改进的 A* 算法，s 赋值成 9 再做

前面的例子：



OPEN表	CLOSED表	f_m
s(0+10)	s(0+10)	10
A(6+1) B(3+5) C(1+8)	s(0+10) C(1+8)	10
A(6+1)B(2+5)D(2+12)	s(0+10) C(1+8) B(2+5)	10
A(3+1) D(2+12)	s(0+10)C(1+8)B(2+5)A(3+1)	10
G(11+0) D(2+12)		

说明：蓝颜色表示在nest中的节点

2 Ch02

2.1 BP

把样例输入网络，计算每个单元u的输出o_u

- 对于输出层单元k，计算误差项： $\delta_k = (t_k - o_k) o_k (1 - o_k)$
- 对于隐含层单元h，计算误差项： $\delta_h = o_h (1 - o_h) \sum_{k \in 后继(h)} \delta_k w_{kh}$
- 更新每个权值： $w_{ji} = w_{ji} + \Delta w_{ji}$
- 其中： $\Delta w_{ji} = \eta \delta_j x_{ji}$

交叉熵损失函数

$$\mathrm{H}_d(w) = -\sum_{k=1}^{M} t_{kd} \log(o_{kd})$$

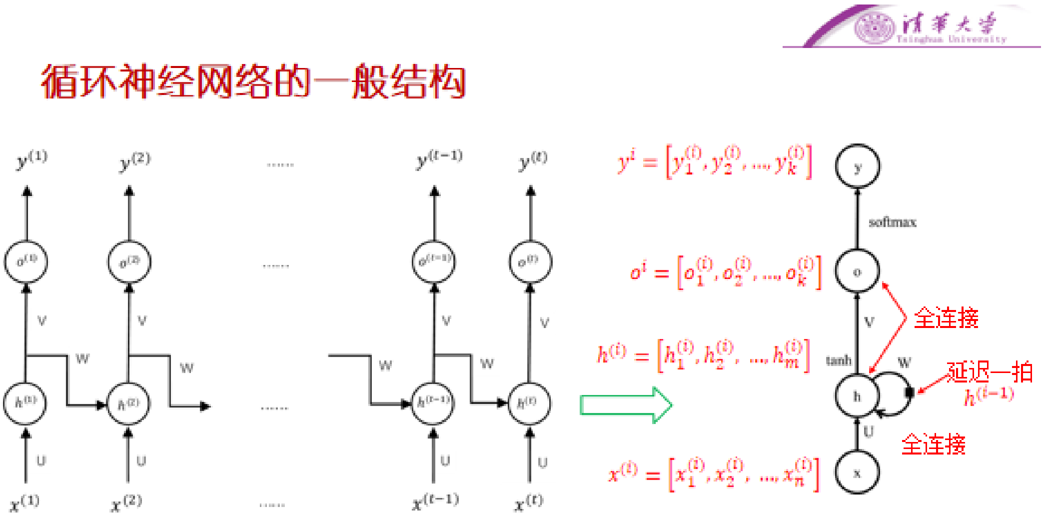
$$H(w) = \sum_{d=1}^{N} \mathrm{H}_d(\mathrm{w}) = -\sum_{d=1}^{N} \sum_{k=1}^{M} t_{kd} \log(o_{kd})$$

t_{kd}： 样本d对应的希望输出值

o_{kd}： 样本d对应的实际输出值，要求o_{kd}是个概率值

2.2 Models

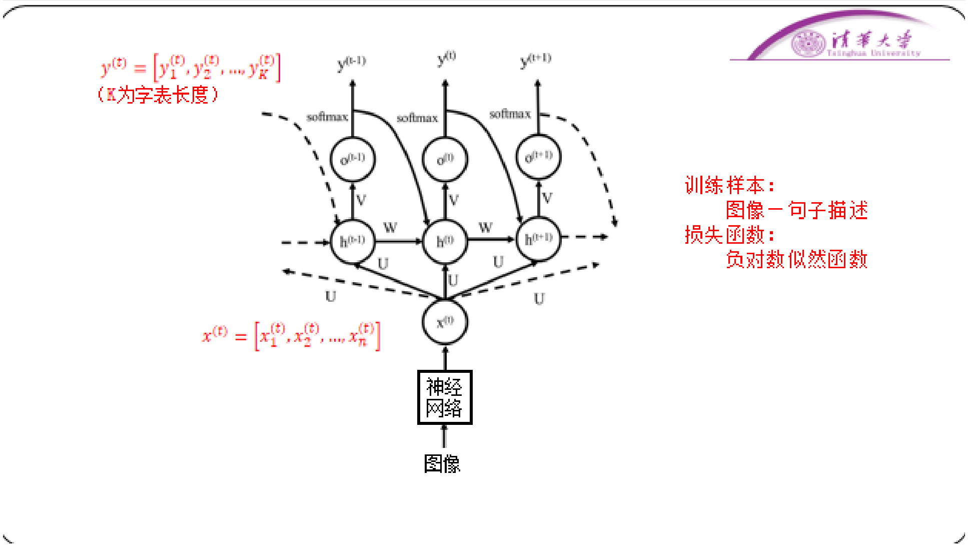
- TextCNN
- RNN



– 应用例子

  ◦ 中文分词

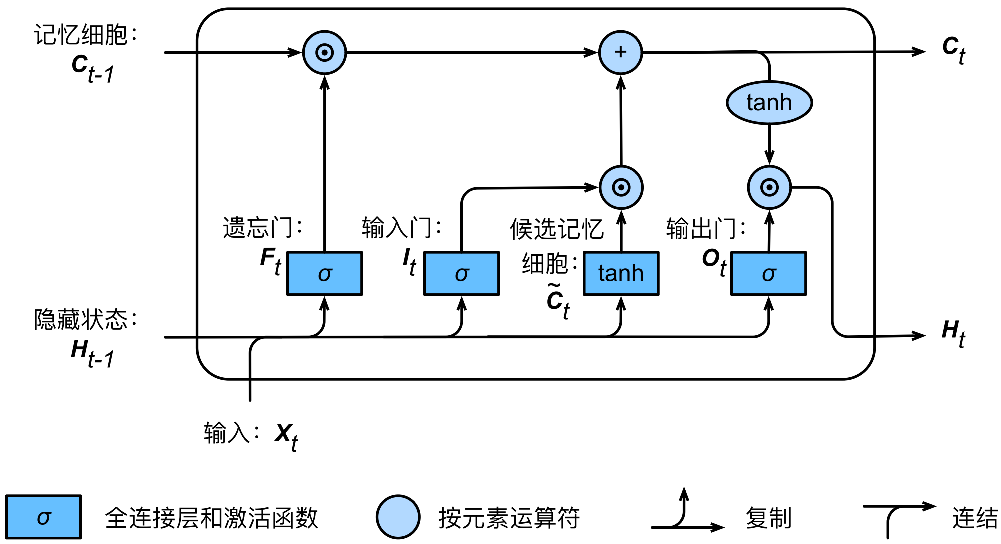
  ◦ 看图说话

  ◦ 

– 双向循环神经网络

– Seq2seq Encoder Decoder

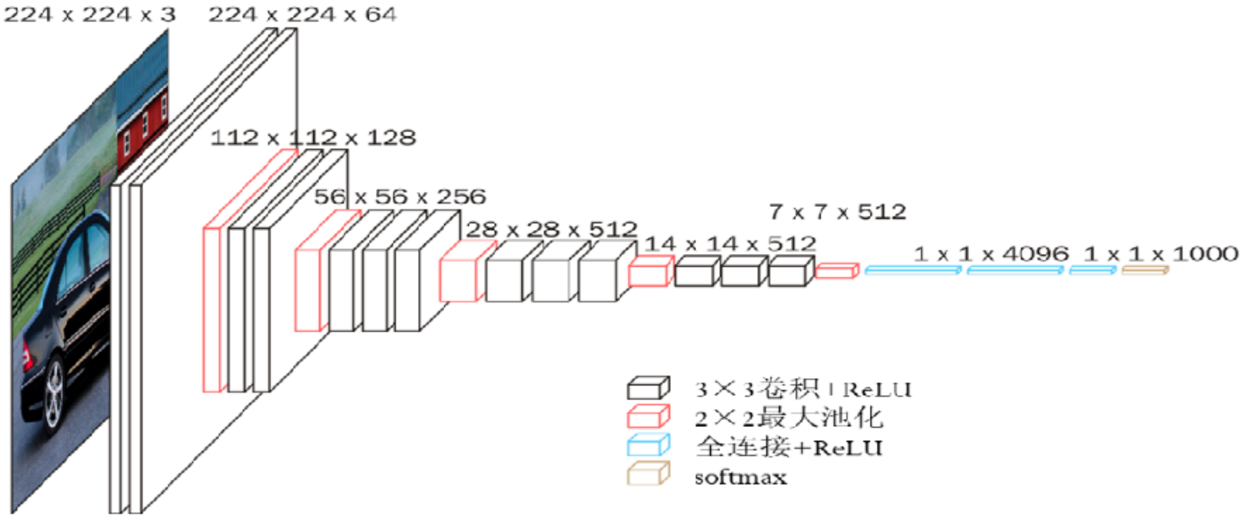
– LSTM

  ◦ 

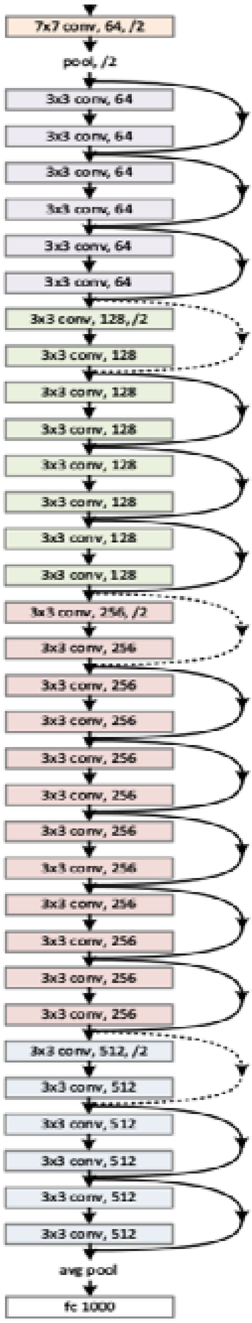
$$
\begin{aligned}
\boldsymbol{I}_t &= \sigma(\boldsymbol{W}_{xi}\boldsymbol{X}_t + \boldsymbol{W}_{hi}\boldsymbol{H}_{t-1} + \boldsymbol{b}_i),\\
\boldsymbol{F}_t &= \sigma(\boldsymbol{W}_{xf}\boldsymbol{X}_t + \boldsymbol{W}_{hf}\boldsymbol{H}_{t-1} + \boldsymbol{b}_f),\\
\boldsymbol{O}_t &= \sigma(\boldsymbol{W}_{xo}\boldsymbol{X}_t + \boldsymbol{W}_{ho}\boldsymbol{H}_{t-1} + \boldsymbol{b}_o),\\
\tilde{\boldsymbol{C}}_t &= \tanh(\boldsymbol{W}_{xc}\boldsymbol{X}_t + \boldsymbol{W}_{hc}\boldsymbol{H}_{t-1} + \boldsymbol{b}_c),\\
\boldsymbol{C}_t &= \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + \boldsymbol{I}_t \odot \tilde{\boldsymbol{C}}_t,\\
\boldsymbol{H}_t &= \boldsymbol{O}_t \odot \tanh(\boldsymbol{C}_t).
\end{aligned}
$$

2.3 Example Models

```python
LeNet = nn.Sequential(
    nn.Conv2d(in_channels=1, out_channels=6, kernel_size=5),
    nn.MaxPool2d(kernel_size=2, stride=2),
    nn.Conv2d(in_channels=6, out_channels=16, kernel_size=5),
    nn.MaxPool2d(kernel_size=2, stride=2),
    nn.Flatten(),
    nn.Linear(400, 120),
    nn.Linear(120, 84),
    nn.Linear(84, 10)
)
```



- GoogLeNet
- ResNet

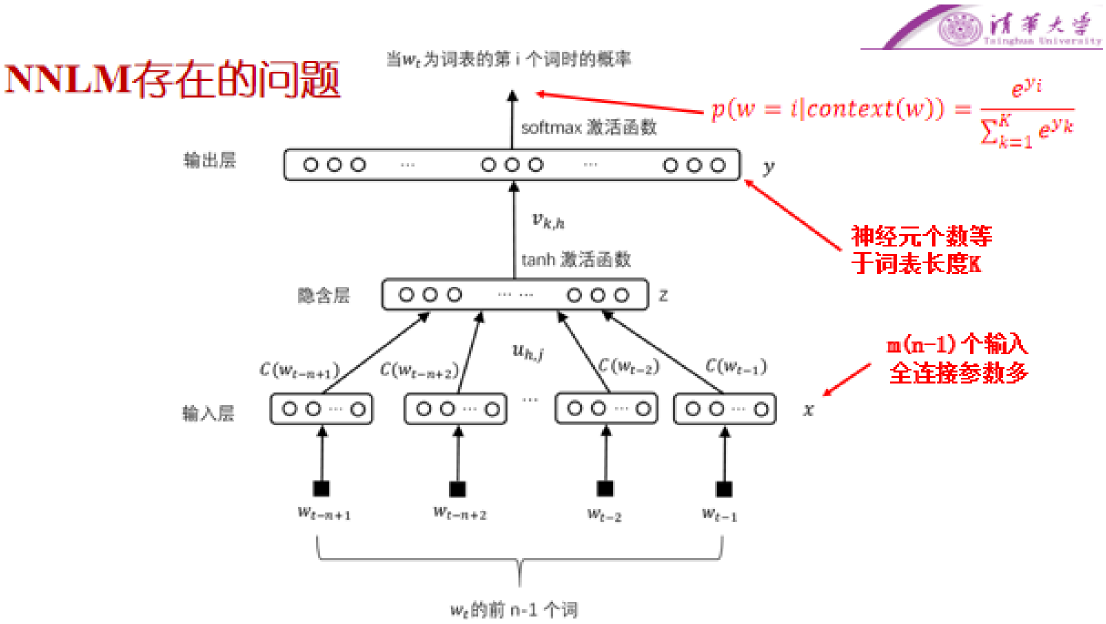


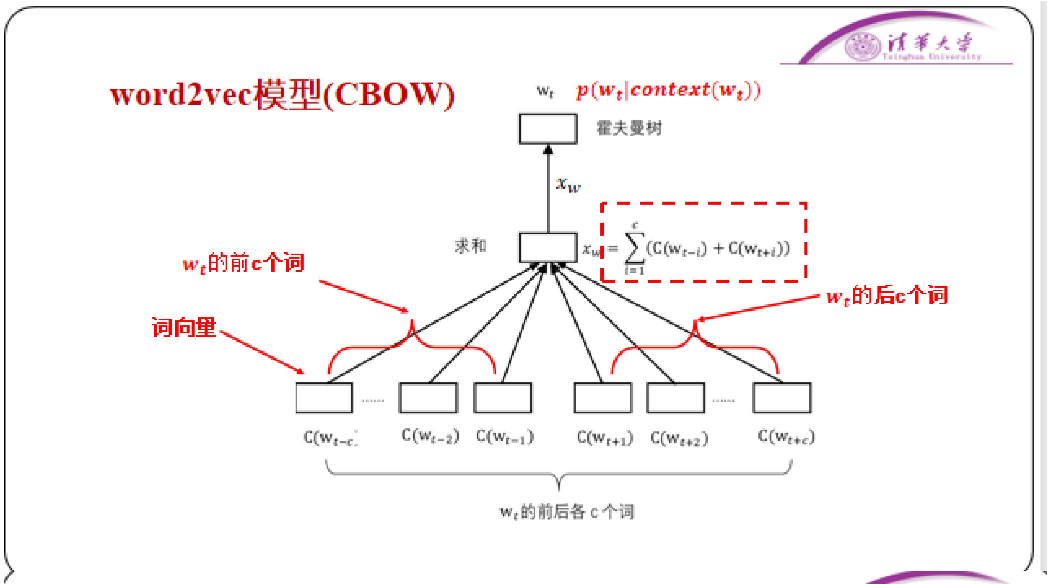
2.4 机器学习中出现的问题

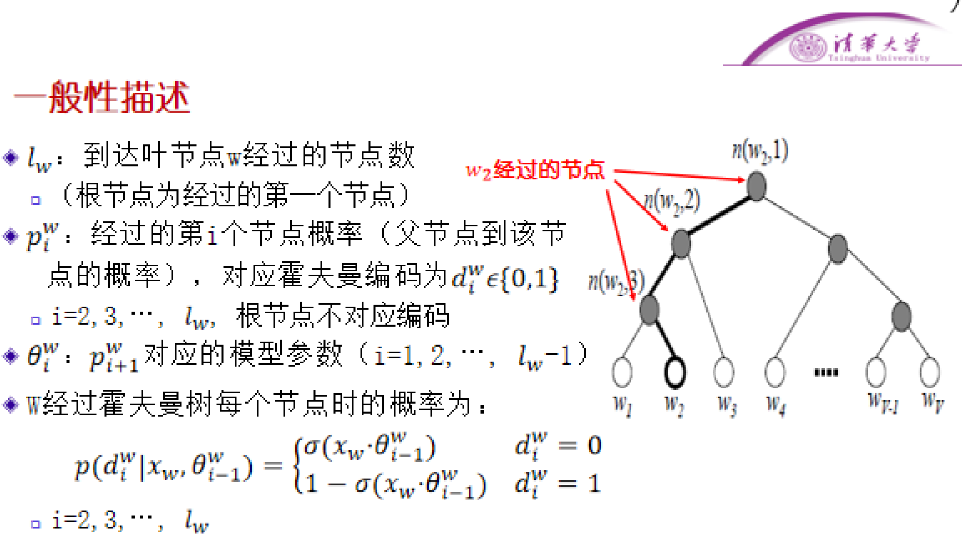
- 梯度消失问题
  - 使用 ReLU 激活函数，多输出，设计残差模块
- 过拟合：减少过拟合的方法
  - 正则项（L1 Loss 与 L2 Loss 的效果）、Dropout，数据增强

2.5 语言模型

- NNLM
  - 词向量的训练：对输入也回传梯度



- Word2vec
  - CBOW
    - 

    - 

单击此处添加标题

- 词w的最大似然函数：

$$\prod_{i=2}^{l_w} p(d_i^w|x_w,\theta_{i-1}^w) = \prod_{i=2}^{l_w}[\sigma(x_w\cdot\theta_{i-1}^w)]^{1-d_i^w}\,[1-\sigma(x_w\cdot\theta_{i-1}^w)]^{d_i^w}$$

- 定义损失函数（负对数似然函数）：

$$\begin{aligned} L &= -\,log \prod_{i=2}^{l_w} p(d_i^w|x_w,\theta_{i-1}^w) \\ &= -\sum_{i=2}^{l_w}\{(1-d_i^w)\,log\,[\sigma(x_w\cdot\theta_{i-1}^w)] + d_i^w\,log[1-\sigma(x_w\cdot\theta_{i-1}^w)]\} \end{aligned}$$

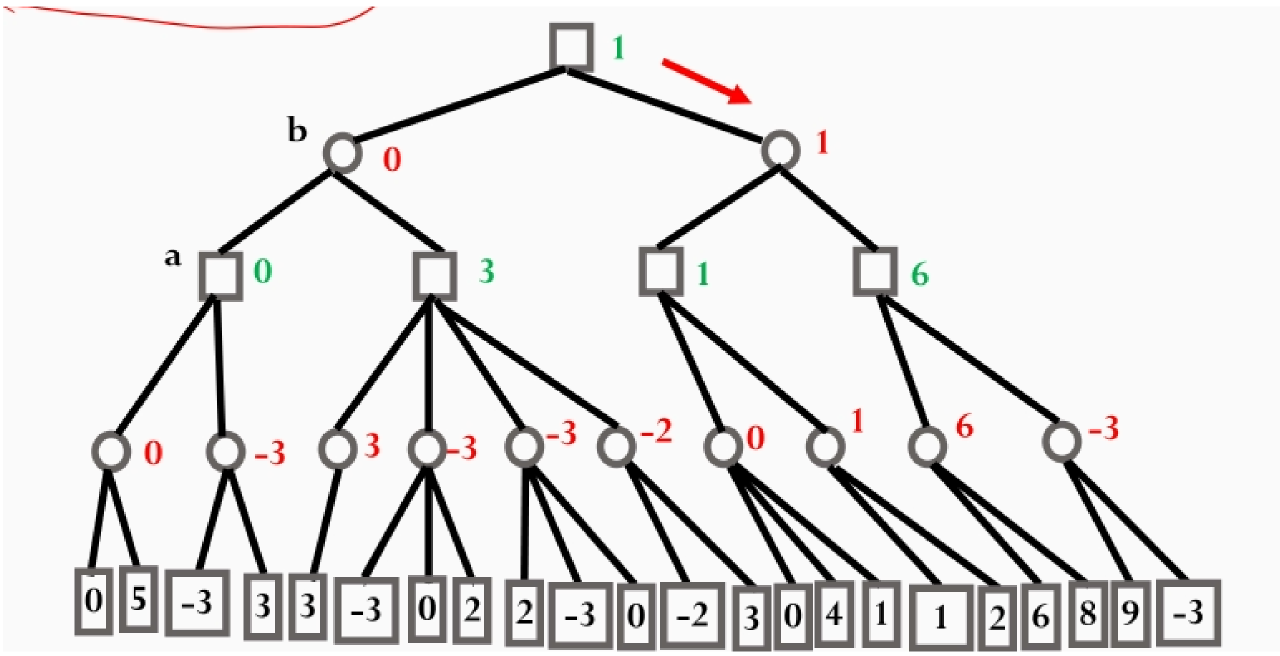
– Skip-Gram Model

3 Ch03

博弈问题：双人，一人一步，双方信息完备，零和

3.1 Minimax / alpha-beta

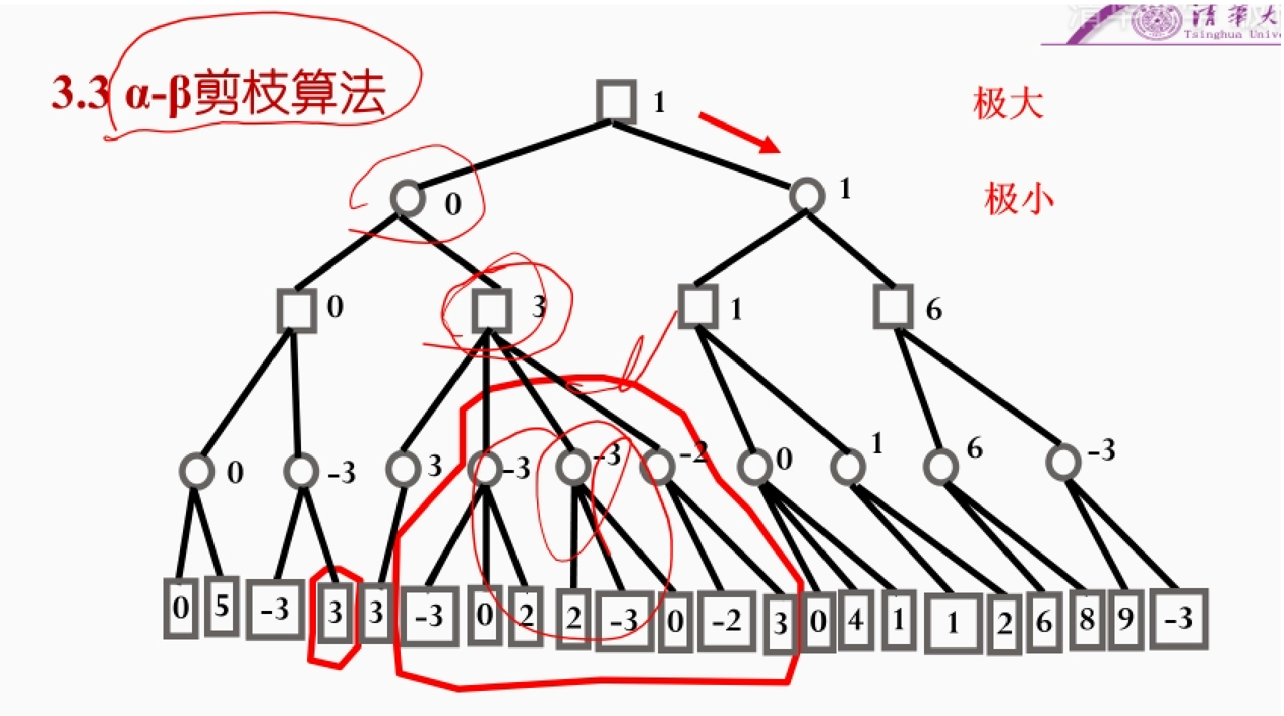
极大极小模型：



alpha-beta 剪枝：

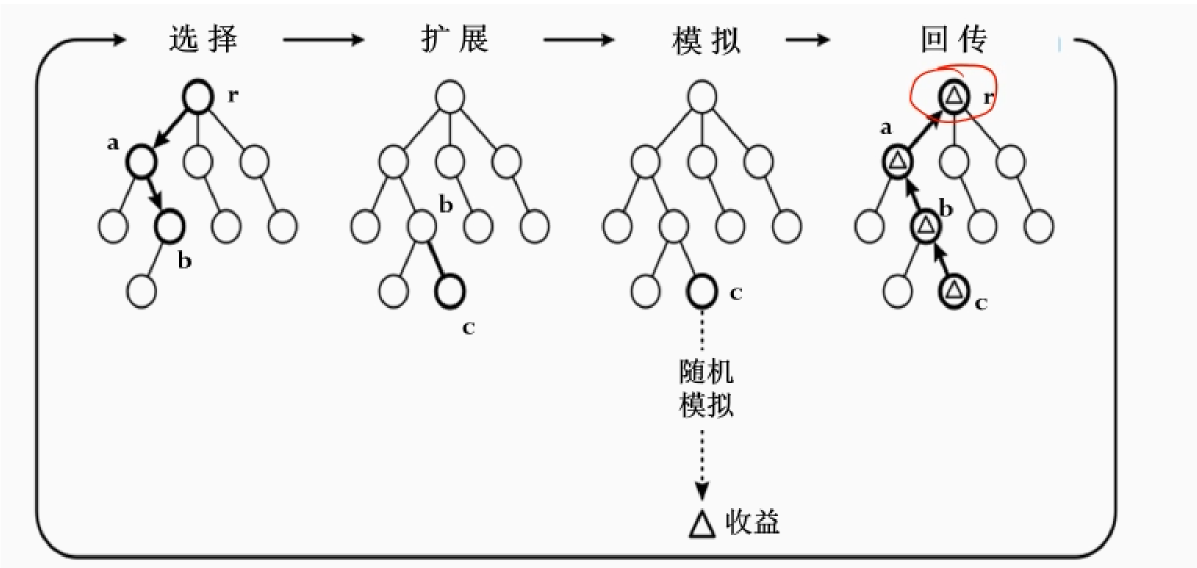
α-β剪枝算法

- 极大节点的下界为α。
- 极小节点的上界为β。
- 剪枝的条件：
  - 后辈节点的β值≤祖先节点的α值时， α剪枝
  - 后辈节点的α 值≥祖先节点的β值时， β剪枝
- 简记为：
  - 极小≤极大，剪枝
  - 极大≥极小，剪枝



```
procedure search(node)
    while (n has children to extend) do
        if branch cut is possible
            cut branches
            break
        endif
        search(first child node not extended)
    endwhile
    update parent node's value
```

3.2 蒙特卡洛方法

- 选择（选第一个有子节点没扩展的节点）
  - 对尚未充分了解的节点的探索
  - 对当前具有较大希望节点的利用
- 扩展（生成子节点）
- 模拟（获得收益）
- 回传（胜负交替）
- 

选择：

信心上限算法（UCB：Upper Confidence Bound）

```
function UCB1
    for each 拉杆j:
            访问该拉杆并记录收益
        end for
        while 尚未达到访问次数限制 do:
            计算每个拉杆的UCB1信心上界Ij
            访问信心上界最大的手臂
        end while
```

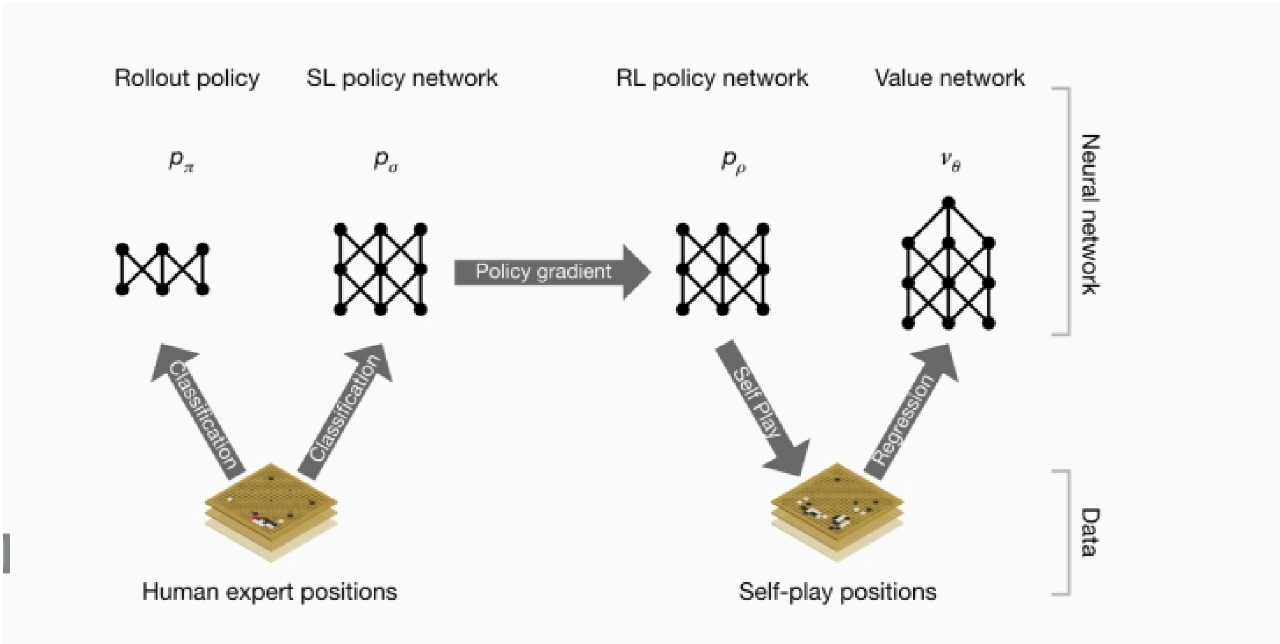
信心上限的计算

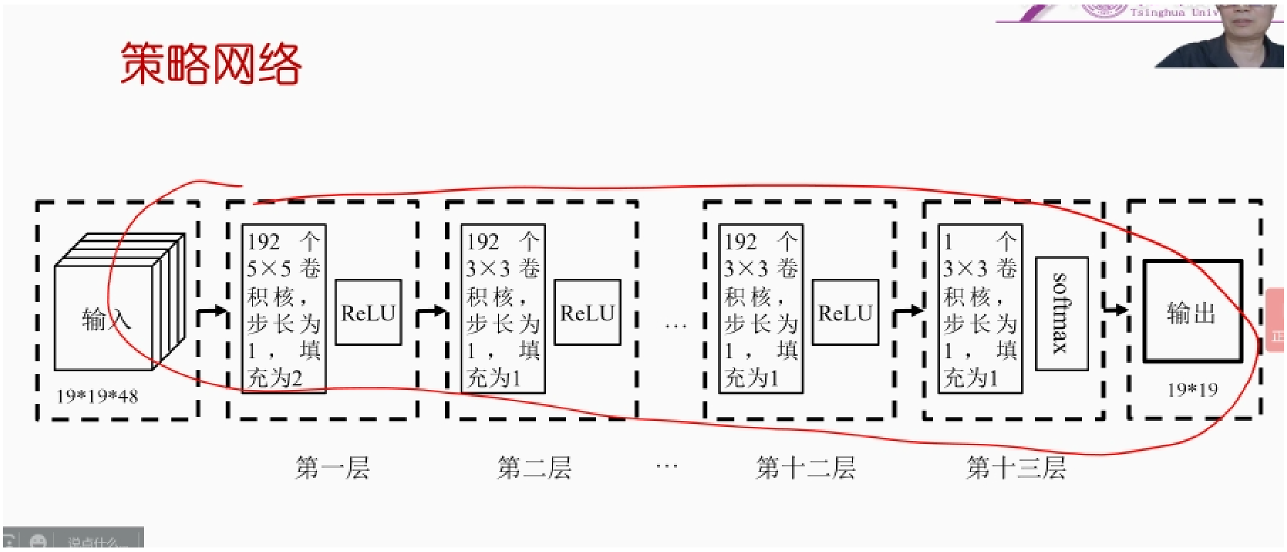
$$I_j = \bar{X}_j + \sqrt{\frac{2\ln(n)}{T_j(n)}}$$

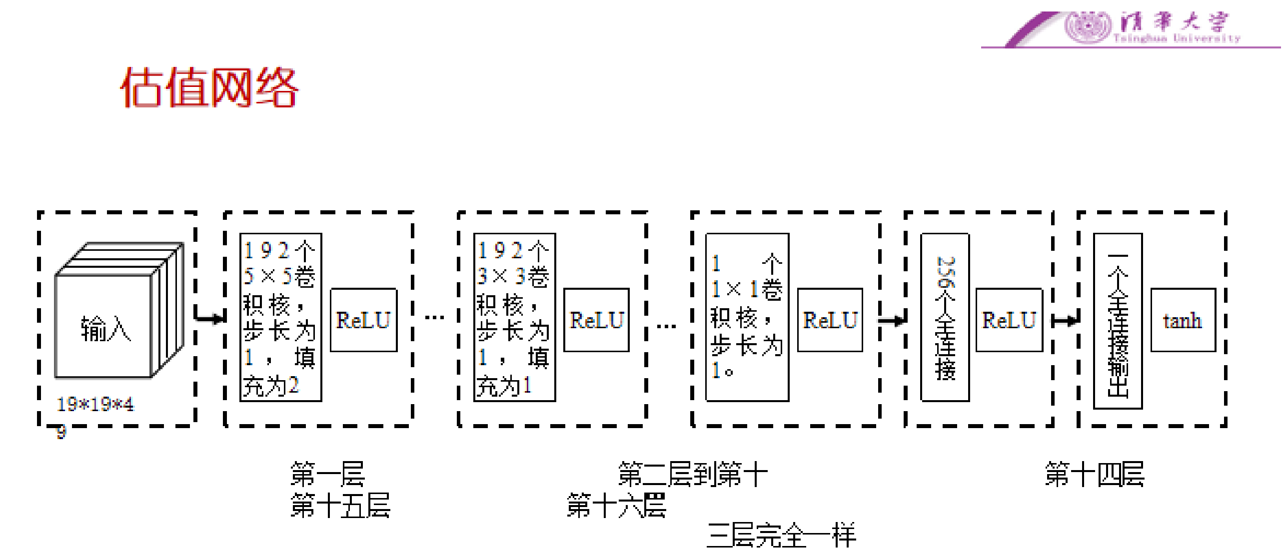
- 其中：
- $\bar{X}_j$ 是拉杆j所获得回报的均值
- n是到当前这一时刻为止所访问的总次数
- $T_j(n)$ 是拉杆j到目前为止所访问的次数
- 上式考虑了“利用”和“探索”间的平衡

3.3 AlphaGo

策略网络（输入当前棋局，输出在某个点的落子概率）、估值网络（输入当前棋局，输出估计的收益 [-1, 1]）

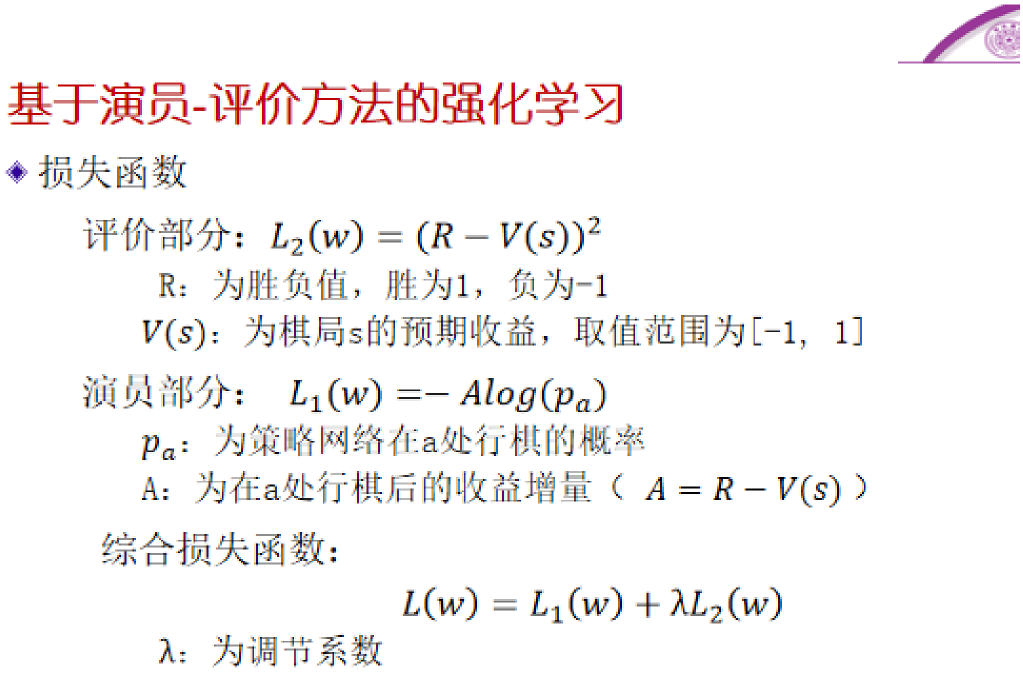

Rollout policy
SL policy network
RL policy network
Value network
p_π
p_σ
p_ρ
v_θ
Policy gradient
Neural network
Classification
Classification
Self Play
Regression
Data
Human expert positions
Self-play positions


策略网络
输入
19*19*48
192 个 5×5 卷积核，步长为1，填充为2
ReLU
192 个 3×3 卷积核，步长为1，填充为1
ReLU
…
192 个 3×3 卷积核，步长为1，填充为1
ReLU
1 个 3×3 卷积核，步长为1，填充为1
softmax
输出
19*19
第一层
第二层
…
第十二层
第十三层


估值网络
输入
19*19*49
192个5×5卷积核，步长为1，填充为2
ReLU
…
192个3×3卷积核，步长为1，填充为1
ReLU
…
1 个1×1卷积核，步长为1。
ReLU
256个全连接
ReLU
一个全连接输出
tanh
第一层
第二层到第十三层完全一样
第十四层
第十五层
第十六层

3.4 RL

- 基于策略梯度的 RL
- 基于价值评估的 RL
- 演员-评价方法



3.5 AlphaGo Zero

- 修改模拟阶段的收益 => 仅采用估值网络
- 引入多样性：对策略网络的输出增加噪声

4 Ch04

4.1 SVM

4.1.1 线性可分支持向量机

线性可分支持向量机（二分类）、函数间隔（超平面关于样本点的，超平面关于训练集 T 的）、几何间隔（欧氏距离）、求解的转化、学习的对偶算法

KKT条件：

$$\nabla_{w,b} L(w, b, \alpha) = 0$$

$$\alpha_i [1 - y_i(w \cdot x_i + b)] = 0$$

$$[1 - y_i(w \cdot x_i + b)] \leq 0$$

$$\alpha_i \geq 0$$

$$i = 1,2, \ldots, N$$

- 目标函数由求极大转换成求极小，得到等价的对偶问题：

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N} \alpha_i$$

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

$$\alpha_i \geq 0, i = 1,2,\ldots, N$$

如何求得w*、b*?

KKT条件(部分):

$$\nabla_{w,b} L(w, b, \alpha) = 0$$

$$\alpha_i[1 - y_i(w \cdot x_i + b)] = 0$$

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

$$\Rightarrow w^* - \sum_{i=1}^{N} \alpha_i^* y_i x_i = 0$$

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

$$\alpha_i[1 - y_i(w \cdot x_i + b)] = 0, i = 1...N$$

$$\Rightarrow b^* = y_j - w \cdot x_j, \text{选择一个}\alpha_j \neq 0$$

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

支持向量的数目

4.1.2 线性支持向量机

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i$$

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

$$0 \le \alpha_i \le \mathrm{C}, i = 1,2,\ldots,N$$

求得最优解 $\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)^T$

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

选择一个 $0 < \alpha_j^* < C$，计算：

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

$\alpha_i^* > 0$ 所对应的样本 x_i 称为支持向量（软间隔的支持向量）

若 $\alpha_i^* < C$，则 $\xi_i = 0$，x_i 在间隔边界上

若 $\alpha_i^* = C$，$0 < \xi_i < 1$，则分类正确，x_i 在间隔边界与分离超平面之间

若 $\alpha_i^* = C, \xi_i = 1$，则 x_i 在超平面上

若 $\alpha_i^* = C, \xi_i > 1$，则 x_i 位于误分一侧

4.1.3 非线性支持向量机

用某个变换将原空间数据映射到新空间，在新空间用线性分类方法学习。

核函数，用核函数 K 找映射 ϕ，

4.1.4 应用

tf-idf 的算法？ i 是词项，j 是文档

4.2 决策树

- 信息增益的计算公式
- 生成决策树的算法
  - ID3
  - C4.5

- 设训练集D，K个类C_k，特征A有n个不同的取值$\{a_i,\ldots,a_n\}$，A的不同取值将D划分为n个子集$D_1\ldots D_n$，D_i中属于类C_k的样本的集合为D_{ik}，$|\cdot|$表示样本个数。
- 信息增益计算如下：

$H(D)=-\sum p_i \log p_i$

$$H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}\log_2\frac{|C_k|}{|D|}$$

$$H(D\mid A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|}$$

$$g(D,A)=H(D)-H(D\mid A)$$

4.2.1 ID3

输入：训练集D，特征集A，阈值ε>0

输出：决策树T

1，若D中所有实例属于同一类Ck，则T为单节点树，将Ck作为该节点的类标记，返回T

2，若A为空，则T为单节点树，将D中实例数最大的类Ck作为该节点的类标记，返回T

3，否则计算A中各特征对D的信息增益，选择信息最大的特征Ag

4，如果Ag的信息增益小于阈值ε，则置T为单节点树，将D中实例数最大的类Ck作为该节点的类标记，返回T

5，否则对Ag的每一可能值ai，依Ag=ai将D分割为若干子集Di，作为D的子节点

6，对于D的每个子节点Di，如果Di为空，则将D中实例最大的类作为标记，构建子节点

7，否则以Di为训练集，以A-{Ag}为特征集，递归地调用步1~步6，得到子树Ti，返回Ti

ID	年龄 A1	有工作 A2	有房子 A3	信贷情况 A4	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

$$H(D) = -\frac{9}{15}\log_2\frac{9}{15} - \frac{6}{15}\log_2\frac{6}{15} = 0.971$$

$$g(D, A_1) = H(D) - \left[\frac{5}{15}H(D_1) + \frac{5}{15}H(D_2) + \frac{5}{15}H(D_3)\right]$$

$$= 0.971 - \frac{5}{15}\left[\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) + \left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) + \left(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}\right)\right] = 0.083$$

$g(D, A_2) = 0.324$

$g(D, A_3) = 0.420$　该信息增益最大

$g(D, A_4) = 0.363$

A_3作为根节点，将D划分为$D_1(A_3 =$是$)$和

$D_2(A_3 =$否$)$, D_1成为叶结点

对D_2从特征A_1、A_2、A_4中选择特征

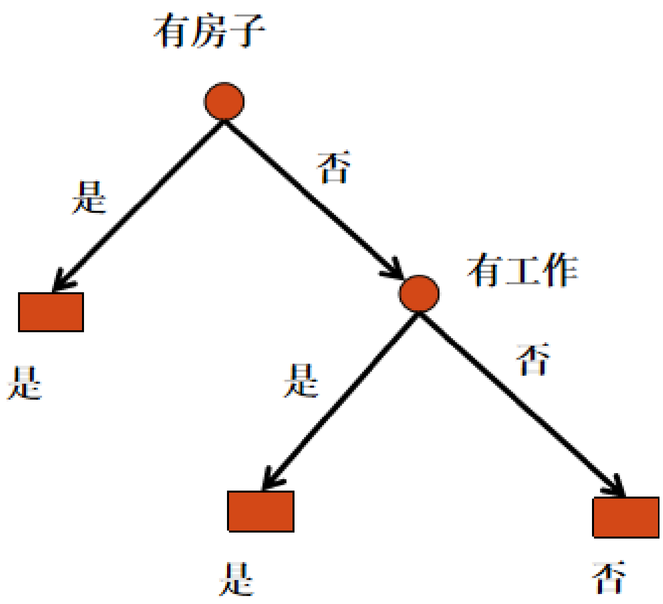
$g(D_2, A_1) = 0.251$

$g(D_2, A_2) = 0.918$　信息增益最大

$g(D_2, A_4) = 0.474$

选取A_2作为节点的特征，根据其两个取值，可以得到两个子节点，一个对应“有工作”，并且是一个叶节点，标记类别为“是”。另一个节点对应“无工作”，且样本属于同一类，也是一个叶节点，标记类别为“否”

◆生成的决策树如下：



4.2.2 C4.5

信息增益比

$$g_R(D,A)=\frac{g(D,A)}{H_A(D)}$$

$$H_A(D)=-\sum_{k=1}^{n}\frac{|D_k|}{|D|}\log_2\frac{|D_k|}{|D|}$$

◆其中A为属性，A的不同取值将D划分为n个子集$D_1 \cdots D_n$