《数据挖掘技术》

⭐ CH10 隐马尔可夫模型

➡️ Create by *Wang JingHui*

➡️ Last Revision Time: *2021.04.02*

主要内容

1. 隐马尔可夫模型的基本概念
2. 概率计算方法
3. 学习算法
4. 预测算法

重点

HMM有两个基本假设和三个基本问题。

隐马尔可夫模型的基本概念

随机变量与随机过程

> 19世纪, 概率论的发展从对(相对静态的)**随机变量**的研究发展到对随机变量的时间序列$s_1, s_2, s_3, ..., s_t, ...$,即**随机过程**(动态的)的研究

马尔可夫链

随机过程有两个维度的不确定性. 马尔可夫为了简化问题, 提出了一种简化的假设, 即随机过程中各个状态s_t的概率分布, 只与它的前一个状态s_{t-1}有关, 即 $P(s_t|s_1,s_2,s_3,...,s_{t-1}) = P(s_t|s_{t-1})$

> 这个假设后来被称为**马尔可夫假设**,而符合这个假设的随机过程则称为**马尔可夫过程**, 也称为**马尔可夫链**.

$$P(s_t|s_1,s_2,s_3,...,s_{t-1}) = P(s_t|s_{t-1})$$

时间和状态取值都是离散的马尔可夫过程也称为马尔可夫链.

隐含马尔可夫模型

$$P(s_1, s_2, s_3, \ldots, o_1, o_2, o_3, \ldots) = \prod_t P(s_t|s_{t-1}) \cdot P(o_t|s_t)$$

隐含的是**状态**s

隐含马尔可夫模型由**初始概率分布**(向量π), **状态转移概率分布**(矩阵A)以及**观测概率分布**(矩阵B)确定.

隐马尔可夫模型λ 可以用三元符号表示, 即

$$\lambda = (A, B, \pi)$$

其中A, B, π称为模型三要素.

两个基本假设

1. 齐次马尔科夫假设(**状态**)

$$P(i_t|i_{t-1},o_{t-1},\ldots,i_1,o_1)=P(i_t|i_{t-1}),t=1,2,\ldots,T$$

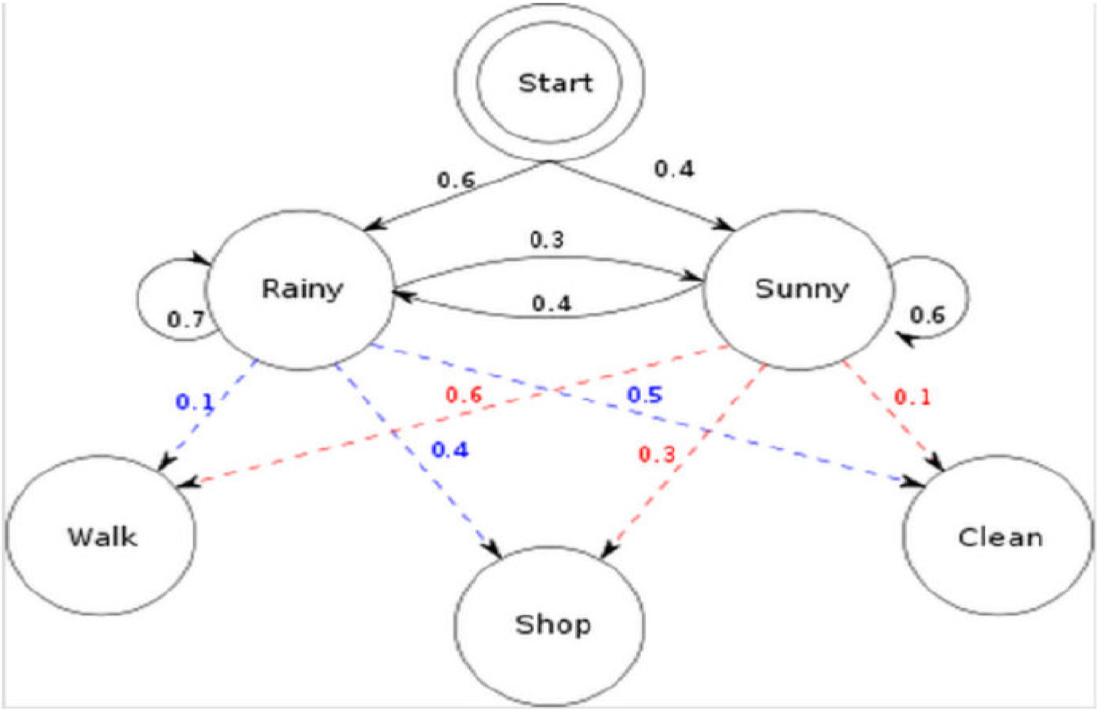
> 假设隐藏的马尔可夫链在**任意时刻t的状态**$\rightarrow i_t$
> 只依赖于其前一时刻的状态$\rightarrow i_{t-1}$
> 与其他时刻的状态 $\rightarrow i_{t-2},\ldots,i_1$
> 及观测无关 $\rightarrow o_{t-1},\ldots,o_1$
> 也与时刻t无关 $\rightarrow t=1,2,\ldots,T$

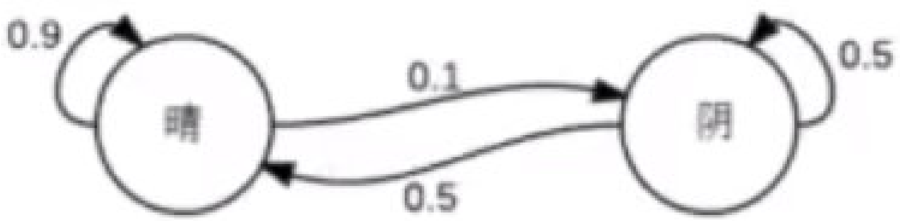
如此烦绕的一句话, 用一个公式就表示了, 数学是如此美妙.

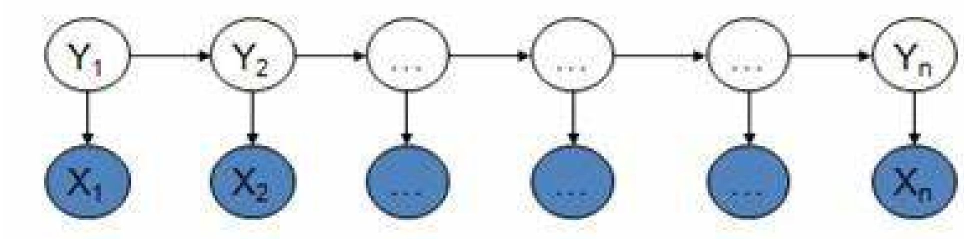
2. 观测独立性假设(**观测**)

$$P(o_t|i_T,o_T,i_{T-1},o_{T-1},\ldots,i_{t+1},o_{t+1},i_t,i_{t-1},o_{t-1},\ldots,i_1,o_1)=P(o_t|i_t)$$

> 假设**任意时刻t的观测**$\rightarrow o_t$
> 只依赖于该时刻的马尔可夫链的状态 $\rightarrow i_t$
> 与其他观测 $\rightarrow o_T,o_{T-1},\ldots,o_{t+1},o_{t-1},\ldots,o_1$
>
> 及状态无关 $\rightarrow i_T,i_{T-1},\ldots,i_{t+1},i_{t-1},\ldots,i_1$


Start
0.6
0.4
Rainy
Sunny
0.3
0.4
0.7
0.6
0.1
0.6
0.5
0.1
0.4
0.3
Walk
Shop
Clean


0.9
晴
0.1
0.5
阴
0.5


Y_1
Y_2
...
Y_n
X_1
X_2
...
X_n

三个基本问题

1. 概率计算问题
   输入: 模型$\lambda = (A, B, \pi)$, 观测序列$O = (o_1, o_2, \ldots, o_T)$
   输出: $P(O|\lambda)$

2. 学习问题
   输入: 观测序列 $O = (o_1, o_2, \ldots, o_T)$
   输出: 输出$\lambda = (A, B, \pi)$

3. 预测问题, 也称为解码问题(Decoding)
   输入: 模型$\lambda = (A, B, \pi)$, 观测序列$O = (o_1, o_2, \ldots, o_T)$
   输出: 状态序列 $I = (i_1, i_2, \ldots, i_T)$

   因为状态序列是隐藏的, 不可观测的, 所以叫解码.

观测序列生成算法

> 输入：$\lambda = (A, B, \pi)$,观测序列长度T
>
> 输出：观测序列$O = (o_1, o_2, \ldots, o_T)$
>
> 1. 按照初始状态分布π产生i_1
> 2. $t = 1$
> 3. 按照状态i_t的观测概率分布$b_{i_t}(k)$生成o_t
> 4. 按照状态i_t的状态转移概率分布$\{a_{i_t, i_{t+1}}\}$产生状态i_{t+1},$i_{t+1} = 1, 2, \ldots, N$
> 5. $t = t + 1$ 如果$t < T$转到3,否则，终止

概率计算算法

前向概率与后向概率

> 给定马尔可夫模型λ, 定义到时刻t部分观测序列为$o_1, o_2, \ldots, o_t$, 且状态q_i的概率为**前向概率**, 记作
>
> $$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, i_t = q_i | \lambda)$$
>
> 给定马尔可夫模型λ, 定义到时刻t状态为q_i的条件下, 从$t+1$到T的部分观测序列为$o_{t+1}, o_{t+2}, \ldots, o_T$的概率为**后向概率**, 记作
>
> $$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | i_t = q_i, \lambda)$$
>
> 关于α和β这两个公式, 前向概率从前往后递推， 后向概率从后向前递推。

前向算法

输入: λ, O

输出:$P(O|\lambda)$

1. 初值

$$\alpha_1(i) = \pi_i b_i(o_1), i = 1, 2, \ldots, N$$

观测值o_1, i的含义是对应状态q_i

这里α 是N维向量, 和状态集合Q的大小N有关系. α是个联合概率.

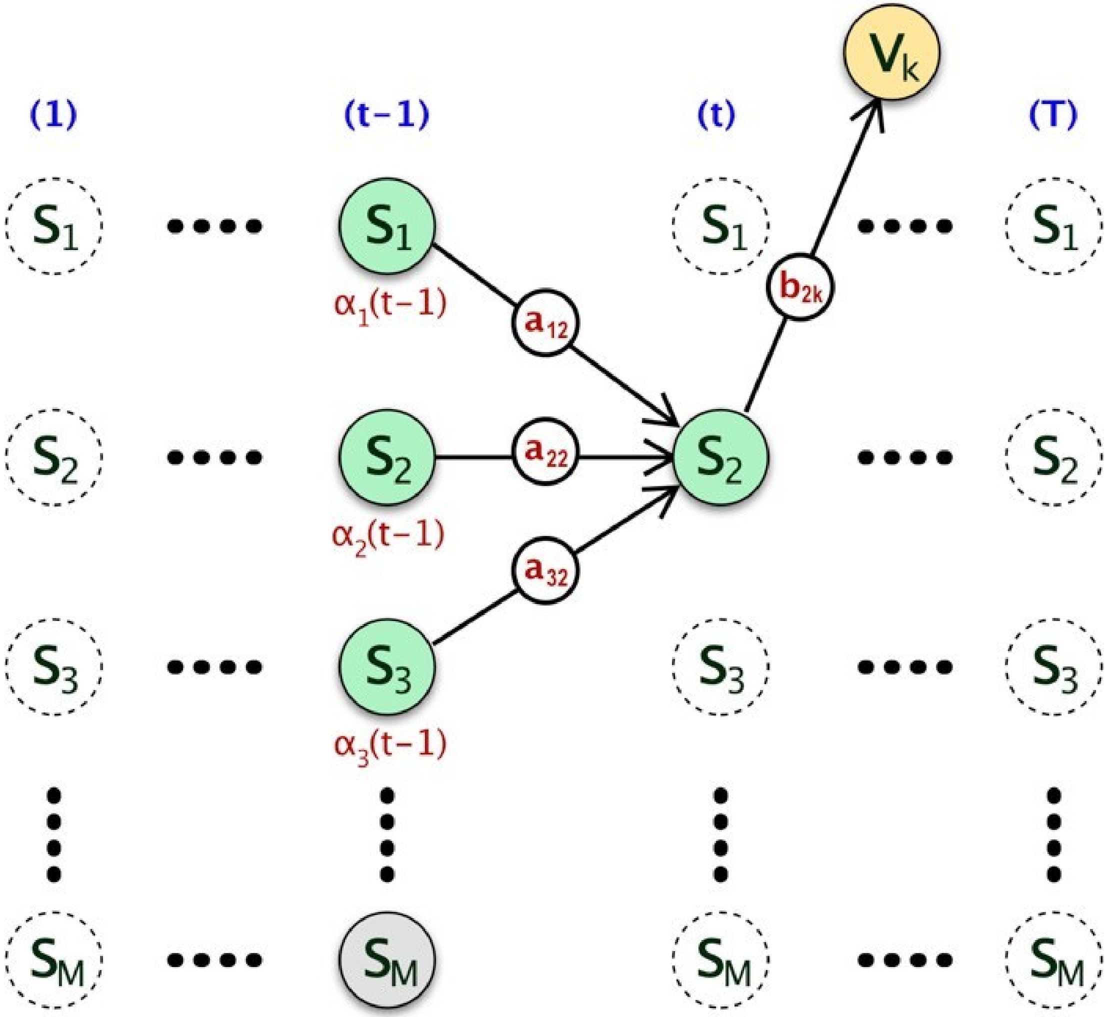
2. 递推

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) a_{ji}\right] b_i(o_{t+1}),\ i = 1, 2, \ldots, N,\ t = 1, 2, \ldots, T-1$$

3. 终止

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) = \sum_{i=1}^{N} \alpha_T(i)\beta_T(i)$$

注意, 这里$O \rightarrow (o_1, o_2, o_3, \ldots, o_t)$, α_i见前面前向概率的定义$P(o_1, o_2, \ldots, o_t, i_t = q_i|\lambda)$, 所以, 对$i$求和能把联合概率中的$I$消掉.

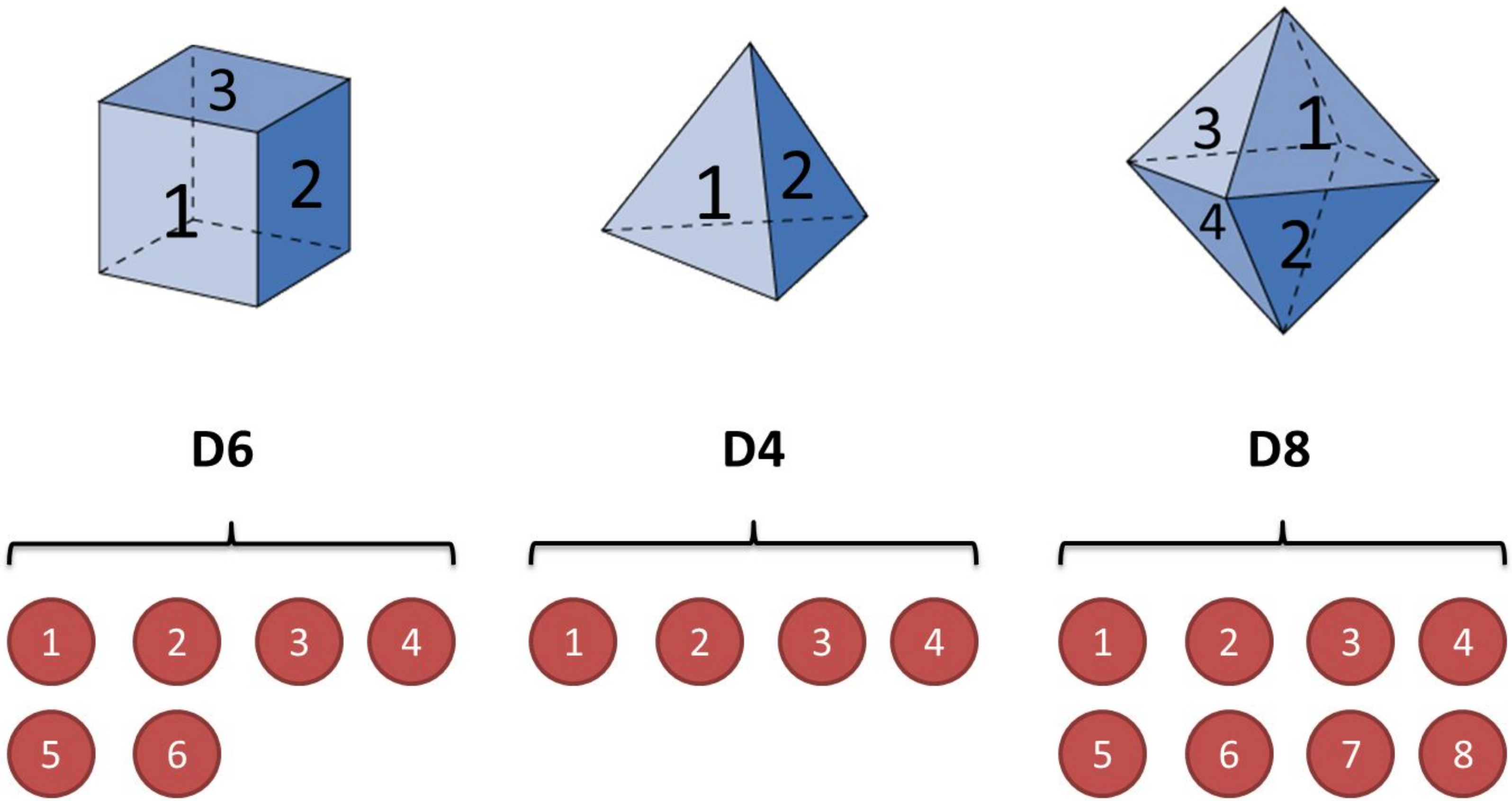


(1)
(t−1)
(t)
(T)
V_k
S_1
S_2
S_3
S_M
α_1(t−1)
α_2(t−1)
α_3(t−1)
a_12
a_22
a_32
b_2k

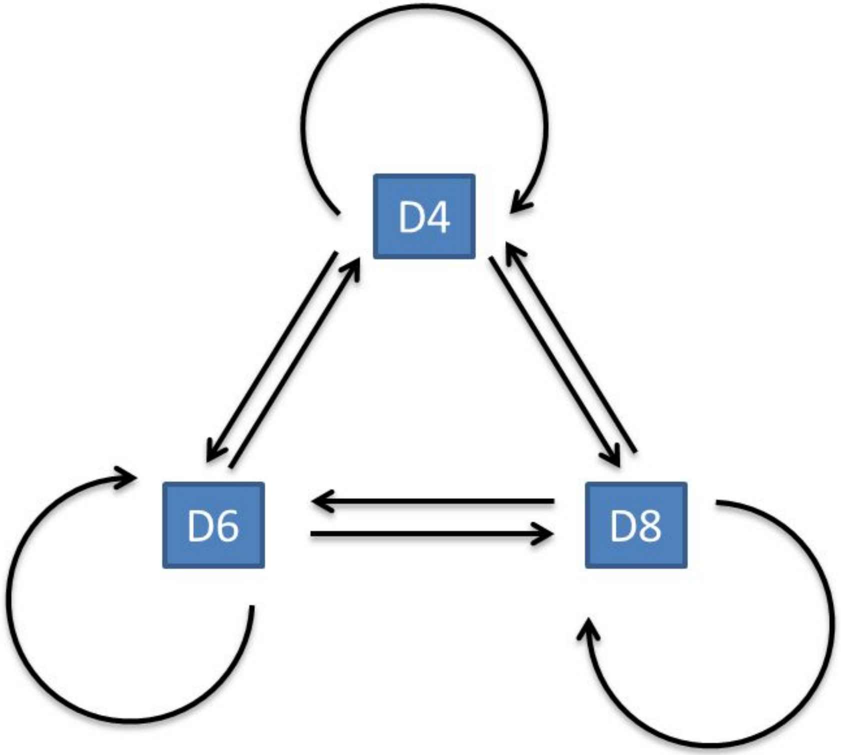
注意：

- 前向算法的关键是其局部计算前向概率, 然后利用路径结构将前向概率"递推"到全局.
- 减少计算量的原因在于每一次计算直接引用前一时刻的计算结果, **避免重复计算**.
- 前向算法计算$P(O|\lambda)$的复杂度是$O(N^2T)$阶的, 直接计算的复杂度是$O(TN^T)$阶, 所以$T=2$时候并没什么改善.

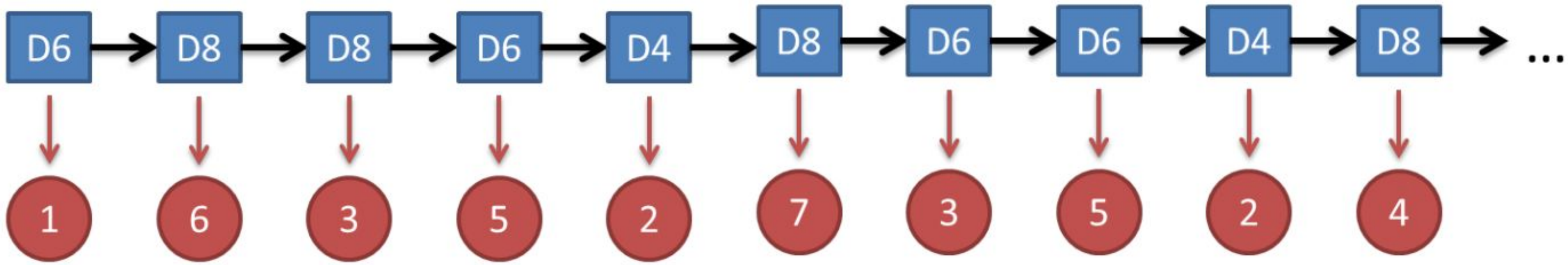
三种骰子和掷骰子可能产生的结果



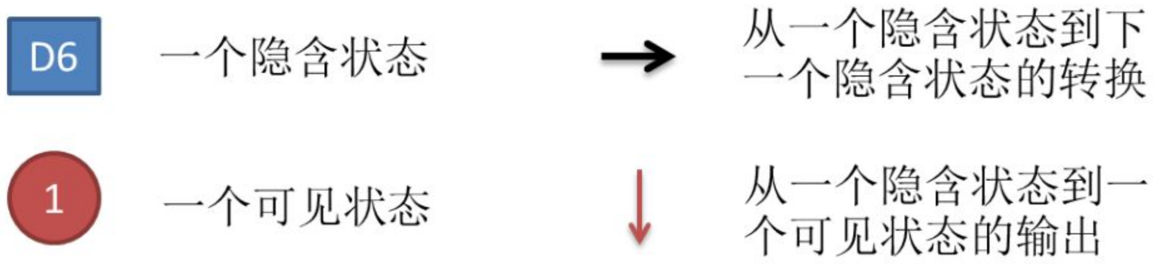
隐含状态转换关系示意图



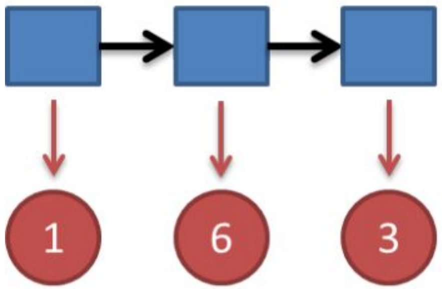
隐马尔可夫模型示意图



图例说明：



前向计算举例



	P1	P2	P3
D6	$\frac{1}{3}*\frac{1}{6}$	$\mathrm{P1(D6)}*\frac{1}{3}*\frac{1}{6}+\mathrm{P1}(D4)*\frac{1}{3}*\frac{1}{6}+\mathrm{P1(D8)}*\frac{1}{3}*\frac{1}{6}$	
D4	$\frac{1}{3}*\frac{1}{4}$	$\mathrm{P1(D6)}*\frac{1}{3}*0+\mathrm{P1}(D4)*\frac{1}{3}*0+\mathrm{P1(D8)}*\frac{1}{3}*0$	
D8	$\frac{1}{3}*\frac{1}{8}$	$\mathrm{P1(D6)}*\frac{1}{3}*\frac{1}{8}+\mathrm{P1}(D4)*\frac{1}{3}*\frac{1}{8}+\mathrm{P1(D8)}*\frac{1}{3}*\frac{1}{8}$	
Total	0.18	0.05	

后向算法

输入: λ, O

输出:$P(O|\lambda)$

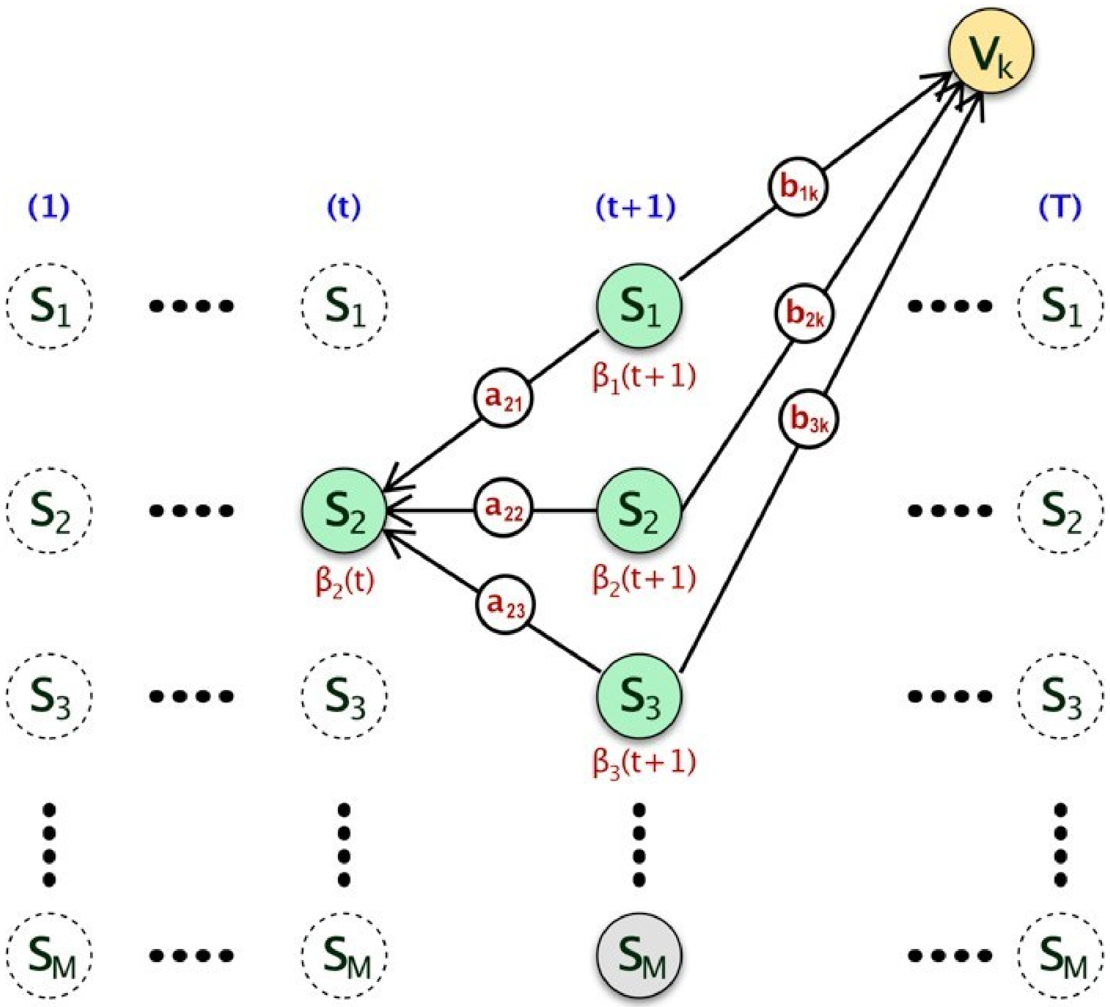
1. 终值$\beta_T(i) = 1, i = 1, 2, \ldots, N$,在$t = T$时刻, 观测序列已经确定.

2. 递推

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), i = 1, 2, \ldots, N, t = T-1, T-2, \ldots, 1$$

3. 得到

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i) = \sum_{i=1} \alpha_1(i) \beta_1(i)$$



$$\beta_i(t) = 1, when t = T$$

$$\beta_i(t) = \sum_{j=0}^{M} a_{ij} b_{jk} \beta_j(t+1) when t less than T$$

- 这里需要**注意**下, 按照后向算法, β在递推过程中会越来越小, 如果层数较多, 怕是$P(O|\lambda)$会消失
- 另外一个要注意的点$o_{t+1}\beta_{t+1}$
- 注意，红色部分为后补充，结合前面的前向概率最后的红色部分一起理解。

其实前向和后向不是为了求整个序列O的概率, 是为了求中间的某个点t, 前向后向主要是有这个关系:

$$\alpha_t(i)\beta_t(i) = P(i_t = q_i, O|\lambda)$$

当$t=1$或者$t=T-1$的时候, 单独用后向和前向就可以求得$P(O|\lambda)$, 分别利用前向和后向算法均可以求解$P(O|\lambda)$, 结果一致.

利用上述关系可以得到下面一些概率和期望, 这些概率和期望的表达式在后面估计模型参数的时候有应用.

概率与期望

1. 输入模型λ与观测O, 输出在时刻t处于状态q_i的概率$\gamma_t(i)$
2. 输入模型λ与观测O, 输出在时刻t处于状态q_i且在时刻$t+1$处于状态q_j的概率$\xi_t(i,j)$
3. 在观测O下状态i出现的期望值
4. 在观测O下状态i转移的期望值
5. 在观测O下状态i转移到状态j的期望值

学习算法

监督学习方法

效果好，费钱，如果有钱能拿到标注数据，不用犹豫。

Baum-Welch算法

马尔可夫模型实际上是一个含有隐变量的概率模型

$$P(O|\lambda) = \sum_I P(O|I,\lambda)P(I|\lambda)$$

> 输入: 观测数据$O = (o_1, o_2, \dots, o_T)$
> 输出: 隐马尔可夫模型参数

1. 初始化

   对$n = 0$, 选取$a_{ij}^{(0)}, b_j(k)^{(0)}, \pi_i^{(0)}$, 得到模型参数$\lambda^{(0)} = (A^{(0)}, B^{(0)}, \pi^{(0)})$

2. 递推

对$n = 1, 2, \ldots,$

$$a_{ij}^{(n+1)} = \frac{\sum\limits_{t=1}^{T-1} \xi_t(i,j)}{\sum\limits_{t=1}^{T-1} \gamma_t(i)}$$

$$b_j(k)^{(n+1)} = \frac{\sum\limits_{t=1, o_t=v_k}^{T} \gamma_t(j)}{\sum\limits_{t=1}^{T} \gamma_t(j)}$$

$$\pi_i^{(n+1)} = \gamma_1(i)$$

3. 终止

得到模型参数$\lambda^{(n+1)} = (A^{(n+1)}, B^{(n+1)}, \pi^{(n+1)})$

预测算法

近似算法(MAP)

每个时刻最有可能的状态i_t^*是

$$i_t^* = \arg\max_{1 \leqslant i \leqslant N} \left[\gamma_t(i)\right], t = 1, 2, \ldots, T$$

得到序列$I^* = (i_1^*, i_2^*, \ldots, i_T^*)$

这个算法, 在输出每个状态的时候, 只考虑了当前的状态.

维特比算法(Viterbi)

> 输入: 模型$\lambda = (A, B, \pi)$和观测$O = (o_1, o_2, \ldots, o_T)$
>
> 输出: 最优路径$I^* = (i_1^*, i_2^*, \ldots, i_T^*)$

1. 初始化

> $$\delta_1(i) = \pi_i b_i(o_1), i = 1, 2, \ldots, N$$
> $$\psi_1(i) = 0, i = 1, 2, \ldots, N$$

2. 递推

$$t = 2, 3, \ldots, T$$
$$\delta_t(i) = \max_{1 \leqslant j \leqslant N} [\delta_{t-1}(j) a_{ji}] \, b_i(o_t), i = 1, 2, \ldots, N$$
$$\psi_t(j) = \arg \max_{1 \leqslant j \leqslant N} [\delta_{t-1}(j) a_{ji}], i = 1, 2, \ldots, N$$

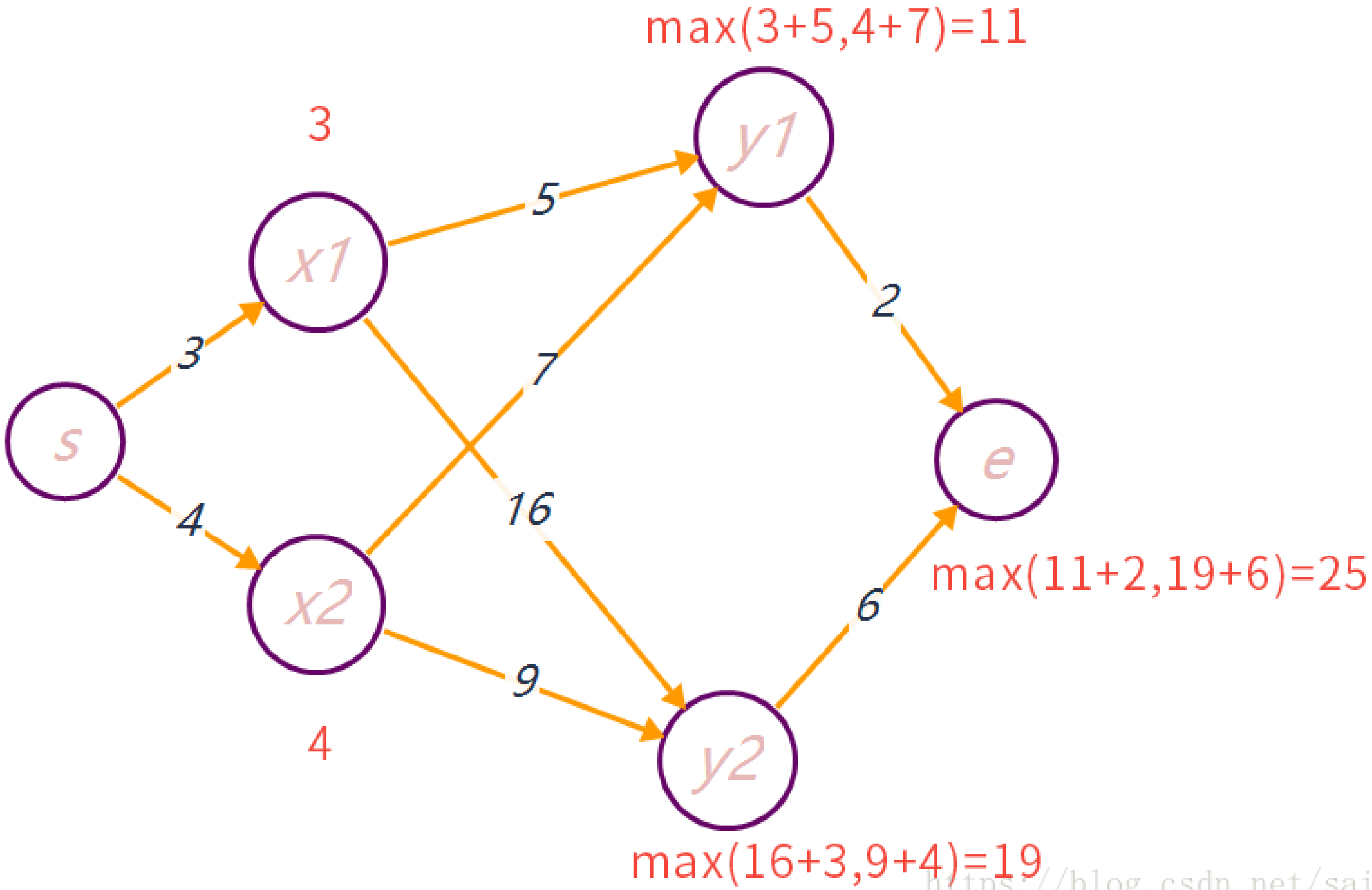
1. 终止

$$P^* = \max_{1 \leqslant i \leqslant N} \delta_T(i)$$
$$i_T^* = \arg \max_{1 \leqslant i \leqslant N} [\delta_T(i)]$$

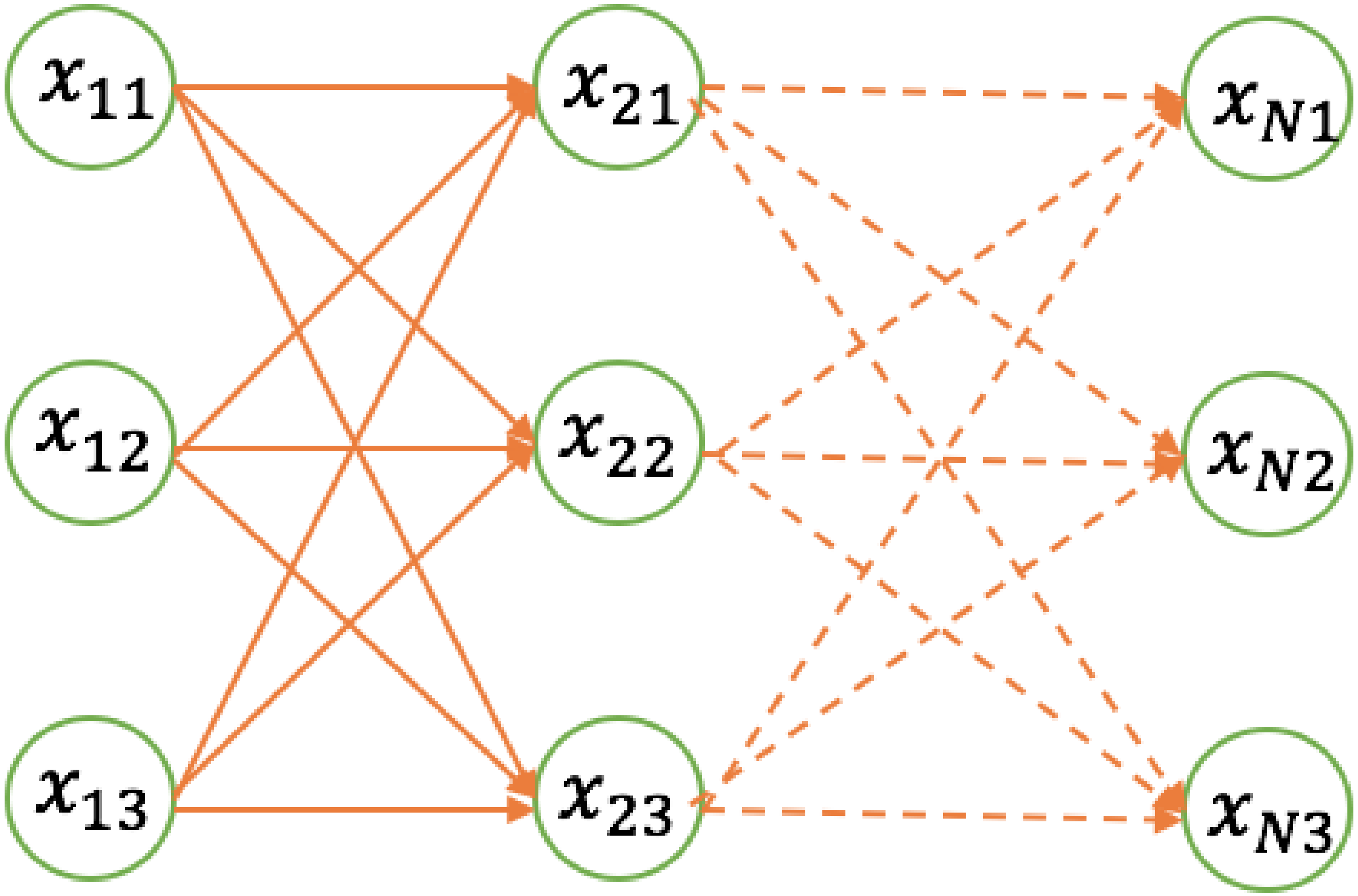
2. 最优路径回溯

$$t = T-1, T-2, \ldots, 1$$
$$i_t^* = \psi_{t+1}(i_{i+1}^*)$$

Viterbi算法理解



HMM Viterbi算法



HMM Viterbi Example, $O = \{$红、白、红$\}$

$$A=\begin{bmatrix}0.5 & 0.2 & 0.3\\ 0.3 & 0.5 & 0.2\\ 0.2 & 0.3 & 0.5\end{bmatrix},\quad B=\begin{bmatrix}0.5 & 0.5\\ 0.4 & 0.6\\ 0.7 & 0.3\end{bmatrix},\quad \pi=(0.2,0.4,0.4)^{\mathrm{T}}$$

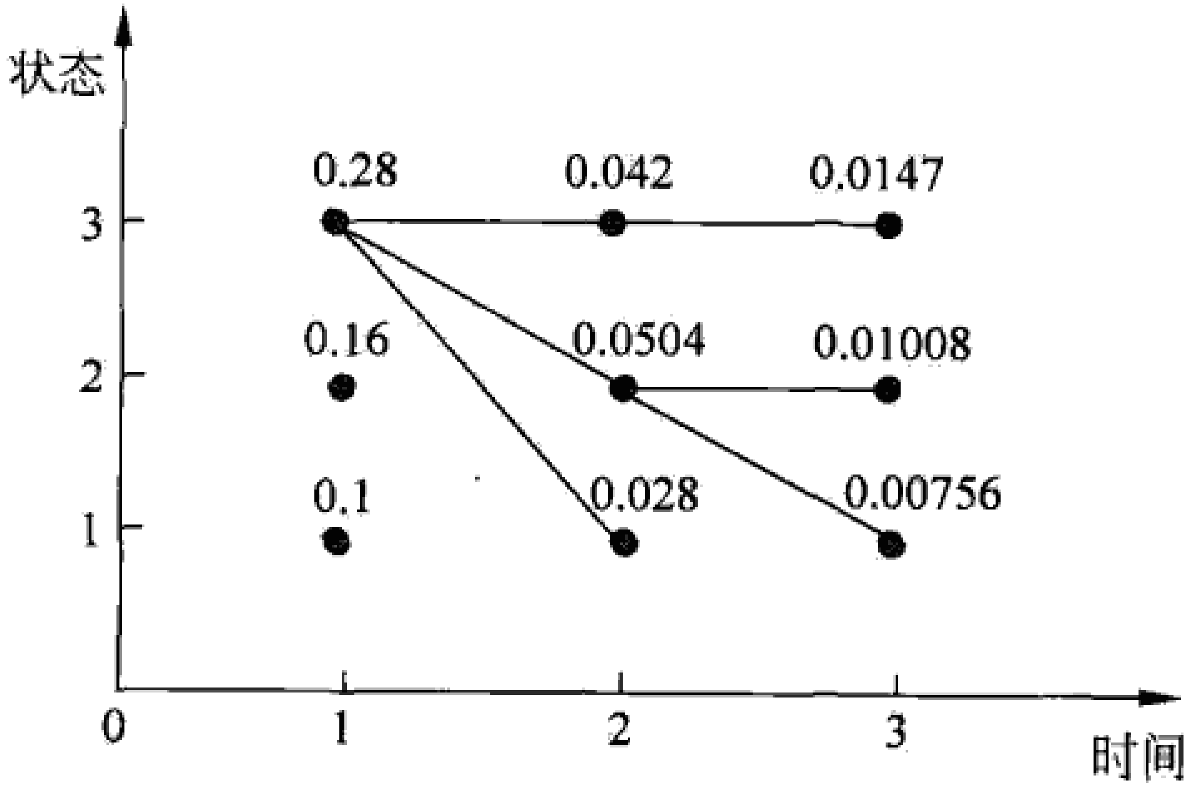


图 10.4　求最优路径

$$
\begin{aligned}
\delta_2(1) &= \max_{1\leqslant j\leqslant 3}[\delta_1(j)a_{j1}]b_1(o_2) \\
&= \max_{j}\{0.10\times 0.5, 0.16\times 0.3, 0.28\times 0.2\}\times 0.5 \\
&= 0.028
\end{aligned}
$$

$$\psi_2(1) = 3$$

$$\delta_2(2) = 0.0504, \quad \psi_2(2) = 3$$

$$\delta_2(3) = 0.042, \quad \psi_2(3) = 3$$

参考

1. [^4]: A tutorial on hidden Markov models and selected applications in speech recognition
2. [^1]: 数学之美-CH05隐含马尔可夫模型, 吴军
3.
4. [^3]: [PRML:13.2](## 参考)
5. Wikipedia: Hidden Markov Model
6. [^5]: An introduction to hidden markov Models
7. CRF++: Yet Another CRF toolkit： https://taku910.github.io/crfpp/
8. https://www.zhihu.com/question/20962240
9. https://nbviewer.jupyter.org/github/lopatovsky/CT-HMM/blob/master/hmms.ipynb

↘ Enjoy your machine learning!

https://github.com/wjssx/Statistical-Learning-Slides-Code

E-mail: csr_dsp@sina.com