大数据机器学习第八次理论作业

2026 年 1 月 22 日

一、　给定盒子和球组成的隐马尔可夫模型 $\lambda = (A, B, \pi)$, 其中,

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

设 $T = 4$，$O = (红, 白, 红, 白)$ 试用维特比算法求最优路径 $I^* = (i_1^*, i_2^*, i_3^*, i_4^*)$。

I. 问题分析与相关知识点

1) 隐马尔可夫模型

隐马尔可夫模型是关于时序的概率模型。描述由一个隐藏的马尔可夫链随机生成不可观测的状态随机序列，再由各状态生成一个观测而产生观测随机序列的过程，序列的每个位置也可看作是一个时刻。隐马尔可夫模型的组成元素：

- Q: 所有可能状态的集合，$Q = \{q_1, q_2, \cdots, q_N\}$
- V: 所有可能观测的集合，$V = \{v_1, v_2, \cdots, v_N\}$
- I: 长度为 T 的状态序列，$I = \{i_1, i_2, \cdots, i_T\}$
- O: 对应的长度 T 的观测序列，$O = \{o_1, o_2, \cdots, o_T\}$
- π: 初始状态概率向量
- A: 状态转移概率矩阵

$$A = [a_{ij}]_{N \times N}$$

$$a_{ij} = P(i_{t+1} = q_j | i_t = q_i), \quad i = 1, 2, \cdots, N; \quad j = 1, 2, \cdots, N$$

- B: 观测概率矩阵

$$B = [b_j(k)]_{N \times M}$$

$$b_j(k) = P(o_t = v_k | i_t = q_i), \quad k = 1, 2, \cdots, M; \quad j = 1, 2, \cdots, N$$

三要素为：

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

两个基本假设为：

1. 齐次马尔可夫性假设，隐马尔可夫链 t 的状态只和 t-1 的状态有关：

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

2. 观测独立性假设，观测只和当前时刻状态有关：

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

2) 隐马尔可夫模型的预测（解码）问题

隐马尔可夫模型的问题可分为计算 $P(O|\lambda)$ 的概率计算问题、估计使 $P=(O|\lambda)$ 最大的 $\lambda=(A,B,\pi)$ 的学习问题以及求使 $P=(I|O)$ 最大的状态序列 $I=(i_1,i_2,\cdots,i_T)$ 的预测问题。

马尔科夫模型的预测的两种算法为：近似算法与维特比算法（Viterbi algorithm）。近似算法计算简单，但不能保证预测的状态序列是整体最有可能的状态序列。

维特比算法使用动态规划解马尔可夫模型，用动态规划求概率最大路径。这时一条路径对应着一个状态序列。

根据动态规划的原理，最优路径的特性为：如果最优路径在时刻 t 通过结点 i_t^*，那么这一路径的从结点 i_t^* 到终点 i_T^* 的部分路径，对从 i_t^* 到终点 i_T^* 的所有可能的部分路径来说必须为最优。根据这一原理，我们从 $t=1$ 时刻开始，递推地计算在时刻 t 状态 i 的各条部分路径的最大概率，直到得到时刻 T 状态为 i 的各条路径的最大概率 $P*$ 和最优路径终结点 i_T^*。之后为了找出最优路径的结点，从终点 i_T^* 开始逐步向前得到结点 $i_{T-1}^*,\cdots,i_1^*$，得到最优路径 $I^*=(i_1^*,i_2^*,\cdots,i_T^*)$。

我们导入两个变量 δ 和 Ψ。

- 定义在时刻 t 状态为 i 的所有单个路径 $(i_1,i_2,\cdots,i_t)$ 的概率最大值为 $\delta_t(i)$

$$\delta_t(i)=\underset{i_1.i_2,\cdots,i_{t-1}}{max}P(i_t=i,i_{t-1},\cdots,i_1,o_t,\cdots,o_1|\lambda),\quad i=1,2,\cdots,N$$

  由定义可得变量 δ 的递推公式：

$$\begin{aligned}\delta_{t+1}(i)&=\underset{i_1.i_2,\cdots,i_{t-1}}{max}P(i_t=i,i_{t-1},\cdots,i_1,o_t,\cdots,o_1|\lambda)\\&=\underset{1\le j\le N}{max}[\delta_t(j)a_{ji}]b_i(o_{t+1}),\quad i=1,2,\cdots,N;t=1,2,\cdots,T-1\end{aligned}$$

- 定义在时刻时刻 t 状态为 i 的所有单个路径 $(i_1,i_2,\cdots,i_{t-1},i)$ 中概率最大路径的第 $t-1$ 个结点为 $\Psi_t(i)$：

$$\Psi_t(i)=\arg\max_{1\le j\le N}[\delta_{t-1}(j)a_{ji}],\quad i=1,2,\cdots,N$$

3) 维特比算法

输入：模型 $\lambda=(A,B,\pi)$ 和观测 $O=(o_1,o_2,\cdots,o_T)$

输出：最优路径 $I^*=(i_1^*,i_2^*,\cdots,i_T^*)$ 过程：

1. 初始化：

$$\delta_1(i)=\pi_ib_i(o_1),\quad i=1,2,\cdots,N$$

$$\Psi_1(i)=0,\quad i=1,2,\cdots,N$$

2. 递推，对 $t=2,3,\cdots,T$：

$$\delta_t(i)=\max_{1\le j\le N}[\delta_{t-1}(j)a_{ji}]b_i(o_t),\quad i=1,2,\cdots,N$$

$$\Psi_t(i)=\arg\max_{1\le j\le N}[\delta_{t-1}(j)a_{ji}],\quad i=1,2,\cdots,N$$

3. 终止：

$$P*=\max_{1\le j\le N}\delta_T(i)$$

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

4. 最优路径回溯，对 $t = T-1, T-2, \cdots, 1$:

$$i_t^* = \Psi_{t+1}(i_{t+1}^*)$$

求得最优路径 $I^* = (i_1^*, i_2^*, \cdots, i_T^*)$。

II. 求解最优路径

1. 初始化，在 $t = 1$ 时，对每个状态 i，$i = 1\ 2\ 3$，求观测状态为 i 观测 o_1 为红的概率。记概率为 $delta_1(i)$，则：

$$\delta_1(i) = \pi_i b_i(o_1) = \pi_i b_i(\text{红}), \quad i = 1, 2, 3$$

代入数据：

$$\delta_1(1) = 0.10, \quad \delta_1(2) = 0.16, \quad \delta_1(3) = 0.28$$

记 $\Psi_1(i) = 0, i = 1, 2, 3$

2. 在 $t = 2$ 时，对每个状态 i，$i = 1\ 2\ 3$，求 $t = 1$ 时状态为 j 观测为红并在 $t = 2$ 时状态为 i 观测 o_2 为白的路径的最大概率，记此最大概率为 $\delta_2(i)$，则

$$\delta_2(i) = \max_{1 \leq j \leq 3} [\delta_1(j) a_{ji}] b_i(o_2)$$

同时，对每个状态 i，记录概率最大路径的前一个状态 j：

$$\Psi_2(i) = \arg \max_{1 \leq j \leq 3} [\delta_1(j) a_{ji}], \quad i = 1, 2, 3$$

计算：

$$\begin{aligned}
\delta_2(1) &= \max\{0.10 \times 0.5, 0.16 \times 0.3, 0.28 \times 0.2\} \times 0.5 \\
&= 0.028 \\
\Psi_2(1) &= 3 \\
\delta_2(2) &= \max\{0.10 \times 0.2, 0.16 \times 0.5, 0.28 \times 0.3\} \times 0.6 \\
&= 0.0504 \\
\Psi_2(2) &= 3 \\
\delta_2(3) &= \max_{1 \leq j \leq 3} [\delta_1(j) a_{j3}] b_3(o_2) \\
&= \max\{0.10 \times 0.3, 0.16 \times 0.2, 0.28 \times 0.5\} \times 0.3 \\
&= 0.042 \\
\Psi_2(3) &= 3
\end{aligned}$$

同理，在 $t=3$ 时：

$$\begin{aligned}
\delta_3(i) &= \max_{1\le j\le 3}[\delta_2(j)a_{ji}]b_i(o_3) \\
\Psi_3(i) &= \arg\max_{1\le j\le 3}[\delta_2(j)a_{ji}] \\
\delta_3(1) &= \max\{0.028\times 0.5, 0.0504\times 0.3, 0.042\times 0.2\}\times 0.5 \\
&= 0.00756 \\
\Psi_3(1) &= 2 \\
\delta_3(2) &= \max\{0.028\times 0.2, 0.0504\times 0.5, 0.042\times 0.3\}\times 0.4 \\
&= 0.01008 \\
\Psi_3(2) &= 2 \\
\delta_3(3) &= \max\{0.028\times 0.3, 0.0504\times 0.2, 0.042\times 0.5\}\times 0.7 \\
&= 0.0147 \\
\Psi_3(3) &= 3
\end{aligned}$$

同理，在 $t=4$ 时：

$$\begin{aligned}
\delta_4(i) &= \max_{1\le j\le 3}[\delta_3(j)a_{ji}]b_i(o_4) \\
\Psi_4(i) &= \arg\max_{1\le j\le 3}[\delta_3(j)a_{ji}] \\
\delta_4(1) &= \max\{0.00756\times 0.5, 0.01008\times 0.3, 0.0147\times 0.2\}\times 0.5 \\
&= 0.00189 \\
\Psi_4(1) &= 1 \\
\delta_4(2) &= \max\{0.00756\times 0.2, 0.01008\times 0.5, 0.0147\times 0.3\}\times 0.6 \\
&= 0.003024 \\
\Psi_4(2) &= 2 \\
\delta_4(3) &= \max\{0.00756\times 0.3, 0.01008\times 0.2, 0.0147\times 0.5\}\times 0.3 \\
&= 0.002205 \\
\Psi_4(3) &= 3
\end{aligned}$$

3. 以 P^* 表示最优路径概率，则

$$P^* = \max_{1\le j\le 3}[\delta_4(i)] = 0.003024$$

最优路径的终点为 i_4^*:

$$i_4^* == \arg\max_i[\delta_4(i)] = 2$$

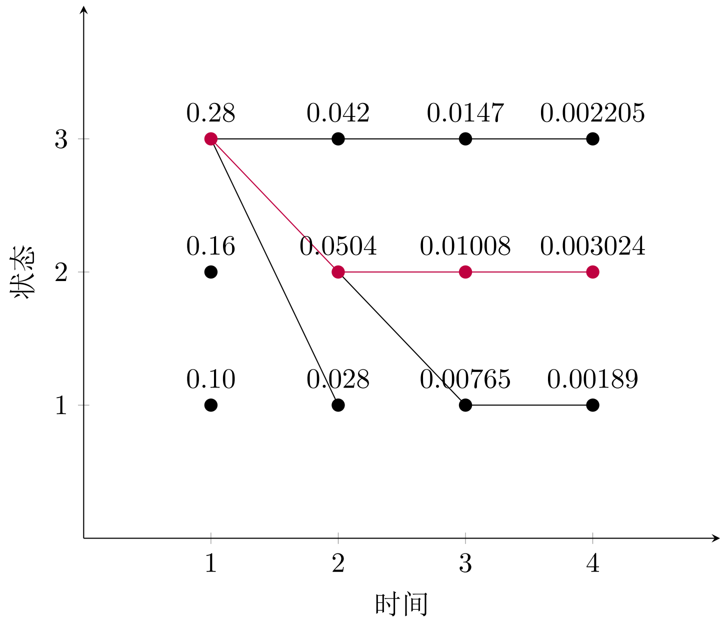
4. 由最优路径的终点 i_4^*，逆向找到 i_3^*, i_2^*, i_1^*:

$$t=3\text{时，}i_3^* = \Psi_4(i_4^*) = \Psi_4(2) = 2$$

$$t=2\text{时，}i_2^* = \Psi_3(i_3^*) = \Psi_3(2) = 2$$

$$t=1\text{时，}i_1^* = \Psi_2(i_2^*) = \Psi_2(2) = 3$$

于是求得最优路径，即最优状态序列 $I^* = (i_1^*, i_2^*, i_3^*, i_4^*) = (3,2,2,2)$。



III. 代码验证

代码实现

```python
import numpy as np

# 输入数据
# lambda= (A, B, pi)
pi = np.array([0.2, 0.4, 0.4])
# N=3
A = np.array([[0.5, 0.2, 0.3], [0.3, 0.5, 0.2], [0.2, 0.3, 0.5]])
B = np.array([[0.5, 0.5], [0.4, 0.6], [0.7, 0.3]])
# T = 4, 红0白1
O = np.array([0, 1, 0, 1])  # 长度为T的观测序列O
Q = np.array([1, 2, 3])  # 所有可能的状态集合Q

def viterbi(A, B, pi, O, Q):
    N =  len(Q)  # 所有可能的状态共有N个
    T =  len(O)  # 观测序列的长度为T
    delta_t = np.zeros([T, N])  # 存储delta
    Psi_t = np.zeros([T, N])  # 存储Psi
    I_star = np.array([0, 0, 0, 0])  # 存储最优路径序列
    print("开始递推")
    # 开始递推
    for t in  range(T):
        print(f't={t + 1}')
        # 初始化 t=1
        if t == 0:
            for i in  range(N):
                delta_t[0][i] = pi[i] * B[i][O[0]]
                Psi_t[0][i] = 0
                print(f'delta_1[{i + 1}]:{delta_t[0][i]:.6f},Psi_1[{i + 1}]:{Psi_t[0][i]}')
        # t=2,3,...,T
        else:
            for i in  range(N):
```

```python
                delta_t[t][i] = np. max([np.
                    max(delta_t[t - 1][j - 1] * A[j - 1][i]) * B[i][O[t]] for j in Q])
                delta_t_A = [delta_t[t - 1][j - 1] * A[j - 1][i] for j in Q]
                Psi_t[t][i] = np.argmax(delta_t_A) + 1
                print(f'delta_{t + 1}[{i + 1}]:{delta_t[t][i]:.6f},Psi_{t + 1}[{i + 1}]:{Psi_t
                    [t][i]}')
    # 终止 t=T
    print('终止')
    P_star = np. max(delta_t[T - 1])
    I_star[T - 1] =  int(np.argmax(delta_t[T - 1])) + 1

    # 回溯
    print('开始回溯')
    for i in  range(T - 1):
        t = T - i - 2
        # 对t=T-1, T-2, ...,1
        I_star[t] =  int(Psi_t[t + 1][I_star[t + 1] - 1])
    print("结束")
    return I_star

if __name__ == '__main__':
    I_star = viterbi(A, B, pi, O, Q)
    print(f'最优序列为: {I_star}')
```

输出结果

```
开始递推
t=1
delta_1[1]:0.100000,Psi_1[1]:0.0
delta_1[2]:0.160000,Psi_1[2]:0.0
delta_1[3]:0.280000,Psi_1[3]:0.0
t=2
delta_2[1]:0.028000,Psi_2[1]:3.0
delta_2[2]:0.050400,Psi_2[2]:3.0
delta_2[3]:0.042000,Psi_2[3]:3.0
t=3
delta_3[1]:0.007560,Psi_3[1]:2.0
delta_3[2]:0.010080,Psi_3[2]:2.0
delta_3[3]:0.014700,Psi_3[3]:3.0
t=4
delta_4[1]:0.001890,Psi_4[1]:1.0
delta_4[2]:0.003024,Psi_4[2]:2.0
delta_4[3]:0.002205,Psi_4[3]:3.0
终止
开始回溯
结束
最优序列为: [3 2 2 2]
```

IV. 总结

代码计算的结果与之前的结果一致，最优状态序列为 $I^* = (3, 2, 2, 2)$。

二、　试用前向概率和后向概率推导：

$$P(O|\lambda)=\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j),\quad t=1,2,\ldots,T-1$$

I. 问题分析与相关知识点

1) 前向概率

给定马尔科夫模型 λ，定义到时刻 t 部分观测序列为：$o_1,o_2,\cdots,o_t$, 且状态为 q_i 的概率为前向概率，记为：

$$\alpha_t(i)=P(o_1,o_2,\cdots,o_t,i_t=q_i|\lambda)$$

可以递推地求前向概率 $\alpha_t(i)$ 以及观测序列概率 $P(O|\lambda)$。

2) 观测序列概率的前向算法

输入：隐马尔可夫模型 λ，观测序列 O
输出：观测序列概率 $P(O|\lambda)$

1. 初始化前向概率：

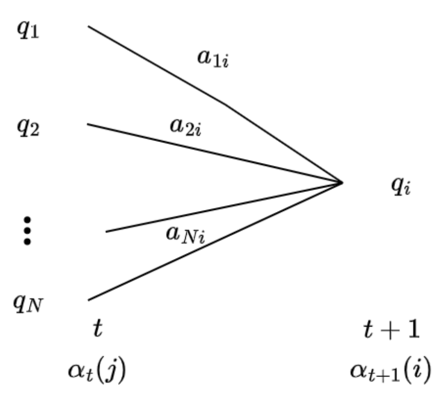
$$\alpha_1(i)=\pi_i b_i(o_1),\quad i=1,2,\cdots,N$$

2. 递推，对 $t=1,2,\cdots,T-1$, 根据马尔可夫模型的两个基本假设，我们基于前一时刻的各状态的前向概率，乘以对应的状态转移概率，以及观测概率，得到时刻 $t+1$ 的各个状态 q_i 的前向概率。

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

3. 终止，将时刻 T 各状态对应的概率相加，得到观测序列概率：

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



前向概率递推公式

3) 后向概率

给定马尔可夫模型 λ，定义到时刻 t 状态为 q_i 条件下，从 $t+1$ 到 T 的部分观测序列为 $o_{t+1}, o_{t+2}, \cdots, o_T$ 的概率为后向概率，记作

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \cdots, o_T | i_t = q_i, \lambda)$$

可以用递推的方法求后向概率 $\beta_t(i)$ 以及观测序列概率 $P(O|\lambda)$

4) 观测序列概率的后向算法

输入：隐马尔可夫模型 λ，观测序列 O
输出：观测序列概率 $P(O|\lambda)$

1. 初始化后向概率：

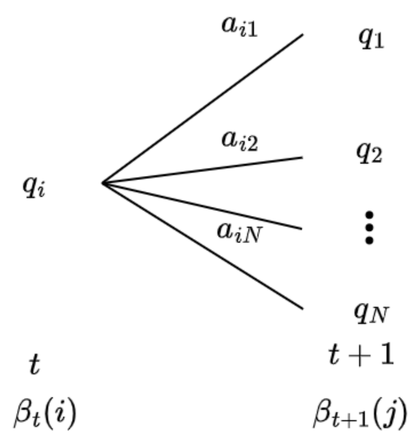
$$\beta_T(i) = 1, \quad i = 1, 2, \cdots, N$$

2. 递推，对 $t = T-1, T-2, \cdots, 1$, 根据后一时刻的各状态的后向概率、对应的状态转移概率以及观测概率计算时刻 t 的各个状态 q_i 的后向概率。

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

3. 终止：

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



后向概率递推公式

II. 问题推导

观测序列概率 $P(O|\lambda)$ 即观测序列 O 在模型参数 $\lambda = (\pi, A, B)$ 下出现的条件概率。I 是可能出现的隐藏的状态序列，对任意一个状态序列 I，有：

$$P(I|\lambda) = P(i_1, i_2, \cdots, i_T|\lambda) = \pi_{i_1} a_{i_1 i_2} \cdots a_{i_{T-1} i_T}$$

且:

$$P(O|I, \lambda) = P(o_1, o_2, \cdots, o_T | i_1, i_2, \cdots, i_T, \lambda) = b_{i_1}(o_1) b_{i_2}(o_2) \cdots b_{i_T}(o_T)$$

$$P(O,I|\lambda) = P(O|I,\lambda)P(I|\lambda) = \pi_{i_1} b_{i_1}(o_1) a_{i_1 i_2} b_{i_2}(o_2) \cdots a_{i_{T-1} i_T} b_{i_T}(o_T)$$

根据联合概率可求边缘概率:

$$\begin{aligned} P(O|\lambda) &= \sum_I P(O,I|\lambda) \\ &= \sum_{i_1,i_2,\cdots,i_T} \pi_{i_1} b_{i_1}(o_1) a_{i_1 i_2} b_{i_2}(o_2) \cdots a_{i_{T-1} i_T} b_{i_T}(o_T) \end{aligned}$$

直接计算上式时间复杂度 $O(TN^T)$ 过高，我们根据前向概率和后向概率的定义推导：

$$\begin{aligned} P(O|\lambda) &= \sum_{i_t,i_{t+1}} P(O,i_t,i_{i+1}|\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} P(O,i_t=q_i,i_{t+1}=q_j|\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} P(o_1,o_2,\cdots,o_{t-1},\cdots,o_T,i_t=q_i,i_{t+1}=q_j|\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} P(o_1,o_2,\cdots,o_{t-1},\cdots,o_t,i_t=q_i|\lambda)P(i_{t+1}=q_j,o_{t+1},o_{t+2},\cdots,\cdots,o_T|\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)P(i_{t+1}=q_j|\lambda)P(o_{t+1},o_{t+2},\cdots,\cdots,o_T|i_{t+1}=q_j,\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)P(i_{t+1}=q_j|\lambda)P(o_{t+1}|i_{t+1}=q_j,\lambda)P(o_{t+2},\cdots,\cdots,o_T|i_{t+1}=q_j,\lambda) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j) \end{aligned}$$

推导完毕。

III. 总结

使用前向概率加上后向概率可以计算整体的观测序列概率，且相比于直接计算法，可以减少时间复杂度。