隐马尔可夫模型(1)

(HMM)

徐静

2018-06-13

0.马尔可夫链

1.隐马尔可夫模型的基本概念

2.概率计算算法

3.参考文献

4.下次预告

5.问题与讨论

0. 马尔可夫链

1. 马尔可夫链的概念

1. 马尔可夫链的概念[1]

1. Markov过程是一类重要的随机过程(啥是随机过程？)。应用场景：物理学，生物学，通信，信息，信号处理，语言处理，自动化控制等领域

2. Markov性：已知随机过程在时刻 t_i 所处的状态的条件下，过程在时刻 $t(> t_i)$ 所处的状态与过程在时刻 t_i 以前的状态无关，而仅与过程在 t_i 所处额状态有关，则称该过程为Markov过程。

马尔可夫，A.A.

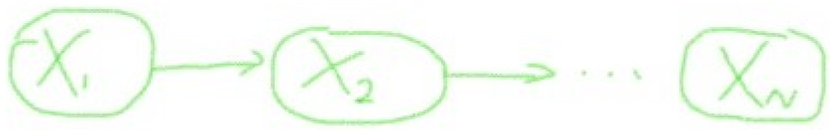
T 表示时间空间 E 表示状态空间

1. T 连续, E 连续 -- 连续Markov过程
2. T 连续, E 离散 -- 离散Morkov过程
3. T 离散, E 连续 -- Markov序列
4. T 离散, E 离散 -- Markov链

[1] 是一种非常经典又相对简单的概率图模型http://dataxujing.coding.me/page2/

一个随机变量的序列称为**马尔可夫链**，如果他们满足**马尔可夫性质**:

$$\Pr(X_i|X_1,\ldots X_{i-1}) = \Pr(X_i|X_{i-1}), \quad \forall\, i$$



Markov Chain的性质:

0.条件独立 $\{X_1, \ldots, X_{i-1}\} \perp \{X_{i+1}, \ldots, X_N\} | X_i$

$$\Pr(X_1, \ldots, X_N) = \Pr(X_1) \prod_{i=2}^{N} \Pr(X_i | X_{i-1})$$

1.马尔可夫性

2.转移概率（转移矩阵)

$$p_{ij}^k = \Pr(X^{k+1} = j | X^k = i, X^{k-1}, \ldots, X^1) = \Pr(X^{k+1} = j | X^k = i)$$

第 k 个时刻从第 i 个状态转移到第 j 个状态的转移概率

3.齐次马尔可夫链

4.n步转移概率

5.稳态分布*(啥是稳态分布？)

小栗子：

假设现有商品ABC今年的市场占率分别为20%、20%和40%，A商品每年流失30%到B，流失30%到C；B商品下一年会流失20%到A，流失30%到C； C商品每年会流失40%到A，流失40%到B，则刚开始ABC的市场占有率形成的矩阵[A0 B0 C0]=[0.2 0.2 0.4]，商品流动率形成的马尔科夫矩阵p=[0.4 0.3 0.3;0.2 0.5 0.3;0.4 0.4 0.2]，（这里要注意一下马尔科夫矩阵的性质：矩阵的每行的和为1，矩阵的每列的和也为1）。然后我们可以利用马尔科夫链推算下一年的商品ABC的市场占有率[A1 B1 C1]=p[A0 B0 C0]=[0.26 0.26 0.24] 。 如果ABC商品市场占有率满足马氏性，那么最终（平稳）的商品市场占有率为[An Bn Cn=pn[A0 B0 C0]=[0.2556 0.2556 0.2556]

Markov Chain现实应用：

1. HTTP服务请求预测：Google 依据用户即将访问某一界面的概率提前准备好该界面以提高查询速度。
2. 关键词集群识别：将关键词按不同集群分类，并根据关键词集群确定用户未来搜索路径。
3. 检索推荐：根据用户行为自动为用户推荐链接和检索
4. 评分：根据马尔科夫链可确定权威搜索模式，判定用户的搜索行为

1.隐马尔可夫模型的基本概念

0.HMM的定义

1.小栗子

2.观测序列的生成过程

3.隐马尔可夫的3个基本问题

0.HMM的定义

定义 (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，再由各个状态生成一个观测而产生观测随机序列的过程。隐藏的马尔可夫链随机生成的状态的序列称为状态序列(state sequence);每个状态生成一个观测，而由此产生的观测的随机序列，称为观测序列(observation sequence)。序列的每一个位置又可以看做一个时刻。

- HMM由初始概率分布，状态转移概率分布，观测概率分布确定
- 设 Q 是所欲可能的状态的集合， V是所有可能的观测的集合

$$Q=\{q_1,\ldots q_N\},\quad V=\{v_1,\ldots,v_M\}$$

  其中 N 是可能的状态数， M是可能的观测数。

- I是长度为 T 的状态序列， O 是对应的观测序列

$$I=(i_1,\ldots,i_T),\quad O=(o_1,\ldots,o_T)$$

- A是状态转移矩阵
- B 是观测概率矩阵

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

其中 $b_j(k)=\Pr(o_t=v_k|i_t=q_j),k=1,\ldots M;j=1,..,N$

- π是初始状态概率向量：

$$\pi=(\pi_i)$$

  其中 $\pi_i=\Pr(i_i=q_i),\quad i=1,..,N$ 是时刻 $t=1$ 处于状态 q_i的概率

- HMM： $\lambda=(A,B,\pi)$, A,B,π称为HMM的三要素

- 状态转移概率矩阵与初始的状态概率向量确定了马尔可夫链，生成不可观测的状态序列。观测概率矩阵确定了如何从状态生成观测，与状态序列综合决定了如何产生观测序列
- 齐次马尔可夫性假设

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

- 观测独立性假设：任意时刻的观测仅依赖于该时刻的马尔可夫链的状态，与其他观测和状态无关

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

1.小栗子

（盒子和球模型）假设有4个盒子，每个盒子里都装有红白两种颜色的球，如下表

Show 10 entries Search:

	盒子	红球数	白球数
1	1	5	5
2	2	3	7
3	3	6	4
4	4	8	2

Showing 1 to 4 of 4 entries Previous 1 Next

按照下面方法抽球，产生一个球的颜色的观测序列：开始，从4个盒子里以等概率抽取1个盒子，从这个盒子里随机抽取1个球，记录其颜色后，放回；然后从当前盒子随机转移到下一个盒子，规则是：如果当前盒子是盒子1，那么下一个盒子一定是盒子2，如果当前盒子是2或3，那么分别依概率0.4,0.6转义到左边或右边的盒子，如果当前是盒子4，那么各以0.5的概率停留在盒子4或转移到盒子3；确定转移的合资后，再从这个盒子里抽出1个球，记录其颜色，放回；如此下去，重复进行5次，得到一个球的颜色的观测序列：{红，红，白，白，红}。在这个过程中，观察者只能观察到球的颜色的序列，观测不到球是从哪个盒子取出的，观测不到盒子的序列。

- 智能观测到球的颜色的序列，观测不到球从哪个盒子取出，观测不到盒子的序列
- 两个盒子的序列：盒子的序列(状态序列)，球的颜色的观测序列(观测序列)，前者是隐藏的后者是可观测的
- 这是一个HMM的例子 $Q=\{盒子1,盒子2,盒子3,盒子4\}$, $V=\{红,白\}$, $T=5$, $\pi=(0.25,0.25,0.25,0.25)^T$

$$A=\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0.4 & 0 & 0.6 & 0 \\ 0 & 0.4 & 0 & 0.6 \\ 0 & 0 & 0.5 & 0.5 \end{bmatrix}$$

$$B=\begin{bmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \\ 0.6 & 0.4 \\ 0.8 & 0.2 \end{bmatrix}$$

2.观测序列的生成过程

输入：隐马尔可夫模型 $\lambda = (A, B, \pi)$, 观测序列长度 T;

输出：观测序列 $O = (o_1, \ldots, o_T)$

(1)按照初始状态分布 π 产生状态 i_1

(2)令 $t = 1$

(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}

(5)令 $t = t + 1$ 如果 $t < T$, 转(3);否则终止

3.HMM的3个基本问题

- 概率计算问题

给定模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, ..., o_T)$ 计算在模型 λ 下观测序列出现的概率 $\Pr(O|\lambda)$

- 学习问题

知道观测序列，估计HMM模型的参数，使得在该模型下观测序列的概率 $\Pr(O|\lambda)$ 最大

- 预测问题

也称为解码问题，知道HMM模型和观测序列，求对给定观测序列条件概率 $\Pr(I/O)$ 最大的状态序列 $I = (i_1, ..., i_T)$, 即给定观测序列，求最优可能的对应的状态序列

2.概率计算算法

0.直接计算算法

1.前向算法

2.后向算法

3.如何自己实现HMM

4.相关的工具包

5.HMM的应用场景

0.直接计算法

- 仅在概念上可行[1]
- 最直接的方法是按照概率公式直接计算
- 列举所有可能的长度为 T 的状态序列 I
- 求每个状态序列与观测序列 O 的联合概率 $\Pr(O, I|\lambda)$
- 然后对所有状态序列对应的联合概率求和，得到 $\Pr(O|\lambda)$

[1].具体公式比较简单我就不列了，计算量是 $O(TN^T)$，显然不可行

1.前向算法

定义(前向概率)： 给定HMM,定义到时刻 t 部分观测序列为 $o_1, o_2, \ldots, o_t$ 且状态为 q_i 的概率为前向概率, 记作

$$\alpha_t(i) = \Pr(o_1, \ldots, o_t, i_t = q_i | \lambda)$$

可以递推的求得前向概率 $\alpha_t(i)$以及观测序列概率 $\Pr(O|\lambda)$

(观测序列概率的前向传播)

输入：隐马尔可夫模型 λ, 观测序列 O

输出：观测序列概率 $\Pr(O|\lambda)$

(1)初值

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

(2)递推 对 $t = 12, \ldots, T-1$

$$\alpha_{t+1}(i) = [\sum_{j=1}^{N} \alpha_i(j) a_{ji}] b_i(o_{t+1})$$

(3)终止

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

[1].这是啥意思?

2.后向算法

定义（后向概率）给定隐马尔可夫模型 λ, 定义在时刻 t状态为 q_i 的条件下，从 $t+1$ 到 T的部分观测序列为 $o_{t+1},...,o_T$ 的概率为后向概率，记作

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

可以用递推的方法求得后向概率 $\beta_t(i)$ 及观测序列的概率 $\Pr(O|\lambda)$

(观测序列概率的后向传播算法)

输入：隐马尔可夫模型 λ， 观测序列 O

输出: 观测序列概率 $\Pr(O|\lambda)$

(1)

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

(2)对 $t = T-1, T-2, \ldots, 1$

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

(3)

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

[1].这是啥意思?

3.如何自己实现HMM

- 观测序列的生成Python实现

```
mport numpy as np
lass HMM():
    '''
    算法首先初始化两个长度为T的向量，接着按照初始状态分布pi生成第一个状态，
    取出状态对应的观测的概率分布生成一个观测。接下来都是按前一个状态取出
    状态转移概率分布生成状态，再取出状态对应的观测的概率分布生成一个观测。
    重复该步骤就得到长度为T的观测和状态向量。
    '''
    def __init__(self, A, B, pi):
        self.A = A
        self.B = B
        self.pi = pi
    def simulate(self, T):
        # draw_from接受一个概率分布，然后生成该分布下的一个样本。
        def draw_from(probs):
            return np.where(np.random.multinomial(1,probs) == 1)[0][0]
        observations = np.zeros(T, dtype=int)
        states = np.zeros(T, dtype=int)
        states[0] = draw_from(self.pi)
        observations[0] = draw_from(self.B[states[0], :])
        for t in range(1, T):
```

- 前向算法Python实现

```
def forward(self, obs_seq):
    """
    前向算法
    """
    N = self.A.shape[0]
    T = len(obs_seq)
    F = np.zeros((N, T))
    F[:, 0] = self.pi * self.B[:, obs_seq[0]]
    for t in range(1, T):
        for n in range(N):
            F[n, t] = np.dot(F[:, t-1], (self.A[:, n])) * self.B[n, o
   return F
```

- 后向算法Python实现

```
def backward(self, obs_seq):
    """后向算法"""
    N = self.A.shape[0]
    T = len(obs_seq)
    M = np.zeros((N, T))
    M[:, T-1] = 1
    # 或者M[:, -1:] = 1，列上表示最后一行
    for t in reversed(range(T-1)):
        for n in range(N):
            M[n, t] = np.dot(self.A[n, :], M[:, t+1]) * self.B[n, obs
    return M
```

4.相关的工具包

R

- RHmm
- depmixS4
- HMM

Python

- hmmlearn
- seqlearn

[1].我们都是“调包侠”

5.HMM的应用场景

- 寻找一个事务在一段时间里的变化模式(规律)
- 这些模式可以发生在很多领域…
- 文本分词，语音识别，金融衍生品,管理决策…

4.下次预告

4.下次预告

1.学习算法

- 问题2的解决办法

2.预测算法

- 问题3的解决办法

3.隐马尔可夫模型和生成语音识别的变体

- 基于GMM-HMM构建声学模型
- 基于轨迹和隐藏动态模型的语音建模和识别
- 基于生成模型HMM及其变体解决语音识别问题

5.参考文献

[1]. 机器学习[M],周志华

[2]. 统计学习方法[M],李航

[3]. 解析深度学习语音识别实践[M],俞凯，钱彦旻

[4]. 深度学习导论及案例分析[M],李玉鑑，张婷

6.问题与讨论

1. Markov链的性质有哪些?
2. 生成学习与传统判别学习的区别在哪?
3. HMM三个基本问题是什么?
4. HMM的应用场景有哪些?

Thanks For Your Attention!