

北京林业大学

2019 学年—2020 学年第 2 学期 数据挖掘 实验报告书

专 业：计算机创新 班 级：计创 17 班

姓 名：窦勇强 学 号：171001209

实验地点：计算中心 任课教师：王建新

实验题目：马尔科夫模型和隐马尔科夫模型

实验环境：Python 3.7, PyCharm by JetBrains

1 实验要求

1. 掌握马尔科夫模型和隐马尔科夫模型的原理和算法。
2. 能够利用模型编写相应的程序。
3. 能够利用编写的程序解决实际问题，包括训练模型参数、识别和预测。

2 实验内容

2.1 实验背景介绍

马尔可夫模型和隐马尔科夫模型不仅仅广泛应用于语音识别和自然语言理解。下面就是一个例子：

中国湖南湖北一带有一句歇后语“下雨天打孩子——闲着也是闲着”。其实“打孩子”是谐音，实际上是“打鞋子”，也就是制作草鞋的意思。表达的大致意思是：下雨天不能出去到地里劳作，就在家利用空闲的时间预备一些草鞋。一段时间的观察结果如下：

序号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
天气	阴	阴	晴	晴	阴	雨	阴	雨	晴	雨	阴	阴	雨	雨	晴	晴	阴	阴	阴
打鞋	打	不	不	打	不	打	不	不	不	打	打	不	打	打	不	不	不	打	不

图 1: 一段时间的观察结果

如果天气情况看作状态，打孩子/不打孩子看作观察结果，那么这是一个典型的隐马尔科夫模型系统。根据上表数据，推算状态转移矩阵 A （是一个 3 乘 3 的矩阵），和状态到观察值的条件概率矩阵（是一个 3 乘 2 的矩阵）。状态的初始概率暂时用整体的概率来表示。如果今天阴天，请计算后天“打孩子”的概率。

有了初步的三种值：初始状态概率、状态转移矩阵和状态到观察值的条件概率，那么在知道一串观察值“打”与“不打”的情况下，要进行模型推断：包括初始状态概率值（是一个向量）、状态转移概率（是一个方阵）、状态到观察值之间的概率（一个矩阵）。

2.2 输入输出及具体要求

首先，关于训练模型：

输入：一个或多个观察值序列，是一个或多个“打-不”串，分别表示（“打”和“不打”）。

输出：通过模型迭代计算，得到状态初始概率值向量、状态转移方阵、状态到观察值之间的概率矩阵。

第二，关于概率计算（在已经获取模型的基础上）：

输入：一个“打-不”串，分别表示（“打”和“不打”）。

输出：该串出现的概率。

第三，关于识别（在已经获取模型的基础上）：

输入：一个观察值序列，是一个“打-不”串，分别表示（“打”和“不打”）。

输出：观察值背后的状态序列，字符串用三个字符“晴”、“阴”、“雨”表示。

3 实现方法

本次实验依旧在之前实验编写的 dmarsenal 的 Python package 包内迭代开发。

关于 HMM 模型的训练采用 Baum-Welch 算法，即 EM 算法
关于模型给定观测序列预测状态（类似于语音识别）的实现
采用 Viterbi 算法，即动态规划算法。

下面列出的是几个实现的核心部分，便于在报告中查看。

3.1 图片插入示例



图 2: 聚类图像压缩测试

3.2 代码片段示例

```
1 from dmarsenal.series.HMM import HMM
2
3 state_sequence = " 阴-阴-晴-晴-阴-雨-阴-雨-晴-雨-阴-阴-雨-雨-晴-晴-阴-阴-阴"
4 obs_sequence = " 打-不-不-打-不-打-不-不-不-打-打-不-打-打-不-不-不-打-不"
5
6 state_seq = state_sequence.split(sep='-')
7 obs_seq = obs_sequence.split(sep='-')
8
9 print("Train state sequence:",state_seq)
10 print("Train observation sequence:",obs_seq)
11
12 S = list(set(state_seq))
13 O = list(set(obs_seq))
14
15 model = HMM(N=len(S), M=len(O), list_states=S, list_obs=O, random_state=None)
16
17 model.train(obs_seq, 100)
18 print("Predict probability:",model.cal_prob(obs_seq))
19 print("Predicted state sequence:",model.predict(obs_seq))
```

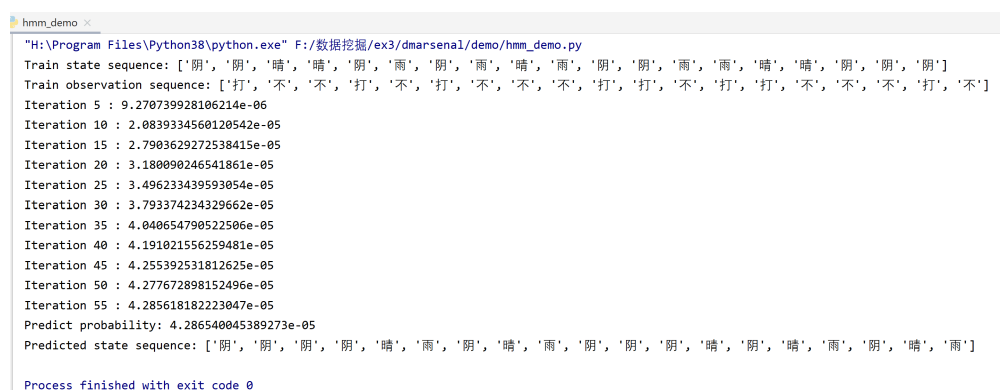
3.3 数据预处理

如上面的代码所示，实现了状态序列内部表示序号，以及序号到状态等的的数据序列转换。

4 实验结果

4.1 模型的训练、概率计算及预测

大概经过 55 次迭代后，模型对于观测序列的预测概率收敛至 $4.285618182223047e-05$ ，具体如下图所示。



```
hmm_demo x
"H:\Program Files\Python38\python.exe" F:/数据挖掘/ex3/dmarsenal/demo/hmm_demo.py
Train state sequence: ['阴', '阴', '晴', '晴', '阴', '雨', '阴', '雨', '晴', '雨', '阴', '阴', '雨', '雨', '晴', '晴', '阴', '阴', '阴']
Train observation sequence: ['打', '不', '不', '打', '不', '打', '不', '不', '打', '打', '不', '打', '不', '不', '打', '不']
Iteration 5 : 9.270739928106214e-06
Iteration 10 : 2.0839334560120542e-05
Iteration 15 : 2.7903629272538415e-05
Iteration 20 : 3.180090246541861e-05
Iteration 25 : 3.496233439593054e-05
Iteration 30 : 3.793374234329662e-05
Iteration 35 : 4.040654790522506e-05
Iteration 40 : 4.191021556259481e-05
Iteration 45 : 4.255392531812625e-05
Iteration 50 : 4.277672898152496e-05
Iteration 55 : 4.285618182223047e-05
Predict probability: 4.286540045389273e-05
Predicted state sequence: ['阴', '阴', '阴', '阴', '晴', '雨', '阴', '晴', '雨', '阴', '阴', '阴', '晴', '雨', '阴', '晴', '雨']
Process finished with exit code 0
```

图 3: HMM 模型的训练、概率计算及预测结果

4.2 模型数据预测结果

原始序列的状态:

['阴', '阴', '晴', '晴', '阴', '雨', '阴', '雨', '晴', '雨', '阴', '阴', '雨', '雨', '晴', '晴', '阴', '阴', '阴']

训练好模型的预测状态结果:

['阴', '阴', '阴', '阴', '晴', '雨', '阴', '晴', '雨', '阴', '阴', '阴', '晴', '阴', '晴', '雨', '阴', '晴', '雨']

5 心得体会

1. Viterbi 算法类似于多段图的最短路径问题，通过动态规划的思想求解出达到最优概率值隐藏状态序列。

2. Baum-Welch 算法 (EM 算法) 可以在无监督的情况下, 同时不知道状态序列和模型参数来进行 E 步和 M 步的迭代求解。并且初始的模型参数不会对于结果造成较大影响, 最终都会收敛到一个比较好的概率值
3. 书上的是关于有一个观测序列去训练 HMM 模型, 当有多个观测序列的时候, 在独立性假设下, 整个的似然值等于每个序列的概率值的乘积, 只需要在 EM 算法更新参数时进行修改为多个序列就好。参考:

<https://www.zhihu.com/question/383437650/answer/1153730978>,

<https://www.zhihu.com/question/49189633>

4. 关于在无监督学习的 Baum-Welch 算法不需要标签去训练和预测的时候隐藏状态的对应问题需要进一步学习和研究。