寒假第四周读书报告 2.24-3.1

61518426 周之遥

·读书进度

统计学习方法完成第10章：隐马尔可夫模型。

隐马尔可夫模型的特点是由一系列不可观测的状态序列生成观测序列，状态遵循马尔科夫链的特征，观测的生成只与当前状态相关。隐马尔科夫模型的三要素是一步状态转移概率矩阵、观测概率矩阵和初始概率分布。

隐马尔科夫模型模型有3个基本问题：概率计算问题、学习问题和预测问题。

概率计算问题：给定模型和观测序列O，求O出现的概率。由于直接计算法复杂度较高，使用前向算法或后向算法。另外可以进一步求得和。

学习问题：给定观测序列O，极大似然估计模型参数。状态序列I可以看作隐变量，使用EM算法迭代求参数值。又称Baum-Welch算法。

预测问题：当I已知时可以直接极大似然估计。当I未知时，使用Viterbi算法。根据动态规划中最优路径的特性，可以递推求出和。当t=T时确定最优路径，后回溯可以求出各结点值。

·提出问题

1. P196 算法10.1中第(5)步“如果t<T”是不是有问题？

·解决问题

1. P202页(10.22)如何推导？

解答：在确定一组(i,j)的情况下，后面的式子就是求出所有观测序列符合条件的概率。然后对所有（i,j）的可能求和，就是所有观测符合条件的概率。

1. 近似算法究竟哪里近似了？

解答：对每个状态单独求概率最大并将其组成的状态序列看作最优，这钟方法实际上用局部最优代替了全局最优。极端一点结果中可能会出现转移概率为零的相邻状态。

·代码实现

1. 概率计算问题和预测问题

**import** numpy **as** np  
**from** hmmlearn **import** hmm  
  
*# 3 个盒子状态*states=[**'box1'**,**'box2'**,**'box3'**]  
*# 2 个球观察状态*observations=[**'red'**,**'white'**]  
*# 初始化概率*start\_probability=np.array([0.2,0.4,0.4])  
*# 转移概率*transition\_probability = np.array([  
 [0.5, 0.2, 0.3],  
 [0.3, 0.5, 0.2],  
 [0.2, 0.3, 0.5]  
])  
  
*#发射状态概率*emission\_probability = np.array([  
 [0.5, 0.5],  
 [0.4, 0.6],  
 [0.7, 0.3]  
])  
  
*# 建立模型，设置参数*model=hmm.MultinomialHMM(n\_components=len(states))  
model.startprob\_=start\_probability  
model.transmat\_=transition\_probability  
model.emissionprob\_=emission\_probability  
  
*# 预测问题*seen=np.array([0,1,0])  
logprob,box=model.decode(seen.reshape(-1, 1),algorithm=**'viterbi'**)  
print(**'The ball picked:'**,**','**.join(map(**lambda** x:observations[x],seen)))  
print(**'The hidden box:'**,**','**.join(map(**lambda** x:states[x],box)))  
  
*# 观测序列的概率计算问题*print(np.exp(model.score(seen.reshape(-1,1))))

结果：

The ball picked: red,white,red

The hidden box: box3,box3,box3

0.13021800000000003

1. 学习问题

**import** numpy **as** np  
**from** hmmlearn **import** hmm  
  
states = [**"box1"**, **"box2"**, **"box3"**]  
n\_states = len(states)  
  
observations = [**"red"**, **"white"**]  
n\_observations = len(observations)  
  
model=hmm.MultinomialHMM(n\_components=n\_states)  
*# 三个观测序列，用来训练*X = np.array([[0,1,0,1],[0,0,0,1],[1,0,1,1]])  
model.fit(X)  
print(**'初始分布为:'**,model.startprob\_)  
print(**'转移概率为:'**,model.transmat\_)  
print(**'发射概率为:'**,model.emissionprob\_)  
*# 概率计算*print(model.score(X))

结果：

初始分布为: [9.99929344e-01 1.08304545e-09 7.06552847e-05]

转移概率为: [[0.00935718 0.69334519 0.29729762]

[0.55591308 0.22702185 0.21706507]

[0.49642482 0.23136411 0.27221106]]

发射概率为: [[0.98638429 0.01361571]

[0.11073428 0.88926572]

[0.35899209 0.64100791]]

-6.7045680827806855