第十五次作业讲评

2022-05-09

郭明非

利用ARMA进行建模

```
# TODO
# 黄楠 1900012126
P, Q = np.meshgrid(np.arange(1, 6), np.arange(1, 6))
best_aic = np.inf
best_p, best_q = 1, 1
for p, q in zip(P.flatten(), Q.flatten()):
    model = ARIMA(data, order=(p, 0, q)).fit()
    aic = model.aic
    if aic < best_aic:
        best_aic = aic
        best_p, best_q = p, q

model = ARIMA(data, order=(best_p, 0, best_q)).fit()
pred = model.predict(start=0, end=len(data) - 1)
scale = np.max((np.max(pred), np.max(data))) - np.min((np.min(data), np.min(pred)))
mse = np.mean(((data - pred) / scale) ** 2)
print(f"MSE: {mse}")

plt.plot(data, label="data")
plt.plot(pred, label="pred")
plt.legend()
plt.title(f"ARMA, MSE: {mse:.4f}")
plt.show()
```

python里的ARMA是要求AR平稳，并且阶次越高，越难保持平稳，所以可以使用较小的order。AIC和BIC只用于平稳数据。本题的数据非常少，也没有判断平稳性，所以算出来的AIC和BIC不一定适用。

HMM

```
# 韦锡宇 2000013085
#时刻t处于状态下生成观测集合元素的概率
Node1=emission_probability['Rainy']['walk']*start_probability['Rainy']\
      /(emission_probability['Rainy']['walk']*start_probability['Rainy']+emission_probability['S
Node2=emission_probability['Sunny']['walk']*start_probability['Sunny']\
      /(emission_probability['Rainy']['walk']*start_probability['Rainy']+emission_probability['S
# TODO
Nodelist=[Node1,Node2]
for k in range(2):
    ob=observations[k+1]
    node11=Node1*transition_probability['Rainy']['Rainy']\
           +Node2*transition_probability['Sunny']['Rainy']
    node12=Node1*transition_probability['Rainy']['Sunny']\
           +Node2*transition_probability['Sunny']['Sunny']
    Node1=emission_probability['Rainy'][ob]*node11\
          /(emission_probability['Rainy'][ob]*node11+emission_probability['Sunny'][ob]*node12)
    Node2=emission_probability['Sunny'][ob]*node12\
          /(emission_probability['Rainy'][ob]*node11+emission_probability['Sunny'][ob]*node12)
print(Node1,Node2)
```

由于最后一天的后向beta是1，所以可以直接使用前向算法+归一化计算最后一天天气的概率。

HMM

```
pbt = np.zeros((4, 2))

pbt[0][0], pbt[0][1] = 0.6, 0.4
for i in range(1, 4):
  for j in range(2):
      pbt[i, j] = (pbt[i-1, :]@tp[:, j])*ep[j, Ob[i-1]]

print("Rainy:",pbt[3][0]/(pbt[3][0]+pbt[3][1]), ",Sunny:",pbt[3][1]/(pbt[3][0]+pbt[3][1]))
```

前向传播的第一步需要直接用start_probability*emission_probability计算，不需要做状态转移。

维特比算法

```
# 对第二天及以后应用维特比算法
for t in range(1, len(obs)+1):
    V.append({})
    newpath = {}
    for y in states:
        #这里即为计算每一组中的最大值，即前一天阴天的的情况下今天是y的概率，以及前一天晴天时今天是y的概率中选择一个最大值
        #隐状态概率 = 前状态是y0的概率 * y0转移到y的概率 * y在t状态是为当前状态的概率
        # 韦锡宇 2000013085
        (prob, state) = max([(V[t - 1][y0] * trans_p[y0][y] * emit_p[y][obs[t-1]], y0) for y0 in states])
        # 记录最大概率
        V[t][y] = prob
        # 记录路径，这里为了方便回溯，记录回溯结果
        newpath[y] = path[state] + [y]

# 记录回溯结果
path = newpath
```

寻找最可能的隐藏状态序列。从t=1时刻开始，不断向后递推到下一个状态的路径的最大概率，直到在最后到达最终的最优路径终点，然后依据终点回溯到起始点。