

作业2: EM算法估计高斯混合模型

谭天一

ZY2203511

billytan@buaa.edu.cn

1. 实验介绍

用给定的student_height.py生成2000个身高数据，包括500个女生身高数据（前500个数据）以及1500个男生身高数据。身高数据按照高斯分布随机生成，女生组的均值为164，标准差为3；男生组的均值为176，标准差为5。

在实验时，认为这些身高数据的分布符合高斯混合模型，即由多个高斯分布加权相加得到。此处由于已知要分类的对象是男生和女生，因此认为高斯混合模型由2个高斯分布加权相加得到，即

$$p_0 N(\mu_0, \sigma_0) + p_1 N(\mu_1, \sigma_1) \quad (1)$$

首先将2000个数据划分为训练集和测试集，然后在训练集中根据EM算法求 μ_0 、 μ_1 、 σ_0 、 σ_1 、 p_0 、 p_1 的值。并用这些值在测试集中，根据身高数据预测性别。

EM算法

首先确定两个分布的初值，由于女生大多身高低，男生大多身高高，因此用极值进行初始化。用所有下标为0的变量指代女生，用所有下标为1的变量指代男生。

$$\mu_0 = \min(x), \quad \mu_1 = \max(x) \quad (2)$$

用整体的方差对两个方差数据进行初始化。

$$\sigma_0 = \sigma_1 = \sqrt{\frac{1}{n-1} \sum_{i=0}^{n-1} (x_i - \bar{x})^2} \quad (3)$$

并设置权重的初值相等。

$$p_0 = p_1 = 0.5 \quad (4)$$

E-step

$$\gamma_0 = \frac{p_0 N_0(x)}{p_0 N_0(x) + p_1 N_1(x)} \quad (5)$$

$$\gamma_1 = \frac{p_1 N_1(x)}{p_0 N_0(x) + p_1 N_1(x)} \quad (6)$$

M-step

$$n_0 = \sum_{n=0}^{n-1} \gamma_0, n_1 = \sum_{n=0}^{n-1} \gamma_1 \quad (7)$$

$$\mu_0 = \frac{\gamma_0 \cdot x}{n_0}, \mu_1 = \frac{\gamma_1 \cdot x}{n_1} \quad (8)$$

$$\sigma_0 = \sqrt{\frac{\gamma_0 \cdot (x - \mu_0)^2}{n_0}}, \sigma_1 = \sqrt{\frac{\gamma_1 \cdot (x - \mu_1)^2}{n_1}} \quad (9)$$

$$p_0 = \frac{n_0}{n}, p_1 = \frac{n_1}{n} \quad (10)$$

重复E步和M步，最终所有参数收敛。

2. 实验过程

2.1 数据加载

```
1 def load_data(filepath):
2     df = np.loadtxt(filepath, skiprows=1)
3     return df
```

2.2 高斯分布的概率密度函数

```
1 def gauss_N(x, mu, sigma):
2     pdf = 1 / np.sqrt(2 * np.pi) / sigma * np.exp(-0.5 * ((x - mu) / sigma) ** 2)
3     return pdf
```

2.3 EM算法计算

函数名为 `cal_EM(x)`

首先初始化参数。设置精度要求，并设置迭代步数计算。

```
1 # Initialize parameters
2 mu0 = np.min(x)
3 mu1 = np.max(x)
4 sigma0 = np.std(x)
5 sigma1 = sigma0
6 p0 = 0.5
7 p1 = 0.5
8 n = len(x)
9 # Set Precision
10 accuracy = 0.0000001
11 # Step count
12 step_count = 0
```

进入循环求解阶段，对于每个循环，首先记录迭代步数，然后记录上一轮的结果。并输出每一步的计算结果用于追踪数据收敛过程。

```

1     while True:
2         step_count += 1
3         print(step_count, ': ', mu0, mu1, sigma0, sigma1, p0, p1)
4         # Previous Memory
5         p0_pre = p0
6         p1_pre = p1
7         mu0_pre = mu0
8         mu1_pre = mu1
9         sigma0_pre = sigma0
10        sigma1_pre = sigma1

```

E-step和M-step计算。

```

1         # E-STEP
2         gamma0 = p0 * gauss_N(x, mu0, sigma0) / (p0 * gauss_N(x, mu0, sigma0) + p1
* gauss_N(x, mu1, sigma1))
3         gamma1 = 1 - gamma0
4         # M-STEP
5         n0 = np.sum(gamma0)
6         n1 = n - n0
7         mu0 = gamma0.dot(x) / n0
8         mu1 = gamma1.dot(x) / n1
9         sigma0 = np.sqrt(gamma0.dot((x - mu0) ** 2) / n0)
10        sigma1 = np.sqrt(gamma1.dot((x - mu1) ** 2) / n1)
11        p0 = n0 / n
12        p1 = n1 / n

```

分别从精度、最大步数、异常结果三个方面判断是否跳出循环。

```

1         # End loop judgment
2         if abs(sigma1_pre - sigma1) < accuracy \
3             and abs(sigma0_pre - sigma0) < accuracy \
4             and abs(p0_pre - p0) < accuracy \
5             and abs(p1_pre - p1) < accuracy \
6             and abs(mu0_pre - mu0) < accuracy \
7             and abs(mu1_pre - mu1) < accuracy:
8             print("结束迭代: 达到预定精度")
9             break
10        if step_count > 100000:
11            print("结束迭代: 达到最大步数")
12            break
13        if math.isnan(mu0) or math.isnan(mu1) or math.isnan(sigma0) or
math.isnan(sigma1) or math.isnan(
14            p0) or math.isnan(p1):
15            print("结束迭代: 出现NaN")
16            break

```

2.4 训练与测试

```
1 def test(test_data, mu0, mu1, sigma0, sigma1):
2     probability_girl = gauss_N(test_data, mu0, sigma0)
3     probability_boy = gauss_N(test_data, mu1, sigma1)
4     considered_boy = 0
5     considered_girl = 0
6     for i in range(len(probability_girl)):
7         if probability_girl[i] >= probability_boy[i]:
8             considered_girl += 1
9         else:
10            considered_boy += 1
11    return considered_boy, considered_girl
```

用概率密度函数作为判断标准，对于测试集中的每个数据，如果属于男生高斯分布的概率密度大于属于女生的高斯分布，则判断为男生；反之亦然。

在划分测试集时，根据7:3的比例进行划分。并最终计算识别成功率。

```
1 # 划分测试集和训练集，比例7:3
2 girl = data[:500]
3 boy = data[500:]
4 girl_train = girl[:350]
5 girl_test = girl[350:]
6 boy_train = boy[:1050]
7 boy_test = boy[1050:]
8 data_train = np.concatenate([girl_train, boy_train])
9 # 用训练集进行训练
10 mu0, mu1, sigma0, sigma1, p0, p1 = cal_EM(data_train)
11 print("女生：均值=", mu0, "；标准差=", sigma0, "；权重=", p0)
12 print("男生：均值=", mu1, "；标准差=", sigma1, "；权重=", p1)
13 # 用测试集进行测试
14 girl_incorrect_identify, girl_correct_identify = test(girl_test, mu0, mu1, sigma0,
15                                                       sigma1)
16 boy_correct_identify, boy_incorrect_identify = test(boy_test, mu0, mu1, sigma0,
17                                                       sigma1)
18 girl_classification_accuracy = ('%.2f' % (girl_correct_identify / len(girl_test) *
19                                           100))
20 boy_classification_accuracy = ('%.2f' % (boy_correct_identify / len(boy_test) *
21                                           100))
22 print("女生测试集，识别正确：", girl_correct_identify, '/', len(girl_test), "正确率：",
23       girl_classification_accuracy, '%')
24 print("男生测试集，识别正确：", boy_correct_identify, '/', len(boy_test), "正确率：",
25       boy_classification_accuracy, '%')
```

3. 实验结果

```
172 : 164.17492981867997 176.22515519323196 2.931055520960746 4.975264011661764 0.26466604662105886 0.7353339533789411
173 : 164.17492971623577 176.2251551003724 2.9310554640315143 4.975264074047932 0.26466603870449784 0.7353339612955022
结束迭代：达到预定精度
女生：均值= 164.17492962338227 ; 标准差= 2.9310554124319337 ; 权重= 0.26466603152907847
男生：均值= 176.22515501620614 ; 标准差= 4.9752641305935565 ; 权重= 0.7353339684709215
女生测试集，识别正确： 144 / 150 正确率： 96.00 %
男生测试集，识别正确： 412 / 450 正确率： 91.56 %
```

可见程序收敛速度很快，在精度0.0000001时仅用173次迭代即收敛。女生的真实参数为均值164，标准差3，权重0.25；男生的真实参数为均值176，标准差5，权重0.75。可见EM算法对参数的估计与真实参数很接近。在测试集中的测试结果也显示，基于EM算法估计数据的模型在150名女生中成功识别144名，正确率96%；在450名男生中成功识别了412名，正确率91.56%。该结果较为理想。

附录：全部代码

```
1  # @Author: 谭天一
2  # @Date: 2023-4-9
3
4  import math
5  import numpy as np
6
7
8  def load_data(filepath):
9      df = np.loadtxt(filepath, skiprows=1)
10     return df
11
12
13 def gauss_N(x, mu, sigma):
14     pdf = 1 / np.sqrt(2 * np.pi) / sigma * np.exp(-0.5 * ((x - mu) / sigma) ** 2)
15     return pdf
16
17
18 def cal_EM(x):
19     # Initialize parameters
20     mu0 = np.min(x)
21     mu1 = np.max(x)
22     sigma0 = np.std(x)
23     sigma1 = sigma0
24     p0 = 0.5
25     p1 = 0.5
26     n = len(x)
27     # Set Precision
28     accuracy = 0.0000001
29     # Step count
30     step_count = 0
31     while True:
32         step_count += 1
33         print(step_count, ': ', mu0, mu1, sigma0, sigma1, p0, p1)
34         # Previous Memory
35         p0_pre = p0
```

```

36     p1_pre = p1
37     mu0_pre = mu0
38     mu1_pre = mu1
39     sigma0_pre = sigma0
40     sigma1_pre = sigma1
41     # E-STEP
42     gamma0 = p0 * gauss_N(x, mu0, sigma0) / (p0 * gauss_N(x, mu0, sigma0) + p1
* gauss_N(x, mu1, sigma1))
43     gamma1 = 1 - gamma0
44     # M-STEP
45     n0 = np.sum(gamma0)
46     n1 = n - n0
47     mu0 = gamma0.dot(x) / n0
48     mu1 = gamma1.dot(x) / n1
49     sigma0 = np.sqrt(gamma0.dot((x - mu0) ** 2) / n0)
50     sigma1 = np.sqrt(gamma1.dot((x - mu1) ** 2) / n1)
51     p0 = n0 / n
52     p1 = n1 / n
53     # End loop judgment
54     if abs(sigma1_pre - sigma1) < accuracy \
55         and abs(sigma0_pre - sigma0) < accuracy \
56         and abs(p0_pre - p0) < accuracy \
57         and abs(p1_pre - p1) < accuracy \
58         and abs(mu0_pre - mu0) < accuracy \
59         and abs(mu1_pre - mu1) < accuracy:
60         print("结束迭代: 达到预定精度")
61         break
62     if step_count > 100000:
63         print("结束迭代: 达到最大步数")
64         break
65     if math.isnan(mu0) or math.isnan(mu1) or math.isnan(sigma0) or
math.isnan(sigma1) or math.isnan(
66         p0) or math.isnan(p1):
67         print("结束迭代: 出现NaN")
68         break
69     return mu0, mu1, sigma0, sigma1, p0, p1
70
71
72 def test(test_data, mu0, mu1, sigma0, sigma1):
73     probability_girl = gauss_N(test_data, mu0, sigma0)
74     probability_boy = gauss_N(test_data, mu1, sigma1)
75     considered_boy = 0
76     considered_girl = 0
77     for i in range(len(probability_girl)):
78         if probability_girl[i] >= probability_boy[i]:
79             considered_girl += 1
80         else:
81             considered_boy += 1
82     return considered_boy, considered_girl

```

```
83
84
85 path = './height_data.csv'
86 data = load_data(path)
87 # 划分测试集和训练集, 比例7:3
88 girl = data[:500]
89 boy = data[500:]
90 girl_train = girl[:350]
91 girl_test = girl[350:]
92 boy_train = boy[:1050]
93 boy_test = boy[1050:]
94 data_train = np.concatenate([girl_train, boy_train])
95 # 用训练集进行训练
96 mu0, mu1, sigma0, sigma1, p0, p1 = cal_EM(data_train)
97 print("女生: 均值=", mu0, "; 标准差=", sigma0, "; 权重=", p0)
98 print("男生: 均值=", mu1, "; 标准差=", sigma1, "; 权重=", p1)
99 # 用测试集进行测试
100 girl_incorrect_identify, girl_correct_identify = test(girl_test, mu0, mu1, sigma0,
101                                                       sigma1)
101 boy_correct_identify, boy_incorrect_identify = test(boy_test, mu0, mu1, sigma0,
102                                                       sigma1)
102 girl_classification_accuracy = ('%.2f' % (girl_correct_identify / len(girl_test) *
103                                           100))
103 boy_classification_accuracy = ('%.2f' % (boy_correct_identify / len(boy_test) *
104                                           100))
104 print("女生测试集, 识别正确: ", girl_correct_identify, '/', len(girl_test), "正确率: ",
105       girl_classification_accuracy, '%')
105 print("男生测试集, 识别正确: ", boy_correct_identify, '/', len(boy_test), "正确率: ",
106       boy_classification_accuracy, '%')
```