深度学习与自然语言处理课程报告
——预测男女身高的高斯混合模型推导

姓名：龙行健

学号：ZY2203110

一、 摘要

高斯混合模型（Gaussian Mixture Model）通常简称 GMM，是一种业界广泛使用的聚类算法，该方法使用了高斯分布作为参数模型，并使用了期望最大（Expectation Maximization，简称 EM）算法进行训练。本文对其原理进行了简单的叙述，同时基于一组由两个不同高斯参数模型生成的混合男女身高数据集，应用 GMM 进行参数分离和预测，结果表明不同的初始参数对于预测的结果存在一定影响，GMM 收敛速度较快，是一种良好的聚类模型。

二、 引言

对于单分布生成的数据集，若知道其生成分布的表达形式，我们很容易通过最大似然估计、最小方差估计等办法算出其未知参数。但现实中的数据集往往并不是由单一的分布就可以生成，在总体分布中可能含有若干个子分布存在，这导致通常的办法没办法对整个样本的分布状况形成良好的解释。GMM 不要求观测数据提供关于子分布的具体信息，就可以计算观测数据在总体分布中的概率，其对于含有隐变量的分布情况有更佳的解释效应。

三、 实验原理

1. 问题分布形式

假设 y 是身高，θ_k 是第 k 个高斯模型的参数，即 $\theta_k = (\mu_k, \sigma_k)$，对于该问题，有如下的具体概率表达式：

$$P(y|\theta) = \alpha_1 \phi(y|\theta_1) + \alpha_2 \phi(y|\theta_2)$$

其中，α_1,α_2代表男女生的概率，有约束$\alpha_1+\alpha_2=1$，$\phi(y\mid\theta_k)$为高斯分布，有：

$$\phi(y\mid\theta_k)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(y-y_k)^2}{2\sigma_k^2}\right)$$

2. 明确隐藏变量

隐变量很好确认，即为某个身高来自哪个高斯模型，为未知参数，我们以γ_{jk}表示：

$$\gamma_{jk}=\begin{cases}1,\text{If j}^{th}\text{ height is from k}^{th}\text{ model}\\ 0,\text{else}\end{cases},j=1,\ldots,N$$

其中，N 代表一共有 N 组身高测量数据，k 最高为 2，因为目前只有男女两组模型。

3. 确定似然函数

要优化的似然函数即为每个数据在某个参数分布下的乘积，如下所示：

$$\begin{aligned}L(\theta)&=\prod_{j=1}^{N}P\left(y_j,\gamma_{j1},\gamma_{j2}\mid\theta\right)\\&=\prod_{k=1}^{2}\prod_{j=1}^{N}\left[\alpha_k\phi\left(y_j\mid\theta_k\right)\right]^{\gamma_{jk}}\\&=\prod_{k=1}^{2}\alpha_k^{n_k}\prod_{j=1}^{N}\left[\phi\left(y_j\mid\theta_k\right)\right]^{\gamma_{jk}}\\&=\prod_{k=1}^{2}\alpha_k^{n_k}\prod_{j=1}^{N}\left[\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(y-y_k)^2}{2\sigma_k^2}\right)\right]^{\gamma_{jk}}\end{aligned}$$

其中，$n_k=\sum_{j=1}^{N}\gamma_{jk}$，$\sum_{j=1}^{2}n_k=N$。这是这是显然的，因为对于某一个数据$j$来说，其所属类别不是$\gamma_{j1}$就$\gamma_{j2}$是 ，所以所有的$\gamma$求和即为数据的总数$N$。

对似然函数进行对数化，则有：

$$\log L(\theta)=\sum_{j=1}^{2}\left\{n_k\log\alpha_k+\sum_{j=1}^{N}\gamma_{jk}\left[\log\left(\frac{1}{\sqrt{2\pi}}\right)-\log\sigma_k-\frac{1}{2\sigma_k^2}\left(y_j-\mu_k\right)^2\right]\right\} \tag{1}$$

将其对各个参变量进行求导，并令其等于零，就可以得到每个参变量的极大值，用以在下一轮迭代中运用。

4. 确定隐变量的期望分布，EM 中的 E 步

为了让对数似然函数可以最大化，首先要估计隐藏变量的估计值，如下所示：

$$\begin{aligned}
\hat{\gamma}_{jk} &= P\left(\gamma_{jk}=1 \mid y,\theta\right) \\
&= \frac{P\left(\gamma_{jk}=1, y_j \mid \theta\right)}{P\left(\gamma_{j1}=1, y_j \mid \theta\right)+P\left(\gamma_{j2}=1, y_j \mid \theta\right)} \\
&= \frac{P\left(y_j \mid \gamma_{jk}=1,\theta\right)P\left(\gamma_{jk}=1 \mid \theta\right)}{P\left(y_j \mid \gamma_{j1}=1,\theta\right)P\left(\gamma_{j1}=1 \mid \theta\right)+P\left(y_j \mid \gamma_{j2}=1,\theta\right)P\left(\gamma_{j2}=1 \mid \theta\right)} \\
&= \frac{\alpha_k \phi\left(y_j \mid \theta_k\right)}{\alpha_1 \phi\left(y_j \mid \theta_1\right)+\alpha_2 \phi\left(y_j \mid \theta_2\right)}
\end{aligned} \tag{2}$$

5. 确定新一轮迭代模型的参数，EM 中的 M 步：

将公式(2)代入到公式(1)中，并对每个模型参数求导可以得到：

$$\hat{\mu}_k = \frac{\sum_{j=1}^{2}\hat{\gamma}_{jk} y_j}{\sum_{j=1}^{2}\hat{\gamma}_{jk}}, k=1,2$$

$$\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{2}\hat{\gamma}_{jk}\left(y_j-\mu_k\right)^2}{\sum_{j=1}^{2}\hat{\gamma}_{jk}}, k=1,2$$

$$\hat{\alpha}_k = \frac{n_k}{N} = \frac{\sum_{j=1}^{2}\hat{\gamma}_{jk}}{N}, k=1,2$$

将得到的参数重新代入回到(2)重新计算，直到收敛即可得到最后的结果。

四、 实验过程

1. 实验环境

带有基础科学运算的 python 环境

2. 数据来源

在确定高斯分布下模拟生成的数据集：

```
# 定义高斯分布的参数
mean1, std1 = 164, 3
mean2, std2 = 176, 5

# 从两个高斯分布中生成各 50 个样本
data1 = np.random.normal(mean1, std1, 500)
data2 = np.random.normal(mean2, std2, 1500)
data = np.concatenate((data1, data2), axis=0)
np.random.shuffle(data)
```

将样本数据统计频次，画出其直方图，可得：

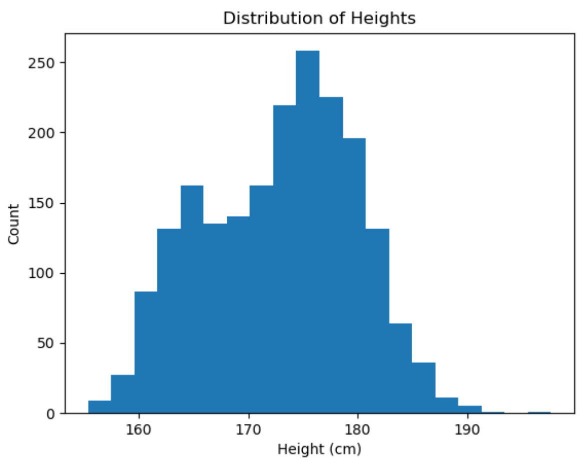

图 1.身高数据直方图分布统计图

从图 1.中可以看出，其并不由一个高斯分布生成，这说明必须采用含有隐变量计算形式的 GMM 等模型对其进行预测。

3. 数据预处理

这部分主要是读取数据集、初始化生成θ,α,γ等参数的存储空间、定义运算精度ε。

```
    with open(file_locaton, "r") as f:
        reader = csv.DictReader(f)
        tempt_h = [row['height'] for row in reader]

    #Read height to data_h
    tempt_h = np.array(tempt_h)
    N = np.size(tempt_h, 0)
```

```
    data_h = np.arange(N, dtype=float)
    for i in range(N):
        data_h[i] = float(tempt_h[i])
    K =  2

    #Initial parameters
    theta = np.array([[170.,5],[155,5]])
    alpha = np.array([0.5, 0.5])
    epi = 1e-4
    print(theta)
    L = 0
    count = 0
    gama = np.arange(N*K,dtype = float).reshape(K,N)
```

4. 更新 γ_{jk}

将公式转化为 python 语言即可：

```
        #update gama
        for k in range(K):
            for j in range(N):
                tempt = 0
                for p in range(K):
                    tempt += alpha[p]*GaussProb(data_h[j], theta[p][0],
math.sqrt(theta[p][1]))
                gama[k][j] = alpha[k]*GaussProb(data_h[j], theta[k][0],
math.sqrt(theta[k][1])) / tempt
```

5. 更新 $\hat{\mu}_k, \hat{\sigma}_k^2, \hat{\alpha}_k$

将公式转化为 python 语言即可：

```
        #update miu
        for k in range(K):
            tempt1 = 0
            tempt2 = 0
            for j in range(N):
                tempt1 += gama[k][j] * data_h[j]
                tempt2 += gama[k][j]
            theta[k][0] = tempt1 / tempt2

        #update sig
        for k in range(K):
```

```
            tempt1 = 0
            tempt2 = 0
            for j in range(N):
                tempt1 += gama[k][j] * (data_h[j] - theta[k][0])**2
                tempt2 += gama[k][j]
            theta[k][1] = tempt1 / tempt2

        #update alpha
        for k in range(K):
            tempt1 = 0
            for j in range(N):
                tempt1 += gama[k][j]
            alpha[k] = tempt1 / N
```

五、 实验结果

给定精度或者给定固定迭代次数进行计算，在不同的初始参数下，得到如图 2.的结果。

```
[[170.    5.]
 [170.    5.]]
[[172.85232236   6.92707644]
 [172.85232236   6.92707644]]
```

(a)

```
[[150.    5.]
 [150.    5.]]
[[172.85232236   6.92707644]
 [172.85232236   6.92707644]]
```

(b)

```
[[130.    5.]
 [190.    5.]]
[[163.97131191   3.02183854]
 [175.94689127   4.94604231]]
```

(c)

```
[[170.    5.]
 [155.    5.]]
[[175.94948184   4.94436287]
 [163.97450386   3.02357124]]
```

(d)

图 2.GMM 不同初始参数和预测结果

结果表明，当初始参数取得相近，最终预测得到的结果有较大的差别，而当初始参数取得较远，预测的结果偏差较小，这表明 GMM 算法对于初始参数的选取较为敏感。这有可能是因为，GMM 算法容易陷入极小值点，要改进模型的性能可以考虑当有可能陷入极小值点的时候，加入随机方向的参数矢量，使得模型参数有机会跃出局部极值点。

六、 结论

上述实验过程和结果表明，GMM 在初始参数选取得当的情况下，是一个较好的混合聚类模型，对于非单分布可以解释的数据集有较好的预测结果。

七、 参考文献

[1]. 高斯混合模型（GMM） - 知乎 (zhihu.com)