**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

# 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。

# 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。

# 实验原理

## 问题分布形式

假设是身高，是第个高斯模型的参数，即，对于该问题，有如下的具体概率表达式：



其中，代表男女生的概率，有约束，为高斯分布，有：



## 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，我们以表示：



其中，代表一共有N组身高测量数据，最高为2，因为目前只有男女两组模型。

## 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：



其中，，。这是这是显然的，因为对于某一个数据来说，其所属类别不是就是 ，所以所有的求和即为数据的总数。

对似然函数进行对数化，则有：

 (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。

## 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：

 (2)

## 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：



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。

# 实验过程

## 实验环境

带有基础科学运算的python环境

## 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

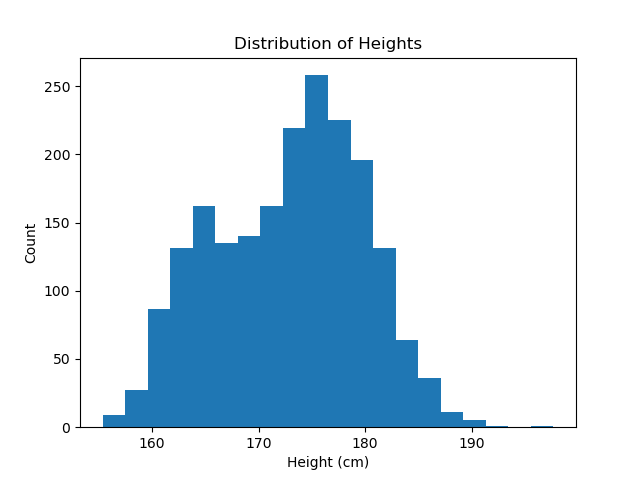
data1 = np.random.normal(mean1, std1, 500)

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。

## 数据预处理

这部分主要是读取数据集、初始化生成等参数的存储空间、定义运算精度。

    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)

## 更新

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

## 更新

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

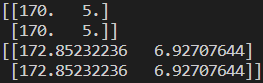
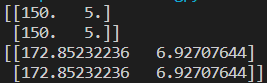
            for j in range(N):

                tempt1 += gama[k][j]

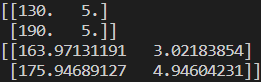
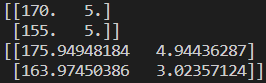
            alpha[k] = tempt1 / N

# 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。

1. (b)

(c) (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。

## 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)](https://zhuanlan.zhihu.com/p/30483076)