哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称： 机器学习
课程类型： 必修
实验题目： K-means和混合高斯模型

学号：1160300507
姓名： 聂晨晞

一.实验目的

1. 实现一个K-means算法和混合高斯模型，并且使用EM算法估计它的参数

二、实验要求及实验环境

实验要求：

1. 用K-means聚类，测试效果
2. 使用混合高斯模型和你实现的EM算法估计参数，观察每次迭代之后似然值的变化情况，考察EM算法是否可以得到正确的结果。

3. 在UCI上下载一个简单的数据集，使用GMM进行聚类

实验环境：

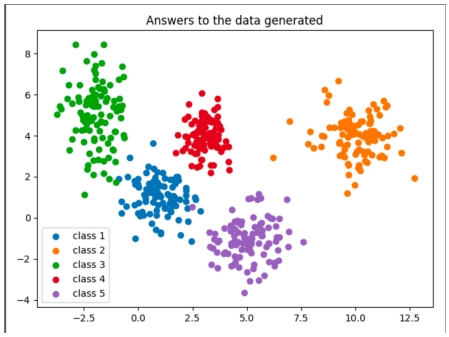
1. 硬件环境：Macbook pro 3.1Ghz i5 + 16G内存
2. 软件环境：MacOS 10.14 + python 3.6.5

三、设计思想（本程序中的用到的主要算法及数据结构）

1．算法原理

1. K-means算法原理

假设数据集 $\mathbf{X} = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$ 为本实验需要用到的数据集，其中$\bar{x}_i$表示第i个数据，每一个数据有m维，并且该数据集被分为k类，下面讨论在数据集$\mathbf{X}$上的K-means聚类算法。



如左图所示是数据集$\mathbf{X}$在标注为k = 5以及数据维度m = 2上的可视化表示，很明显，该数据被分为5类，K-means算法的优化任务就是通过某种机器学习的方法，在没有着色的左图中将左图的数据集分为5类。

由于K-means算法是一种特殊的EM算法，所以在本节中我们略去对K-means算法的数学原理的讨论，转而直接叙述K-means算法的步骤。

1. 首先，随机的（或者具有某种策略的）选择某k个点作为“样本中心”，记为集合$\mathbf{C} = \{\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_k\}$,这之中的每一个向量$\bar{c}_i$被称作第i类的样本中心，然后K-means算法通过不停迭代的两步来完成对该数据的聚类工作。
2. 首先，K-means算法扫描整个数据集，对于每一个数据点$\bar{x}_i$计算它和每一个数据中心$\bar{c}_i$的距离（在本例中使用欧氏距离），找到距离该数据点最近的数据中心，然后我们将该点标记为“数据中心$\bar{c}_i$的随从点”。于是执行完第二步之后，我们就完成了将数据集上所有的点都分配给了某一个特定的数据中心的操作，即每一个数据中心

都有了若干个点，记这些点构成的集合为$\mathbf{Class_i} = \{\bar{x}_{i1}, \bar{x}_{i2}, \ldots \bar{x}_{im}\}$其中m表示数据中心i下属的数据点的个数

3. 其次，对于每一个数据中心的数据集合$\mathbf{Class_i}$来说，我们通过这些数据对真实数据中心进行重新的估计，对于K-means算法来说，就是简单的计算该数据集合中所有点，对每一个维度的平均值，即

$$C_{i_{new}} = \frac{\sum_j^m \bar{x}_{ij}}{m}$$

这样我们就有了一组新的数据中心集合，然后我们将这一组新的数据中心集合带入到步骤2就可以得到在这样一个新的数据中心集合下每一个数据中心的新的“下属数据点”于是我们使用新的下属数据点又可以根据步骤3得到新的数据中心聚合。

如果我们重复执行第2步和第3步，直到新产生的数据中心集合和上一次的数据中新集合的差距足够小，我们就说该算法收敛，并且得到了最后的聚类结构

2. EM算法原理

EM算法相对于K-means算法更加复杂，下面结合混合高斯模型，对它的数学原理进行稍微详细一些的叙述。

2.1 混合高斯模型

设有随机变量$\mathbf{X}$，那么混合高斯模型可以用下式表示

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

其中$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$被称为混合模型中的第k个分量。$\pi_k$表示混合系数，并且满足:

$$\sum_{k=1}^{K} \pi_k = 1$$

$$0 \leq \pi_k \leq 1$$

可以看到π_k相当于每个分量$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$的权重

2.2 GMM的参数估计过程

2.2.1GMM的贝叶斯理解

在介绍GMM参数估计之前，我们先改写GMM的形式，改写之后的GMM，模型可以方便的使用EM估计参数。GMM的原始形式如下所示

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

前面提到π_k可以看成是第k类被选中的概率。我们引入一个新的K维随机变量z，$z_k(1 < \mathrm{k} < \mathrm{K})$只能取0或1两个值；$z_k = 1$表示第k类被选中的概率，即$P(z_k = 1) = \pi_k$；$z_k = 0$表示第k类没有被选中的概率，更数字化一点，$z_k$需要满足以下两个条件

$$z_k \in \{0,1\}$$

$$\sum_k z_k = 1$$

我们假设z_k之间是独立同分布的，那么我们可以写出z的联合概率分布形式：

$$p(\boldsymbol{z}) = p(z_1)p(z_2)\ldots p(z_K) = \prod_{k=1}^{K} \pi_k^{z_k}$$

对于混合高斯模型来说，我们有第k类数据服从高斯分布，

$$p(\boldsymbol{x} \,|\, z_k = 1) = \mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

进而上式可以写成如下的形式

$$p(\boldsymbol{x} \,|\, \boldsymbol{z}) = \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}$$

上面分别给出了$p(\boldsymbol{z})$和$p(\boldsymbol{x} \,|\, \boldsymbol{z})$，于是根据全概率公式，我们可以得到$p(\boldsymbol{x})$的取值如下所示：

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{\boldsymbol{z}} p(\boldsymbol{z}) p(\boldsymbol{x}|\boldsymbol{z}) \\ &= \sum_{i=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \ (z_k = 0\text{的项为}1\text{，省略}) \end{aligned}$$

知道了以上的几个等式之后，我们可以计算在贝叶斯公式的理解下的后验概率$p(\boldsymbol{z}|\boldsymbol{x})$如下

$$\begin{aligned}\gamma(z_k) &= p(z_k = 1|\boldsymbol{x}) \\ &= \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{p(\boldsymbol{x}, z_k = 1)} \\ &= \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\boldsymbol{x}|z_j = 1)} \text{ (Full probability formula)} \\ &= \frac{\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu_j}, \boldsymbol{\Sigma_j})} \text{ (combining (3)(4))}\end{aligned}$$

上式中我们定义$\gamma(z_k)$来表示第k个分量的后验概率。在贝叶斯的观点下，π_k可以视为$z_k = 1$的后验概率。

上述内容改写了GMM的形式，引入了隐变量z和一只x之后的后验概率$\gamma(z_k)$，这样做是为了方便后面的EM算法来估计GMM的参数

2.2.2 使用EM算法来估计的参数

EM算法分两步，第一步先求出要估计参数的粗略值，第二部使用第一步的值最大化似然函数，因此要求出GMM的似然函数。

假设$\mathbf{x} = \{\bar{x}_1, \ldots \bar{x}_n\}$,GMM的概率模型如上一节讨论所示，那么，我们知道这里一共有π, μ, Σ等三个参数需要估计，于是我们将该式重新改写一下，写成如下的形式：

$$p(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

为了估计这三个参数，需要分别求出这三个参数的最大似然函数，先求解μ_k的最大的似然函数，对上式取对数后再对

μ_k求导，并且令导数等于零，得到如下的等式

$$0 = -\sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k (\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

注意到上式中分数的一项正好是上一节中讨论的后验概率的形式，两边同乘Σ_k^{-1},重新整理，我们可以得到如下的等式:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n$$

其中

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

同理我们求$\mathbf{\Sigma_k}$的最大似然函数，可以得到:

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \boldsymbol{\mu}_k)(x_n - \boldsymbol{\mu}_k)^T$$

最后剩下π_k的最大似然函数，注意到π_k有限制条件

$\sum_{k=1}^{K} \pi_k = 1$,因此我们需要加入拉格朗日算子如下:

$$\ln p(\boldsymbol{x} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda (\sum_{k=1}^{K} \pi_k - 1)$$

求上式关于π_k的最大似然函数，得到如下所示:

$$0 = \sum_{n=1}^{N} \frac{\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda$$

上式两边同乘以π_k,可以得到$\mu = -N$,进而可以得到π_k更简洁的表达形式。

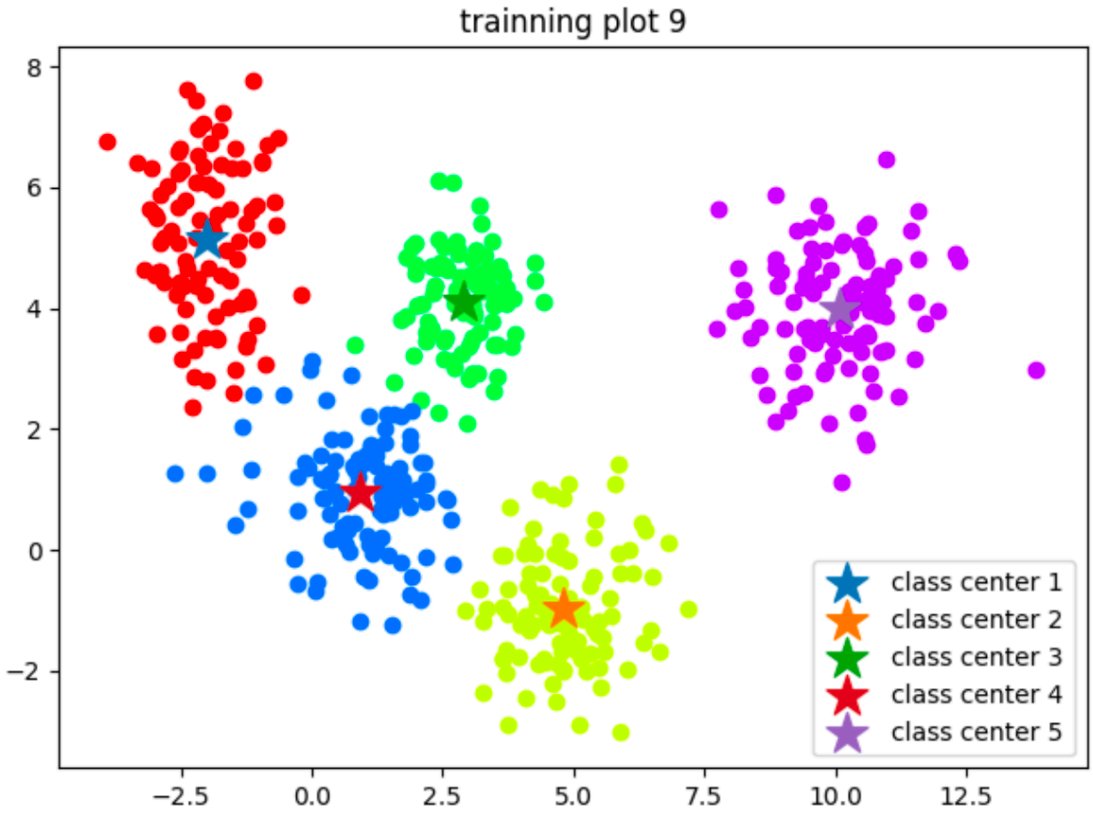
$$\pi_k = \frac{N_k}{N}$$

EM算法估计GMM参数即最大化以上三个似然函数，我们先指定π, μ, Σ的初值，带入后验概率公式中计算出后验概率（E步），然后将后验概率分别带入相应的似然函数进行计算新的一组π, μ, Σ，在代入计算后验概率，在得到新的一组

π, μ, Σ（M步），在计算后验概率，如此往复，知道算法收敛

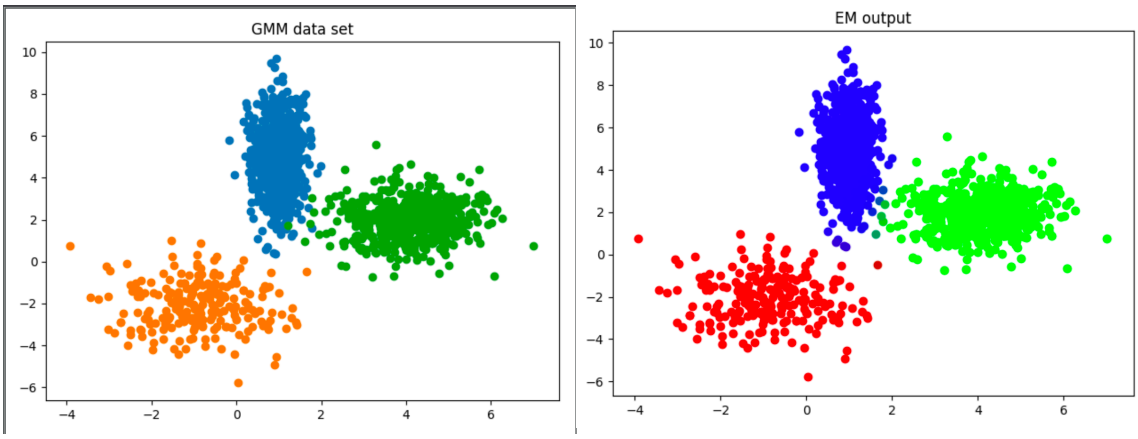
四、实验结果与分析

对于K-means算法，在产生的如上一节中的数据集中，在聚类算法完成之后，我们得到了如下的结果



显然该答案和我们在上一节中产生的数据差不多，为了进一步测试本程序，我们对UCI上的鸢尾花数据集进行了K-means聚类算法，最后得到的准确度为0.89

对于EM算法，我使用了相同的方法产生了三组数据，隐变量的取值分别为0.5，0.2，0.3，使用EM算法估计得到的隐变量在收敛之后为0.52.0.35，0.13， 然后得到的聚类和答案分别如下所示，左边是答案，右边是EM算法的输出，如图所示，说明我们的EM算法还是很成功的



在UCI数据集中运行EM算法对于三个不同的类，正确率分别文0.9，1，1，可以看到EM算法比K-means算法更加准确，但是运行的时间也同样更长一些。

五、结论

在本文中我们实现了EM算法和K-means算法并且对于算法的准确度进行了初步的测量，得到的结论是EM算法比K-means算法更加准确

六、参考文献

【1】EM算法介绍https://blog.csdn.net/jinping_shi/article/details/59613054

七、附录：源代码（带注释）

源代码在https://github.com/Billy-Nie/machine_learning_HIT 这个github仓库中可以找到