哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

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 聚类和 GMM

学号：1171800323

姓名： 杨富祥

一、实验目的

实现一个 k-means 算法和混合高斯模型，并且用 EM 算法估计模型中的参数。

二、实验要求及实验环境

实验要求：

用高斯分布产生 k 个高斯分布的数据（不同均值和方差）（其中参数自己设定）。

（1）用 k-means 聚类，测试效果；

（2）用混合高斯模型和你实现的 EM 算法估计参数，看看每次迭代后似然值变化情况，考察 EM 算法是否可以获得正确的结果（与你设定的结果比较）。

应用：可以 UCI 上找一个简单问题数据，用你实现的 GMM 进行聚类。

实验环境： Windows10 + matlabR2018a

三、设计思想

1．算法原理

EM-算法：是常用的估计参数隐变量的利器，它是一种迭代式的方法。其基本思想是：若参数θ已知，则可根据训练数据推断出最优隐变量 Z 的值（E 步）；反之，若 Z 的值已知，则可以方便地对参数θ做极大似然估计（M 步）。

于是，以初始值θ^0为起点，可以迭代执行以下步骤直至收敛：

1）基于θ^t推断隐变量 Z 的期望，记为Z^t；

2）基于以观测变量X和Z^t对参数θ做极大似然估计，记为θ^{t+1}；

这就是 EM 算法的原型。本次实验两个算法都是 EM 算法的应用。

K-means 算法：给定样本集$D = \{x_1, x_2, \ldots, x_m\}$，k 均值算法针对聚类所得簇划分$C = \{C_1, C_2, \ldots, C_k\}$最小化平方误差

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||_2^2$$

其中$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$是簇$C_i$的均值向量。上式刻画了簇内样本围绕簇均值向量的紧密程度，E 值越小则簇内样本相似度越高。

如果暴力求解，找到它的最优解需考察样本集 D 所有可能的簇划分，这是一个 NP 难问题。因此，可以采用 EM 算法优化。

优化目标可写为：$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$。

1）固定μ，优化C（E 步）

$$\sum_{j=1}^{m} \min_{i=1,2,\ldots,k} ||\mu_{C_i} - x_j||^2$$

对每一个样本点x_j，找到离它最近的均值向量μ_{C_i}，从而确定样本点的簇标记i，将其划分入相应的簇C_i.

2）固定C，优化μ（M 步）

$$\sum_{i=1}^{k} min \sum_{x \in C_i} ||x - \mu_i||_2^2$$

有了簇划分，只需计算簇C_i的均值向量作为新的均值向量μ即可。

不断重复 1）2）两步，直到均值向量不再变化则算法收敛。

多元高斯分布：对 n 维样本空间X中的随机向量x，若x服从高斯分布，其概率密度函数为

$$p(x|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

其中μ是 n 维均值向量，Σ是$n \times n$的协方差矩阵，高斯分布完全由均值向量μ和协方差矩阵Σ这两个参数决定。

GMM：高斯混合模型聚类采用概率模型来表达聚类原型。

高斯混合模型有 k 个混合成分，成分 i 对应一个高斯分布。第 i 个高斯混合成分的参数是均值向量μ_i和协方差矩阵Σ_i.

假设样本的生成过程由高斯混合分布给出，每一个数据根据以下步骤产生：

1）随机选择一个高斯混合成分，选择成分 i 的概率是$P(y = i)$；

2）$x \sim N(\mu_i, \Sigma_i)$，根据被选择的高斯混合成分的概率密度函数进行采样，生成相应的样本。

从而我们知道$p(x|y = i) \sim N(\mu_i, \Sigma_i)$，故可以得到高斯混合分布的概率密度函数

$$p(x) = \sum_{i=1}^{k} P(x|\mu_i, \Sigma_i) P(y = i)$$

其中$P(y = i)$是混合比例，$P(x|y = i)$是高斯混合成分。

若训练集$D = \{x_1, x_2, \ldots, x_m\}$由上述过程生成，随机变量$y \in \{1,2,\ldots,k\}$表示生成样本$x_j$的高斯混合成分，其取值未知。$y$的先验概率是$P(y = i)$，记为$\alpha_i$. 根据贝叶斯定理，$y$的后验分布对应于

$$p(y=i|x_j)=\frac{P(y=i)p(x_j|y=i)}{p(x_j)}$$
$$=\frac{\alpha_i p(x_j|\mu_i,\Sigma_i)}{\sum_{l=1}^{k}\alpha_l p(x_j|\mu_l,\Sigma_l)}$$

$p(y=i|x_j)$给出了样本x_j由第 i 个高斯混合成分生成的后验概率，记为$\gamma_{ji}(i=1,2,\dots,k)$.

当高斯混合分布已知时，高斯混合聚类把样本集 D 划分为 k 个簇$C=\{C_1,C_2,\dots,C_k\}$，每个样本x_j的簇标记λ_j如下确定：

$$\lambda_j=\arg max_{x\in\{1,2,\dots,k\}}\gamma_{ji}$$

因此，高斯混合聚类是采用概率模型（高斯分布）对数据进行刻画，簇划分则根据数据对应的后验概率确定。

对于高斯混合分布 $p(x)=\sum_{i=1}^{k}\alpha_i P(x|\mu_i,\Sigma_i)$的模型参数 $\{(\alpha_i,\mu_i,\Sigma_i)|1\le i\le k\}$的求解采用极大似然估计，最大化对数似然

$$LL(D)=\ln\Pi_{j=1}^{m}p(x_j)=\sum_{j=1}^{m}\ln(\sum_{i=1}^{k}\alpha_i p(x_j|\mu_i,\Sigma_i))$$

可以采用 EM 算法进行迭代优化。在每步迭代中，先根据当前参数来计算每个样本属于每个高斯成分的后验概率γ_{ji}（E 步），再用最大似然估计更新模型参数$\{(\alpha_i,\mu_i,\Sigma_i)|1\le i\le k\}$（M 步）.

更新公式推导略，结果如下：

$$\mu_i=\frac{\sum_{j=1}^{m}\gamma_{ji}x_j}{\sum_{j=1}^{m}\gamma_{ji}}$$

$$\Sigma_i=\frac{\sum_{j=1}^{m}\gamma_{ji}(x_j-\mu_i)(x_j-\mu_i)^T}{\sum_{j=1}^{m}\gamma_{ji}}$$

$$\alpha_i=\frac{1}{m}\sum_{j=1}^{m}\gamma_{ji}$$

可以看出来，各混合成份的均值可通过样本加权平均来估计，样本权重是每个样本是属于该成分的后验概率，协方差矩阵也有类似的形式。选择每个高斯成分的概率α_i是由属于该成分的平均后验概率确定。

收敛条件可以是似然函数增长很小甚至不在增长。

高斯混合模型的决策面是二次的。

$$\ln\frac{P(y=i|x)}{P(y=j|x)}=\ln\frac{P(x|y=i)P(y=i)}{P(x|y=j)P(y=j)}$$

$$=\ln\frac{P(y=i)}{P(y=j)}+\ln\frac{\frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_{\mathrm{i}}|^{\frac{1}{2}}}e^{-\frac{1}{2}(x-\mu_i)^T\Sigma_{\mathrm{i}}^{-1}(x-\mu_i)}}{\frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_{\mathrm{j}}|^{\frac{1}{2}}}e^{-\frac{1}{2}(x-\mu_j)^T\Sigma_{\mathrm{j}}^{-1}(x-\mu_j)}}$$

$$=\ln\frac{P(y=i)}{P(y=j)}+\frac{1}{2}\ln\frac{|\Sigma_j|}{|\Sigma_i|}+[-\frac{1}{2}(x-\mu_i)^T\Sigma_{\mathrm{i}}^{-1}(x-\mu_i)+\frac{1}{2}(x-\mu_j)^T\Sigma_{\mathrm{j}}^{-1}(x-\mu_j)]$$

$$=x^TWx+w^Tx$$

由于各个高斯混合分布的协方差矩阵不同，二阶项没有消掉，所以决策面是二次的。而 k-means 的协方差矩阵是相同的，只有一阶项，决策面是线性的。

2. 算法的实现

k-means 伪代码及 GMM 聚类算法伪代码如下图：.

1: 从 D 中随机选择 k 个样本作为初始均值向量 $\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$
2: **repeat**
3: 令 $C_i=\varnothing\ (1\leqslant i\leqslant k)$
4: **for** $j=1,2,\ldots,m$ **do**
5: 计算样本 $\boldsymbol{x}_j$ 与各均值向量 $\boldsymbol{\mu}_i\ (1\leqslant i\leqslant k)$ 的距离: $d_{ji}=\|\boldsymbol{x}_j-\boldsymbol{\mu}_i\|_2$;
6: 根据距离最近的均值向量确定 $\boldsymbol{x}_j$ 的簇标记: $\lambda_j=\arg\min_{i\in\{1,2,\ldots,k\}}d_{ji}$;
7: 将样本 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$;
8: **end for**
9: **for** $i=1,2,\ldots,k$ **do**
10: 计算新均值向量: $\boldsymbol{\mu}_i'=\frac{1}{|C_i|}\sum_{\boldsymbol{x}\in C_i}\boldsymbol{x}$;
11: **if** $\boldsymbol{\mu}_i'\neq\boldsymbol{\mu}_i$ **then**
12: 将当前均值向量 $\boldsymbol{\mu}_i$ 更新为 $\boldsymbol{\mu}_i'$
13: **else**
14: 保持当前均值向量不变
15: **end if**
16: **end for**
17: **until** 当前均值向量均未更新
输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
高斯混合成分个数 k.
过程:
1: 初始化高斯混合分布的模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$
2: **repeat**
3: **for** $j=1,2,\ldots,m$ **do**
4: 根据式(9.30)计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率, 即
$\gamma_{ji}=p_{\mathcal{M}}(z_j=i\mid\boldsymbol{x}_j)\ (1\leqslant i\leqslant k)$
5: **end for**
6: **for** $i=1,2,\ldots,k$ **do**
7: 计算新均值向量: $\boldsymbol{\mu}_i'=\frac{\sum_{j=1}^m\gamma_{ji}\boldsymbol{x}_j}{\sum_{j=1}^m\gamma_{ji}}$;
8: 计算新协方差矩阵: $\boldsymbol{\Sigma}_i'=\frac{\sum_{j=1}^m\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i')(\boldsymbol{x}_j-\boldsymbol{\mu}_i')^{\mathrm{T}}}{\sum_{j=1}^m\gamma_{ji}}$;
9: 计算新混合系数: $\alpha_i'=\frac{\sum_{j=1}^m\gamma_{ji}}{m}$;
10: **end for**
11: 将模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$ 更新为 $\{(\alpha_i',\boldsymbol{\mu}_i',\boldsymbol{\Sigma}_i')\mid 1\leqslant i\leqslant k\}$
12: **until** 满足停止条件
13: $C_i=\varnothing\ (1\leqslant i\leqslant k)$
14: **for** $j=1,2,\ldots,m$ **do**
15: 根据式(9.31)确定 $\boldsymbol{x}_j$ 的簇标记 λ_j;
16: 将 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$
17: **end for**
输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

matlab 实现：

```
function kmeans(X,k)
close all;
[m,n] = size(X);
iter = 0;
% 初始化k个均值向量
mu = zeros(k,n);
previous_mu = mu;
R = randperm(m,k);
for i = 1:k
    mu(i,:) = X(R(i),:);
end
while true
    iter = iter + 1;
    % 对每一个样本找到离它最近的中心点，E步
    index = zeros(m,1);
    distance = zeros(1,k);
    for i = 1:m
        for j = 1:k
            distance(j) = norm(X(i,:) - mu(j,:)); % 计算矩阵的2-范数
        end
        [~, index(i)] = min(distance);
    end
    % 更新k个均值向量，M步
    for i = 1:k
        idx = find(index == i);
        number = length(idx);
        mu(i,:) = sum(X(idx,:)) / number;
    end
    % 如果均值向量和先前的一样，跳出循环
    if sum(abs(previous_mu - mu)) == 0
        break;
    end
    previous_mu = mu;
    % 绘图
    figure(iter);
    palette = hsv(k);
    colors = palette(index, :);
    scatter(X(:,1),X(:,2),15,colors);
    hold on;
    plot(mu(:,1),mu(:,2),'x','MarkerEdgeColor','k','MarkerSize',10);
end
```

k-means 算法实现

```
function GMM(X, k)
close all;
[m,n] = size(X);
iter = 0;
% 初始化
previous_LLD = 0;
alpha = zeros(1,k) + 1 / k;
mu = zeros(k,n);
R = randperm(m,k);
for i = 1:k
    mu(i,:) = X(R(i),:);
end
sigma = cell(1,k);
for i = 1:k
    sigma{1,i} = diag(zeros(1,n)+0.1);
end
px = zeros(m, k);
```

GMM 初始化参数$\{(\alpha_i, \mu_i, \Sigma_i) | 1 \leq i \leq k\}$

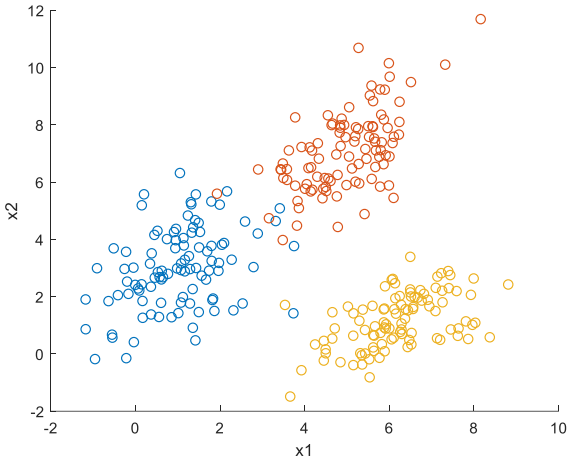
```
while true
    iter = iter + 1;
    % 算法E步，计算每个样本属于每个高斯成分的后验概率
    for i = 1:k
        px(:,i) = mvnpdf(X,mu(i,:),sigma{1,i});
    end
    gamma = repmat(alpha, m, 1) .* px;
    gamma = gamma ./ repmat(sum(gamma,2),1,k);
    % 绘图
    index = zeros(m,1);
    for i = 1:m
        [~,index(i)] = max(gamma(i,:));
    end
    figure(iter);
    palette = hsv(k);
    colors = palette(index, :);
    scatter(X(:,1),X(:,2),15,colors);
    hold on;
    % 算法M步，更新模型参数，使得LLD最大
    Nk = sum(gamma,1);
    temp = gamma' * X;
    for i = 1:k
        mu(i,:) = temp(i,:) / Nk(:,i);   % 更新mu
    end
    for i = 1:k
        XminusMu = X - repmat(mu(i,:),m,1);
        sigma{1,i} = XminusMu' * diag(gamma(:,i)) * XminusMu / Nk(:,i); % 更新sigma
    end
    alpha = Nk / m; % 更新alpha
    % 绘图，均值向量落点
    plot(mu(:,1),mu(:,2),'x','MarkerEdgeColor','k','MarkerSize',10);
    % 停止条件，如果LLD增加很少，则停止
    LLD = sum(log(alpha * px'));
    if abs(LLD - previous_LLD) < 1e-6
        break;
    end
    previous_LLD = LLD;
end
```

GMM 的核心步骤：EM 算法实现

四、实验结果与分析

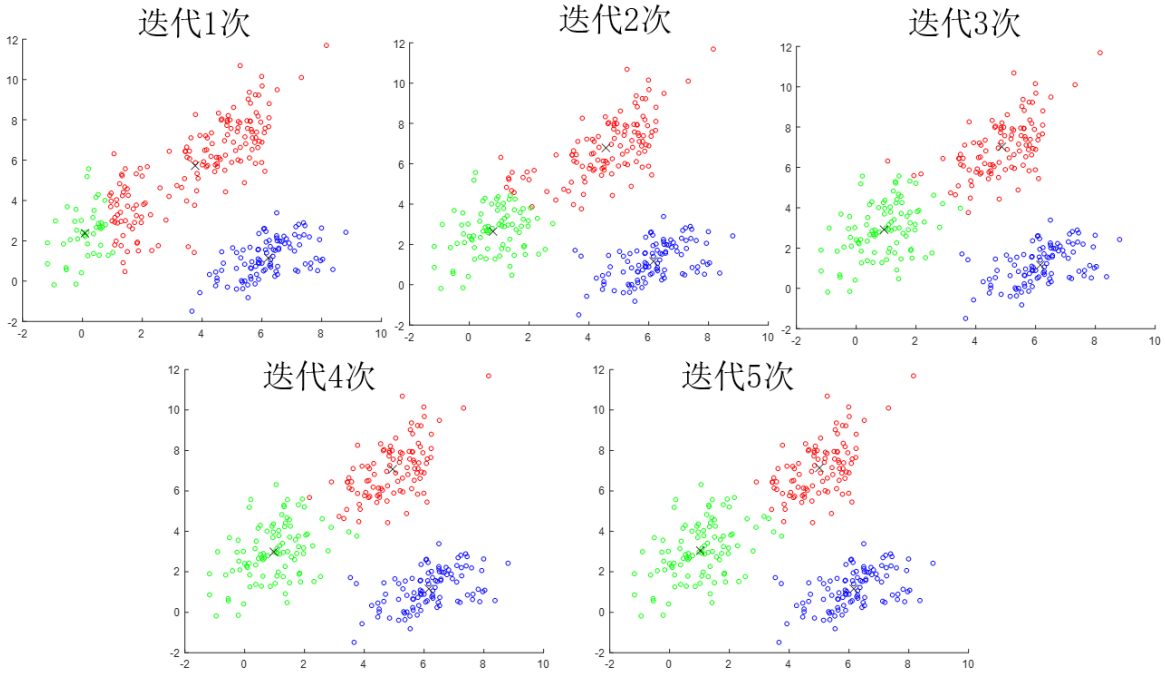
生成数据：利用三个高斯分布产生三个类别的数据作为聚类的数据。

```
function X = productData()

number = 100;                % 产生数据个数

mu = [1 3];                  % 生成数据的均值
sigma = [1 0.5;0.5 2];           % 数据的协方差矩阵
R1 = mvnrnd(mu, sigma, number);

mu = [5 7];
sigma = [1 0.8; 0.8 2];
R2 = mvnrnd(mu, sigma, number);

mu = [6 1];
sigma = [1 0.5;0.5 1];
R3 = mvnrnd(mu, sigma, number);

X = [R1;R2;R3];
```

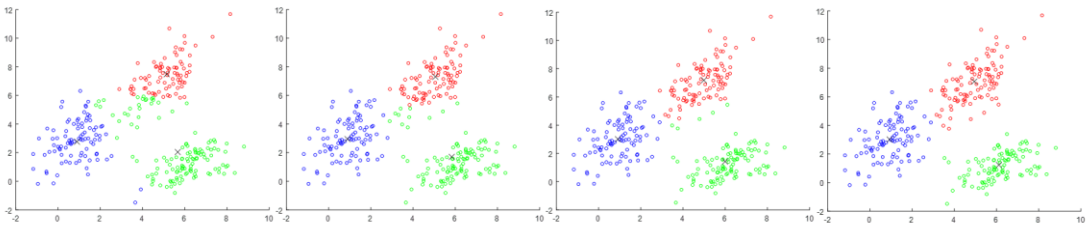


原始数据及正确的分类

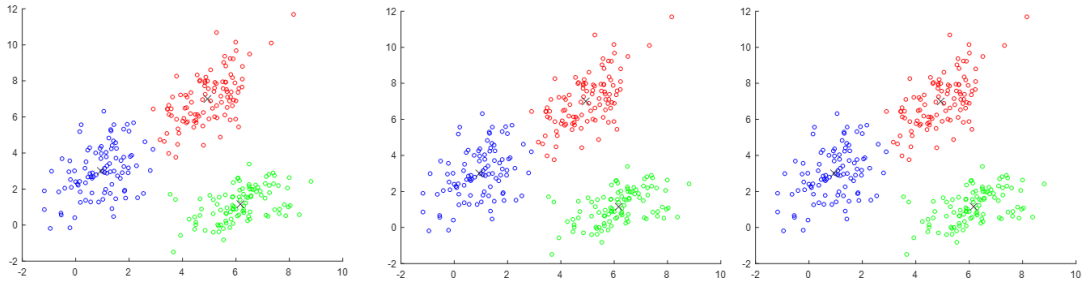
k-means：由于初始化μ的时候是随机选择的，会影响迭代至收敛的次数，但大致在 2-10 次以内（仅以本次数据为例），下面展示进行了 5 次迭代收敛的结果。



GMM： 同样 GMM 算法由于初始均值向量选择不同，收敛时迭代次数不同，但次数普遍高于 k-means 算法，在几十次左右。以 13 次迭代后收敛为例，见下图：



迭代1-4次



迭代11-13次

UCI 数据集：使用 iris 数据集， 总共三类，但有 4 个特征，无法可视化。因此，下面将计算其分类正确率。

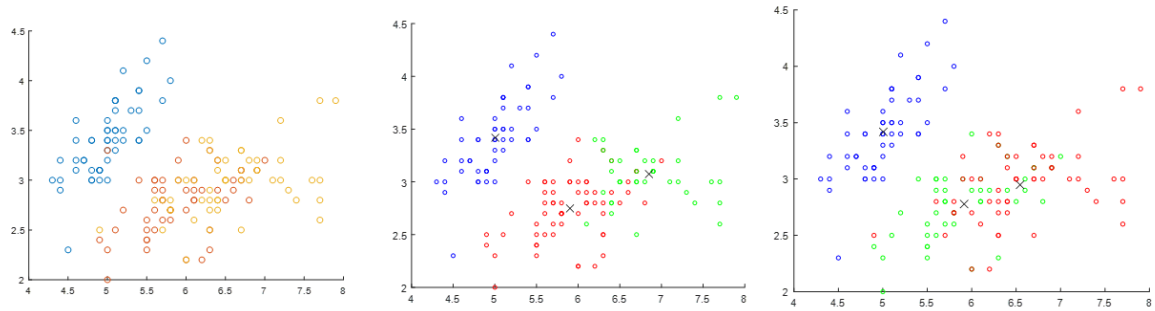
算法	setosa	versicolor	virginica	总计
k-means	50	48	36	134
GMM	50	45	50	145

上表仅列出了用两个方法实验的一组数据。由于初始化参数的不同，最后聚类结果也有所不同。

第一种花和其他两类是线性可分的，k-means 聚类正确率比较高，而后两类线性不可分，会有交错，而 kmeans 只能线性区分，导致正确率偏低。

GMM 聚类是二次决策面，则能够较好区分后两类，正确率更高。

仅以前两个特征作图可视化：



原始数据　　K-means　　GMM

五、结论

1. k-means 收敛快，聚类效果比较好。但是对某些线性不可分的、集群有重叠、一些集群分布比其他集群分布更广等的数据不太适用。

2. GMM 聚类运行速度较慢，但簇可以呈现任何椭圆形状，而不是被限制为圆形（二维）。

3. 由于随机选取的初始值不同，迭代到收敛的次数不相同。

六、参考文献

1. 周志华.(2016).机器学习.清华大学出版社.北京
2. 刘杨.(2019).机器学习课件(7).哈尔滨工业大学.哈尔滨
3. 博客：k-means 算法及其 matlab 实现

七、附录：源代码（带注释）

此处略，前文已经展示所有代码。