一、 实验目的与要求

1.熟练掌握 k-Means 方法对手写数字图像进行分类；
2.用 Matlab 编写代码，熟悉其画图工具，进行实验，并验证结果；
3.锻炼数学描述能力，提高报告的叙述能力。

二、 问题

手写数字图像数据分类问题：文件train_images.mat包含大小为28×28的手写数字图像，共60000张；文件train_labels.mat是其对应的数字标签。文件数据的具体读写和数据格式，请参考附件DataRead.m文件。实验要求对手写数字图像进行聚类，并讨论其性能：

(MNIST DATABASE下载网址: http://yann.lecun.com/exdb/mnist/)

(1) 对train_images.mat的前100张手写数字图像进行聚类，共10类；

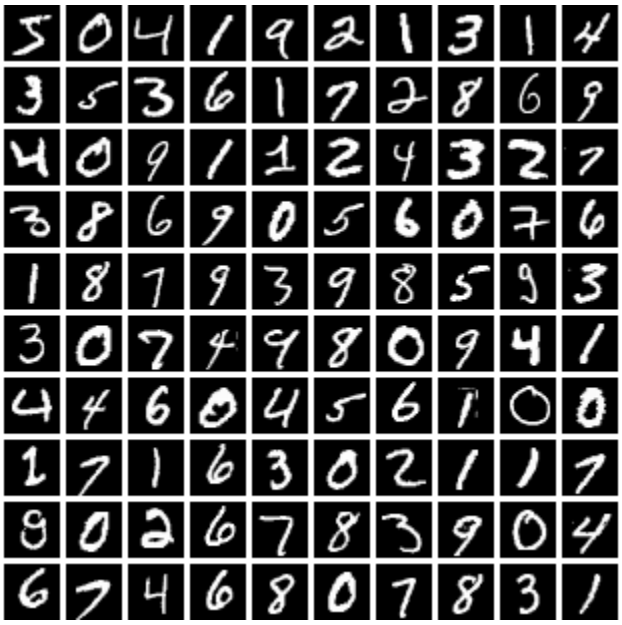

图1. 文件前100张手写数字图像

(2) 对train_images.mat的前1000张手写数字图像进行聚类，共10类；
(3) 根据实际情况，讨论k-Means能对多少张手写图像进行聚类，性能如何？

三、模型建立及求解

本次实验实现过程的思维逻辑如图 1 所示：

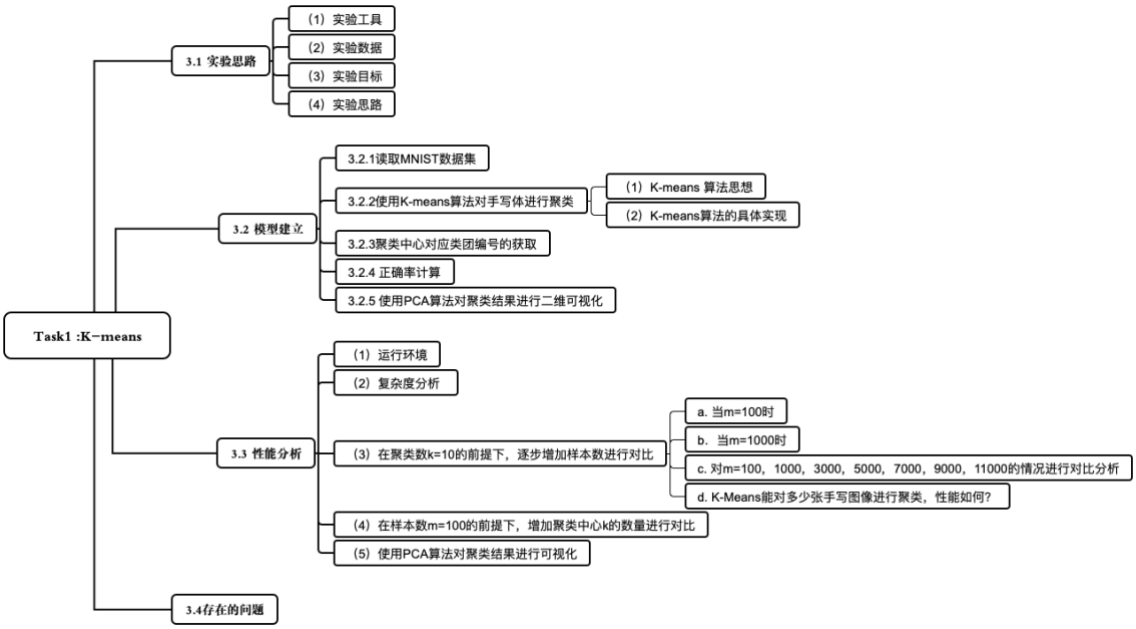

图 1 K-means 实验实现过程思维逻辑图

3.1 实验思路

（1）实验工具：由于笔者对 MATLAB 语言不太熟悉，在进行了一些**尝试**之后，为了更好地完成这次实验，还是决定使用笔者相对熟悉的 **Python** 进行实验。

（2）实验数据： **MNIST 数据集**是机器学习领域中一个经典数据集，由 60000 个训练样本和 10000 个测试样本组成，每个样本都是一张 **28 * 28** 像素的灰度手写数字图片。[1]本次实验使用到 MNIST 的手写体的图像数据（train_images.mat）和图像数据对应的标签（train_labels.mat）。

（3）实验目标：本次实验要求使用 **K-means 算法**建立模型，对手写体进行聚类，再对聚类的结果来进行**性能对比和分析**。

（4）实验思路：

1. 读取并分析 MNIST 数据集
2. 使用 K-means 算法对手写体进行聚类
3. 使用 PCA 算法对聚类结果进行二维可视化
4. 对聚类结果进行性能分析（聚类时间+准确率）

3.2 模型建立

3.2.1 读取 MNIST 数据集

由于所给的手写体文件是以 **mat 格式**来存储的，在 Python 中需要使用相关的工具库进行导入，这里笔者使用 scipy.io 中的函数 **loadmat()**来读取 mat 文件。最终得到的 images 数据是一个**(28,28,60000)**的三维数组，labels 数据是一个**（1，60000）**的二维数据。图 2 展示了导入成功后，其中一张手写体及其标签的灰度像素图片。

[1] 参考：https://www.cnblogs.com/xianhan/p/9145966.html

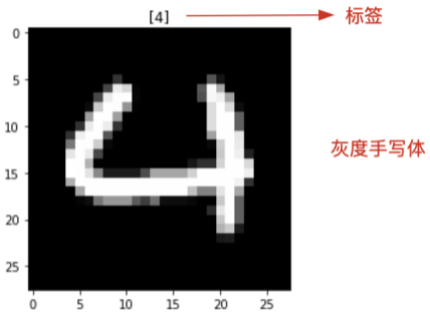

图 2：手写体及其标签的灰度像素图片展示

3.2.2 使用 K-means 算法对手写体进行聚类

（1）K-means 算法思想

经典 K-means 算法的基本工作流程：首先随机选取 K 个数据作为的聚类的**聚类中心**，计算各个数据对象到所选出来的各个中心的**距离**，将数据对象指派到最近的**类**中；然后计算每个类的**均值**，**循环**往复执行，直到满足聚类的**收敛条件**为止。具体的执行步骤如表 1 所示：[2]

表 1 K-means 算法流程

步骤	操作
1	确定聚类的数量，即K值；
2	在数据集中随机初始化K个样本作为初始聚类中心 $a = a_1, a_2, \dots a_k$；
3	针对数据集中每个样本 x_i 计算它到K个聚类中心的欧几里得距离并将其分到距离最小的聚类中心所对应的类中；
4	针对每个类别 a_j，重新计算它的聚类中心 $a_j = \frac{1}{\lvert c_i \rvert}\sum_{x \in c_i} x$ (即属于该类的质心)；
5	反复迭代上面 3、4 两步操作，直达到某个中止条件

（2）K-means 算法的具体实现

a．参数设置

- times：因为 K-means 算法每次运行的结果都不一致，为了减少极端准确率的影响，所以定义变量 **times 表示算法重复进行的次数**（后面在计算精确度时可以对每次运行结果的精确度取平均得到）；
- m：表示使用数据集的**前 m 张图片进行聚类**；
- k：**表示聚类的类别数**，这里因为是手写数字 0-9，因此把 k 设置为 10；
- num_iters：**表示算法迭代的次数**，当达到迭代次数的时候，则退出算法；

b. 初始化聚类中心，从选择的前 m 张手写体数据集中随机挑选出 k 个样本作为质心；

c. 计算出每一张手写体图片与聚类中心的**距离**，挑选出每张手写体图片距离最近的**聚类中心**，并把聚类中心的下标按顺序储存到数组 C 中；

d. 接着继续**更新质心**。重新计算质心后，若质心改变，则更新质心，这时候**迭代次数** num_iters+1。笔者在这里设定当迭代次数达到 **20** 时，则退出迭代。

3.2.3 聚类中心对应类团编号的获取

[2] 参考：埃塞姆. 机器学习导论[M]. 北京：机械工业出版社，2016.

（1）通过 C_labels 列表，在内部嵌套K个列表 list,形成“列表中列表”的数据结构。将 K-Means 函数得到的K个类团中**每个类团的实际 labe1 进行存储**。

（2）根据每个类团存储的 label 数进行**计数**,把 **labe1 数最多的作为本类团的编号**（如：类团[1,2,3,3,3,3,4,4],其中编号出现最多的是“3”,因此 3 作为本类团的聚类中心）。

（3）如果 list 中 label 与计数最多的**相同的个数有多个**，则**取第一个 label** 作为本类团的聚类中心（如：类团[1,2,3,3,4,4],其中编号出现最多的是“3”和“4”,因为 3 排在前面，因此讲 3 作为本类团的聚类中心）。

3.2.4 正确率计算

利用存储图片对应数字的数组和上一步求得的聚类中心数字可以得到利用 K-means 算法识别的数字结果，再将其与 train_labels（每张图片真正的 label）对照并计算识别正确率。

$$\text{精确度} = \frac{\text{正确数}}{\text{总数}}$$

3.2.5 使用 PCA 算法对聚类结果进行二维可视化

为了对聚类的效果进行映射到**二维平面的可视化**，以直观显示出**聚类效果的散点图**，所以需要将维度为 $28 \times 28 = 784$ 的手写体图片**降到 2 维**。

主成分分析（PCA）是最常用的一种降维方法，通常用于**高维数据集的探索与可视化**，还可以用作数据压缩和预处理等。PCA 可以把具有相关性的高维变量合成为**线性无关的低维变量**，称为主成分，主成分能够尽可能保留原始数据的信息。原理就是其协方差矩阵对应的**特征向量**，按照对应的特征值大小进行排序，最大的特征值就是第一主成分，其次是第二主成分，依次类推。[3] PCA 算法流程如表 2 所示：

表 2　PCA 算法流程[4]

输入： 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;

低维空间维数 d'.

过程：

1：对所有样本进行中心化：$\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$;

2：计算样本的协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$;

3：对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 做特征值分解;

4：取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'}$.

输出： 投影矩阵 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$.

这里，笔者使用 **Python 的 sklearn 库**，调用 **decomposition.PAC** 加载 PCA 进行降维，使用**参数 n_components 指定主成分的个数为 2**，即降维后数据的纬度为 2。封装成 **display() 函数**，然后分别对聚类前后的样本进行可视化。

3.3 性能分析

（1）运行环境

MacOS Big Sur + Intel I5 + 16G 内存 + Pycharm

[3] 参考：https://blog.csdn.net/qq_41663800/article/details/90084418

[4] 参考：周志华. 机器学习[M]. 北京：清华大学出版社，2016.

（2）复杂度分析

k个聚类中心,m个数据,计算两数据之间二范数的复杂度为$O(d)$,则聚类一次的时间复杂度为$O(kmd)$。对于dim*dim像素的灰度图像,两数据间的欧氏距离的复杂度为$3*\text{dim}*\text{dim}$。所以时间复杂度可以写为$O(3*dim*dim*km)$。重新确定聚类中心所需的时间复杂度为$O(dim*\ dim\ *m)$,少于第一步。**k-Meams 的时间复杂度则为$\boldsymbol{O(dim*dim*km)}$**。

（3）在聚类数 k=10 的前提下，逐步增加样本数进行对比

a. 当 m=100 时

首先设置参数，**m=100，k=10，times=100**，将前 100 张手写数字图像聚成 10 类，并且重复运行 100 次，得到如图 3 所示的可视化图表。可以看到，100 次聚类测试中，**聚类的平均正确率为 59.50%，最高达到 70%，最低为 46%，方差为 0.002，每次聚类的平均时间为 0.051s。**

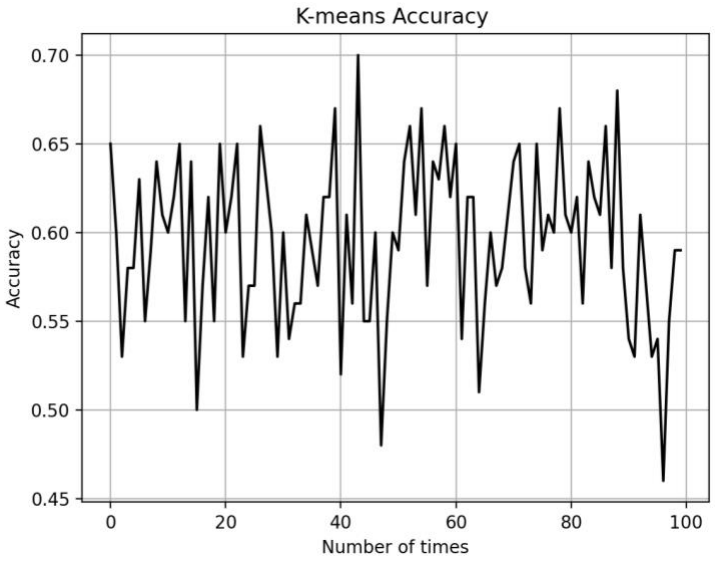

图 3：m=100，k=10，times=100 时的正确率

如表 3 所示，笔者选取其中 5 次运行结果进行**详细展示**：

表 3 m=100，k=10，times= 5 运行详细结果展示

聚类次数	聚类中心	迭代次数	聚类时间(s)	正确率
1	[3, 9, 3, 0, 1, 2, 8, 7, 4, 6]	8	0.062	68.00%
2	[6, 4, 3, 0, 3, 8, 9, 0, 1, 1]	4	0.032	66.00%
3	[1, 4, 7, 7, 7, 0, 6, 8, 3, 0]	5	0.039	66.00%
4	[7, 1, 3, 3, 0, 0, 8, 6, 9, 4]	9	0.075	69.00%
5	[8, 4, 0, 7, 1, 0, 6, 9, 3, 1]	7	0.066	66.00%
平均	-	6.6	0.055	67.00%

接下来，笔者将探究：**在 10 个聚类中心的前提下，随着聚类的样本数的增加，K-means 算法的匹配正确率是否会有显著差异。**

b. 当 m=1000 时

设置参数，**m=1000，k=10，times=100**，将前 1000 张手写数字图像聚成 10 类，并且重复运行 100 次，得到如图 4 所示的可视化图表。可以看到，100 次聚类测试中，**聚类的平均**

正确率为 54.45%，最高达到 62.70%，最低为 45.10%，方差为 0.001，每次聚类的平均时间为 1.392s。

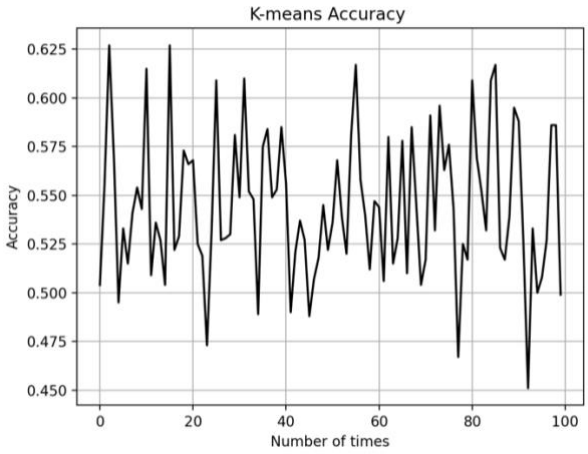

图 4：m=1000，k=10，times=100 时的正确率

c. 对 m=100，1000，3000，5000，7000，9000，11000 的情况进行对比分析

由于篇幅所限，笔者就不在这里详细画出其他情况的**正确率折线图**。这里随着样本数目不断增多，算法运行时间不断增加，**受制于硬件性能**，当样本增加到 9000 时，100 次的运行次数已经较难完成了，所以当$m \geq 9000$时，本实验的测试次数减少为 10 次，但 10 次取平均得到的正确率也可以较好地说明问题了。**结果总结如表 4 所示，可以看到，在 10 个聚类中心的前提下，随着聚类的样本数的增加，K-means 算法的平均正确率都集中在 55%-60%，样本量的增加没有带来正确率的显著差异，但是平均运行时间却显著不断增加。这可能是因为本身聚类算法的**

表 4 :k=10 的前提下，样本数不断增加的测试结果

聚类样本	测试次数	聚类中心	平均正确率	最高正确率	最低正确率	方差	平均时间(s)
100	100	10	**59.50%**	70.00%	46.00%	0.0020	**0.051**
1000	100	10	**54.45%**	62.70%	45.10%	0.0010	**1.392**
3000	100	10	**57.35%**	62.57%	50.13%	0.0007	**4.971**
5000	100	10	**56.72%**	62.90%	50.38%	0.0007	**8.576**
7000	100	10	**56.17%**	62.57%	52.60%	0.0008	**12.53**
9000	10	10	**56.55%**	59.04%	54.77%	0.0002	**16.411**
11000	10	10	**55.88%**	58.69%	52.30%	0.0005	**21.554**

d. K-Means 能对多少张手写图像进行聚类，性能如何？

为了探究本次设计的 K-Means 能对多少张手写图像进行聚类，在 **k=10** 的情况，我选取数据集中**全部 60000 张图片进行聚类**，如图 5 所示，可以看到聚类的**正确率为 58.97%**，仍处于 55%-60%的区间，**运行时间为 249.778s**。**所以本 K-Means 算法能完成对全部 60000 张手写图像进行聚类，性能正确率可以达到 58.97%。**

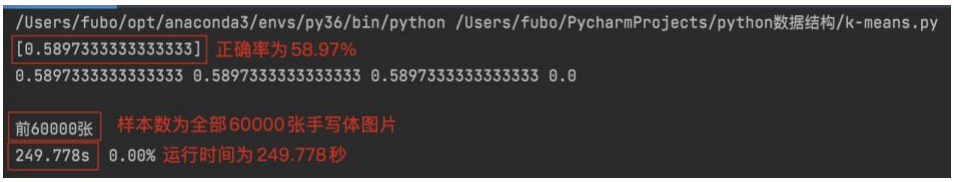

图 5：m=60000，k=10 时的运行结果

（4）在样本数 m=100 的前提下，增加聚类中心 k 的数量进行对比

接下来，笔者还将探究：**在聚类样本数 m=100 的前提下，随着聚类中心 k 的增加，K-means 算法的匹配正确率是否会有显著差异。**

设置参数，**m=100，k=15，times=100**，将前 100 张手写数字图像聚成 15 类，并且重复运行 100 次，得到如图 6 所示的可视化图表。可以看到，100 次聚类测试中，**聚类的平均正确率为 67.21%，最高达到 79.00%，最低为 58.00%，方差为 0.002，每次聚类的平均时间为 0.067s。**可以看到随着**聚类中心的增加（从 10 到 15），聚类的平均正确率显著提升，但聚类的平均时间也有所增加。**（如前面图 2 所示，当 k=10 时，聚类的平均正确率为 59.50%，最高达到 70%，最低为 46%，方差为 0.002，每次聚类的平均时间为 0.051s。）

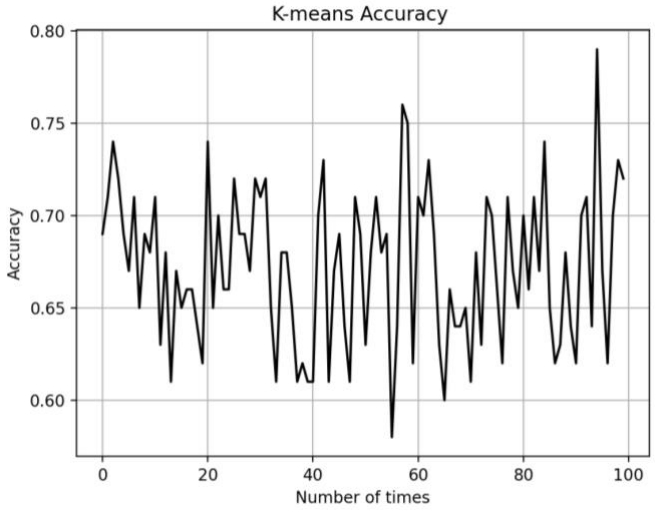

图 6：m=100，k=15，times=100 时的正确率

如表 5 所示，笔者选取其中 **5 次运行结果进行详细展示**，可以看到跟 k=10 的情形进行相对，聚类**迭代的次数也明显下降**（从 6.6 下降为 4）

表 5 m=100，k=15，times= 5 运行详细结果展示

聚类次数	聚类中心	迭代次数	聚类时间(s)	正确率
1	[3,0,5,7,2,9,8,1,6,8,7,4,0,3,4]	3	0.044	67.00%
2	[0,3,6,8,0,3,4,5,2,1,7,1,9,4,0]	3	0.053	73.00%
3	[3,2,3,0,9,4,4,8,7,3,6,1,7,1,3]	5	0.091	65.00%
4	[5,1,7,1,1,9,0,9,3,0,6,2,4,3,4]	5	0.083	62.00%
5	[4,4,3,7,9,1,8,0,1,8,1,0,3,7,4]	4	0.058	66.00%
平均	--	4	0.066	66.60%

（5）使用 PCA 算法对聚类结果进行可视化

当 m=100，k=10 时，运行程序，正确率为 57%，得到如图 7 的聚类可视化结果：

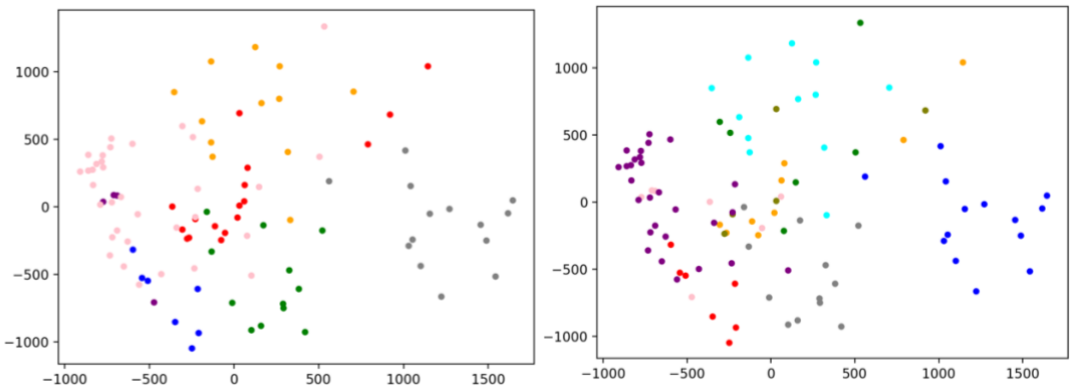

图 7：m=100，k=10 的聚类前（左边）和聚类后（右边）的可视化结果

当 m=1000，k=10 时，运行程序，正确率为 55.80%，得到如图 8 的聚类可视化结果：

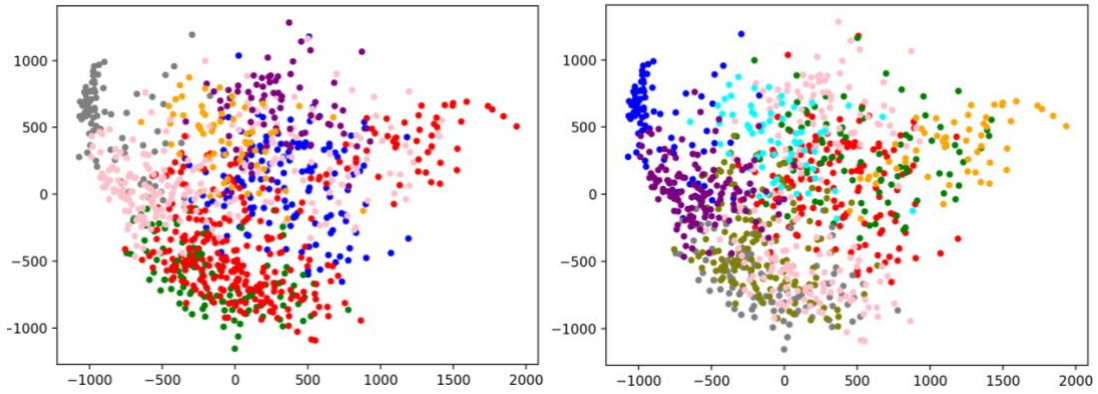

图 8：m=1000，k=10 的聚类前（左边）和聚类后（右边）的可视化结果

在经过 **PCA 算法的可视化**后，可以轻易看出数据集使用 K-Means 进行聚类**并不能有很好的效果**。虽然有一部分是因为 K-means 是根据 28*28 来度量距离的，**取前两维**之后可视化的效果不会特别好。但是其实作为一种**无监督的基于距离的聚类算法**，K-Means 算法虽然易于理解并被广泛运用，但算法并**没有完美**一说，K-Means **时间复杂度较高**并且在实际问题中**聚类中心的数量和选取**在一开始难以精准确认，这都使得 K-Means 算法的聚类结果并没有表现的很好，仍**存在着一些问题**。

3.4 存在的问题

因为未加入随机重启机制仅运行一次聚类算法，会使得运行结果有很大概率仅为**局部最优解而不是全局最优解**。所以这种用局部最优解代替全局最优解的方法，虽然能**显著降低时间复杂度**，但代价是聚类结果对初始**聚类中心的选取**比较**敏感**。

对于 k-Means 初始聚类中心的选择方法主要有两种。一，选择批次距离**尽可能远的 k 个点**；二，选用**层次聚类或者 Canopy 算法**进行初始聚类，然后利用这些类的中心点作为 k-Means 算法初始类团的中心点。

在本次实验中，关于 K-means 算法，还存在**正确率待提高的问题**，若还有时间笔者会进一步对上述的**优化算法进行实验验证**。

四、小结（可含个人心得体会）

4. 1 实验小结

（1）本次实验，笔者在**完成了 K-means 算法后，根据对不同聚类样本数和聚类中心数进行测试和对比，从而分析算法的性能，最后还扩展实现了 PCA 算法对聚类结果进行可视化对比。**

（2）**随着聚类样本数的增加**（m = 100，1000，3000，5000，7000，9000，11000），**聚类的平均正确率并没有显著差异**。这可能是因为随着聚类样本数的增加，在 10 个聚类中心的前提下，随着带来的**错误匹配概率也增加**，因此聚类样本数增加并不意味着聚类结果正确率更高。

（3）**随着聚类中心数的增加**（k=10，15），**聚类的平均正确率有明显提高**，且**迭代次数降低，收敛变快**，这可能是因为聚类中心数的增加提高了**程序可容错性**，不过也导致了**平均聚类时间有一些增加**。

（3）经过 PCA 算法的可视化后，可以轻易看出数据集使用 K-Means 进行聚类**并不能有很好的效果**，期待之后使用 **KNN 等算法**进一步对手写体数据进行实验对比。

4.2 个人体会

（1）本次实验，笔者在刚开始尝试使用 Matlab 完成实验，但后发现自己确实不太会用 matlab，比较难将完成度做得好一些，然后转向用 python 来进行实现。在 **K-means 算法和 PCA 算法**的具体实现和逻辑连接上，笔者也花费了挺多时间来思考代码的整体框架，各种 debug，最后还算是比较成功地完成了任务，**收获颇丰**。

（2）在测试的过程中，当样本量增加的时候，在跑数据的过程中，笔者还同步写实验报告“这里尽管样本数目不断增多,但为了数据的对比准确度,笔者不惜等待时间,仍然对每种情况都进行了 100 次测试。”但数据实在跑了很久都没有完成…最后我妥协了，当$m \geq 9000$时，笔者将测试次数减少为 10 次。现在想想，因其他杂事众多，生活中不能按照自己理想主义般的方式去探索世界，而是**向现实妥协**真不是很好受…

（3）最后，**感谢两位助教的耐心且热情解答笔者的问题，也感谢老师！审批辛苦了！**

指导教师批阅意见： 成绩评定： 指导教师签字： 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
　　2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。