机器学习与数据挖掘 作业 3

林宇浩 21311274

一、K-Means 算法

（1）算法流程简述

K 均值用于将数据集划分为 K 个不同的簇，使得每个数据点属于离它最近的簇中心。下面是 K 均值算法的基本流程：
1.选择簇的数量： 首先，确定希望将数据集划分为多少个簇。这通常是通过先验知识或者使用一些启发式方法来选择的，也可以通过最终结果使用手肘法得知。
2.初始化簇中心： 随机选择 K 个数据点作为初始的簇中心。这些簇中心将作为后续迭代中数据点分配和更新簇中心的起点。
3.分配数据点到簇： 对于每个数据点，计算其与所有簇中心的距离，将其分配到距离最近的簇中心所对应的簇。
4.更新簇中心： 对每个簇，计算其所有成员的平均值，将这个平均值作为新的簇中心。
5.重复步骤 3 和步骤 4： 重复执行步骤 3 和步骤 4，直到满足某个停止条件，例如簇中心不再发生明显变化，或者达到预定的迭代次数。
6.输出结果： 最终，算法收敛并确定每个数据点属于哪个簇。输出包括簇中心和每个数据点的簇分配。

（2）实验

训练设置为每个模型均训练 100 轮，若聚类中心已不变化则提前结束训练。训练完成后标签匹配采用了匈牙利算法，利用训练集上得到的结果，计算每个簇中各个类别的数量，构建一张 10*10 的二分图，通过二分图匹配得到预测标签和真实标签的映射关系。测试集的结果利用训练集得到的匹配函数，不重新进行匹配。

使用了 4 种初始化方法，如下所示：

1. random：
初始聚类中心数据随机生成

2. random2：
初始聚类中心数据在数值 0~255 中随机生成

3. kmeans++：
K-means++初始化方法通过改善初始簇中心的选择来减少算法对初始值敏感的问题，步骤如下：
a 选择第一个簇中心： 从数据集中随机选择一个数据点作为第一个簇中心。
b 选择后续簇中心： 对于每个数据点，计算它与已经选择的簇中心的最短距离（即与最近簇中心的距离）。然后，选择下一个簇中心时，以概率正比于每个数据点的最短距离的平方，越远离已有簇中心的点被选中的概率越大。
c 重复选择： 重复步骤 2，直到选择出 K 个初始簇中心。
d 应用 K-means 算法： 使用选择的初始簇中心运行标准的 K-means 算法。

4. forgy：
Forgy 初始化方法由计算机科学家 Willis J. Forgy 于 1965 年提出的，是 K-means 聚类算法的一种最简单的初始簇中心选择方法。在 Forgy 初始化方法中，簇中心直接从数据集中随机选择。

同时使用了 4 种距离度量方法，如下所示：

1.欧式距离： $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$

2.曼哈顿距离：$d(x,y)=\sum_{i=1}^{n}|x_i-y_i|$

3.切比雪夫距离：$d(x,y)=\max_i|x_i-y_i|$

4.闵可夫斯基距离(p=4)：$d(x,y)=(\sum_{i=1}^{n}|x_i-y_i|^p)^{1/p}$

实验结果如下：

训练集上的聚类精度

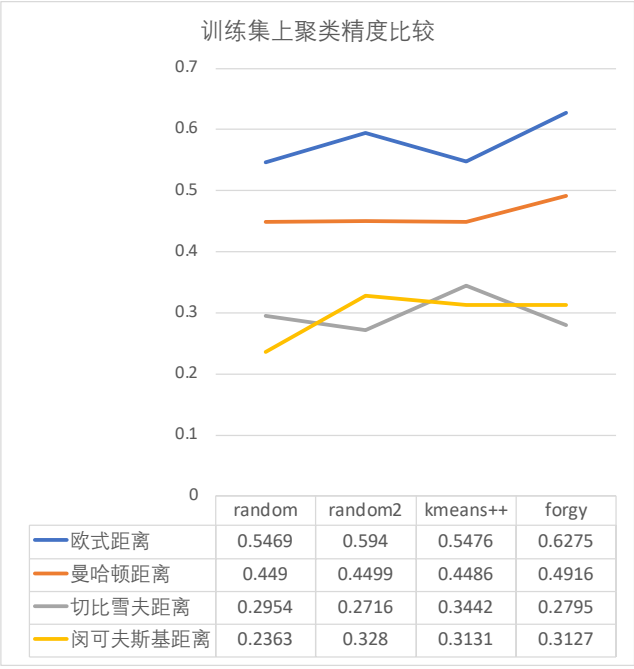
	random	random2	kmeans++	forgy
欧式距离	0.5394	0.5827	0.5402	0.6159
曼哈顿距离	0.4505	0.4504	0.4495	0.4819
切比雪夫距离	0.2913	0.2703	0.3389	0.2852
闵可夫斯基距离	0.2284	0.3149	0.3043	0.3036

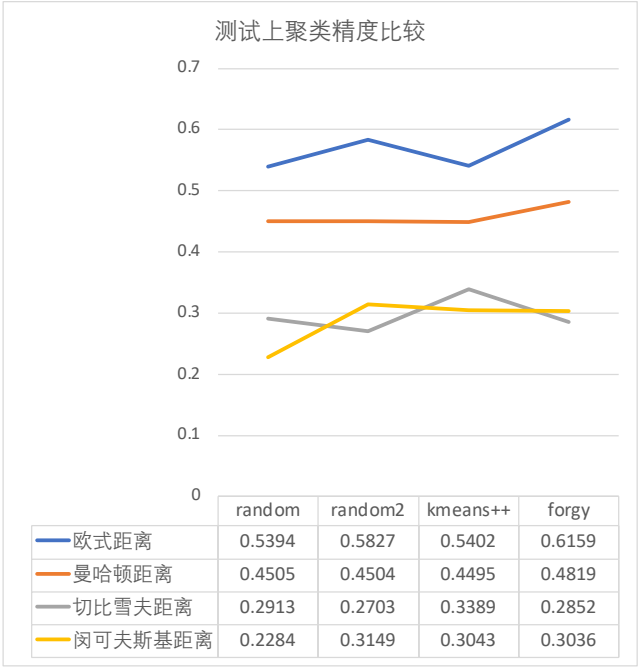
测试集上的聚类精度

	random	random2	kmeans++	forgy
欧式距离	0.5469	0.5940	0.5476	0.6275
曼哈顿距离	0.4490	0.4499	0.4486	0.4916
切比雪夫距离	0.2954	0.2716	0.3442	0.2795
闵可夫斯基距离	0.2363	0.3280	0.3131	0.3127

耗时（单位秒）

	random	random2	kmeans++	forgy
欧式距离	130.47	66.77	147.45	136.80
曼哈顿距离	712.75	1890.24	221.76	252.64
切比雪夫距离	205.19	194.81	206.83	207.54
闵可夫斯基距离	1350.82	786.91	818.28	755.60

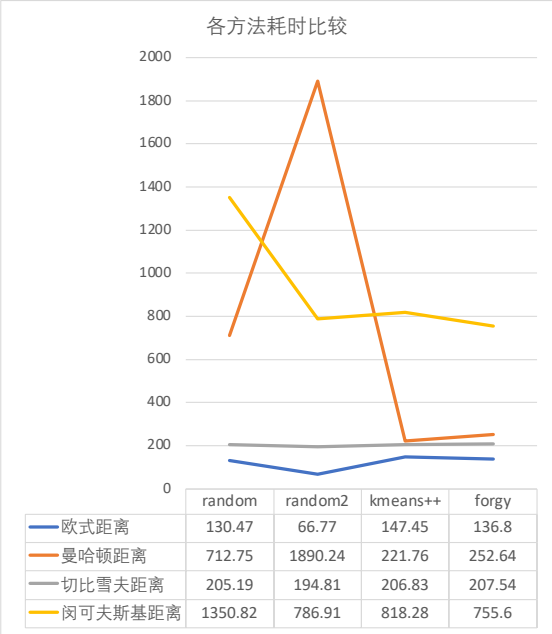




可以看到距离度量方法中，欧式距离在本数据集上的的效果最好，这与数据分布的特点有关，在其他数据集上不一

定也是欧式距离效果最好。然后可以看到对于不同初始化方法，其效果差异并不像距离度量方法导致的差异这么显著，原因可能是数据簇的数量比较多，且数据量比较大，数据分布范围比较广，导致随机初始就已经可以使聚类中心均匀地分布在样本空间中，所以 kmean++相比其他几种随机方式并没有带来显著的提升。不过在训练中，随机初始化可能会出现某次聚类精度较低的情况，这主要是因为可能出现某次初始随机选择的数据不太好，而 kmeans++理论上则会更加稳定。

下图为训练耗时比较，可以看到欧式距离是耗时最少的，从而在这个数据集上比较适合使用欧式距离，在耗时最少的同时还有最高的聚类精度。



	random	random2	kmeans++	forgy
欧式距离	130.47	66.77	147.45	136.8
曼哈顿距离	712.75	1890.24	221.76	252.64
切比雪夫距离	205.19	194.81	206.83	207.54
闵可夫斯基距离	1350.82	786.91	818.28	755.6

二、EM 算法训练 GMM 模型

（1）算法流程简述

期望最大化算法通常用于训练混合高斯模型。GMM 是一种概率模型，用于表示多个高斯分布的线性组合。以下是 EM 算法训练 GMM 模型的基本流程：

1.初始化参数： 随机初始化 GMM 的参数，包括每个高斯分布的均值（μ）、协方差矩阵（Σ）、和混合系数（π）。

2.E 步（Expectation）： 对于每个数据点，计算它属于每个高斯分布的概率（后验概率）。这通常使用贝叶斯定理来完成。E 步的目标是计算隐变量的期望。计算公式如下：

$$Q_i(z^{(i)}=k)=\frac{\phi_k\cdot\frac{1}{\sqrt{2\pi}\sigma_k}\exp[-\frac{(x^{(i)}-\mu_k)^2}{2\sigma_k^2}]}{\sum_{k=1}^{K}\phi_k\cdot\frac{1}{\sqrt{2\pi}\sigma_k}\exp[-\frac{(x^{(i)}-\mu_k)^2}{2\sigma_k^2}]}$$

3.M 步（Maximization）： 使用 E 步中计算得到的后验概率，更新高斯分布的参数。具体来说，对于每个高斯分布，更新均值、协方差矩阵和混合系数，使得似然函数最大化。计算公式如下：

$$\mu_k=\frac{\sum_{i=1}^{N}Q_k^{(i)}x^{(i)}}{N_k}$$

$$\sigma_k=\frac{\sum_{i=1}^{N}Q_k^{(i)}(x^{(i)}-\mu_{(k)})(x^{(i)}-\mu_k)^T}{N_k}$$

$$\phi_k=\frac{\sum_{i=1}^{N}Q_k^{(i)}}{N}$$

$$N_k=\sum_{i=1}^{N}Q_k^{(i)}$$

4.计算似然值： 计算当前参数下数据的似然值，用于后续的收敛判断。
5.收敛判断： 检查似然值的变化是否足够小，或者检查参数的变化是否足够小。如果满足停止条件，则算法收敛；否则，返回第 2 步，继续迭代。
6.输出结果： 当算法收敛时，输出训练得到的 GMM 模型的参数，包括每个高斯分布的均值、协方差矩阵和混合系数。

（2）实验

实验设置仍然是训练 100 轮，标签匹配也采用匈牙利算法通过训练数据集得到匹配函数，测试集上使用训练集得到的匹配函数，不重新匹配。

训练技巧：
1.使用了 log-sum-exp trick 进行对数值的稳定计算，在某些比较极端的初始化情况下，后验概率值会很小，在还没有归一化之前就已经下溢导致数值为 0，通过取对数计算，计算完成后再还原可以防止在计算过程中出现下溢。
2.获得初始协方差矩阵后在其对角线上添加一个极小值，由于某些初始化矩阵不是正定矩阵无法计算后验概率，通过加值操作可以保证协方差矩阵正定。
3.使用 nan_to_num 函数和除 0 检测防止异常值，某些情况下可能出现违法运算规则的操作，比如某时归到某个类中样本数量为 0，这就会导致在最大化中出现除 0 的错误，通过给除数加一个极小值可以防止出现除 0。一些异常运算也可以在检测到后跳过对应操作保持原值。

使用了下面四种协方差矩阵初始化方法：
方法一：使用训练集数据的协方差矩阵作为每个分布的初始协方差矩阵
方法二：将训练数据的方差缩小构造出方差更小的协方差矩阵作为每个分布的初始协方差矩阵
方法三：使用单位矩阵作为每个分布的初始协方差矩阵
方法四：方法三的基础上冻结权重参数，训练中权重均为 0.1

训练集上的聚类精度

	方法一	方法二	方法三	方法四
kmeans++	0.2001	0.1599	0.2762	0.2782
forgy	0.2269	0.1665	0.2877	0.3196

测试集上的聚类精度

	方法一	方法二	方法三	方法四
kmeans++	0.1973	0.1530	0.2749	0.2837
forgy	0.2224	0.1690	0.2876	0.3240

方法一使用训练数据的协方差矩阵作为分布的初始协方差矩阵，可以看到性能不能理想。使用全部类整体的协方差会导致分布的覆盖范围过大，容易出现某几个处于核心位置的高斯分布覆盖了全部样本，导致第一轮训练得到的后验中，样本属于某个核心高斯分布的概率接近 1，属于其他高斯分布的概率接近 0，这在后面的训练中很难改变，最终会导致少数几个核心高斯分布权重很大，其他高斯分布无法发挥作用，所以效果并不好。
方法二尝试减小协方差矩阵的大小，但是效果并没有提升，分析发现原因主要是其效果受减小比例这个超参数影响较大，比较好的减小确实观察到第一轮后验中分布更加分散了，样本能够被划分到各个高斯分布中，有助于后面进一步的划分。但是很多情况减小后反而分布更加集中了，样本有时全被分到一个高斯分布中，这样后面训练中其他高斯分布很难划分到样本。
方法三直接使用了单位矩阵初始化，观察其第一轮的后验可以看到其分布是比较好的，样本被比较均匀的归类到不同高斯分布，但是其后验概率呈现一家独大的状况，即属于某高斯分布的概率直接为 1，属于其他高斯分布的概率为 0，并且在后面的训练中这种情况不会被改变。实际上这几种方法都有这种情况，方法三由于直接使用了很小的方差所以会更为明显，然而直觉上认为，初始时应样本属于某几个类的概率都比较高会好一点，这样有利于训练中各类之间的博弈。

方法四是针对于权重过小的情况，通过分析发现，某些类由于本来划分到的样本数不是很多，随着训练的进行会权重可能会被减小，到最后权重接近 0，这相当于直接失去了一个簇。直觉上 10 分类中各高斯分布的权重差异不应该过大，于是这里把权重均冻结为了 0.1，使得每个高斯分布都必须被得到重视。可以看到修改后性能有明显提升。

这里还调用了 sklearn 库尝试了一下已有的成熟流程所能达到的效果，结果如下：

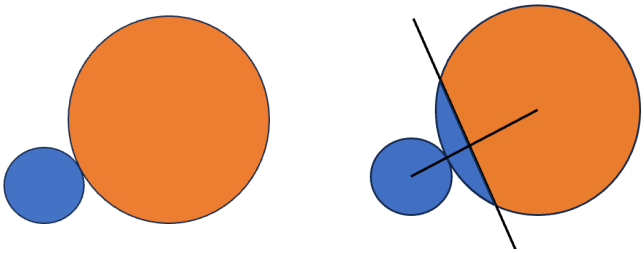
协方差矩阵类型	full	tied	diag	spherical
训练集上聚类精度	35.76%	34.80%	34.84%	50.95%
测试集上聚类精度	34.66%	34.49%	34.43%	50.95%
耗时（秒）	921.94	492.09	29.89	53.74

其中，sklearn 库中的协方差矩阵类型参数含义如下：
full：每个分布都有自己的协方差矩阵，每个协方差矩阵是一个完整的 n x n 矩阵，其中 n 是特征的数量。
tied：所有分布共享一个协方差矩阵，全局共享的协方差矩阵也是一个 n x n 矩阵。
diag：每个分布有一个对角协方差矩阵
spherical：每个分布有一个标量方差，而不是协方差矩阵。

可以看到，我们使用的方法相当于 full 方法，在 sklearn 库的结果中，使用协方差矩阵的三类方法 full、tied、diag 聚类精度均在 35%左右，性能并不理想。使用标准方差的 spherical 方法反而性能最好，聚类精度最高。这说明了样本各维度之间的差异是比较相近的，利用这个启发信息能够在减少耗时的同时达到更好的效果。

三、K-means 聚类方法和 EM 训练的 GMM 聚类方法比较

通过上面的实验结果我们可以发现，EM 训练的 GMM 聚类方法性能不如 K-means 聚类方法，然而理论中 EM 训练的 GMM 聚类方法拟合效果是比 K-means 聚类方法好的，比如在下面左图的两个类中

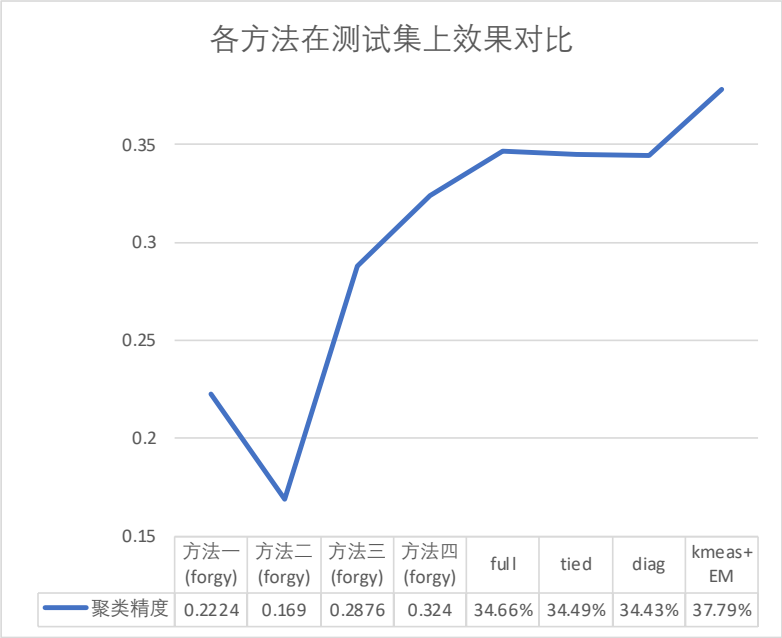


K-means 聚类方法是无法拟合出这种情况的，因为即使聚类中心碰巧能达到对应位置，但是由于数据分布的特点，K-means 聚类方法是无法收敛到期望的位置的，中间图所示。但是 EM 训练的 GMM 聚类方法是能够拟合出这种分布的，由于高斯分布的方差可变，所以能够左下的高斯分布通过较小的方差将对应数据覆盖。

但是由于高斯混合分布的拟合效果更好，所以导致其也更容易陷入了一些局部最优解当中，比如上面分析中出现的样本全被分到一个高斯分布中，这在 kmeans 算法中是几乎不会出现的。我们可以发现，这有点类似于前一个实验中全连接神经网络和卷积神经网络的对比，EM 训练的 GMM 聚类方法类似于用全连接神经网络训练高维数据，虽然其拥有更多参数，理论上全连接神经网络只要超过三层就能拟合任意函数，同样高斯混合模型也有更好的簇拟合效果，但是正是因为其解空间过大，导致出现了一样的问题，就是训练时间非常长，损失下降困难，性能反而还更差。而 kmeans 算法在这里反而类似卷积神经网络，其拥有更多的先验信息，比如刚才看到将权重冻结为 0.1 性能能够提升，而 kmeans 本身就自带了这个先验信息，还有在 sklearn 库的结果中可以看到只使用标准方差性能最好，而 kmeans 计算距离刚好也是各维度操作一致，这导致 kmeans 的解空间会更符合数据的分布，所以效果会更好。

为了进一步进行说明，这里再增加了一个 GMM 模型，其使用 kmeans 聚类完成后的聚类中心来初始化 GMM 的均值，训练结果如下：

	在训练集上	在测试集上
聚类精度	0.3771	0.3779



我们可以看到，在使用了 kmeans 算法的结果作为先验信息后，在所用使用协方差矩阵的方法中，这种 kmeans+EM 的方法取得了最好的效果，性能甚至超过了 sklearn 库中已被大量优化过的函数。如果在 spherical 方法上进一步训练，效果可能最终超过 kmeans 算法。我们可以得知，EM 训练的 GMM 聚类方法主要的问题仍然是缺少先验信息，模型过于复杂，参数量很大，但是最终效果反而是欠拟合，尽管其拟合能力很强，同时训练时间非常久也是一个问题。如果能够为其提供先验信息和一些启发信息，其效果能够得到明显的提升。但这种先验信息一般是较难明确得知的，在本次实验中，使用的这种先简单模型再进一步复杂模型的方法，或许能够在其它网络中尝试其能否取得良好效果，比如将一个复杂模型逐步拆解，再从小到大依次训练。由于时间有限，这方面的资料将在之后再进一步查看。