Assignment 3

part 1 生成数据

因为和 Assignment 1 中的数据类似，可以重复使用代码。

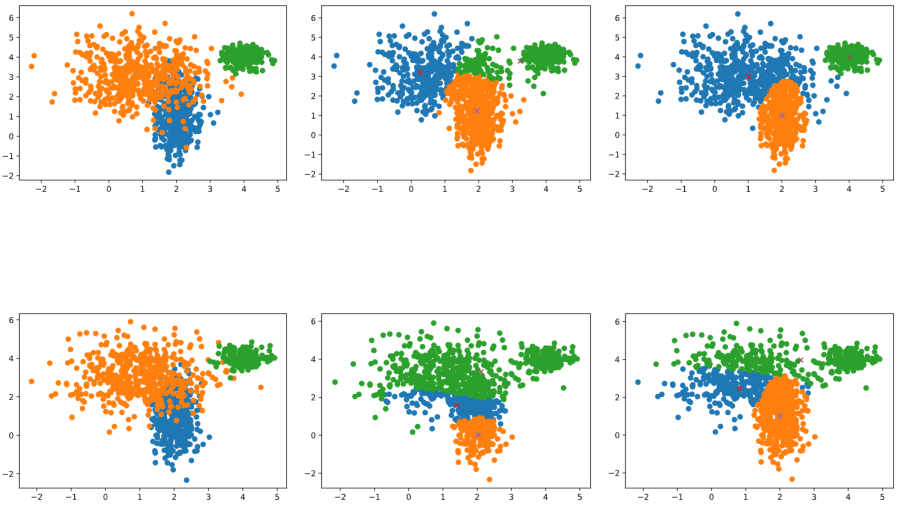
part 2 GMM 模型

1.模型介绍

使用高斯混合模型对数据进行聚类，反复执行 E 步与 M 步，直到模型收敛。两步计算方式如课件中所述，不断迭代，对 μ, σ, π 进行更新。

1. 生成数据集
2. 随机若干个数据点作为 μ 的初值，设置 $\sigma = 0.1I$，其中 I 是单位矩阵，$\pi = 1/k$，其中 k 是高斯分布的数量。
3. E 步，根据现有的参数，计算每个点属于每个高斯分布的概率 $\gamma_{n,k}$
4. M 步，由现有概率更新参数
5. 重复 E 步， M步直到模型收敛。当连续 20 个 epoch 都满足相邻两个 epoch 判定结果不同的数据点个数小于 $n/150$ 时，则判定为收敛。其中 n 为总数据点数。

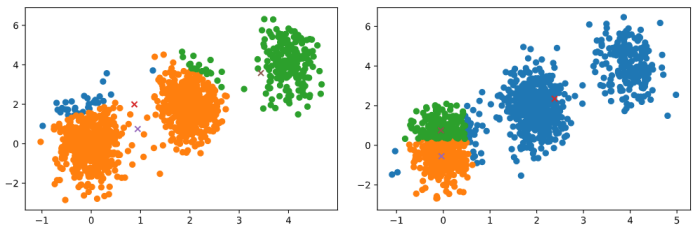
使用这种模型，可以将多数在肉眼可分的数据集成功聚类，但是随机到不利初值时，会收敛到效果很差的极值点。如下图所示，其中左、中、右图分别为采样点、epoch 0 的聚类结果、收敛的聚类结果。



可以发现，如果初值点能分别取在正确的高斯分布上，那么 GMM 模型可以得到很好的结果，但是如果初值点不好，那么有可能收敛到正确的结果，也可能收敛到聚类效果不好的结果。

2.初值选取

为了改进选取初值的办法，先考察一类特殊的数据。



在这样 k 个并排且几乎不交的高斯分布中，如果初值成功在每个高斯分布中选择了一个，那么几乎一定能够正确收敛。如果某个高斯分布中存在多于一个初值点，那么有可能这个高斯分布中的数据被聚类为大小相等的两个类别。

一种可行的办法是多次选取初值进行训练，在最终结果中选择对数边际似然函数最大的一套参数。但是这种方法在高斯分布变多时成功概率并不高。

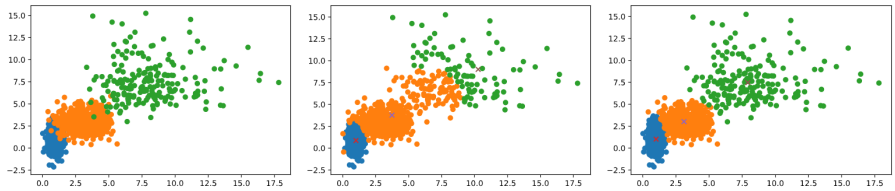
设有 k 个高斯分布，那么能够成功在每个分布中选到一个点的概率是 $\frac{k!}{k^k}$。当 k 变大时，成功率快速降低。

田野同学提出了使用 kmeans++ 初始化的方法。

3.改进的初值选取与模型分析

1. 生成数据集
2. 首先对于数据进行 $epoch_0$ 次 kmeans++ 的迭代，将聚类中心作为 μ 的初值，设置 $\sigma = 0.1I$，其中 I 是单位矩阵，$\pi = 1/k$，其中 k 是高斯分布的数量。
3. E 步，根据现有的参数，计算每个点属于每个高斯分布的概率 $\gamma_{n,k}$
4. M 步，由现有概率更新参数
5. 重复 E 步， M步直到模型收敛。当连续 20 个 epoch 都满足相邻两个 epoch 判定结果不同的数据点个数小于 $n/150$ 时，则判定为收敛。其中 n 为总数据点数。

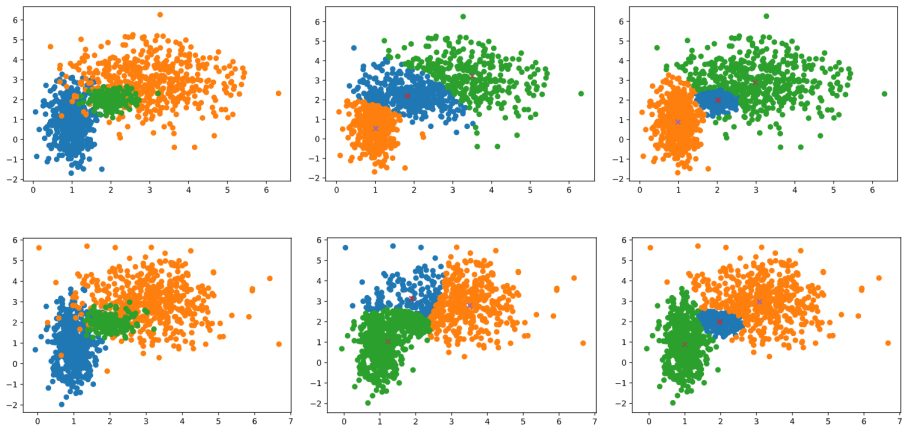
通过这种方式，上述的数据集在初始化步就可以成功将 3 个类别区分开，从而可以得到较好的聚类结果。甚至在非高斯分布中，也能得到一些成功的聚类。



如图，其中蓝色和橙色数据点为高斯分布的采样结果，绿色数据点为首先从高斯分布中采样出 (x_i, y_i)，然后作用 $exp()$ 函数，得到 (e^{x_i}, e^{y_i}) 作为采样点。

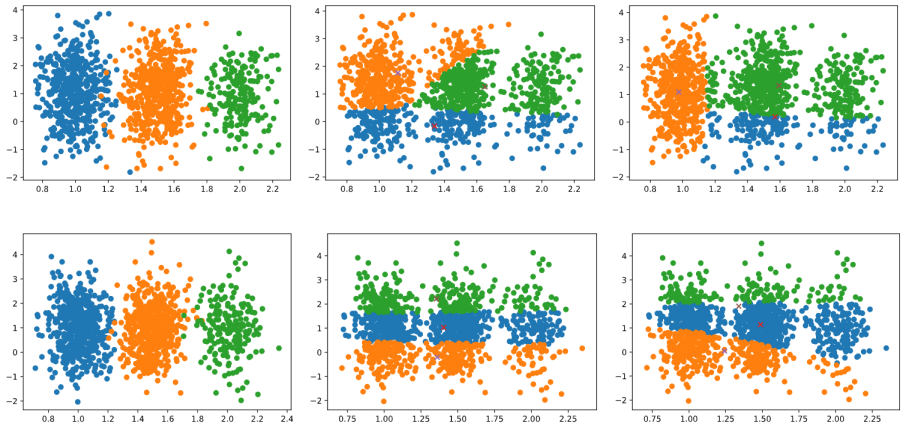
可以看出，kmeans++ 初始化倾向于在空间中平均分配各个分布。但是经过 GMM 的迭代，可以将橙色与绿色数据区分开。

下面两组中，第一组为 $epoch_0 = 10$ 的正常取值的结果（这里也可以看到 kmeans++ 的劣势）。第二组为 $epoch_0 = 0$的结果，即仅挑 k 个距离尽可能远的点作为初值。这两种初值的选择都不是非常好，但是也不是很差的极值点。GMM 都可以成功将中间一块分布密集的高斯分布区分出来。



3.仍有的不足

当上述困难数据变得更加困难时，这个模型仍然会得到和答案不同的聚类。如图，上图参数为 $epoch_0 = 0$ 随机挑选初值，下图参数为 $epoch_0 = 10$ 用 kmeans++ 聚类的结果作为初值。二者都会得到与答案不同的聚类，并且这两种聚类也不同。



并且在 kmeans++ 表现较差的数据上，如果 $epoch_0$ 过大，使得模型初值收敛到 kmeans++ 给出的聚类上时，可能会导致 GMM 模型随之进入了较差的极值点。所以在实际操作中，如果需要使用上述模型，应该小心地选择 $eppoch_0$，以得到较好的结果。

part 3 总结

综上，结合 kmeans++ 和 GMM 模型可以弥补两种算法的缺点，较好地解决聚类问题，并且当数据边界明显时也会大概率收敛至相应的分布。但是仍有些对抗样本能够使得模型收敛到和预期不同的聚类。

part 4 代码运行方式

```
python source.py
```

将自动生成数据并以 $epoch_0 = 2$ 运行改进后的模型。其中数据分布如下：三个分布的协方差都为单位矩阵，中心分别为 $(1,1),(-2,3),(4,4)$，采样数量分别为 $500,500,200$。该模型以大概率成功将这组数据聚类。