机器学习 hw1

1. K-mean vs GMM

传统的 k-mean 算法通过初始化 k 个点，然后进行迭代，不断更新 k 个点的位置，直到每个簇的质心不再改变，此时 k-mean 算法结束。

而 GMM 算法则是一种混合模型，模型里面的单一分布都是高斯分布。采用 EM 算法来求解变量。

相对于 k-mean 的硬分类，GMM 算法实现的是软分类，因此具有更好的鲁棒性。

传统的 k-mean 的算法有一个不足的点就是对于 k 个点的随机初始化会导致一些失败的情况，比如某些点被饿死或者错误的聚类。因此改进的算法中可以修改 k 个聚类点的初始化来防止以上的情况发生。

算法想法 :采用从 1 个开始增加聚类点的数目。即初始化时确定第一个聚类点(即质心)，此时第一个质心即为所有点的均值。然后再将所有的簇一分为二 (刚开始时只有一个簇)，选择一分为二后误差平方和 (SSE) 最小的划分，即为当前最有的划分。继续循环，知道簇的个数满足所要求的个数。

Algorithm 1:

输入：需求的聚类点个数 k

输出：正确的聚类个数下的坐标

伪代码：

```
while(当前簇的数目<k)
{
    for(每个簇)
    {
        记录当前簇的总误差
        对这个簇进行 k=2 的 k-mean 算法
        计算一分为二之后的总误差
        与当前最小的 SSE 进行比较
    }
    找到当前数目下 SSE 最小的划分，并划分，进行下一次循环
}
```

优点：相对于传统的 k-mean 算法，具有更好的分类效果。不会出现初始聚类中心会聚类到一个簇中，一定程度上避免了局部最优解的状态。

缺点：仍不能避免出现局部最优解的情况。

2. K-mean vs CL

相似点：

1. 都是一种分类的算法，并且都是需要比较好的初始化才会有比较不错的分类效果
2. 都是一种 hard-cluster
3. 使用的分类依据是根据欧拉距离

不同点：

1. CL 在运行时会动态地调整中心点的位置，这样会不断显著的分离出每一类别
2. K-mean 的实现以及整体想法更为简洁，而 CL 以及 RPCL 的算法实现会相对复杂
3. K-mean 的算法是通过不断地迭代实现完成的

RPCL 在 K-mean 中的应用：

K-mean 算法在实现的过程中会遇到几个无法避免的问题，一个是 k 值大小的选定以及 k 个点的初始化的坐标的确定。这两个关键的因素决定了整个 k-mean 算法最后的效果。而 RPCL（次者受罚竞争学习）可以解决 k-mean 的这两个问题。RPCL 算法中的输入样本在将获胜节点引向它的同时，会将次胜节点推开。最终，每个类簇将一个权向量引向其中心，而以斥力阻止相近的权向量靠近它。

作业中算法的实现：

首先用了 sklearn 库中的数据生成函数自定义生成了 100 个，分为三类的数据，这是之后用来测试的数据。

RPCL 的核心算法具体分为三个部分。第一部分是随机生成几个中心(暂定为 5 个)，然后通过次者受罚竞争学习来不断调整 5 个中心点的位置。第二部分是给每个调整后的中心点分配所有数据点(通过距离判断)。第三部分是选择出几个比较好的初始化点，同时确定 k 的个数。

普通的 K-mean 和使用过 RPCL 优化过后的 K-mean 最后的 cluster 的优劣：

以下每一组图依次为：生成的数据点、普通 K-mean 聚类、RPCL-K-mean 聚类

第一组：

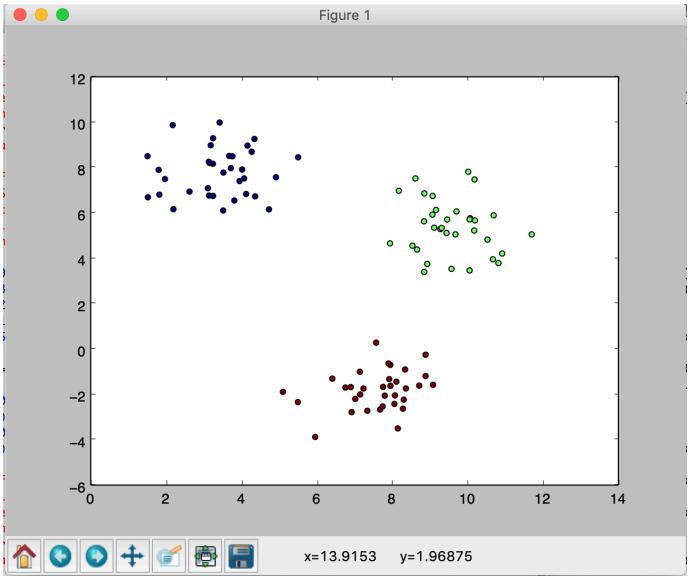


Figure 1
x=13.9153 y=1.96875

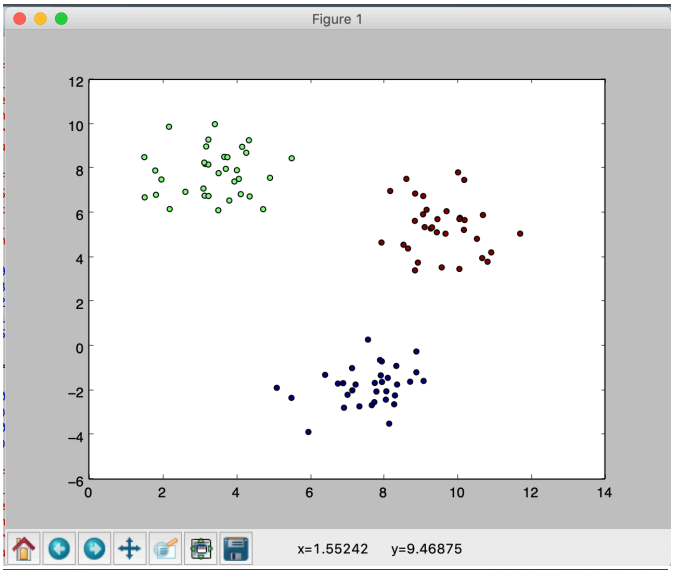


Figure 1
x=1.55242 y=9.46875

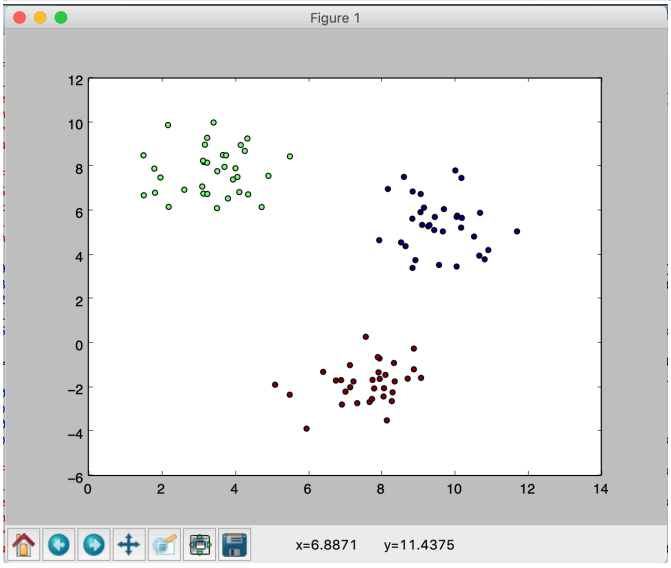
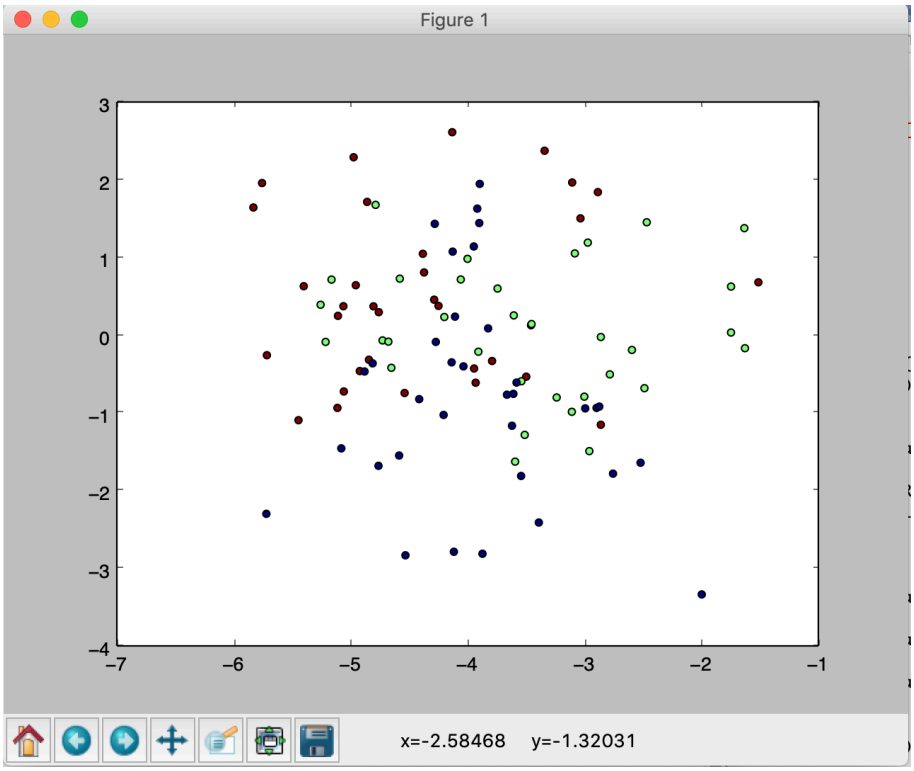
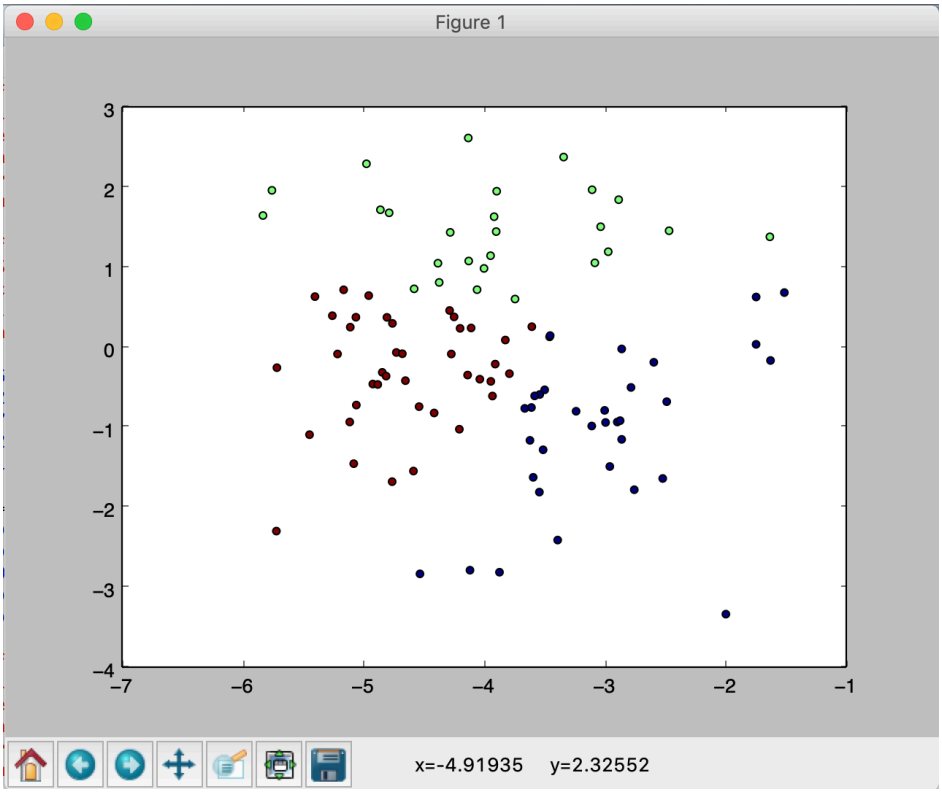


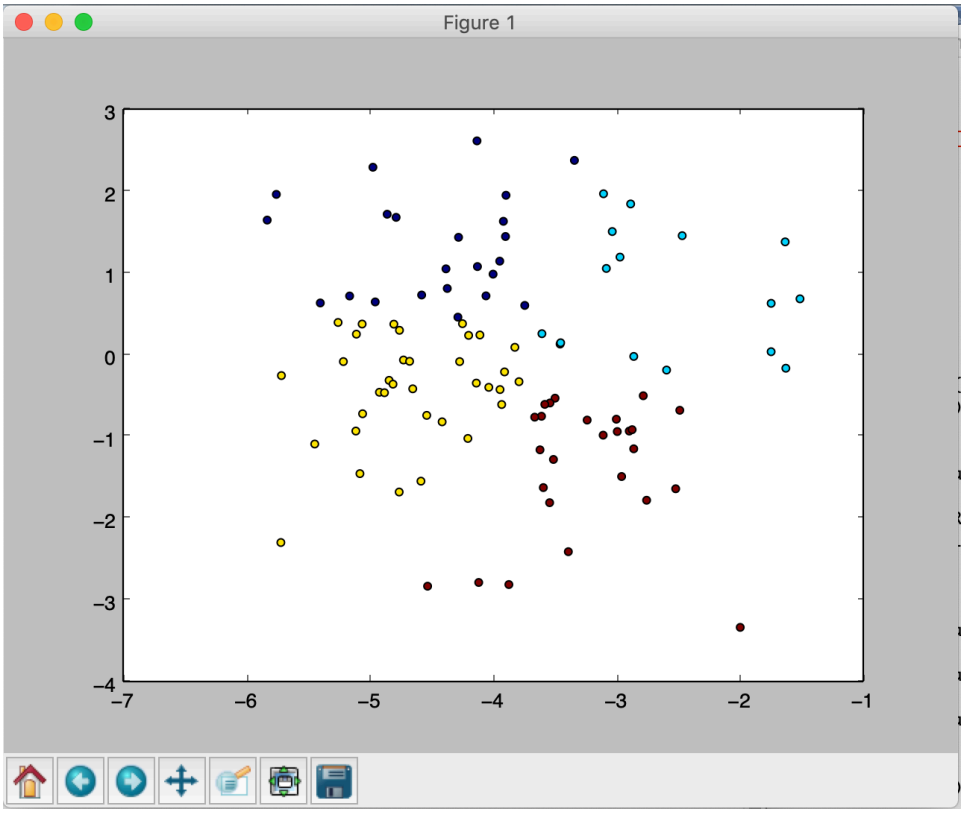
Figure 1
x=6.8871 y=11.4375

分析：从第一组来看，在原始数据分散比较开且聚类比较明显的时候，k-mean 和 RPCL-k-mean 的聚类效果差不多，都能准确的分成三类，其中 RPCL-k-mean 可以自己判断聚类个数而不用手动设定。

第二组：

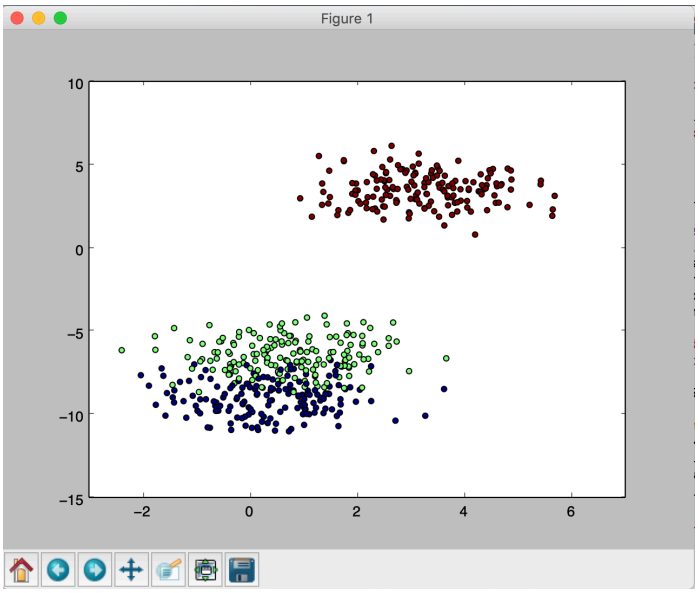


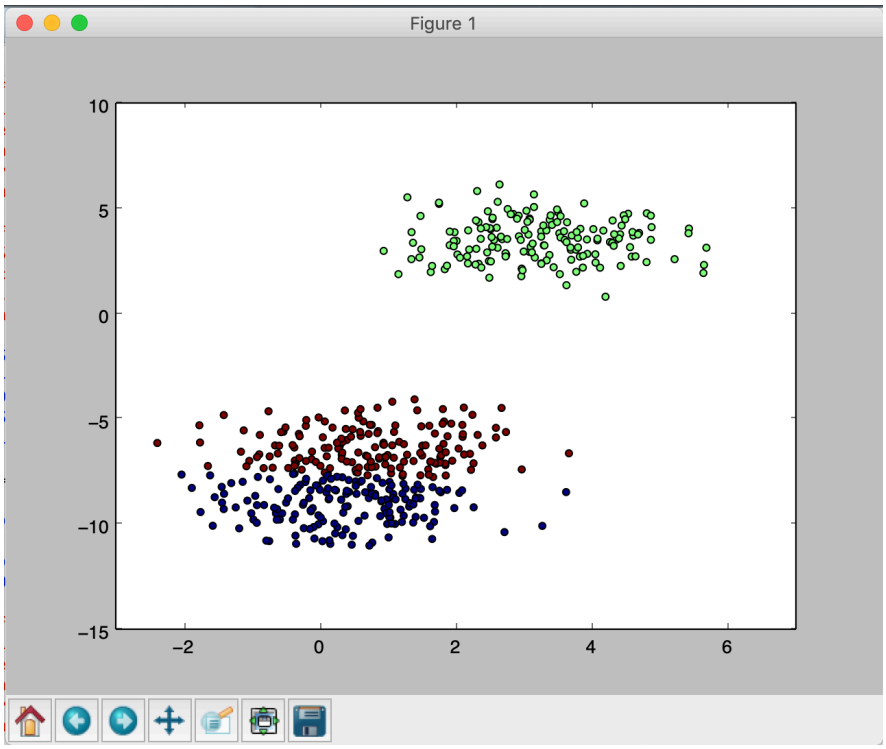


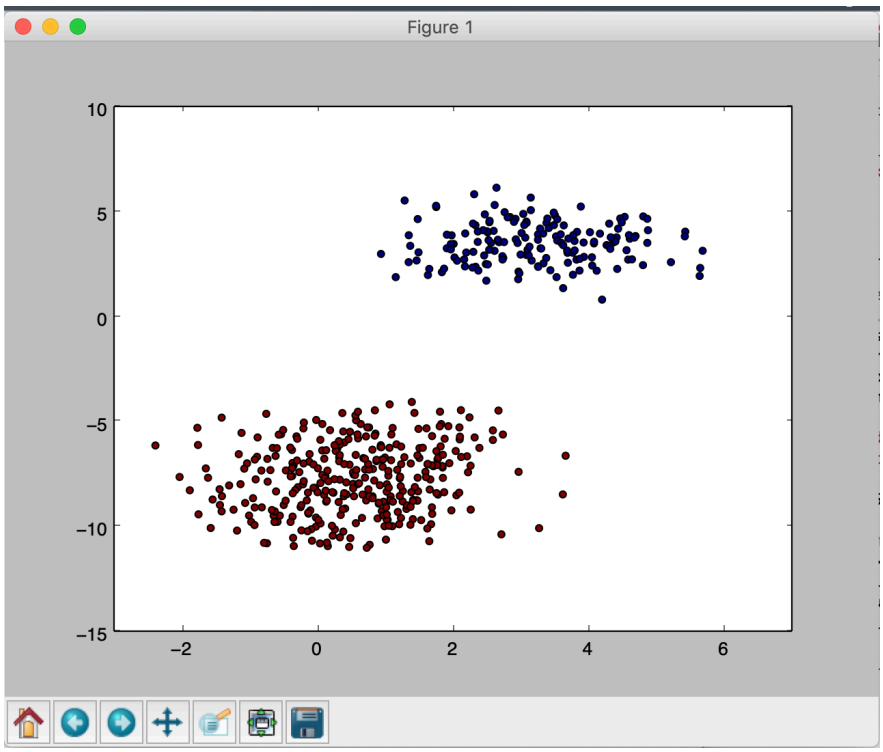


分析：从第二组数据看来，我个人认为这是 RPCL-K-mean 对于普通的 K-mean 比较有优势的一张。可以看出第一张图里的初始数据的生成的点非常杂乱，虽然是按照 3 个中心点生成的数据，但是直观上已经不能靠三个中心点进行聚类。而普通的 k-mean 只能按照预先给定的 k=3 进行聚类，而 RPCL-K-mean 因为是根据所有数据点自动进行初始点个数的确定和初始中心点的确定，所以判断出来所有点应该分成四类，最后分类结果是图三，我认为这是一个比较好的聚类。

第三组：







分析：这一组的数据从生成的数据来看显然是两个聚类，但是同样普通的 k-mean 仍然只能用给定的三个聚类进行分类，而 RPCL-k-mean 仍然成功的确定了 k 应该为 2 以及成功的聚类。

RPCL-K-mean 的优点与局限：

优点：

1. 能够根据生成的数据点自动确定 K 的取值一直初始化中心点位置的选择。
2. 通过 RPCL 与 k-mean 的结合改进了 k-mean 的聚类效果

局限：

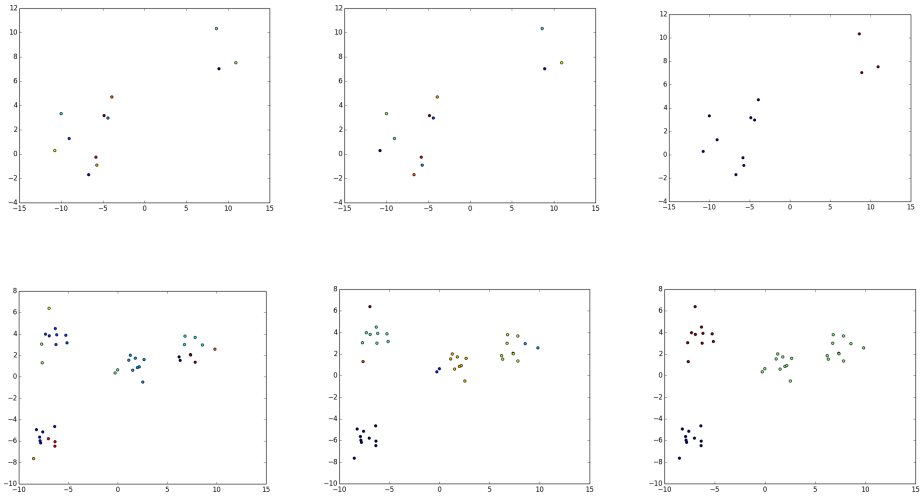
1. RPCL 算法的使用提高了整个算法的复杂性，本次作业中测试的 100 个数据点，而当我将数据点改成 500 个后，虽然聚类的效果仍然很好，但是运行程序所花费也会增加，这是一个不足的地方
2. 没有考虑到原来数据点在整个空间中的分布的密度是不用的，也许之后可以通过点分布的密度来改进算法
3. 确定聚类个数的条件在现阶段也有些不足（具体详见代码）

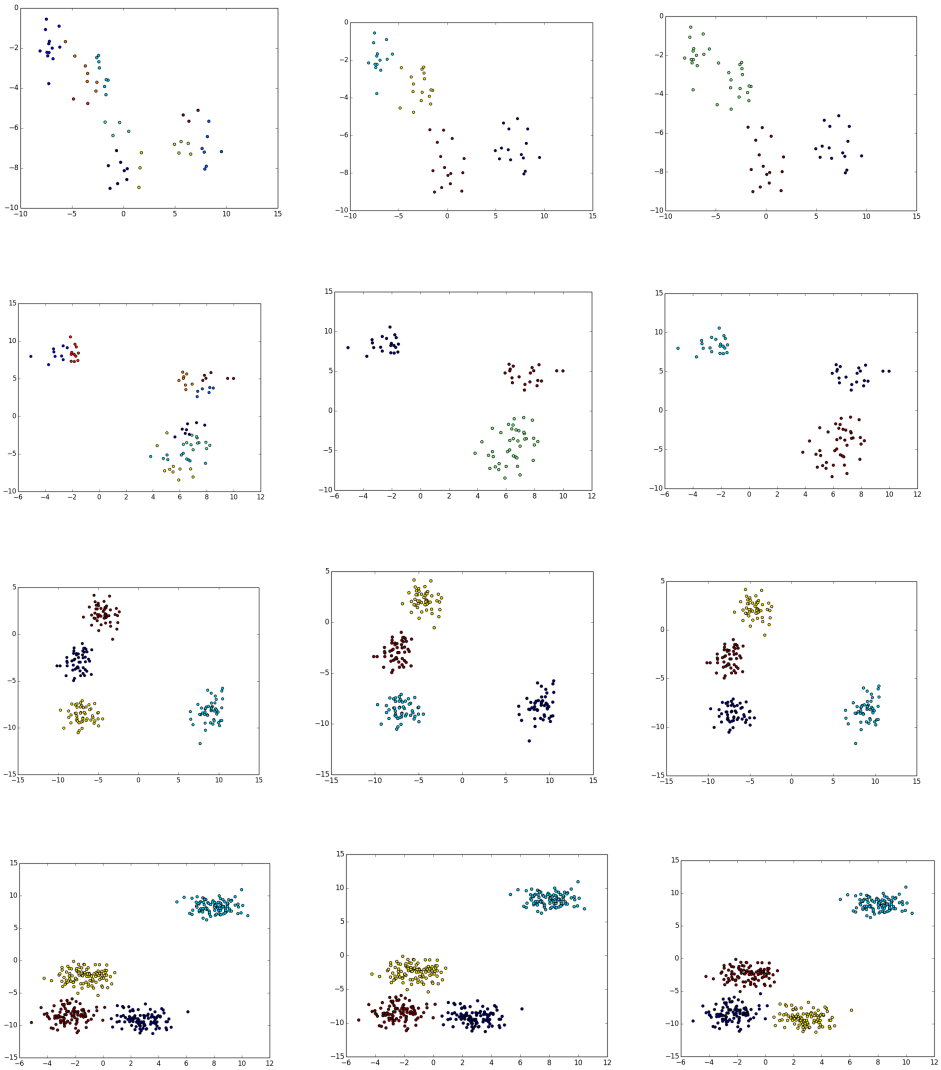
3. model selection of GMM

在这个任务中，我比较了 AIC，BIC 以及 VBEM 这三个不同的模型在不同情况下对于不同 sample size 以及不同 cluster 数目下的聚类效果。

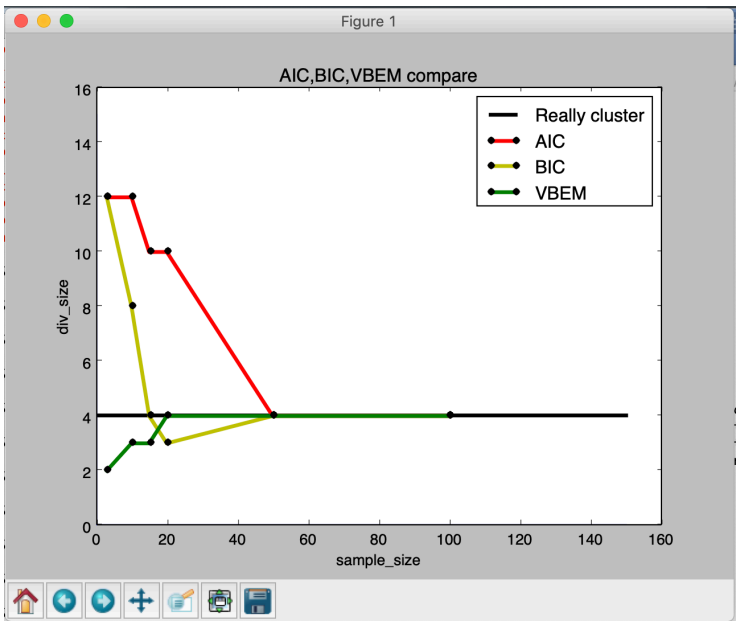
不同的 sample size：

我将随机产生的原始数据集确定为 4 个中心点，sample size 从小到大变化(3,10,15,20,50,100)，共有 6 组对比。





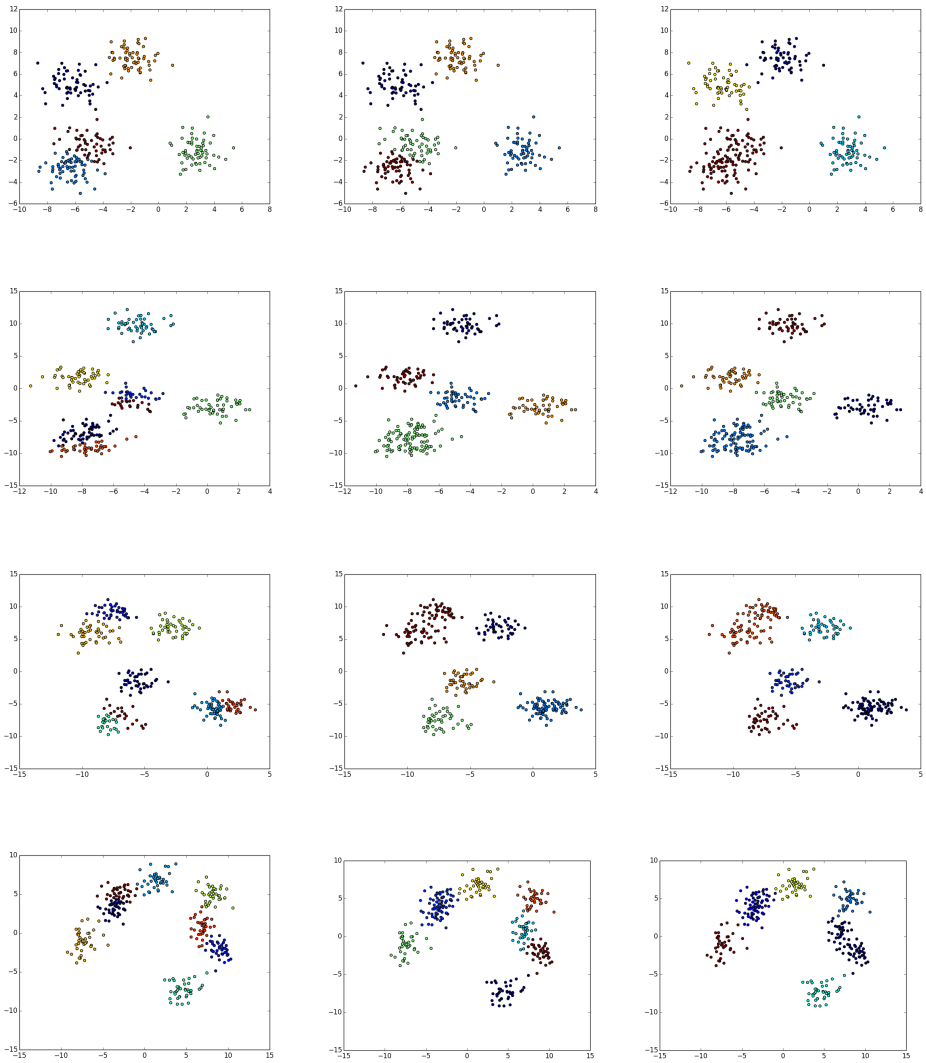
我将实验结果放到一张折线图中更好的显示：

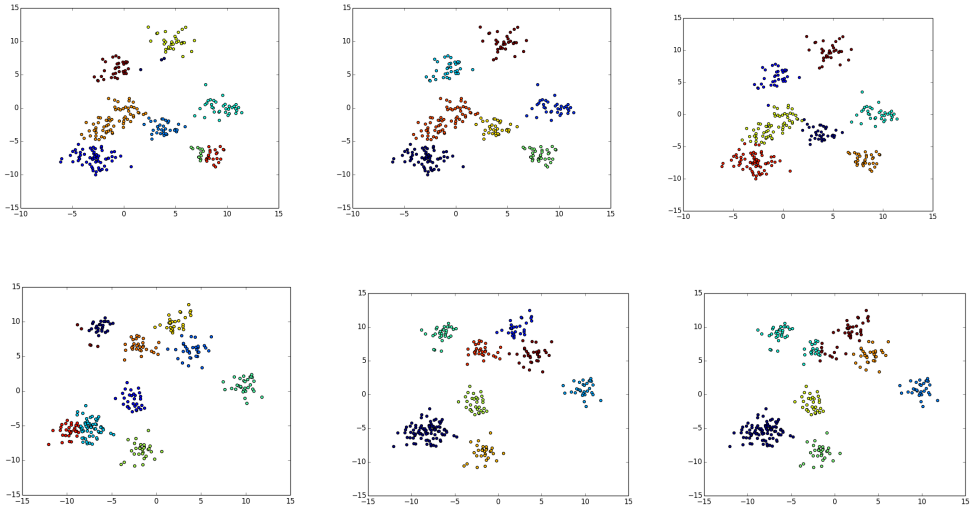


分析：生成的原始数据中，是以 4 个中心点生成的随机数据。而随着 sample size 从小到大(3,10,15,20,50,100)变化，AIC，BIC，VBEM 的分类效果逐渐变好，一般当 sample size 到 50 个左右，这三种分类方法能够较好的完成聚类。而比较这三种方法，可以看出聚类的效果从好到差依次是 VBEM>BIC>AIC，所以从实验数据来看，VBEM 在不同的 sample size 下是一种更好的聚类方法。

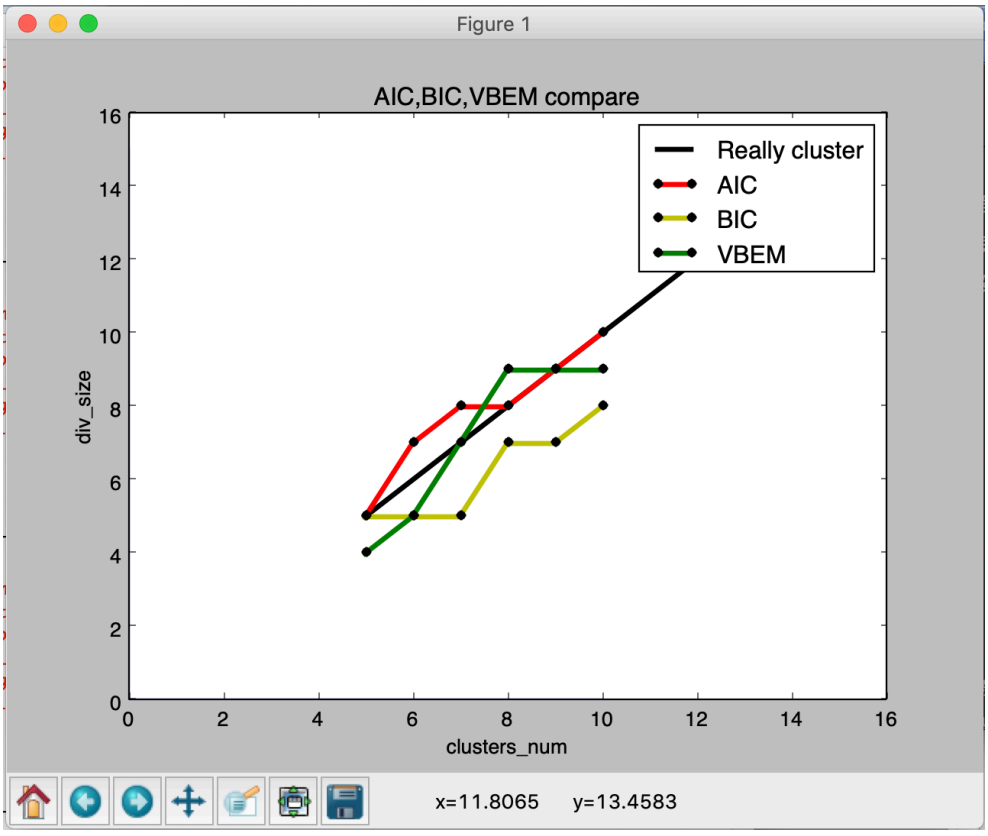
不同的 clusters 数量：

我将数据的中心点从 5 个一直加到 10 个，来判断每种模型在每个中心点数目下的聚类情况。共有 6 组对比。





同样将实验结果放到一张折线图中展示。



分析：我们需要的最好的聚类效果，是希望拟合到黑色的标准线上。从这个实验结果中可以看出，AIC 相对于 BIC 和 VBEM 是一种更好的模型。

4. 总结

在这次作业中，我完成了 k-mean 算法的改良，将 RPCL 算法与 k-mean 结合从而改进 k-mean 算法，以及比较 AIC，BIC，VBEM 这三个模型在 GMM 中的应用。

在这次作业中，我学到了更多的关于机器学习聚类方面的理论知识以及实践上的操作，对于每种聚类算法有了更深的了解。

但在这次作业中也有许多不足的地方，比如算法中仍有一些粗糙的地方还需要改进。

同时也非常感谢在这次作业中对于我给予很大帮助的涂老师和助教。