数据挖掘实验报告

成绩：

姓名	沈建鑫	学号	18051624
实验时间	12.4	实验地点	四教北 404
实验名称	聚类技术		

一、实验过程与问题

请列举实验过程中遇到的问题与对应的解决方案。

1.癌症数据集中有？数据，聚类的时候会出现问题，因为数量不多，直接删除即可。

二、实验结果与分析

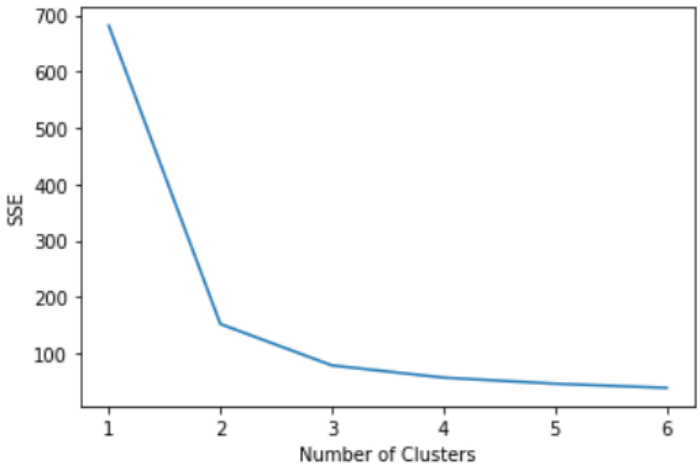
请回答实验代码文件（ipynb 文件）中的思考题

思考：1.对以下数据集使用 K 均值聚类算法，1）观察实验结果是否符合预期；2）利用 SSE 标准确定 K 值；3）自行调参并观察对聚类结果的影响。注意：需要把类别信息去掉。

1. “tutorial3_Data Exploration”中的鸢尾花数据集“iris.data”
2. “tutorial4_Data Preprocessing”中的癌症数据集“breast-cancer-wisconsin.data”

答：1.对于鸢尾花数据集，我们采用 SSE 对 K 值作了一个选择：

```
Out[15]: Text(0, 0.5, 'SSE')
```

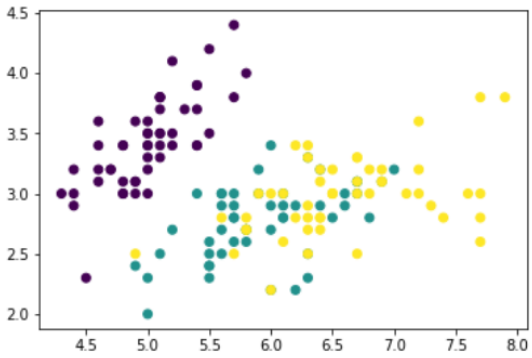


从图中可以看到，有两个拐点，而实际上鸢尾花数据集有三类数据，因此实验得到的结果和实际是符合的，k 值取 3 的时候效果比较好，说明 k_means 是有效的。

接下来利用可视化实现了一下效果图：

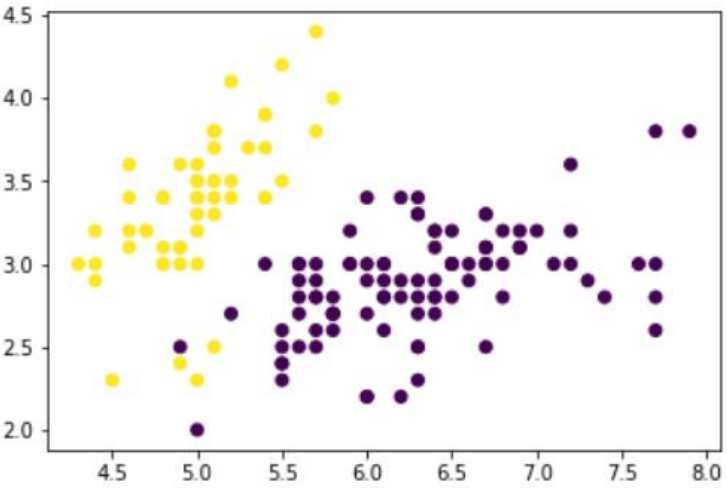
首先是原始数据（使用了前两维数据进行可视化，因为这里感觉没必要降维）：

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

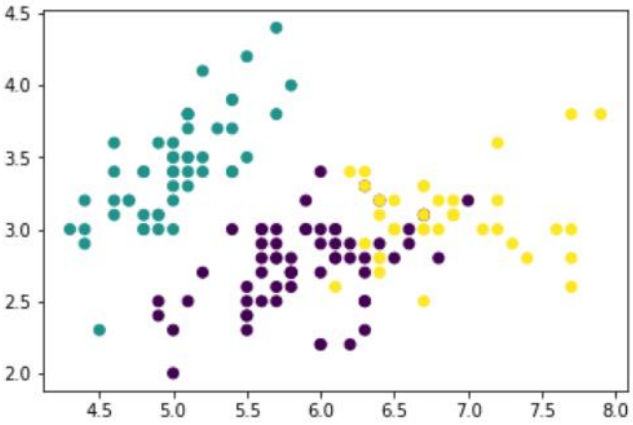


图中显示了原始数据集的前两维分布。

接下来我们分别用 k=2，k=3 看一下效果：

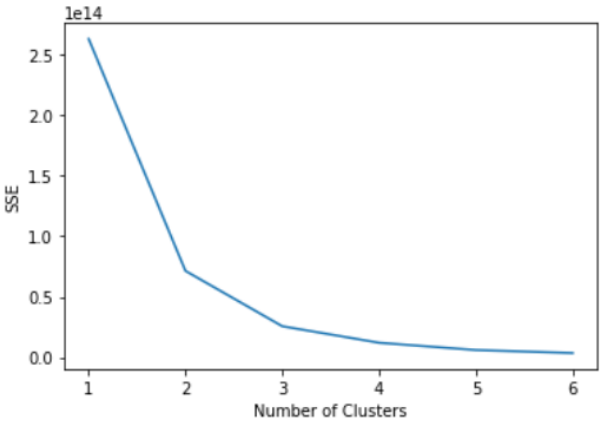


K=2,可以看到最左边一类差不多分开来了，但右边的一些还有待划分。



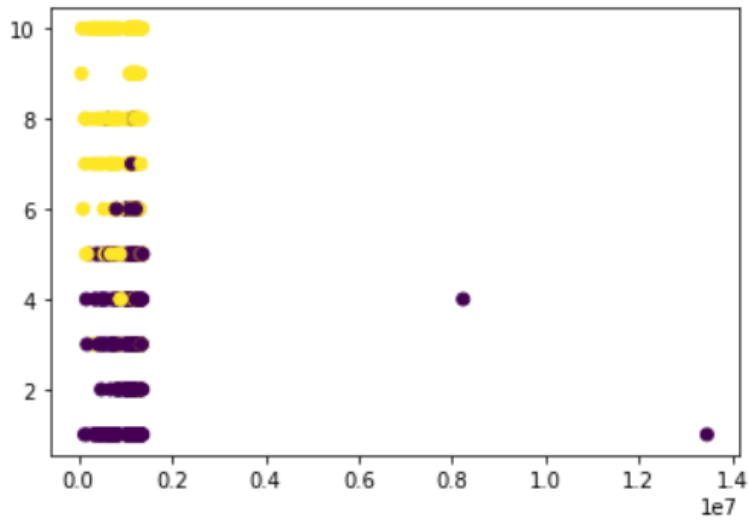
K=3，可以看到结果与原始数据基本一致了，只有个别点还存在问题，这在可接受范围内，因此 K-means 起到了不错的效果。

2.对于癌症数据集，用同样的方法，先查看 SSE

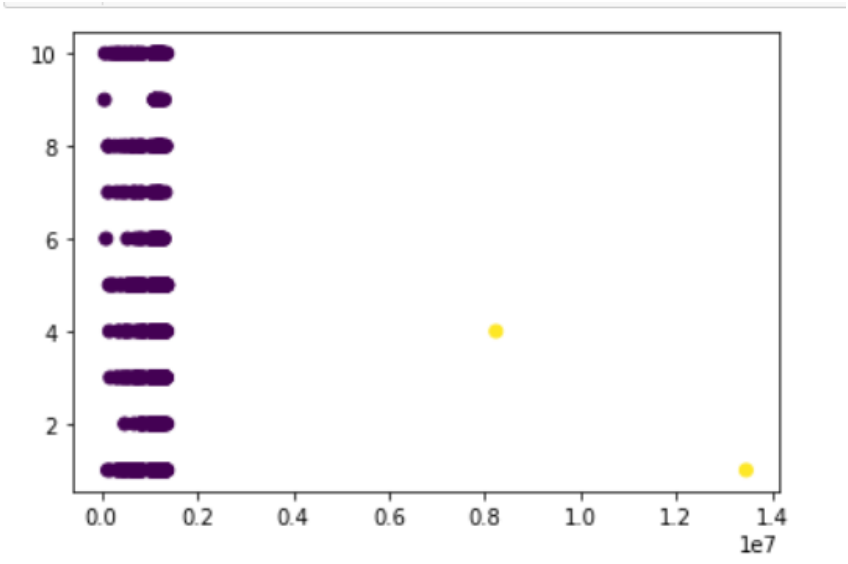


原始数据结果，百度了一下这个数据集最后一行是类别，根据这个做一下可视化：

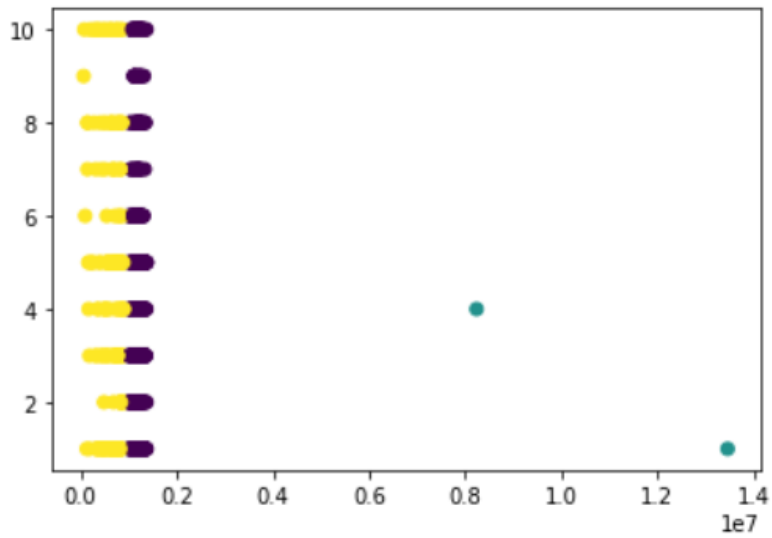
```
[[1002945 5 4 ... 2 1 2]
 [1015425 3 1 ... 1 1 2]
 [1016277 6 8 ... 7 1 2]
 ...
 [888820 5 10 ... 10 2 4]
 [897471 4 8 ... 6 1 4]
 [897471 4 8 ... 4 1 4]]
```



然后是 K=2：



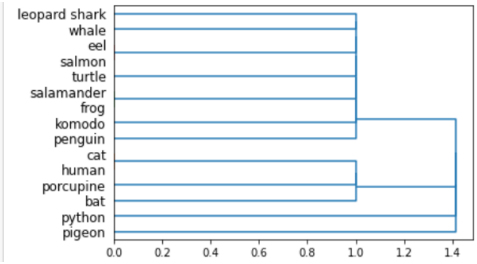
K=3：

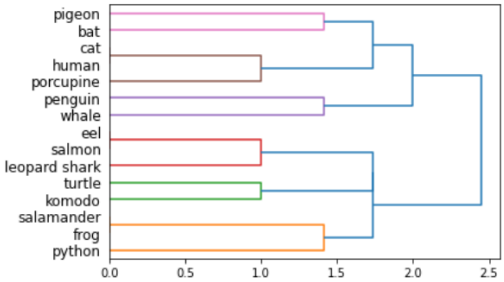


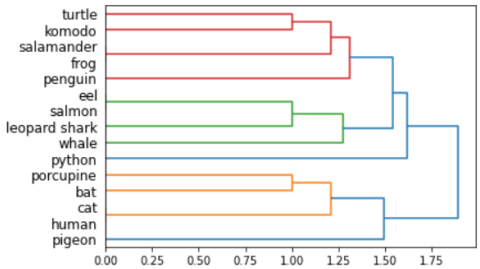
经实验发现，实验结果是达不到预期的，这可能和数据的原始分布有关系，说明 K-means 并不适合该数据分布的聚类。

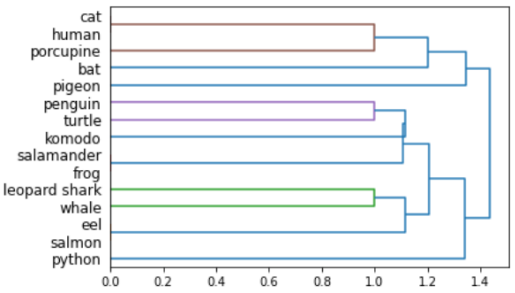
思考：2.结合你的生物知识，分析五种算法的结果是否符合你对生物分类体系的理解；对比五种算法的结果的异同，并结合其工作原理分析原因。

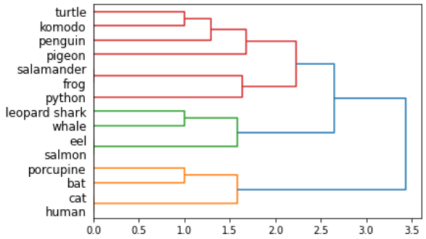
答：从左往右，从上往下依次为单链接、全链接、组平均值、质心、ward











针对我对生物分类体系的理解，首先，对于单链接的方法来讲，这个分类效果是非常差的，从图上我们可以看到，这个基本只能分成四类，而第一类中把 whale、komodo、frog、eel 等分在了一起，很明显是错误分类。再看全链接方法，虽然分的类别多了，但是效果还是不太行。将（pigeon、bat）分在一起很明显是错误的，（penguin 和 whale）也有问题，总的来说虽然不太行，但相比于单链接方法还是将一部分类别分开来了。对于组平均值，观察后可以发现大部分类别区分是正确的，但是还是有少部分类别分错，总的来说比之前两种方法要好。对于质心和 ward 的方法，结果和组平均值差不多，但是分类正确的类别不一样，可见其侧重点不同。

分析：

1.对于单链接方法，其实现原理是取两个类中距离最近的两个样本的距离作为这两个集合的距离，这是一种很明显的以偏概全的方法，在类中数据不聚集的情况下很容易造成聚类结果很差的情况，就如我们的实验结果，第一类中几个脊椎动物都有一维非常接近，因此非常容易分到一起。

2.全链接方法，其实现原理和单链接方法正好相反，是取两个集合中距离最远的两个点的距离作为两个集合的距离。这个方法和单链接的方法都有一个共同的问题，就是没有充分考虑类内距离，很难达到好的效果，与我们的实验结果也相符。

3.组平均值方法，这种方法就是把两个集合中的点两两的距离全部放在一起求均值，相对也能得到合适一点的结果，这种方法感觉上就比较合适一点，把每个维度的信息都考虑到了，只是会受到某些偏离值的影响会比较大，而实验证明确实是这种方法得到的结果比较好。

4.质心，重心距离，利用的是不同类的质心之间的距离最小值，这个和取中间值感觉差不多，实验结果也表明确实是侧重点不一样。

5.ward，这个方法是最短最长平均，离差平方和，也就是说任意两个 cluster 之间的距离就是这两个 cluster 合并后新 cluster 的 ESS，实验结果也和 3、4 差不多，侧重点不同。

思考：3. 查阅 scikit-learn 文档中的数据生成器（Samples generator，https://scikit-learn.org/stable/modules/classes.html#module-sklearn.datasets）请至少生成 5 种不同（形状或者分布）的数据集，并使用 DBScan 和谱聚类进行聚类分析，观察实验结果，结合算法原理进行分析。

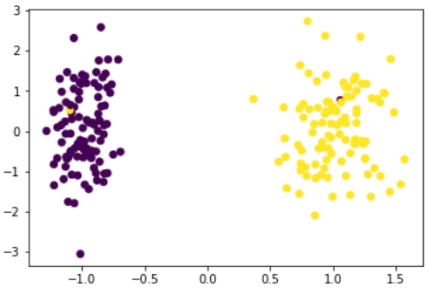
答：

一、采用 make_classification 生成，样本数为 200，特征为 2，n_redundant=0, n_informative=1, n_clusters_per_class=1

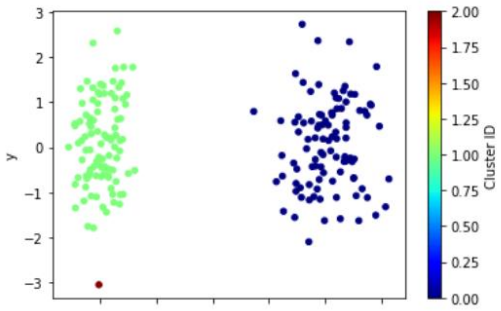
原始数据：

```
]:  1 X1, Y1=datasets.make_classification(n_samples=200, n_features=2, n_redundant=0, n_informative=1, n_clusters_per_class=1)
    2 plt.scatter(X1[:,0], X1[:,1], marker='o', c=Y1)
    3
```

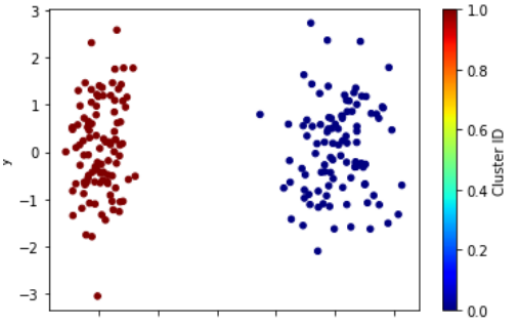
```
]: <matplotlib.collections.PathCollection at 0x24f830225f8>
```



DBScan 聚类:



谱聚类:



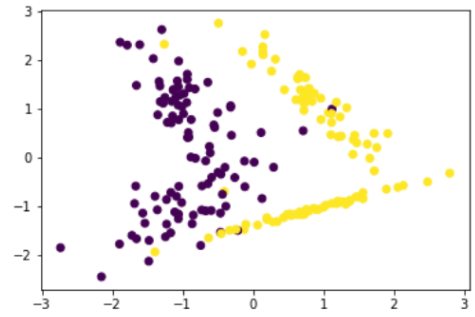
对于该数据分布，因为数据界限较为明显，因此两种方法通过调参之后都得到了比较好的结果，密度聚类因为通过低密度区域分隔的高密度区域来划分，因此会识别出噪声点，如图所示，最下方那个点即被识别出来，然后才是左右两边各自的类别。而谱聚类中通过高斯径向基函数来作为亲和度划分，不区分噪声点，直接划分为两类。

二、采用 make_classification 生成，样本数为 200，特征为 2，n_redundant=0, n_informative=2, n_clusters_per_class=2

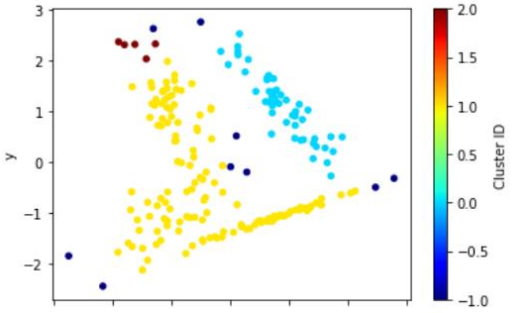
原始数据：

```
X2, Y2 = datasets.make_classification(n_samples=200, n_features=2, n_redundant=0, n_informative=2, n_clusters_per_class=2)
plt.scatter(X2[:,0], X2[:,1], marker='o', c=Y2)
```

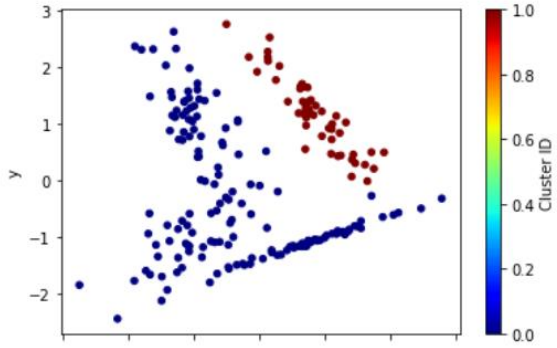
```
<matplotlib.collections.PathCollection at 0x24f830f6198>
```



DBScan 聚类：



谱聚类：

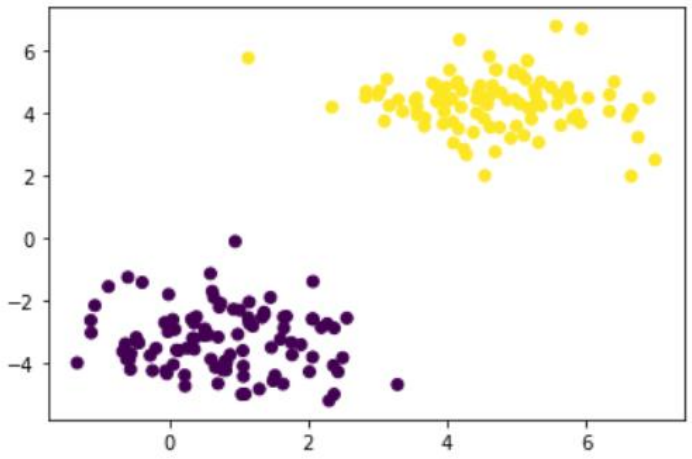


根据实验结果可以发现，当数据分布不是很明显的时候，密度聚类经过调参生成的结果最好如上图，会产生不少的噪声点，效果一般，而基于高斯径向基函数的谱聚类效果比密度聚类好上一点，这和它是基于矩阵相似度的原理有关系。虽然结果和我们预期的分类还是差了一些，但是单从数据角度来看的话似乎也还合理，只有右下角那一条划分错误。

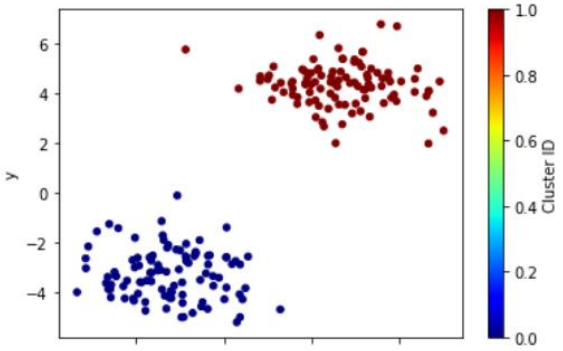
三、make_blobs,样本数 200,特征数为 2,centers=2

```
1  X3, Y3 = datasets.make_blobs(n_samples=200, n_features=2, centers=2)
2  plt.scatter(X3[:,0], X3[:,1], marker='o',c=Y3)
```

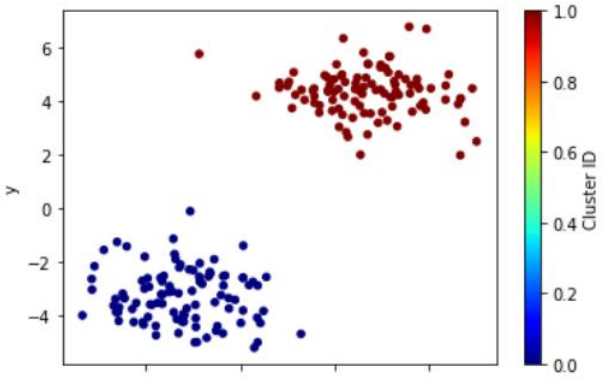
```
<matplotlib.collections.PathCollection at 0x24f824a6908>
```



DBScan 聚类:



谱聚类:

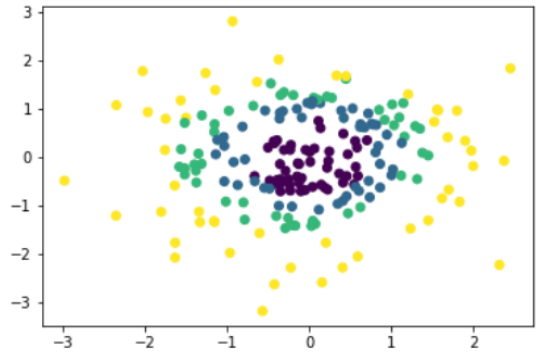


这个情况和一差不多，两个类别的分界线比较明显，在经过调整参数后两种聚类方法都能得到不错的结果。

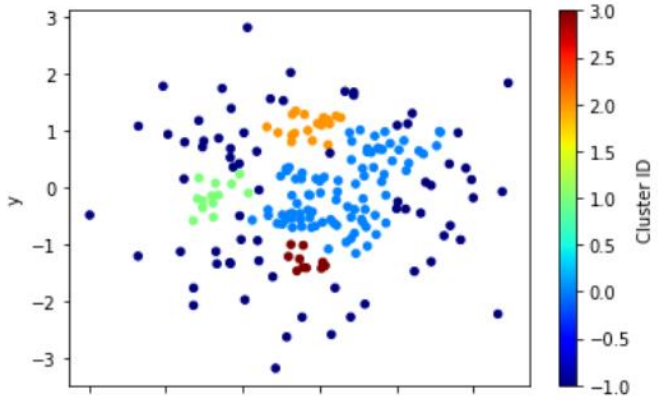
四、make_gaussian_quantiles，样本数为 200，特征数为 2，这里分成了四类

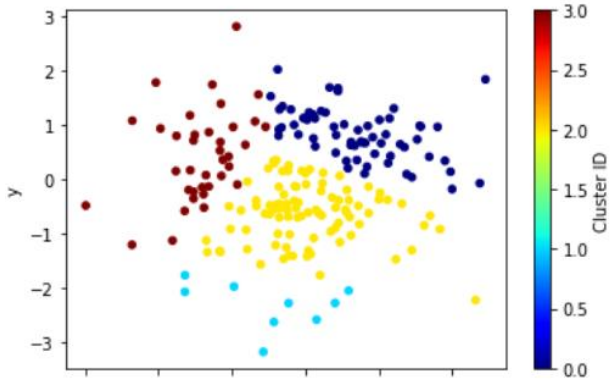
```
X4, Y4 = datasets.make_gaussian_quantiles(n_samples=200, n_features=2, n_classes=4)
plt.scatter(X4[:,0], X4[:,1], marker='o', c=Y4)
```

```
<matplotlib.collections.PathCollection at 0x24f824426a0>
```



DBScan 聚类：





谱聚类：

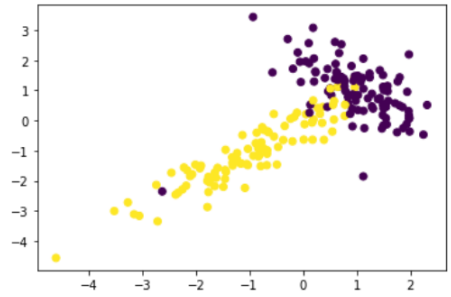
对于该类数据集，密度聚类和谱聚类都难以取得较好的结果，这里还对谱聚类的方法函数做了替换，采用了’ploy’以及’sigmoid’方法，效果基本差不多。首先密度聚类是根据密度来划分，对于该类数据集确实难以划分，而谱聚类效果不好的原因可能还是亲和度函数不对，要改成适应该数据集的方法应该可以。由此可见，两种方法都有一定局

限性。

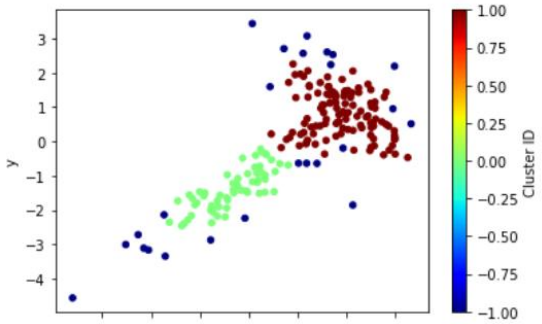
五、make_classification，样本数为2，特征数为2，n_redundant=0,n_informative=2,n_clusters_per_class=1

```
X5, Y5 = datasets.make_classification(n_samples=200, n_features=2, n_redundant=0, n_informative=2, n_clusters_per_class=1)
plt.scatter(X5[:,0], X5[:,1], marker='o', c=Y5)
```

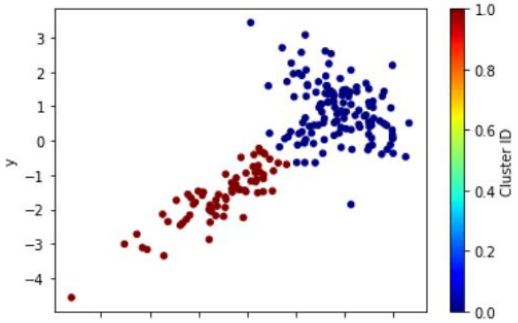
```
<matplotlib.collections.PathCollection at 0x24f83083780>
```



DBScan 聚类：



谱聚类：



对于该数据，可以看到密度聚类即使调参后也难以避免噪声点过多的问题，不过大致可以区分出主体，而谱聚类解决了这个问题，但是在边界处还是聚类的和原始数据有点差距，不过大致正确。

三、意见和建议
（如有，请写出对本次实验的具体意见和建议，包括但不限于教学内容、实验内容、教学 PPT 等）