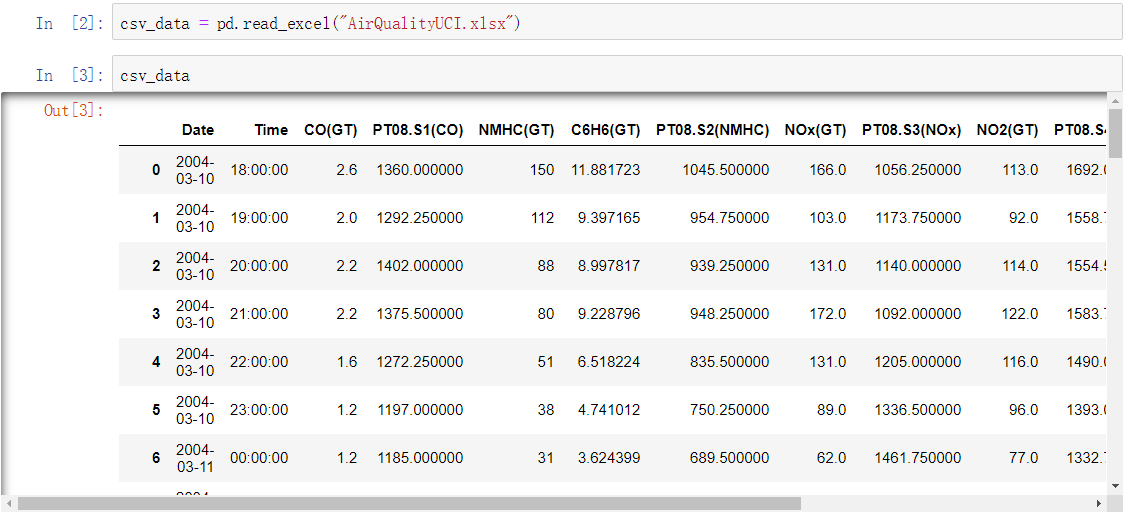
# 机器学习实验报告

10174503110 印张悦

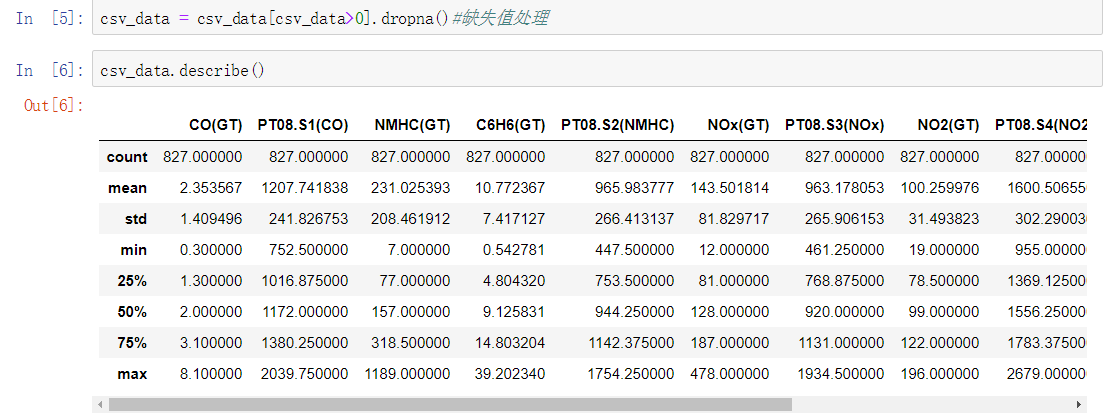
本实验主要采用sklearn进行建模，pandas进行数据分析，TSNE进行数据可视化。

**Air quality dataset回归：线性回归**

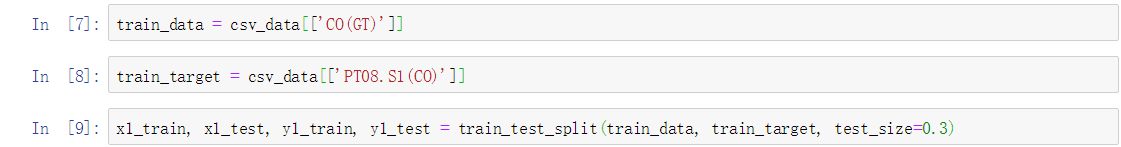
首先载入数据，发现大量缺失值：



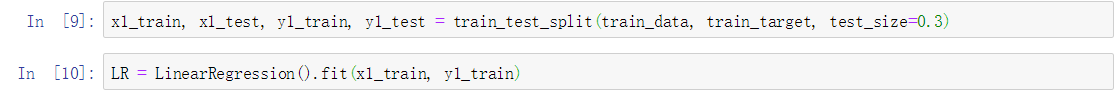
进行缺失值处理：



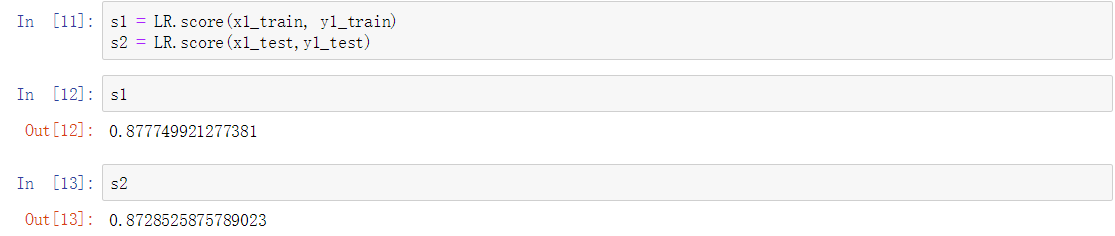
对CO(GT)和PT08.S1(CO)进行线性回归，用CO(GT)的变化预测PT08.S1(CO)的变化：



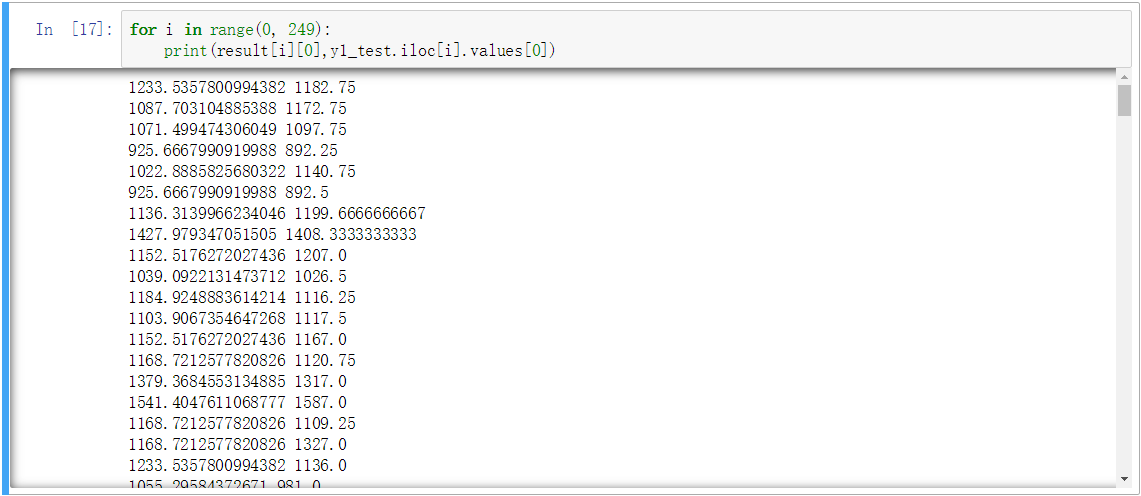
导入模型进行训练：



查看模型准确率：

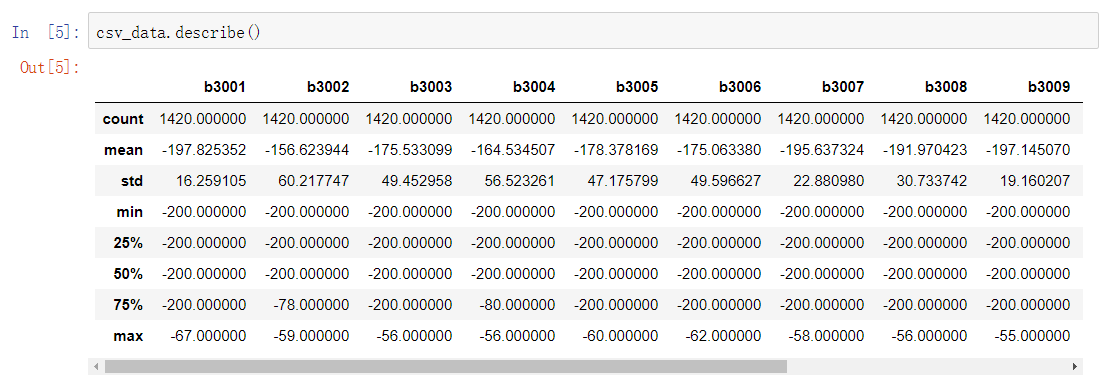
可见，模型对训练集的准确率为87.7%，对测试集的准确率为87.2%，得到了不错的效果。

查看拟合结果（预测值和真实值）：



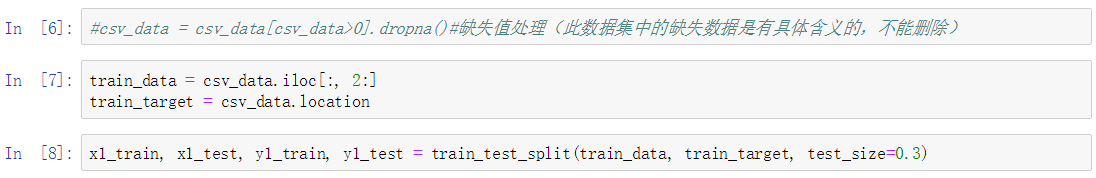
**BLE&RSSI dataset分类：SVM、决策树、随机森林；**

分析数据：

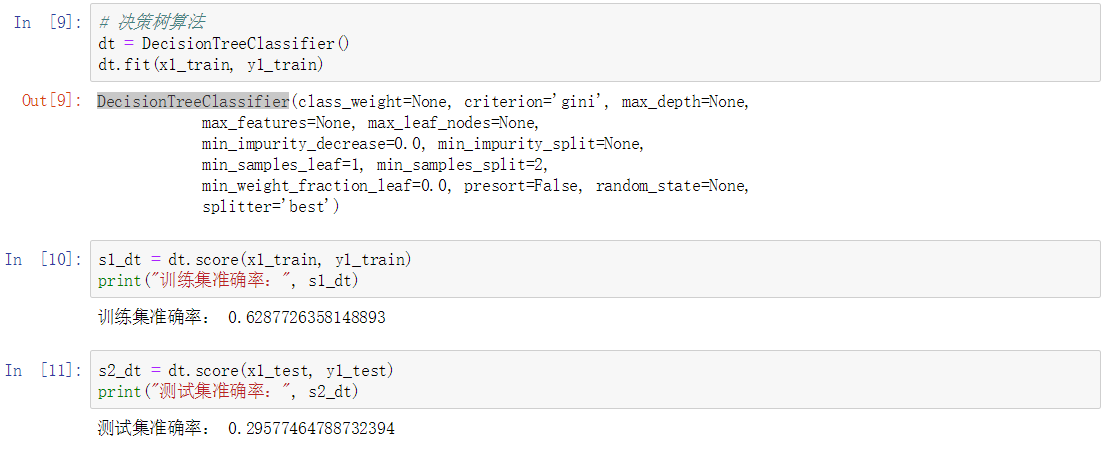


发现其中有大量的缺失值-200，但这里的缺失值都有具体的含义，表示没有收到发射器的信号，因此不能随意删除。

处理数据：

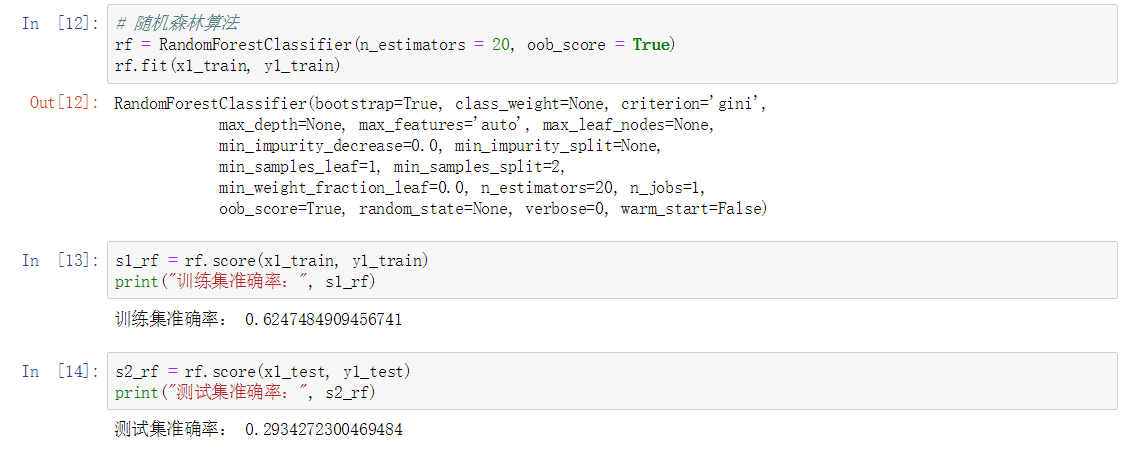


决策树算法：



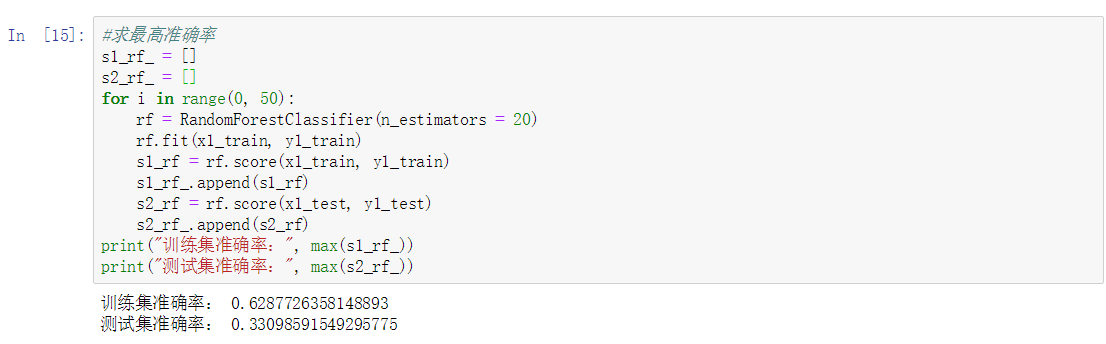
由于我们有100多个标签，但只有1000多个数据，显然数据量太少，而且标签的分布也非常不均匀，因此测试集的准确率30%都不到。

随机森林算法：



同样的原因，随机森林的结果也同样只有30%不到。

我们尝试进行多次随机森林取最优结果：



可以看到最优的测试集准确率达到了33%，已经是一个相对不错的结果了。

SVM算法：



可以看到SVM的结果非常糟糕，只有27%，是所有分类算法里面最低的。

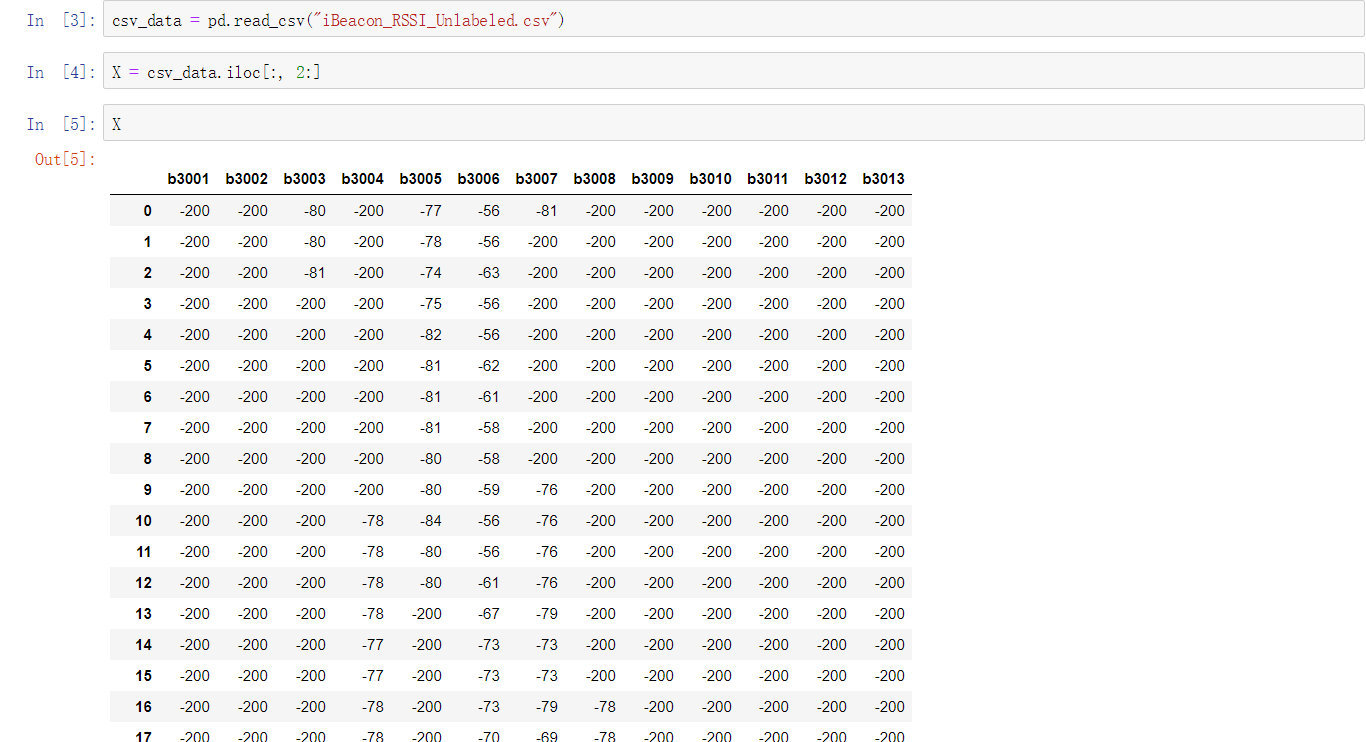
我们尝试通过网格搜索交叉验证的方式寻找最优参数，然后传递给模型进行训练，最后结果如下：



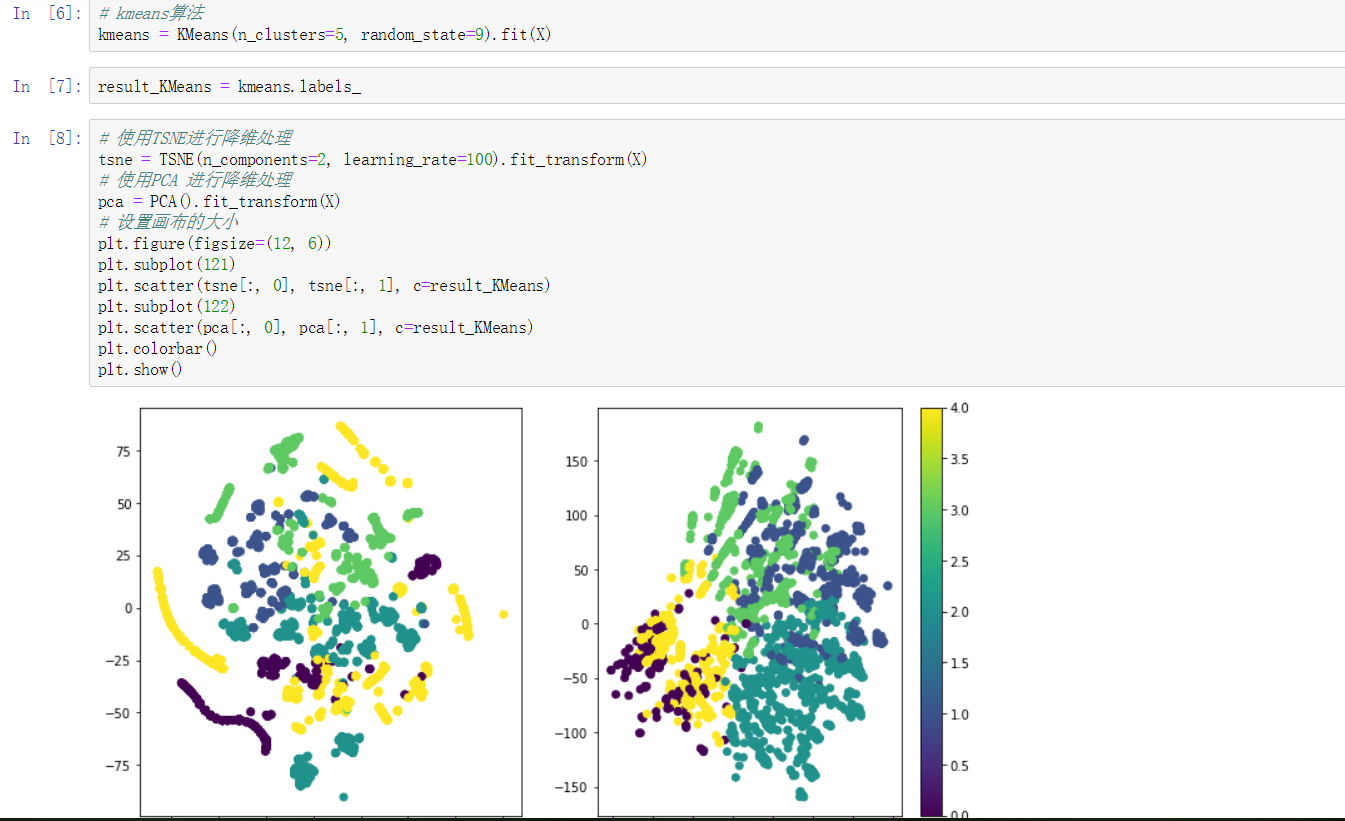
可以看到我们的准确率终于逼近了30%，得到了相对不错的改进。

**BLE&RSSI dataset聚类：DBScan、kmeans、GMM、层次聚类算法**

导入数据集：



采用KMeans算法，分为5类：



左图为采用TSNE进行降维处理的结果，右图为采用PCA进行降维的结果，我们发现采用PCA降维结构边界比较清晰，结果很理想。

采用DBScan算法：



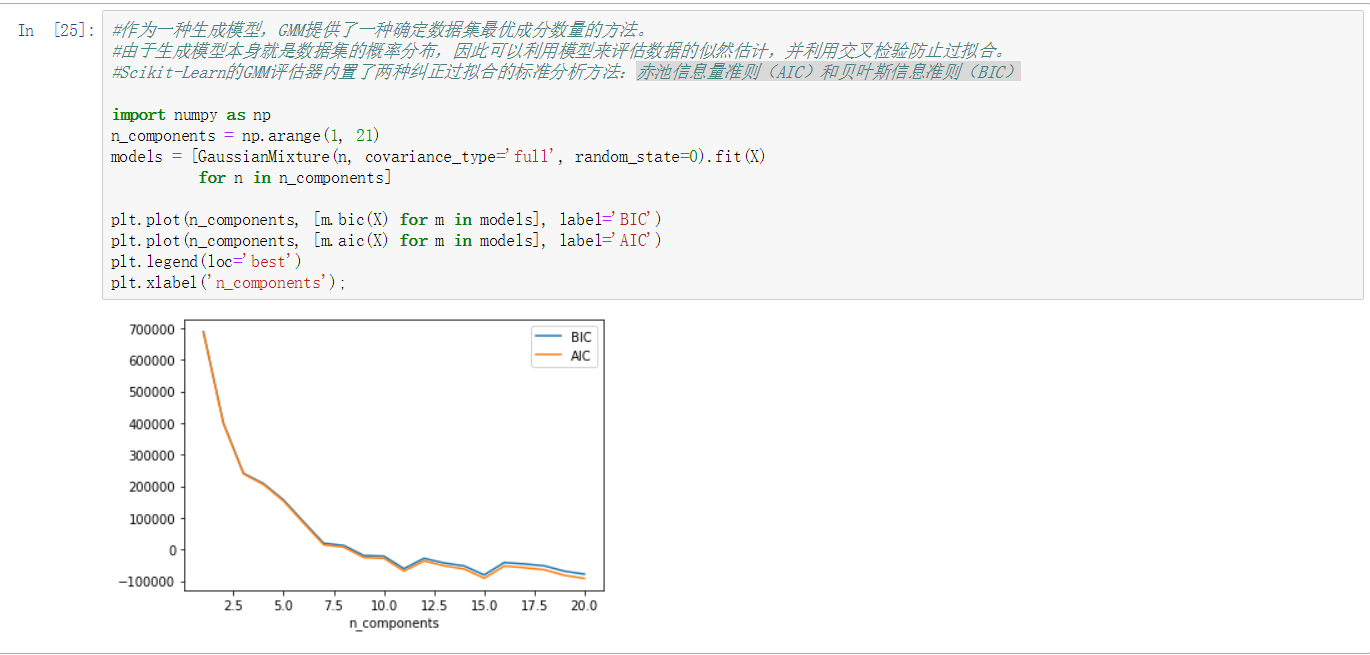
我们发现DBSCAN最终的结果非常不理想，都聚在一起了，采用TSNE降维紫色点都分布在外面一圈，可能是由于该算法没有很好的处理从高纬度降到二维的原因。

采用GMM算法，分为5类：

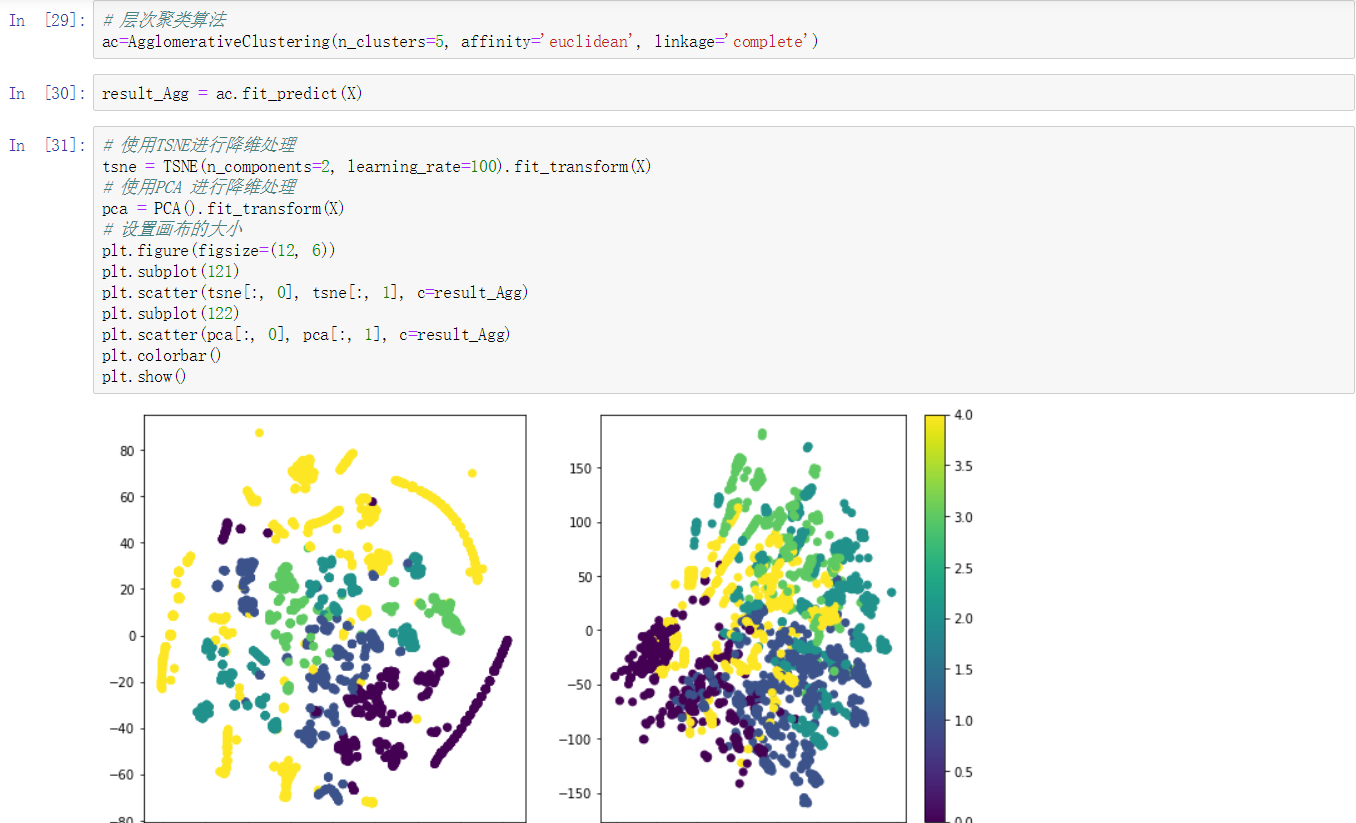


同样，采用GMM算法时也遇到了这个问题，这次外围的点被表示成了绿色，而且边界模糊，结果不是很理想。

纠正过拟合：赤池信息量准则（AIC）和贝叶斯信息准则（BIC）。

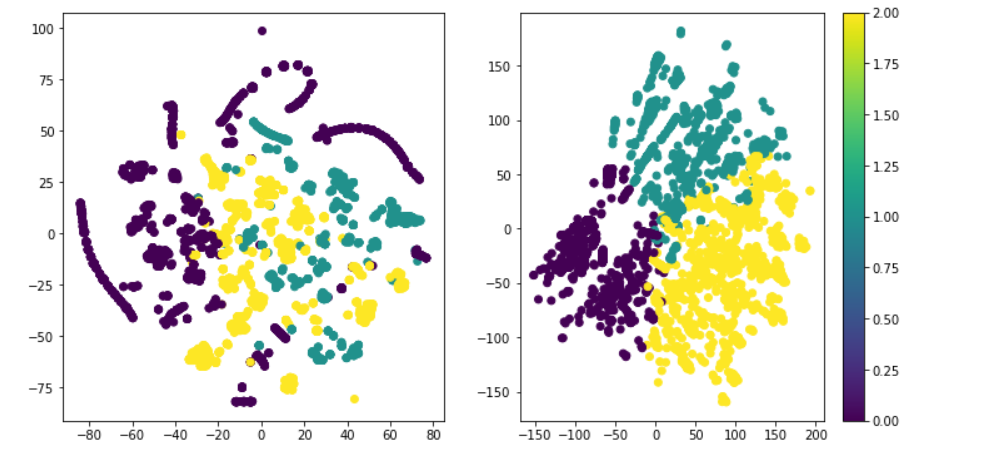


采用层次聚类算法，分为5类：



采用PCA降维得到的结果相对不错，采用TSNE就遇到了之前的问题，但相比之前的DBSCAN和GMM还是不错的，没有重叠到一起，边界还是可以辨别的。

我们采用情况最优的KMeans，改成3类：



可以看到边界均非常清晰，说明较小的分类数量可能对本数据集更加合适，而且TSNE可能由于降维算法的原因聚类情况的展现远远没有PCA来得清晰直观。

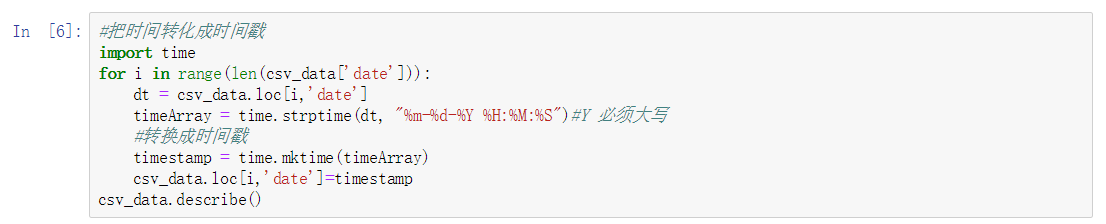
**总结：**

本次实验通过使用sklearn，熟悉了机器学习中的回归、分类和聚类算法，以及如何在python中使用它们，对于实际的数据集如何应用它们，通过对不同算法的比较，对于算法的应用范围也有了一个大致的了解。在实验过程中遇到了很多问题，在此十分感谢同学和助教的帮助。

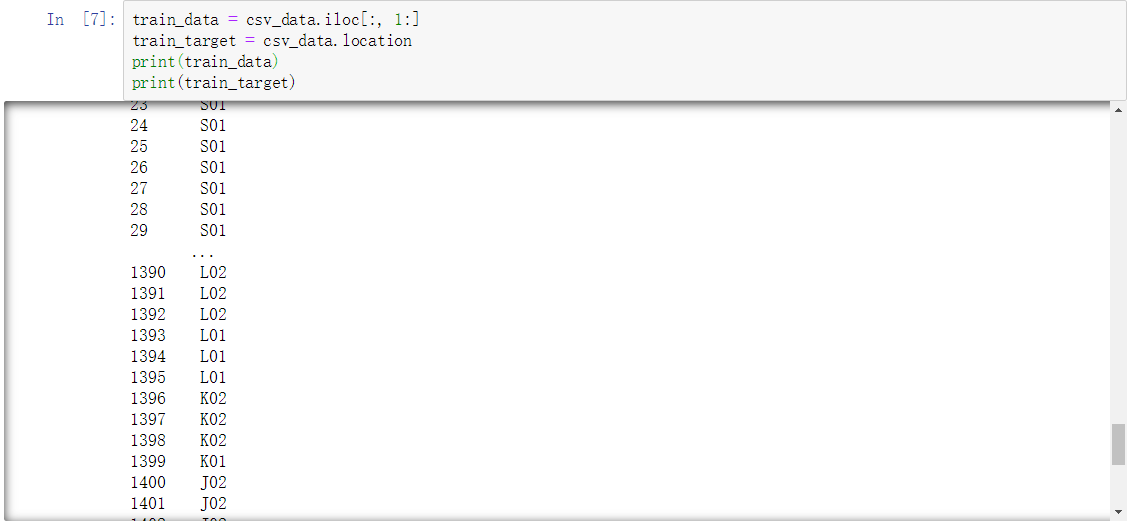
**5.13更新：**

之前做分类时并没有注意信号源与时间的关系，而只是简单的把时间删除了。同学建议可以把时间也考虑在内，可以提高准确率。由于信号源可能是周期性发出信号的，某一时间可能最近的信号源处于休眠状态，稍微远一点的信号源则处于活跃状态，因此信号源与时间也有一定的关联性。

先把时间转换成时间戳：



把时间也包括在内：

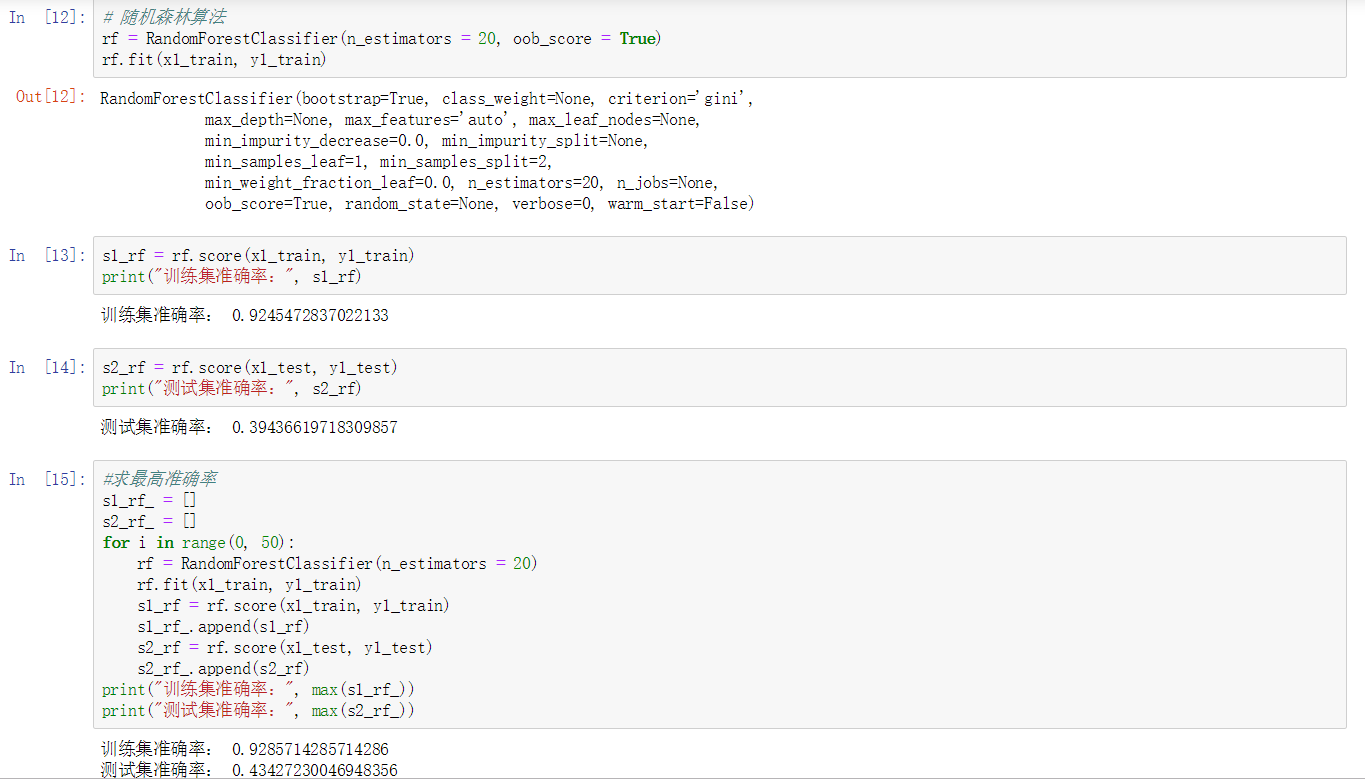


决策树算法：

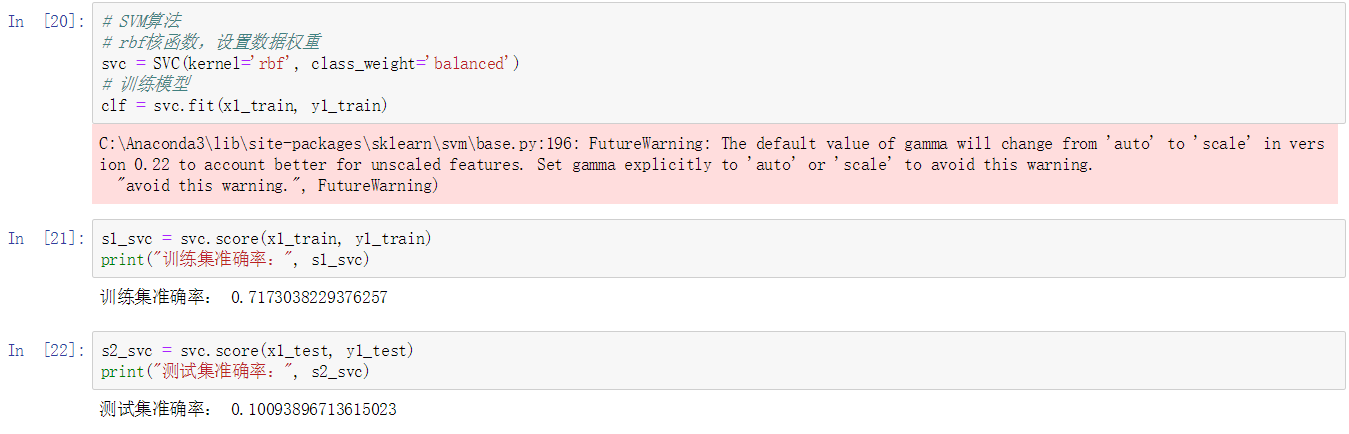


可以看到训练集的准确率从60%提升到了92%，测试集也从30%提升到了40%，提升非常明显。

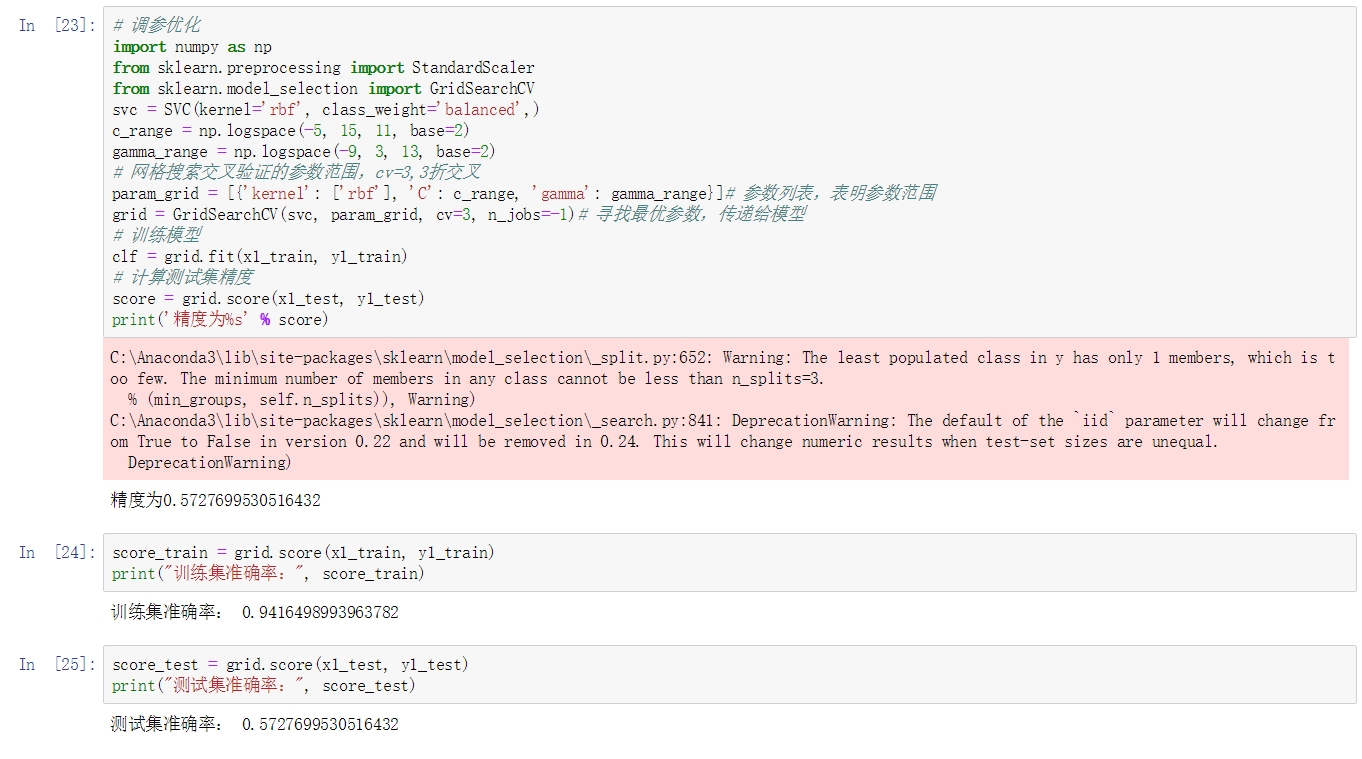
随机森林算法：

同样，随机森林也得到了非常显著的提升，训练集从60%提升到92%，测试集从30%提升到43%。

SVM算法：



啊偶，改进后的SVM算法的训练集虽然有所提升，但测试集准确率居然只有10%，非常糟糕的结果，我们尝试调参改进，同样使用之前的交叉验证法：



结果非常理想，我们得到了最高的准确率，训练集高达94%，测试集也同样高达57%，调参对模型的改进非常明显，这里也显示出调参的重要性。