**第二章实验报告**

**刘森栋 2019213420 2019215119**

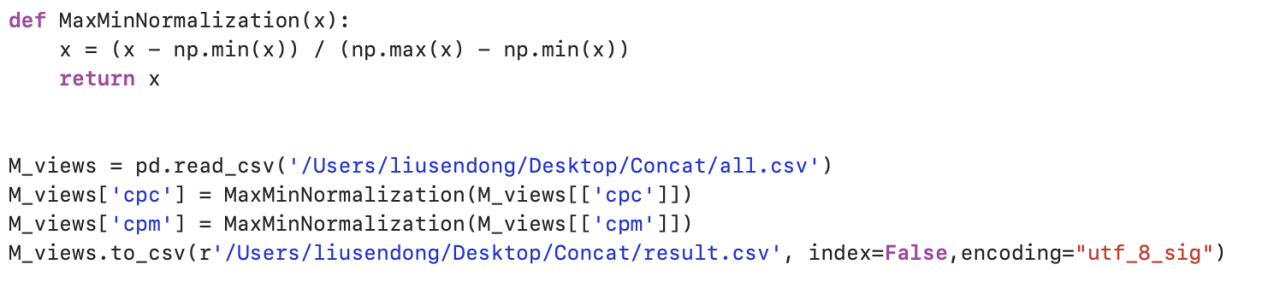
实验环境：语言：python，编译环境：python3.8，电脑系统：MacOS

**第一步，数据预处理**

1. 填补缺值，通过公式计算数据集cpc和cpm的特征匀值：分别为：cpc： 0.1417，cpm: 0.8345，然后通过识别两个数据集的空值部分，用各自的特征匀值替换
2. 合并数据（按时间），如下图代码所示，得到了一个初步的完整的文件，二者具有公有属性timestamp，直接合并即可



1. 数据规格化（Max-Min标准化），如下图代码所示，首先利用Max-Min标准化公式，写出函数，然后读取文件，用标准化后的数值替代原数值，将文件保存为result.csv

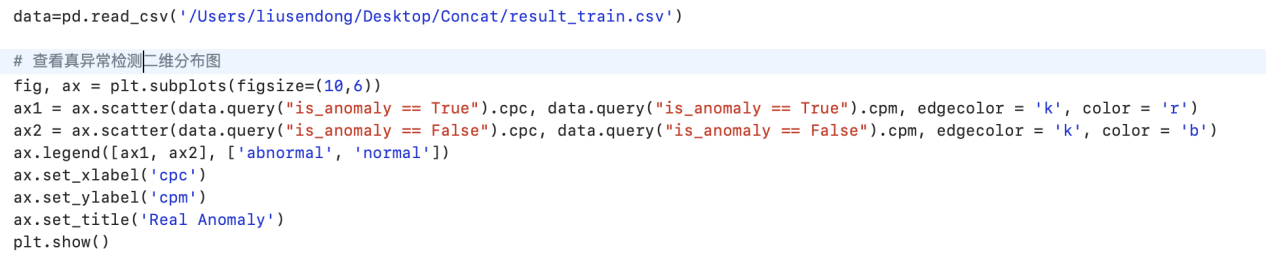


1. 划分训练集和测试集，按大致7比3比例划分了训练集和测试集

**第二步，算法实现**

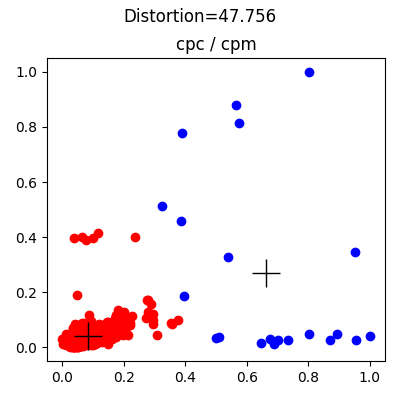
**首先编写如下代码得到真异常检测图：**

**A.**

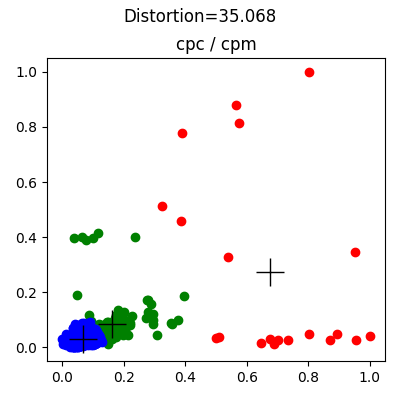
****

1 . K-means 算法实现，首先编写好将数据集聚类的算法，见LAB2\_Kmeans部分的代码，然后随机取K=2，3，4，观察聚类效果。

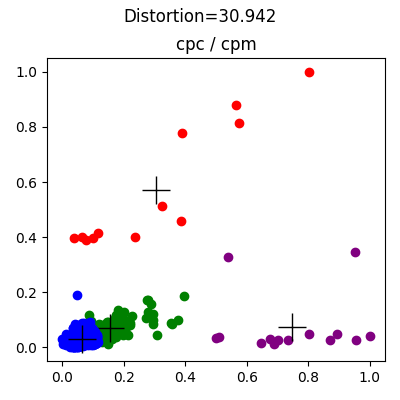
1. K=2



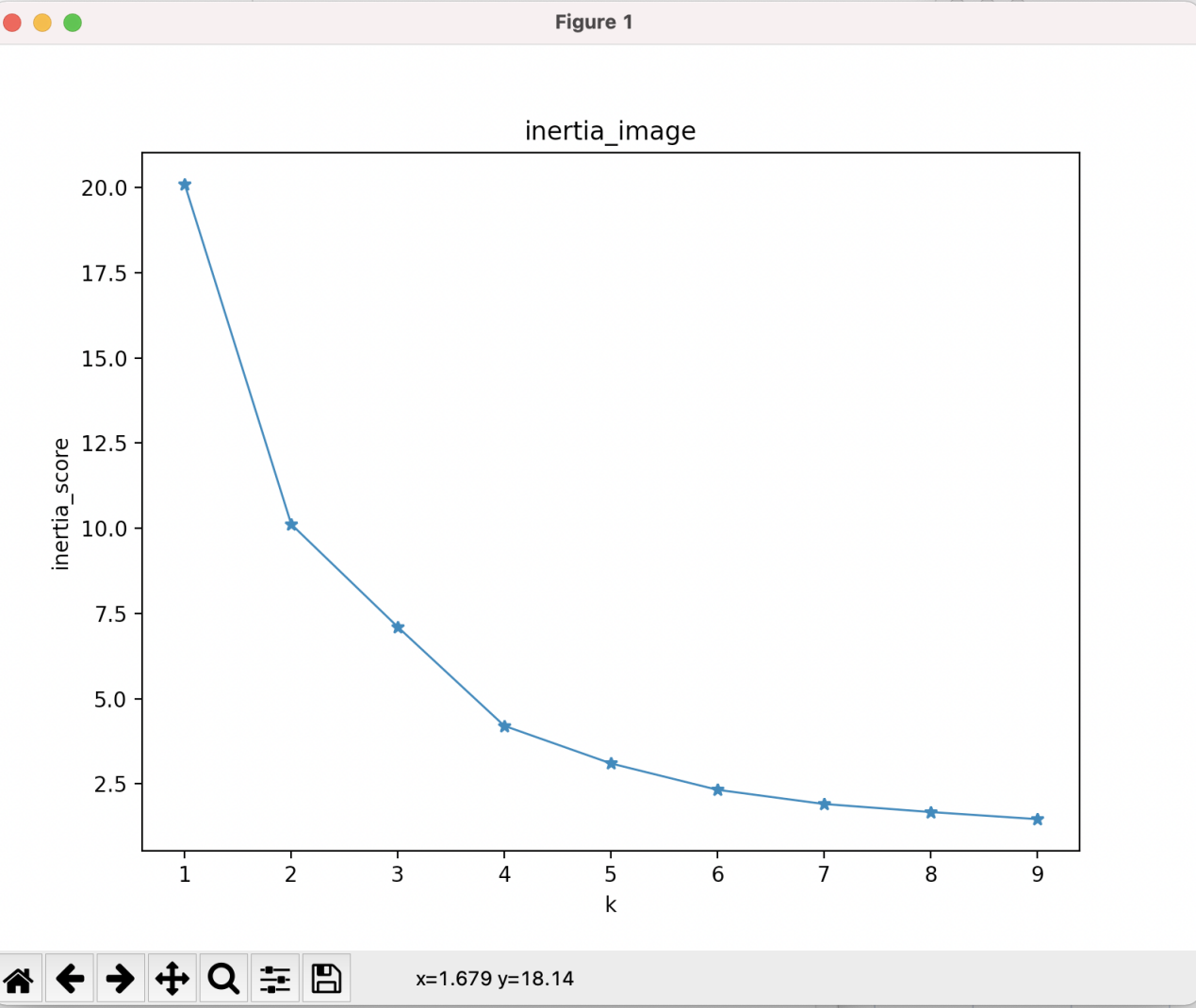
1. K=3



1. K=4



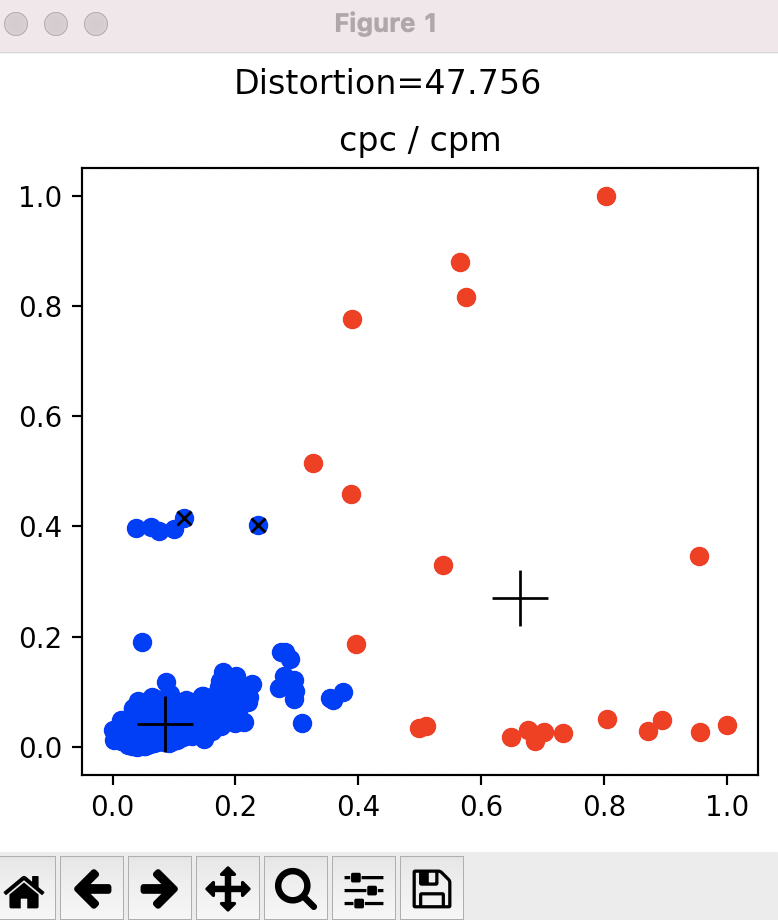
### 同时，可以利用肘部法则来粗略的确定K的取值，如下图



最终综合观察各聚类效果图和分析得到，**K=2**时，聚类效果最好。

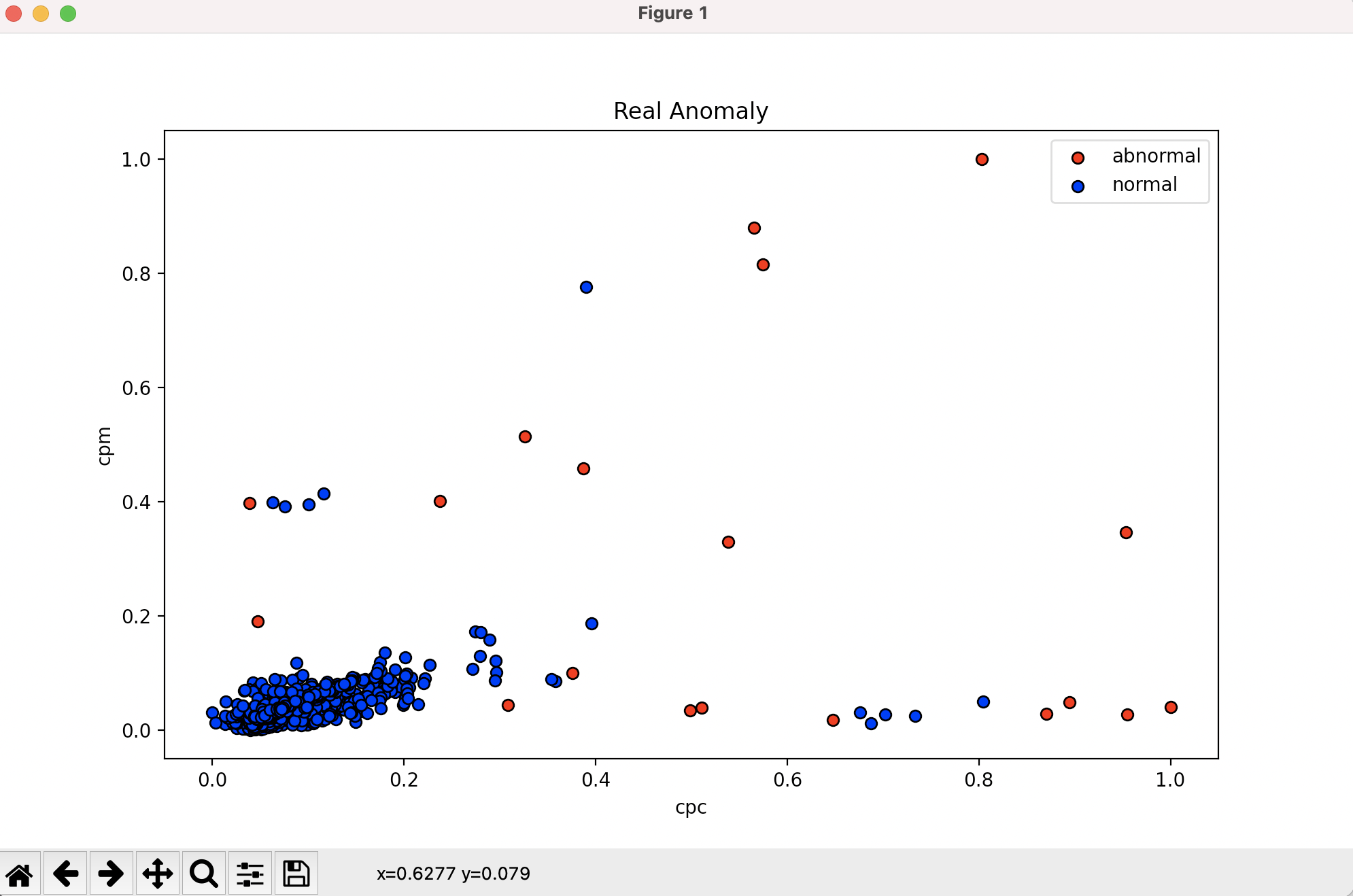
接下来进行基于Kmeans算法的异常检测代码的编写，核心思路是确定各聚类质心的坐标centerxy，求出各聚类的所有点到相应质心的平均距离avgDis，然后利用公式求出所有点到质心相对距离d = d（Data，centerxy）/avgDis，通过观察不同阀值时，离群点的位置和所处范围，得到最终确定的离群阀值 threshold = 8.

最后得到的结果图为



其中，蓝色部分为聚簇1，红色部分为聚簇2，黑色的加号为各自的质心，带黑色叉号的蓝色聚簇点以及红色聚簇点为异常点

Distortion 为畸变函数的值

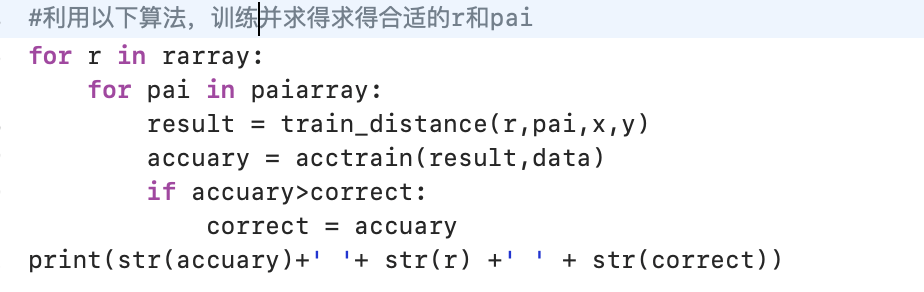


与真实异常检测图进行比对，计算得到 正确率为 **98.9%**

**结果分析：Kmeans异常检测算法对于参数的选择比较敏感，当K取3，4时，正确率，效果很差，同时当离群阀值太小时，如小于4，正确率也会降低，需要选择较大的离群阀值（如8）和K = 2，才能使得算法的准确率较高。**

1. **Distance-based 算法实现**。

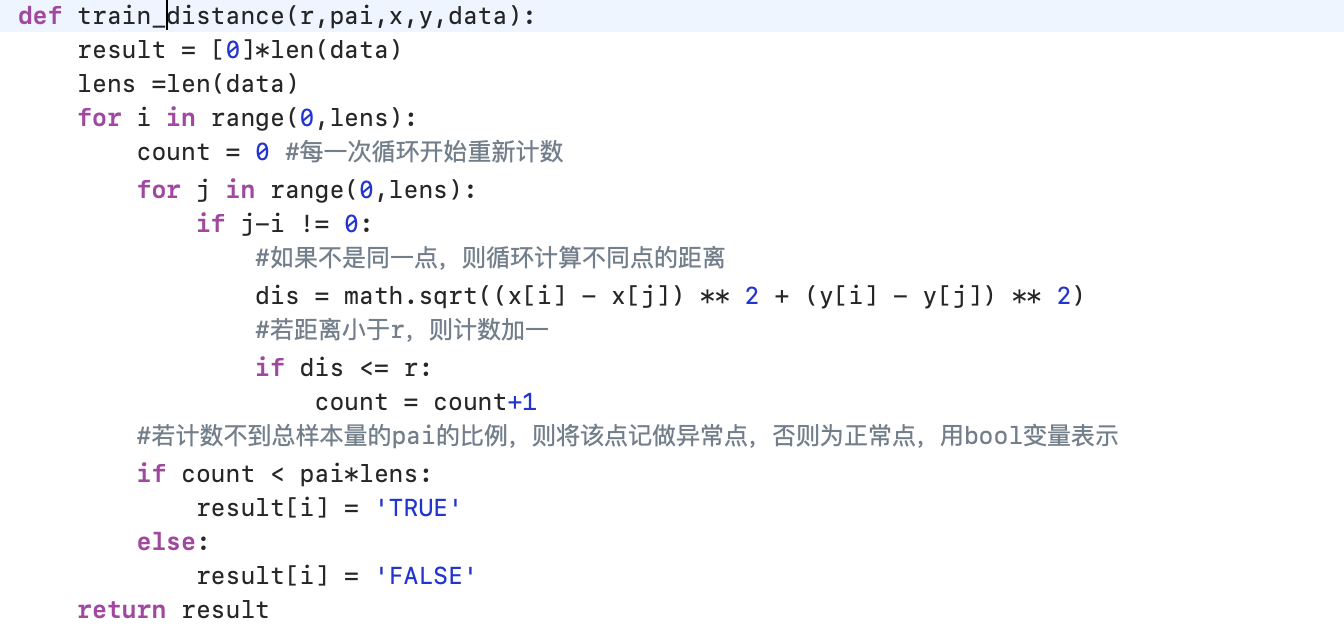
**首先根据以下算法不断训练，以找到合适的r和pai**



通过不断训练来得到合适的pai和r值

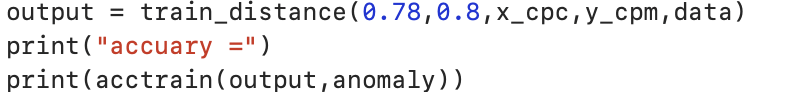
记录下此时的值 r：0.78，pai：0.8

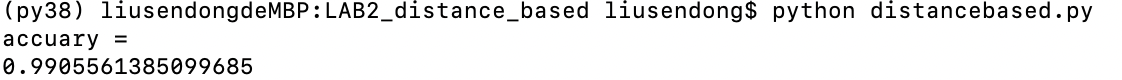
然后进行基于Distance—based算法的异常检测代码的编写，代码如下所示



遍历整个data数组，如果不是同一点，则循环计算不同点的距离，若距离小于r，计数加一，若计数小于π\*data长度，则说明有较少的值在范围内，则说明该点异常，反之则正常，将结果标记在result数组中

然后执行以下代码，计算出当前r和pai值情况下的正确率：99.05%

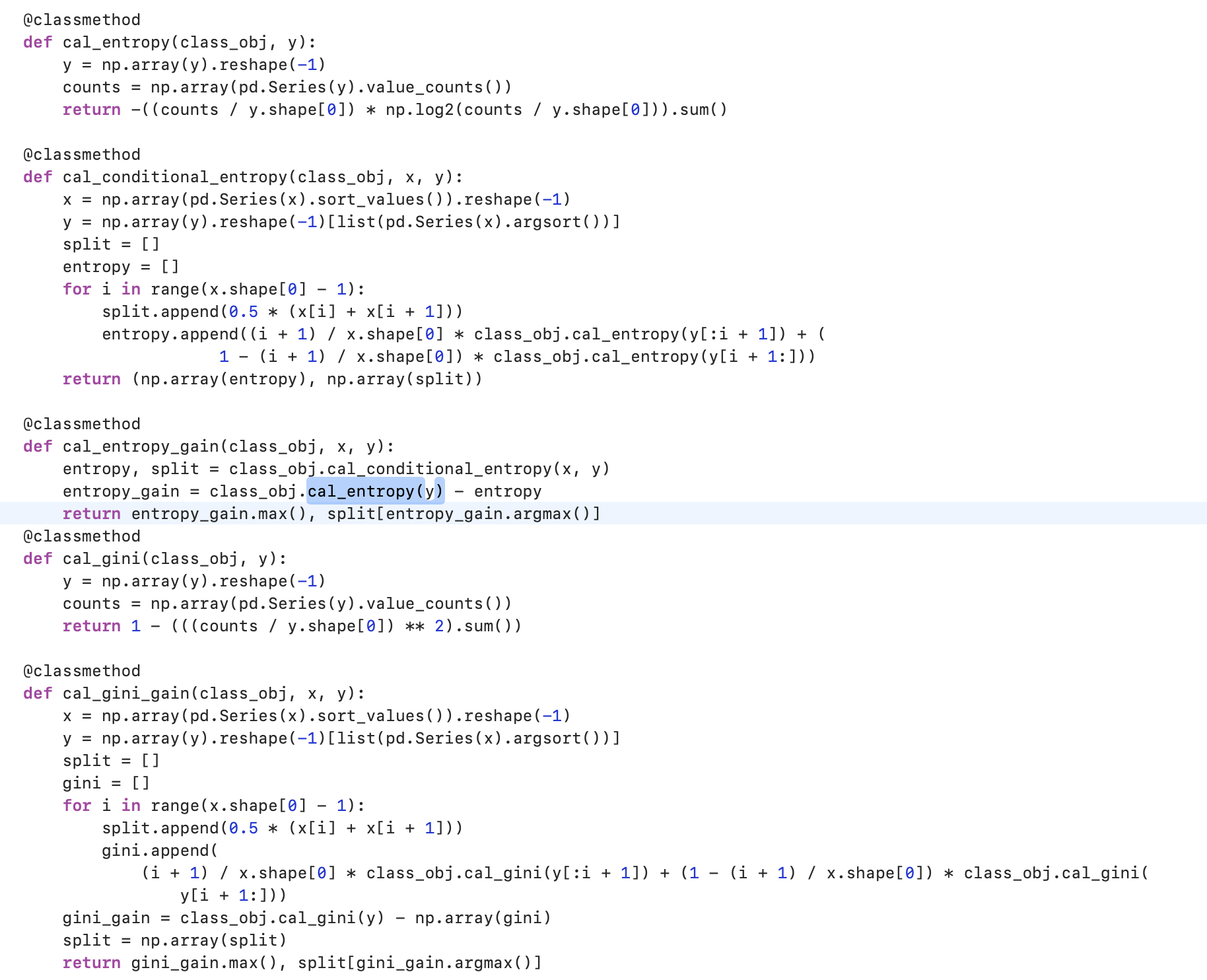


****

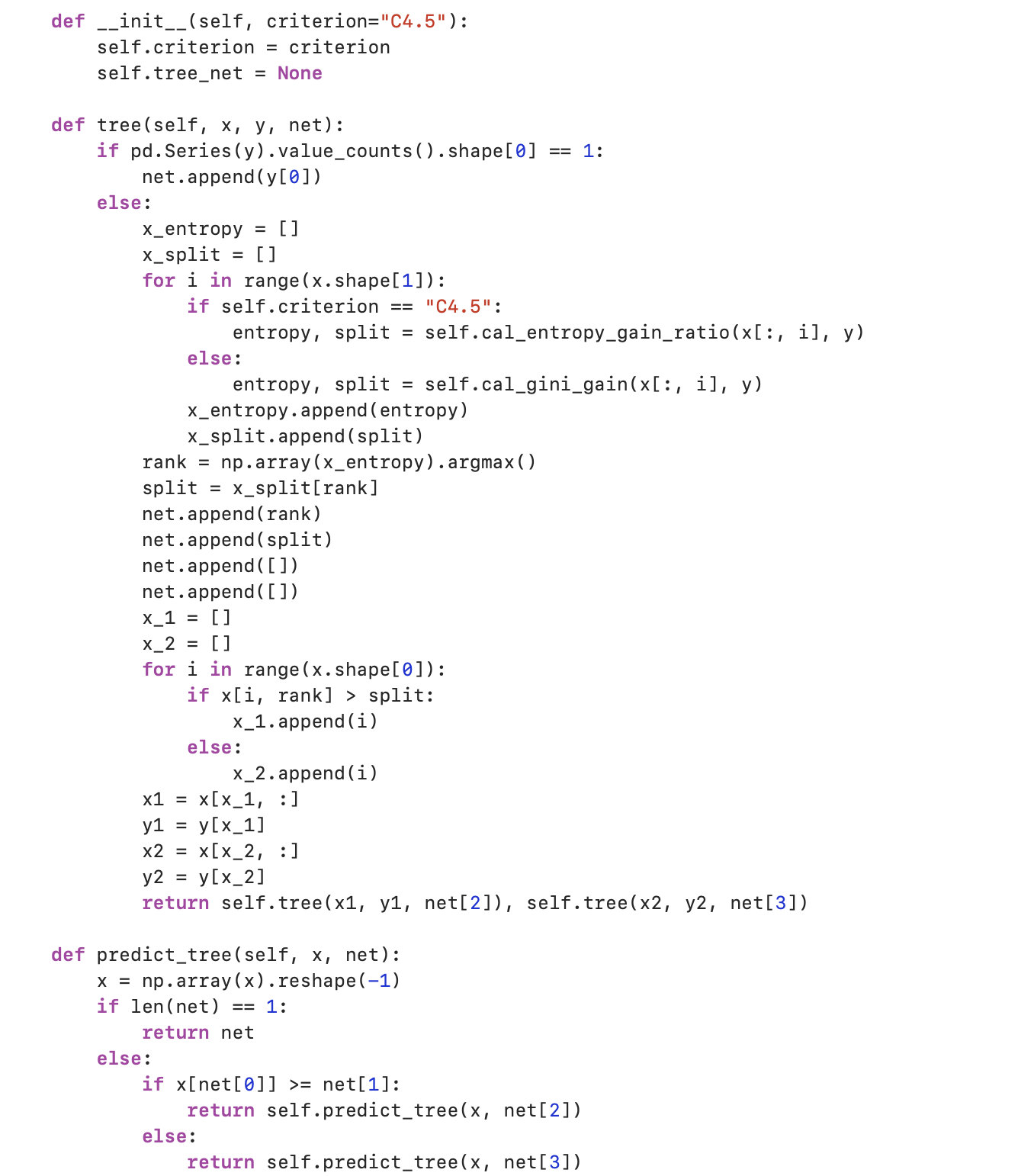
**结果分析：Distance-Based异常检测算法相对来说比较简单，通过不断训练得到效果更好的r和pai值，改进准确率，其算法原理也比较简单易懂，但执行效率较慢，时间复杂度高（O（N2）），适合数据量小的数据集训练和测试。**

1. DTI算法实现

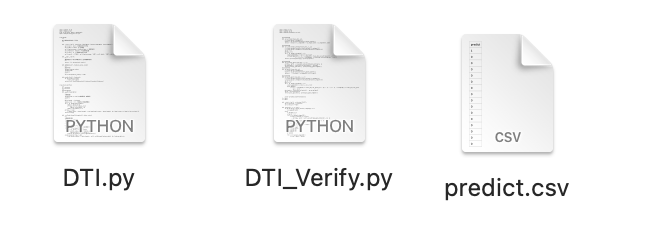
首先编写计算经验熵和条件经验熵的函数及其相关函数



然后编写决策树结构的代码：包括初始化树，定义树的分支和节点。

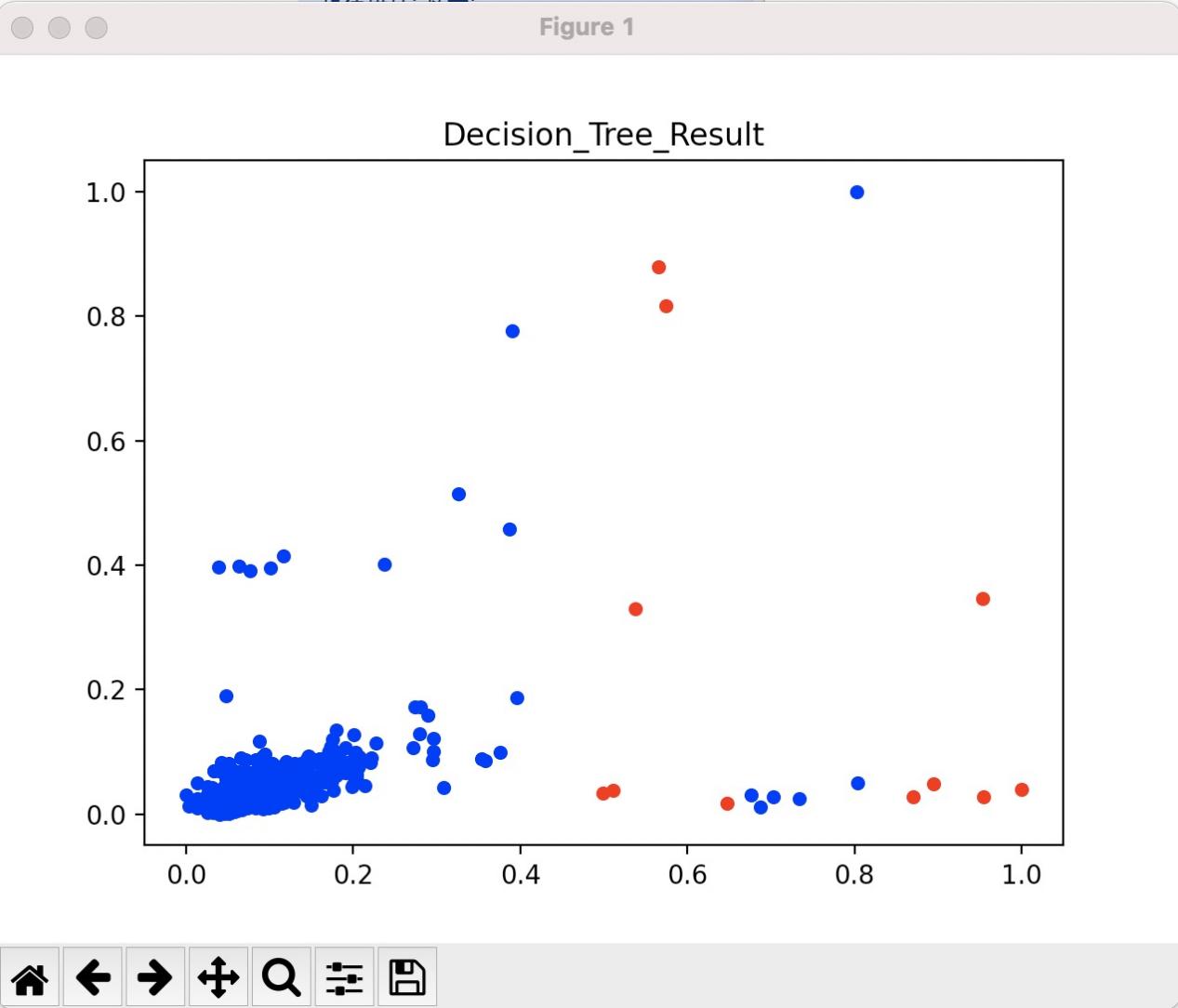


最后编写预测函数和main函数实现基于DTI的异常检测代码：DTI.py与真实验证结果DTI\_Verify.py进行比对，并得到预测集predict.csv

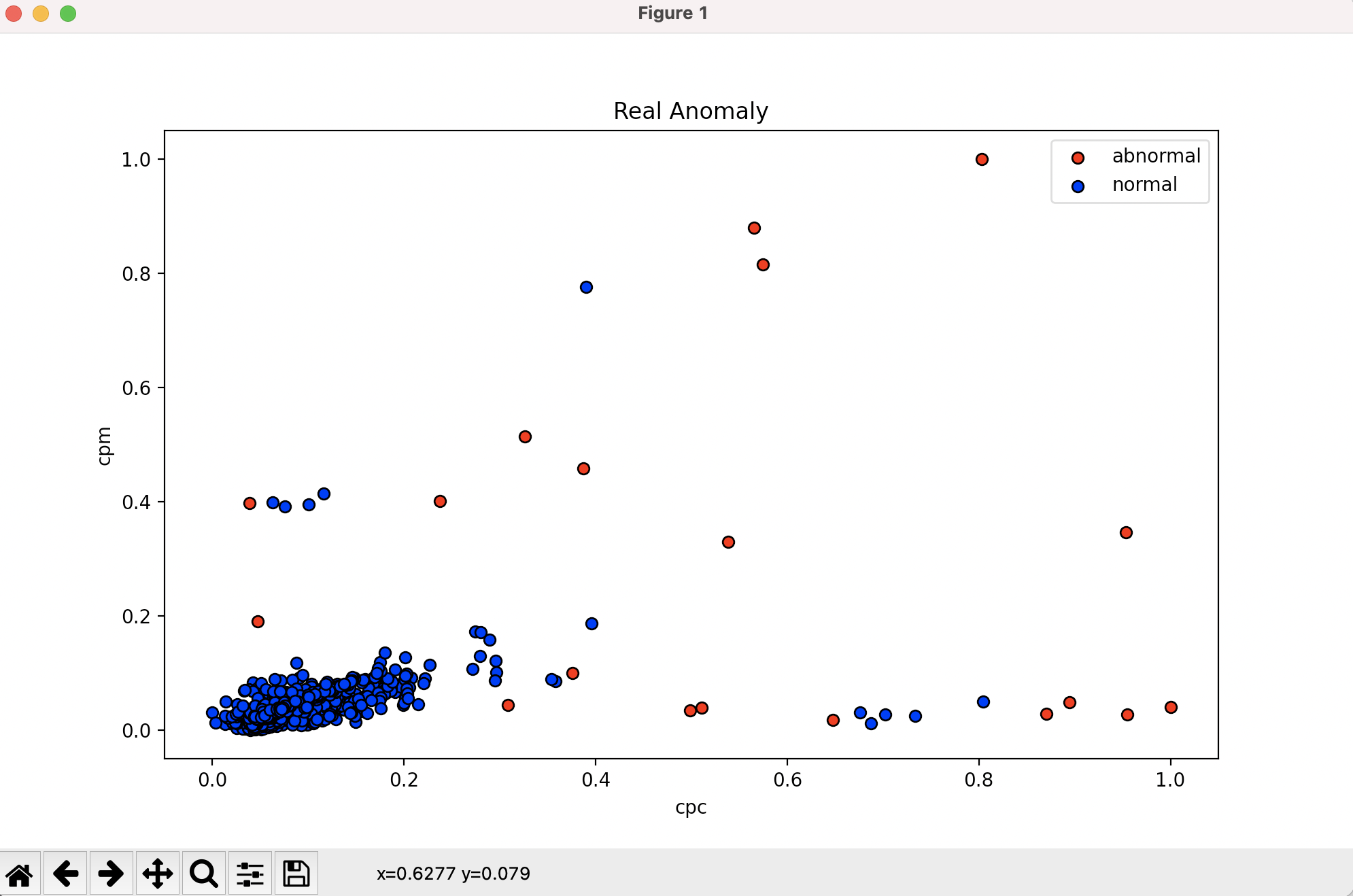


最终得到如下结果：

A.决策树异常检测结果：



B.真异常检测图（用于验证）



通过对两图的数据同时进行统计，最终得到基于决策树的异常检测算法正确率 为 **99.06%**

**比对结果如下：**



**结果分析：DTI异常检测算法的训练过程比较重要，训练好分类的模型，会使得测试过程进行较快，整个算法较为复杂，但总体健壮性好，更为稳定，对中间值的缺失不敏感。**

**基于本次实验的数据集，综合来看Distance-Based算法更好，因为本次实验数据集总量不多（1000条左右），Distance-Based异常检测算法更简单易懂，方便使用，正确率也很高，很适合小规模数据集。**

**第三步：分析三类方法在异常检测场景中的优劣：**

1.基于聚类的异常检测方法分析：

聚类和异常检测目标都是估计分布的参数，以最大化数据的总几率。

优点：

（1）无监督

（2）有坚实的统计学理论基础，当存在充分的数据和所用的检验类型的知识时，这些检验可能很是有效；

（3）测试阶段速度较快

（4）聚类密集时，类与类之间的区别明显，检测效果好

缺点：

（1）对于多纬数据，可用的选择少一些，**而且对于高维数据，这些检测可能性不好**

（2）如果异常数据自己成簇，将难以发现异常

（3） 对孤立点敏感

1. 基于临近度的异常检测方法分析：

这种方法比统计学方法更通常、更容易使用，由于肯定数据集的有意义的邻近性度量比肯定它的统计分布更容易

优点：

1. 简单可用；

（2）适应不同的类型数据类型方便，只需定义合适的便是数据间距离的方法即可

****缺点：****

1. ****基于邻近度的方法须要O(m2)时间，大数据集不适用；（2）该方法对参数的选择也是敏感的；****
2. ****不能处理具备不一样密度区域的数据集，由于它使用全局阈值，不能考虑这种密度的变化。****
3. 基于分类的异常检测方法分析：

****复杂度：****

****训练阶段： 决策树会较快，基于二次最优化的算法****

****测试阶段： 分类技术在测试阶段会较快—分类的模型已经训练好****

****优点：****

****（1）可以处理多分类问题****

****缺点：****

1. ****多分类问题需要精确的多分类标签****
2. ****直接将一个确定的标签赋给测试数据有时候会不恰当****