数据挖掘理论与算法
课程实验报告

肖一凡
计算机系
2014210871

本次实验使用 Python 3.4 语言独立编程，除了实现算法之外，还进行了两次数据实验：根据网络数据 Wine.txt 对 178 种酒进行分类，根据某密度函数生成数据集进行数据模拟实验。过程中，对比了ρ_i采用“阀值核”以及“高斯核”的优劣，利用 MDS 算法对数据进行了可视化呈现等等，证明了该算法的有效性，但其标记“噪声点”的方法仍有改进空间。

一、算法的实现细节

1．1　ρ 和 δ的确定

首先需要确定一个重要参数d_c，根据原文“As a rule of thumb, one can choose dc so that the average number of neighbors is around 1 to 2% of the total number of points in the data set”，程序中将所有两点之间的距离升序排序，取前面第 2%位置的距离作为d_c。文中也说过“For large data sets, the results of the analysis are robust with respect to the choice of dc.”，所以具体选择 1%还是 2%或者其他的值，甚至用其他的方式选择d_c，对最后的结果影响不大。

关于ρ_i：原文中最先介绍的是$\rho_i = \sum_j \chi(d_{ij} - d_c)$，$\chi(x) = \begin{cases} 1, & x < 0 \\ 0, & x \geq 0 \end{cases}$。这个就是“阀值核”，文中后面又提到了“指数核”以及“高斯核”等，这其中高斯核使用得比较广泛，在实验中还专门进行了对比。

高斯核：$\rho_i = \sum_{j \neq i} e^{-\left(\frac{d_{ij}}{d_c}\right)^2}$。后面有专门的章节介绍这两个核的区别，它们各有优劣，适用于不同的数据情况。

关于δ_i：这个是最不需要自由发挥的参数，文中定义的也很明确。

$$\delta_i = \min_{j:\rho_j > \rho_i} d_{ij} \text{ 或 } \delta_i = \max_j d_{ij}$$

1．2　聚类个数的确定

文中确定聚类个数的原则是根据$\rho_i - \delta_i$散点图，判断右上角的“离群点”个数。程序中，将绘制后的$\rho_i - \delta_i$散点图沿 x 轴、y 轴平均分成 20 份，形成 20×20 的网格，然后从右上角开始依次查找右上角正方形内的离群点个数，根据查找结果确定断层发生的位置，进而确定聚类个数以及ρ_i和δ_i的临界值。

但是有些数据可能不适用本聚类方法，所以生成的$\rho_i - \delta_i$散点图不存在右上角明显断层的离群点。此时，可以更改网格密度，比如改为 50×50 的密度再做尝试（需要手动修改代码中的 N_grid 变量的值）。

聚类个数的确定是自动的过程，会得到类似下面的结果：（虚线就是ρ_i和δ_i的临界值）

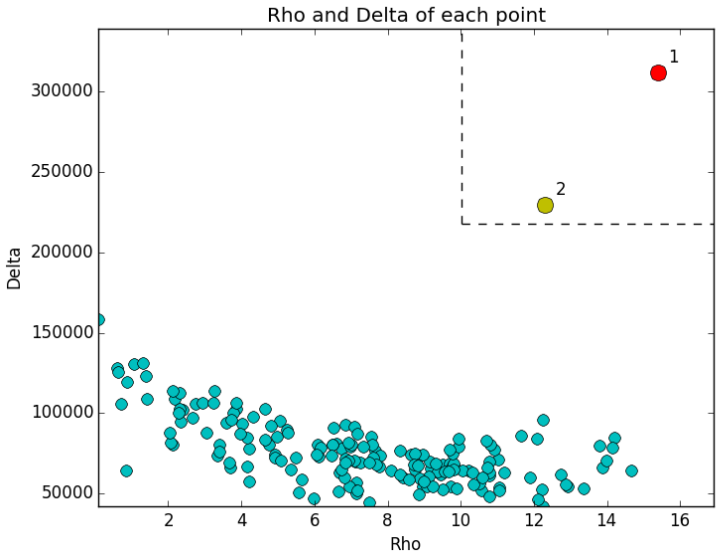


图 1.1　$\rho_i - \delta_i$散点图示例

1．3　聚类结果的确定

将所有点分为三类：类核心点、类边缘点、离散点。

聚类：首先根据已经选取的ρ_i和δ_i的临界值确定各类的中心点，然后根据“After the cluster centers have been found, each remaining point is assigned to the same cluster as its nearest neighbor of higher density.”确定每个点的类别。

类核心点与类边缘点：原文如下“We first find for each cluster a border region, defined as the set of points assigned to that cluster but being within a distance dc from data points belonging to other clusters. We then find, for each cluster, the point of highest density within its border region. We denote its density by rb. The points of the cluster whose density is higher than rb are considered part of the cluster core (robust assignation).”。根据上一步得到的聚类结果，搜索所有和其他类距离小于d_c的点，取其中最大的ρ_i作为本类别的阀值，高于此ρ_i的本类别的点被认为是“核心点”，其余为“边缘点”。

1．4　可视化与 MDS

为了可视化数据聚类的结果，得到和论文中相似的结果，需要使用 MDS 算法。MDS 全称是 Multi-Dimensional Scaling，多维标度分析，可以根据点集中任意两点间距离反向推算出点的坐标。该过程可以得到点的二维坐标、三维坐标或更高维坐标，但是维度较低时会有信息损失。本程序中引用网上成熟的 MDS 代码，得到点的二维坐标并作图。

实现这部分的代码在 Main.py 文件中，执行之前需要新建 Output 文件夹。原始数据的格式仅支持一种：每行 3 个数“点 A 标号　点 B 标号　点 AB 间的距离”。而且所有点的标号必须从 1 开始连续整数标号。

执行过程会在 Output 中创建 Readme.txt、Decision Rho and Delta.txt、Cluster Result.txt 三个文本存储数据处理结果，创建 Rho and Delta.png、Cluster Result.png 两个图片文件。

二、网络数据实验：178 种酒的分类

2．1　数据集介绍

Wine.txt 来自 http://cs.joensuu.fi/sipu/datasets/，包含 178 种酒的特征数据。这些酒分别来自 3 个品种，每个酒都拥有 13 个特征。这些特征的具体含义网站未给出介绍，但是估计是某些特殊物质的含量。特征取值都是 0 到 65535，所以不需要剔除量纲影响。Wine.txt 的内容格式如表 2.1，散点图见图 2.1。

表 2.1　Wine.txt 内容格式

55187	12563	37499	16890	40603	41129	37607	18548	38866	24380	29837	63615	36787	36787
37424	13470	27335	2027	21370	37739	33459	16075	17986	17334	30370	51132	36086	36086
36734	20982	45910	27025	22082	41129	40095	21021	49616	24604	29304	45610	42397	42397
57602	15671	39952	20944	30630	64857	43552	13602	36592	36458	20247	52332	56186	56186
38114	23960	52919	35132	34192	41129	32491	32149	29150	16999	29837	39849	21362	21362

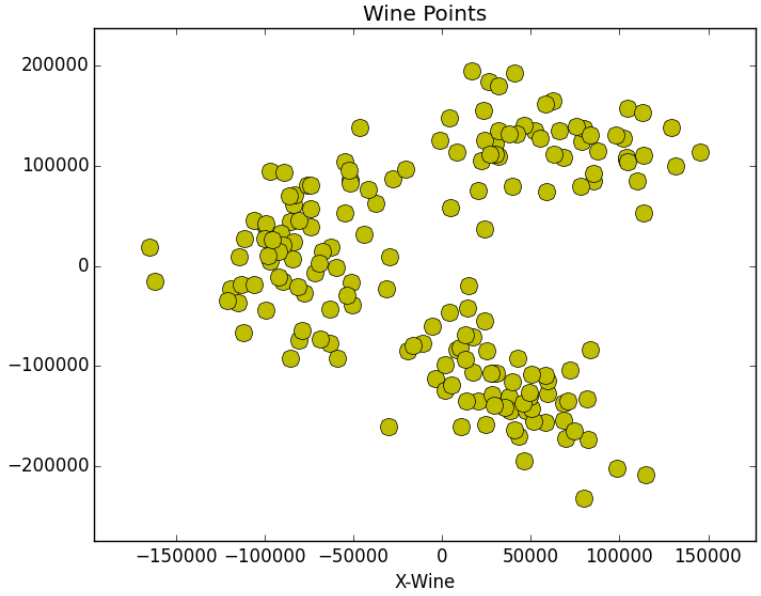


图 2.1　Wine.txt 数据散点图

处理 Wine.txt 的代码见 Data-Wine.py，需要提前新建 Output 文件。过程中会产生 Distance-Wine.txt 包含任意两点间的距离，以及 Wine Points.png 表示酒的散点图。

2．2　阀值核与高斯核

在酒的聚类实验中，测试了两种ρ_i的定义——阀值核与高斯核——的区别。图 2.2 中展示了两种核的区别，左上角是高斯核的$\rho_i - \delta_i$散点图，左下角是阀值核的$\rho_i - \delta_i$散点图。可以看出阀值核的ρ_i取值太离散，有很多点的取值是相同的，而且右上角的离散点也不是很明显，这不利于确定聚类的个数。从右上角的高斯核聚类结果与右下角的阀值核聚类结果对比来看，高斯核的效果也更好。

本程序中默认使用高斯核，如果聚类结果不理想，可以尝试使用阀值核再次尝试。

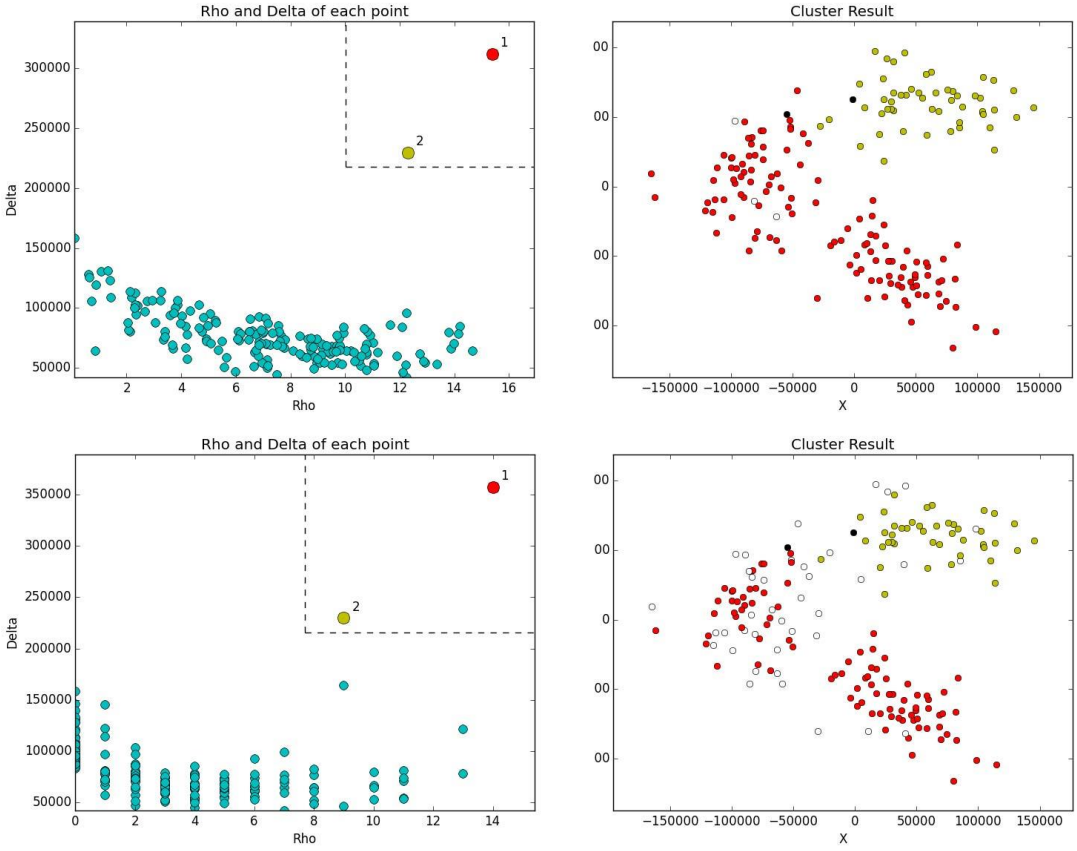


图 2.2　两种核的效果对比

（左上高斯核 右上高斯核聚类结果 左下阀值核 右下阀值核聚类结果）

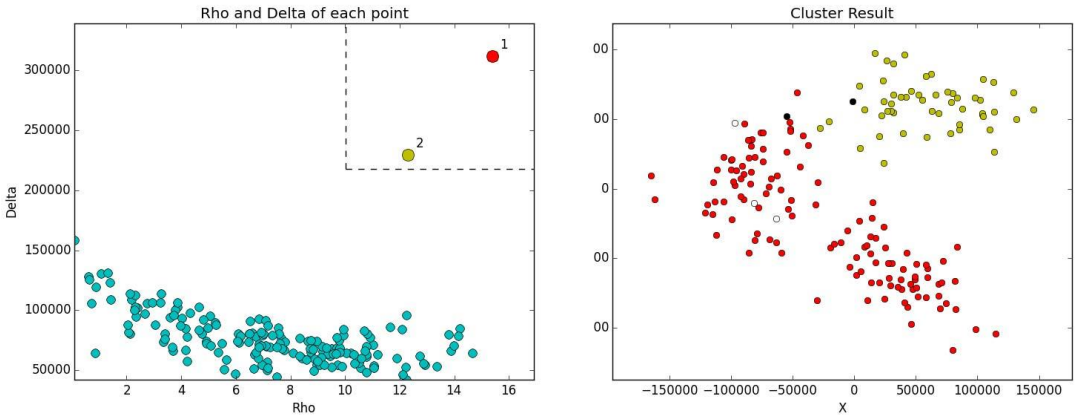
2．3　实验结果

程序从 Distance-Wine.txt 中读出 15753 条记录，共 178 个点。在 2%的位置取阀值，得到$d_c = 82595$。

最终分成 2 类，类中心的ρ_i阀值是 10.03，类中心的δ_i阀值是 217648.9。

类别 1：共 125 个点，类核心有 124 个，类边界有 1 个。类中心的点标号是 36 号。

类别 2：共 53 个点，类核心有 52 个，类边界有 1 个。类中心的点标号是 149 号。



三、数据模拟实验

3．1　数据集构造方法

这一部分的实验使用一个密度函数来生成数据集，直接生成点集的坐标，再根据坐标计算任意两点间的距离，再执行本聚类方法。

使用的密度函数z = $\frac{sin(x^2+y^2)}{0.1+r^2}+\frac{1}{2}\times e^{1-r^2}\times(x^2+y^2)$，$r^2=x^2+y^2$。该函数未进行归一化，最大值略小于 4，最小值略大于-0.2。该函数的负数绝对值都很小，绝对值大于 0.1 的数主要集中在-3 到 3 之间。函数图像见图 3.1。

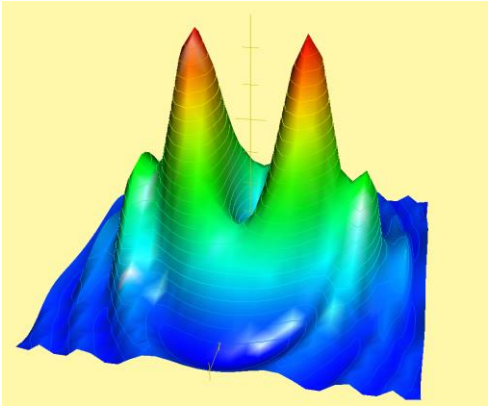


图 3.1　密度函数图像

在 x 轴的-3.5 到 3.5 之间均匀生成 150 个点，在 y 轴上做相同处理，两两相交得到 150*150=22500 个点，计算这些点的密度函数值。同时生成一个-0.1 到 6.9 之间的随机数，如果这个随机数比该点处的密度函数值小的话，那么就保留这个点。

经过几次模拟，得到了 1402 个点，其散点图如图 3.2。

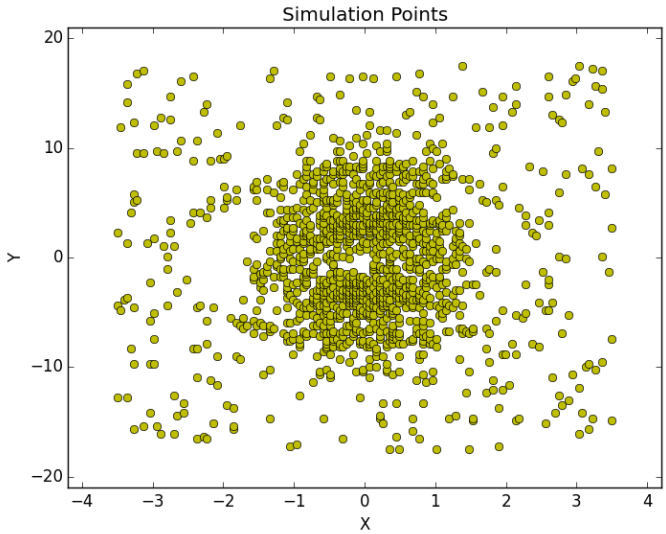


图 3.2　数据模拟原始数据散点图

生成模拟数据的代码见 Data-Simulation.py，需要提前新建 Output 文件。过程中会产生 Distance-Simulation.txt 包含任意两点间的距离，以及 Simulation Points.png 表示酒的散点图。

3．2　实验结果

使用 Main.py 读取 Distance-Simulation.txt 文件，进行聚类分析。

程序从 Distance-Simulation.txt 中读出 982101 条记录，共 1402 个点。在 2%的位置取阀值，得到$d_c = 0.706$。

最终分成 2 类，类中心的ρ_i阀值是 17.31，类中心的δ_i阀值是 5.32。

类别 1：共 689 个点，类核心有 659 个，类边界有 30 个。类中心的点标号是 675 号。

类别 2：共 713 个点，类核心有 691 个，类边界有 22 个。类中心的点标号是 739 号。

图 3.3 右图中的黑色点表示类的边缘点，黄色和红色表示类的核心点。

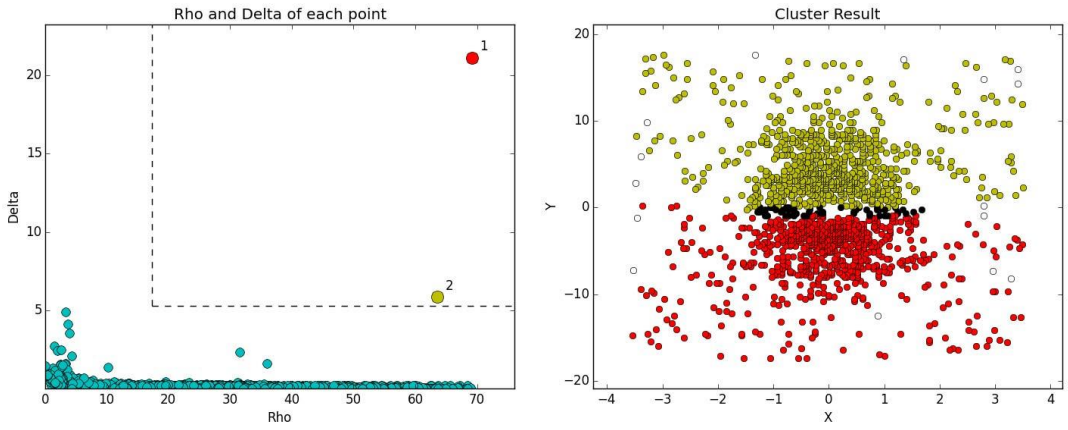


图 3.3　数据模拟结果

肖一凡
计算机系
2014210871
2014 年 12 月 29 日