华东师范大学计算机科学技术系上机实践报告

课程名称：人工智能 **年级：**2016 级 **上机实践成绩：**

指导教师：周爱民 **姓名：**汪春雨 **创新实践成绩：**

上机实践名称：数据聚类 **学号：**10152150127 **上机实践日期：**2018/5/15

上机实践编号：No.5 **组号：** **上机实践时间：**

一、 问题介绍

（本节介绍需要求解的问题是什么，为什么要采用我们介绍的方法求解）

1.1任务描述

实验数据如给定excel文件所示，真实数据分为2类，要求：根据特征向量x=(x1,x2,…,x6)采用聚类分析这些数据的特点，并根据真实数据验证聚类的准确性。可将excel数据格式化后导入文本文件，程序从文本文件读入数据，并输出类别及对应数据编号。

	A	B	C	D	E	F	G	H	I
1	样品编号	蛋白质mg/100g（x1）	PPH自由基1/IC50（g/L）（x2）	总酚(mmol/kg)（x3）	葡萄总黄酮（mmol/kg）（x4）	PH值（x5）	果皮质量（g）（x6）		类型
2	1	555.455	0.4314	23.576	9.509	3.54	0.120		红葡萄
3	2	624.094	0.4659	26.026	13.720	3.88	0.193		红葡萄
4	3	580.273	0.4102	21.479	10.853	3.80	0.160		红葡萄
5	4	527.438	0.2660	10.783	4.394	3.36	0.173		红葡萄
6	5	590.651	0.3972	18.547	10.333	3.58	0.260		红葡萄
7	6	532.026	0.2755	10.469	6.867	3.31	0.213		红葡萄
8	7	489.320	0.1758	9.181	3.497	3.13	0.136		红葡萄
9	8	556.091	0.4160	15.343	8.454	2.90	0.240		红葡萄
10	9	703.300	0.6689	31.767	20.433	3.68	0.150		红葡萄
11	10	547.695	0.3263	9.191	4.603	3.66	0.210		红葡萄
12	11	545.034	0.2796	6.197	2.545	3.46	0.125		红葡萄
13	12	491.265	0.1975	11.924	3.926	3.37	0.253		红葡萄
14	13	603.686	0.4420	14.572	7.360	3.91	0.170		红葡萄
15	14	597.274	0.3606	15.661	7.780	3.46	0.256		红葡萄
16	15	531.431	0.2193	12.001	5.598	3.16	0.208		红葡萄

图1 数据集示意图

1.2求解算法

1.2.1 *k*-means聚类算法

K-means算法是硬聚类算法，是典型的基于原型的目标函数聚类方法的代表，它是数据点到原型的某种距离作为优化的目标函数，利用函数求极值的方法得到迭代运算的调整规则。K-means算法以欧式距离作为相似度测度，它是求对应某一初始聚类中心向量x最优分类，使得评价指标V最小。算法采用误差平方和准则函数作为聚类准则函数。

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

K-means算法是很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。该算法认为簇是由距离靠近的对象组成的，因此把得到紧凑且独立的簇作为最终目标。

k个初始类聚类中心点的选取对聚类结果具有较大的影响，因为在该算法第一步中是随机的选取任意k个对象作为初始聚类的中心，初始地代表一个簇。该算法在每次迭代中对数据集中剩余的每个对象，根据其与各个簇中心的距离将每个对象重新赋给最近的簇。当考察完所有数据对象后，一次迭代运算完成，新的聚类中心被计算出来。如果在一次迭代前后，V的值没有发生变化，说明算法已经收敛。算法具有较好的效果，但是计算量很大。

1.2.2层次聚类

k-means算法却是一种方便好用的聚类算法，但是始终有K值选择和初始聚类中心点选择的问题，而这些问题也会影响聚类的效果。为了避免这些问题，我们可以选择另外一种比较实用的聚类算法-层次聚类算法。顾名思义，层次聚类就是一层一层的进行聚类，就划分策略可分为自底向上的凝聚方法（agglomerative hierarchical clustering），比如AGNES。自上向下的分裂方法（divisive hierarchical clustering），比如DIANA。

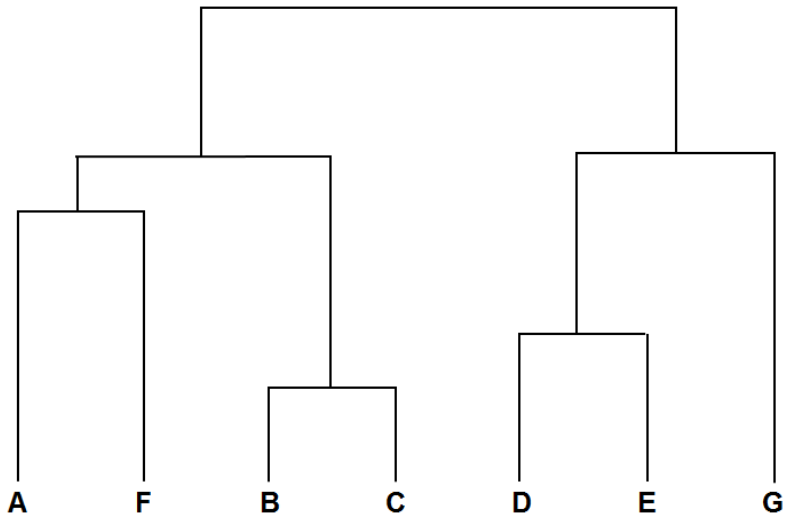

图2 自底向上的层次聚类

如何判断两个cluster之间的距离：

1.最小距离，单链接Single Linkage

两个簇的最近样本决定。

2.最大距离，全链接Complete Linkage

两个簇的最远样本决定。

3.平均距离，均链接Average Linkage

两个簇所有样本共同决定。

1和2都容易受极端值的影响，而方法3计算量比较大，不过这种度量方法更合理。

二、 程序设计与算法分析

（本节介绍程序设计的分析过程，如数据结构定义，算法描述，算法框图等）

2.1数据预处理

将数据读入Dataframe，去除无关列（数据中有空列），第一列作为数据index，最后一列红葡萄类标为0，白葡萄类标为1。

得到如下数据（前五行）：

id	x1	x2	x3	x4	x5	x6
1	555.454722	0.431417	23.575516	9.508588	3.540000	0.120000
2	624.094318	0.465868	26.026218	13.720350	3.883333	0.193333
3	580.272532	0.410171	21.479025	10.853316	3.800000	0.160000
4	527.437998	0.266041	10.783139	4.394164	3.360000	0.173000
5	590.651064	0.397196	18.547204	10.333078	3.576667	0.260000

数据标准化化（normalization）处理：采用**Max-Min Normalization**

$$a_i' = \frac{a_i - min(a_i)}{max(a_i) - min(a_i)}$$

id	x1	x2	x3	x4	x5	x6
1	0.509054	0.584349	0.682254	0.369932	0.587156	0.108108
2	0.736983	0.644656	0.769298	0.547701	0.902141	0.366627
3	0.591466	0.547159	0.607790	0.426690	0.825688	0.249119
4	0.416020	0.294857	0.227890	0.154065	0.422018	0.294947
5	0.625930	0.524446	0.503657	0.404732	0.620795	0.601645
6	0.431255	0.311489	0.216734	0.258458	0.373089	0.437133

数据类标（按顺序）：

[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]

2.2 *k*-means聚类算法

2.1.1经典的*k*-means算法

k-means算法以k为参数，把n个对象分成k个簇，使簇内具有较高的相似度，而簇间的相似度较低。k-means算法的处理过程如下：首先，随机地选择k个对象，每个对象初始地代表了一个簇

的平均值或中心，即选择K个初始质心;对剩余的每个对象，根据其与各簇中心的距离，将它赋给最近的簇;然后重新计算每个簇的平均值。这个过程不断重复，直到准则函数收敛，直到质心不发生明显的变化。通常，采用平方误差准则，误差的平方和SSE作为全局的目标函数，即最小化每个点到最近质心的欧几里得距离的平方和。此时，簇的质心就是该簇内所有数据点的平均值。

（1） 从 n个数据对象任意选择 k 个对象作为初始聚类中心；

（2） 根据每个聚类对象的均值（中心对象），计算每个对象与这些中心对象的距离；并根据最小距离重新对相应对象进行划分；

（3） 重新计算每个（有变化）聚类的均值（中心对象）

（4） 循环（2）到（3）直到每个聚类不再发生变化为止

2.1.2优化后的*k*-means算法

由于该算法具有较大的计算量，我们可以对该算法进行优化，不需要每次交换两个元素之后都立即算误差平方和，这样计算量很大，我们可以通过一些数学变换，减少计算量，具体请参考《模式识别》第二版，p236，边肇祺，清华大学出版社。

下面给出优化后的*k*-means算法：

(1) 选择把 N 个样本分成 C 个聚类的初始划分，计算每个聚类的均值 $\boldsymbol{m}_1,\boldsymbol{m}_2,\cdots,\boldsymbol{m}_c$ 和 J_e。

(2) 选择一个备选样本 $\boldsymbol{y}$，设 $\boldsymbol{y}$ 现在在 Γ_i 中。

(3) 若 $N_i=1$，则转(2)，否则继续。

(4) 计算

$$\rho_j=\begin{cases}\dfrac{N_j}{N_j+1}\|\boldsymbol{y}-\boldsymbol{m}_j\|^2 & j\neq i\\ \dfrac{N_i}{N_i-1}\|\boldsymbol{y}-\boldsymbol{m}_i\|^2 & j=i\end{cases} \tag{10-37}$$

(5) 对于所有的 j，若 $\rho_k\leqslant\rho_j$，则把 $\boldsymbol{y}$ 从 Γ_i 移到 Γ_k 中去。

(6) 重新计算 $\boldsymbol{m}_i$ 和 $\boldsymbol{m}_k$ 的值，并修改 J_e。

(7) 若连续迭代 N 次 J_e 不改变，则停止，否则转到(2)。

2.3 层次聚类

凝聚型层次聚类（自底向上）的策略是先将每个对象作为一个簇，然后合并这些原子簇为越来越大的簇，直到所有对象都在一个簇中，或者某个终结条件被满足。这里给出采用平均距离，均链接Average Linkage的凝聚层次聚类算法流程：

(1)将每个对象看作一类，计算两两之间的最小距离；

(2)将距离最小的两个类合并成一个新类；

(3)重新计算新类与所有类之间的距离；

(4)重复(2)、(3)，直到所有类最后合并成一类（此问题中为两类）。

分裂型层次聚类（自顶向下），和上述过程相反，对于二分类问题的简单算法如下：

（1） 从数据集中找到距离最远的两个数据点x1,x2,类标分别为0,1

（2） 选择一个样本，距离x1近，类标设为0，否则类标设为1

（3） 重复（2），遍历所有的点

三、 实验结果

（本节列出实验的结果，必要时加入一些自己的分析）

3.1 *k*-means

分类结果为（按顺序的类标，类标可以是相反的，下同）

[0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 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, 0, 1, 1, 0, 1]

误差平方和：

err_sum 20.73089248987008

准确率：

acc = 0.6545454545454545

3.2 sklearn *k*-means

这是python提供的机器学习库sklearn中的库方法

分类结果为：

[0 0 0 1 0 1 1 0 0 1 1 1 0 0 1 1 0 1 0 1 0 1 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 0 1 1 0 1]

准确率：

acc = 0.6909090909090909

3.3 层次聚类（自底向上方法）

分类结果为：

[0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 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]

准确率：

acc = 0.5454545454545454

3.4 层次聚类（自上向下方法）

分类结果为：

[1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
1 1 1 0 1 1 0 1 1 1 1 0 1 1 1 1 1 0]

准确率：

acc = 0.5454545454545454

3.5结果分析

对于此次的的聚类任务来说，聚类的结果并不理想，有可能是数据集过小的缘故。

我认为此次任务并不适合无监督学习，因为各个特征的权重是不相同的，可能果皮质量这一特征对聚类结果影响较大，而且不同特征之间可能具有某种关联规则。

此次的任务用有监督学习的方法，如分类（二分类）会更加合适，具体方法如**决策树，SVM,Adaboost**等，相信会有更好的效果。

四、 附件

（本节非必须的，可以列出源代码等，但是要把格式组织好）

工程文件夹： Clustering

Config.py:	配置文件，存放文件路径常量
k-means.py:	自己实现的k-means算法
sklearn_kmeans.oy:	sklearn库提供的API实现聚类
hierarchy.py:	层次化聚类的实现，包括凝聚和分裂
utils.py:	数据处理方法工具集合
data:	数据存取文件夹