ML实验报告：聚类

霍超凡

2019 年 12 月 28 日

1 实验内容

K-means算法设计与实现

2 实验要求

- 编程实现K-means算法，并在红酒数据集上运行。
- 设置不同K值，不同初始中心，在红酒数据集上进行实验比较。
- 分析k-means的优缺点，并对其中一个或几个缺点进行改进。
- 演示实验并提交代码，统计分析实验结果并上交实验报告；

3 实验环境

Jupyter Notebook + Python3

4 实验原理

聚类属于无监督学习，这种方法假设相同类别中的不同实例存在相似的特征，特征越相似的实例越有可能属于同一个类别，通过挖掘已给实例的相似性，判断隐藏类别标签。聚类中有两大问题需要解决，其一是如何度量实例之间的相似性，度量相似度的常用方法是使用欧式距离，两个实例相似度与两个实例间的距离呈负相关，此外还有基于密度的度量方法、基于实例间关联路径的方法等。其二是如何将相似的实例聚为一类，解决这个问题的主要算法有层次聚类、谱聚类、k-means、KNN聚类等。一个聚类算法应用到实际问题中还需要讨论特征选择与提取的问题，本次实验从这个三个方面出发，在Wine数据集上对比了不同的算法和策略。

4.1 特征选择与提取

在机器学习中，特征工程主要分为三个方面：特征表示、特征选择、特征提取。特征表示研究的主要问题是，如何定义特征、如何量化表示特征；特征选择要求我们从定义好的特征中筛选出与我们所研究的问题相关的特征；特征提取将已有的特征转化成新的特征。在聚类中特征选择和提取尤为关键，一些无关特征会影响到聚类结果，我们进行特征选择

的最终目的是为了消除冗余特征、无效特征，冗余特征会对我们的算法效率造成影响，而无效特征会影响我们的算法性能。特征提取最常用的方法是奇异值分解，经过奇异值分解能够消除一些冗余属性。特征选择的一般思路有三种：一是制定一个描述特征好坏的指标，从已有的特征中选取具有最优子集，对于聚类来说，我们的目标要尽可能将相似的实例聚为一类，而不同类的实例差异较大，这可以使用方差来表示。然而聚类算法和监督算法不同，在特征选择阶段我们没有每个实例的类别标签，所以特征选择通常会和聚类算法相结合，将选取的特征在算法中运行一遍，通过运行的结果来判断所选取特征的好坏。在实验中我们使用逐步先前搜索的方法，从一个空的属性集合出发，逐步加入能够提高指标的属性，直到加入属性后不能再提高性能或者已经没有可选取的属性。关于评价聚类性能的指标，我们将在后面的章节中讨论。

4.2 相似度衡量

衡量两个样本之间相似度的常用方法是使用欧式距离，欧式距离是Minkowski距离的特例，Minkowski距离的定义如下：

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt[p]{\sum_{k=1}^{d} (x_{ik} - x_{jk})^p} \tag{1}$$

参数p的选择与所讨论的问题相关，p的值制约较大值属性对距离的影响程度，越高的p值越重视最大值，当$p = +\infty$时，变成Chebyshev距离，较低的p值会削弱最大值对距离的贡献，当$p = 0$时，所有分量对结果的贡献均为1。一些算法并不直接使用以上距离公式，而是将距离经过高斯平滑，消除一些距离较大的值，将距离转换成相似度，

$$similarity(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}} \tag{2}$$

对于不同的问题σ通常会取不同的值，σ取较大的值，会将一些距离较远的样本聚为一类，相反σ取较小的值时，同一类别样本会被划分到不同的类别中，σ的确定可以根据距离直方图来确定，[3]提出了一种自适应的方法，这个方法对不同的样本赋予不同的σ，假定一个样本至少需要和η个其它样本相邻接，第i个样本的σ由下式确定,

$$e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_i^2}} = \eta \tag{3}$$

σ可以使用优化搜索的方法确定。 Minkowski距离适合于刻画那些凸区域，但是对于那些非凸区域，需要使用其它的距离来衡量，比如基于密度的距离衡量方式，[3]使用了一个比较奇特的方法，受电子电路的启发，将样本看成电路中的节点，在两个节点通入相同的电流，使用两个节点的电压差来表示两个节点的相似度，通入节点的电流，会经过不同路径从源头流向终点，两个节点的电阻使用样本的距离来衡量，这种方法不仅考虑到两个节点的直接相邻关系，还考虑到两个节点的亲邻情况，两个节点如果能够通过较多的路径“轻松”抵达，那么这两个节点就具有较高的相似度。

4.3 聚类算法

层次聚类

层次聚类是一种逐步融合相似度较高的簇的方法，根据簇间距离计算的方式不同可以分为single-link层次聚类、complete-link层次聚类、average-link层次聚类，single-link层次

聚类将两个簇间节点最短距离作为两个簇的距离，complete-link层次算法将两个节点最长距离作为两个簇的距离，average-link层次聚类算法将分别位于两个簇的结点间平均距离作为两个簇的距离。层次聚类首先将所有样本各自作为一类，之后逐步将具有最小距离的两个簇融合，直到所有样本融合成一个簇，在融合过程中会将融合过程存储起来，之后根据不同的阈值选取不同程度融合的簇。层次聚类有两个问题需要解决，其一是计算量太大，其二超参数不易确定。

k-means聚类

k-means聚类是一种简单高效的聚类算法，这个算法从最小化类内方差出发，逐步移动中心节点使最终各个簇的簇内方差之和最小。k-means需要确定人为确定类别个数k，k值的确定可以通过观察距离直方图粗略估计，k-means算法首先随机选取k个初始中心点，按照距离远近将剩余样本点划分到不同簇中，之后重新计算中心点，簇内中心点的选取遵循最小化簇内方差原则选取簇内“质心”作为新的簇，重复进行划分剩余节点、更新中心节点两个步骤，直到中心节点收敛。k-means算法的收敛速度和结果的正确性受初始节点选择的影响，一个简单的解决办法是多次重复运行k-meads算法，选取最优的结果作为最终结果。另外一种方法是限制初始点的选择，选择距离相距较远的节点使初始点尽可能位于不同的簇中。

谱聚类

谱聚类利用矩阵谱分解原理，对节点之间的相似度矩阵进行奇异值分解，0奇异值所对应的特征向量为指示向量，之后通过分析指示向量来获得各个类的类别。 [5]十分详细地介绍了谱分解的算法及其原理，从三种角度解释谱聚类算法。谱聚类算法在实际表现中并不好，谱聚类对相似度矩阵的要求较为苛刻，相似度矩阵必须能够通过矩阵行列变换操作形成块对角矩阵，否则很容易将所有节点视为一个簇，而实际问题中的相似度矩阵很难形成严格的块对角矩阵，[3]通过修改样本点之间的距离计算方式来提高谱聚类的性能，其简要思路已经在前面的章节介绍过，具体的细节请参照原文。

4.4 聚类结果评价

我们最初的假设是一个类内样本的特征相似度高，类间样本特征相似度差异较大，我们的聚类目标自然是将相似度高的样本聚为一类，将相似度差异较大的分成不同的类，可以使用类内所有样本的方差来表示类内样本的差异，不同类中心点之间的距离来表示类间差异，将类间方差比上类内方差作为最终评价指标，这个值越大聚类效果越好。但是这种衡量标准不适用于k值不同的聚类结果对比，被分成较多类别的聚类结果明显占优势，在实验中我们使用它的另外一个更复杂的形式，这种衡量准则出自于[1]，定义类内聚类差异

$$S_w = \sum_{j=1}^{k} \pi_j E[(X-\mu_j)(X-\mu_j)^T|\omega_j] = \sum_{j=1}^{k} \pi_j \Sigma_j \tag{4}$$

类间差异

$$S_b = \sum_{j=1}^{k} \pi_j (\mu_j - M_o)(\mu_j - M_o)^T \tag{5}$$

其中，M_o为全部样本的中心点，即$M_o = E[X] = \sum_{j=1}^{k} \pi_j \mu_j$。最终的评价指标被定义为

$$SSC = trace(S_w^{-1} S_b) \tag{6}$$

另外一种不受k值影响的衡量标准是Dunn指数，这个指数总是考虑最坏的分类结果，将两个簇之间的距离定义为分别位于两个簇的两个点之间的最小距离，即

$$d(C_i, C_j) = \min_{x_i \in C_i, x_2 \in C_j} distance(v_1, v_2) \tag{7}$$

一个簇内的差异度使用簇内两个点的最远距离表示，

$$diam(C_k) = \max_{v_1 \in C_k, v_2 \in C_k} distance(v_1, v_2) \tag{8}$$

Dunn指数被定义为

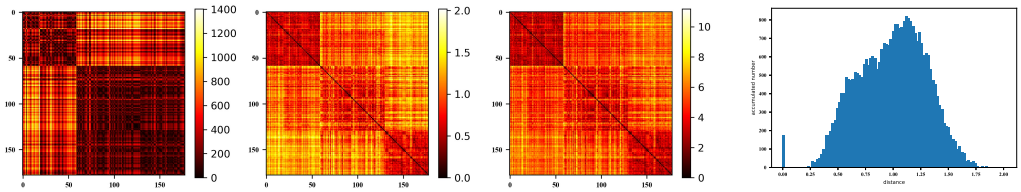
$$DI = \frac{\min_{i \neq j} d(C_i, C_j)}{\max_k diam(C_k)} \tag{9}$$

DI指数只考虑到最坏的分类情况，不受分类簇个数的影响。

5 实验细节及结果

5.1 数据预处理

加载数据集Wine后，查看数据分布，发现各个属性的取值差异较大，需要对数据归一化处理，归一化有两种方式，一种是将取值范围压缩到0和1之间，另外一种方式是不仅压缩取值，而且使各个属性的方差扩大或缩小到1。在实验中分别使用两种方法处理数据，绘制其样本间距离矩阵如图1(a)所示，可以看出经过处理后的数据中不同类别间的差异拉大，不归一化方差相比归一化后差异更明显，之后实验的全部数据均经过压缩取值范围处理，而没有对方差作出处理。



(a) 距离矩阵　　(b) 距离直方图

图 1: 在1(a)中，从左往右分别是原始数据、压缩取值范围的数据、压缩取值范围和方差的数据的距离矩阵，横纵坐标为样本的索引，矩阵中每个点表示两个样本间的距离，距离使用欧式距离计算。由于原始数据集中的样本已经按照类别标签排好顺序，所以距离矩阵出现非常明显的块对角的性质。图1(b)为图1(a) （中图）的距离直方图，直方图中没有形成明显的多峰值分布，从直方图不容易观察出不同类别间距离的阈值。

5.2 相似性度量

在实验过程中我使用了不同距离度量方式，在k-means算法中使用Minkowski距离，并且讨论了不同p值对结果的影响，并且通过优化搜索的方法找到p的最优取值。在谱聚类中使用了高斯平滑函数，数据集中不同类别之间的差异性不大，相似度矩阵无法形成明显的块对角分布，无法使用谱聚类，在实验过程中为了理解谱聚类的算法，我将k-meas的分类结果标签作为一个属性加到原始数据上，然后使用谱聚类的方法在这个有标记的数据集上聚类，效果较好，并且从结果能够发现k-means算法一些被错误聚类的样本。关于细节部分将会分散在之后各个小节中介绍。

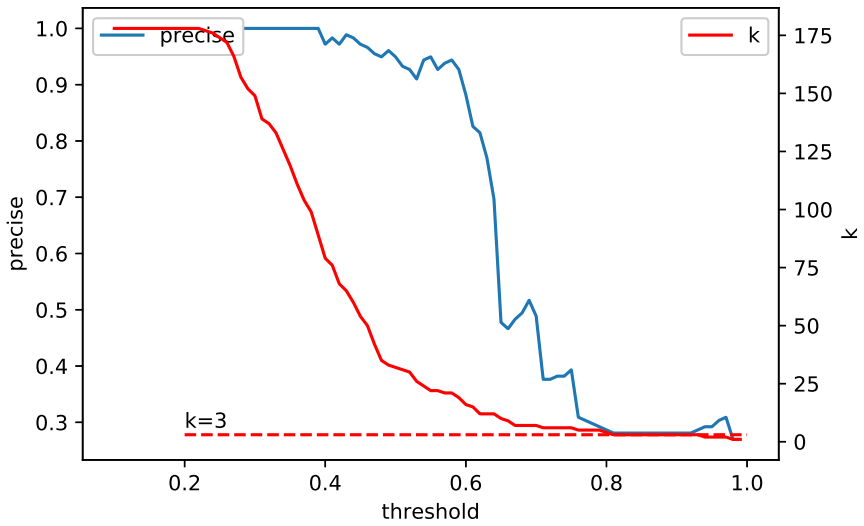


图 2: 基于距离阈值的聚类结果，随着阈值的最大，聚类精度逐渐减低，聚类的类别数目逐渐减少。

5.3 不同聚类算法

层级聚类

实验中分别实现基于不同计算簇间距离计算方式的层级聚类，层级聚类的计算量较大，我根据层级聚类的思想首先实现了基于距离阈值的聚类方法，这种方法逐步将距离小于阈值的点聚为一类，这种方法在实际运行中效果并不好，若取较小的阈值划分的类别太多，若取较大的阈值，精确度太低。如图3(a)所示，当类别数为3时，精度不足0.3。层级聚类算法的复杂度较高，设置合适的阈值使用层次聚类算法将数据聚成3类，得到的如果如表1所示，效果不好，这是由于数据自身分布不适于使用层次聚类。

表 1: 层级聚类的三种方法结果

方法	single-link	complete-link	average-link
精度	0.398	0.713	0.28

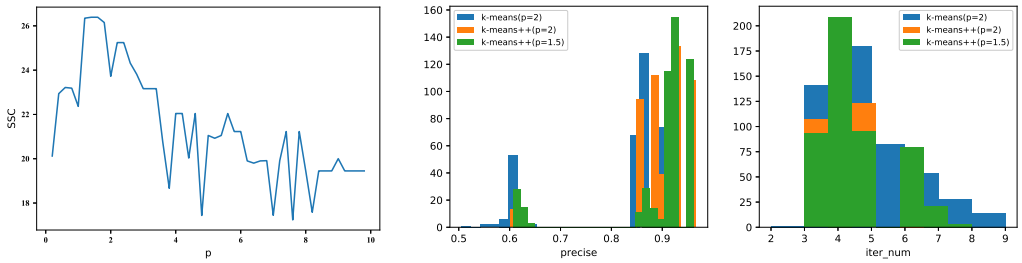
k-means聚类

实验对比了随机选取初始点的k-means算法和选取距离较远的点作为初始点的k-means++算法，重复运行两个算法500次，统计准确度和迭代次数，结果如图 3(b)所示，k-means++算法具有较大概率得到更高的准确度，迭代次数集中在4左右。

为了检验距离计算方式对k-means结果的影响，我使用不同p值的Minkowski距离距离计算方式多次运行k-means算法，采用0.618搜索方法找到p的最优阈值为1.5左右。从3(b)可以看出，p=1.5比p=2更能在较大的概率获得更高的准确度。

谱聚类

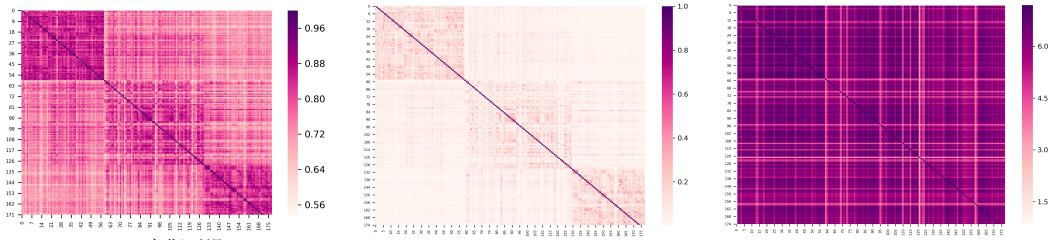
在实验中尝试谱聚类时，谱聚类总是将数据集聚为一类，这是由于数据集样本所形成的相似度矩阵不是块对角矩阵，如图4所示，相似度矩阵虽然肉眼可以看出存在块对角，谱



(a) p值对结果的影响　　(b) k-means

图 3: 图3(a)是式1中p值对k-means算法的影响，纵坐标为式6所计算出的衡量聚类算法效果的标准。 3(b)是k-means聚类算法聚类的结果，k-means算法随机选取初始点，k-means++选取距离较远的初始点。

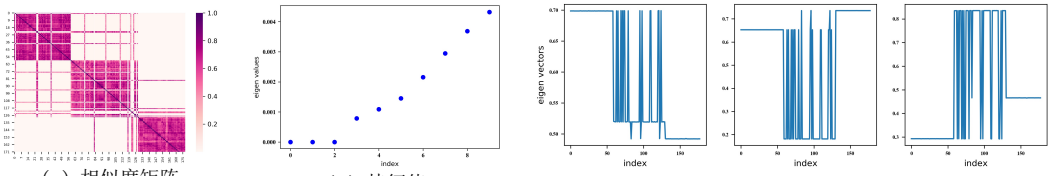
聚类算法要求那些类间的相似度要近乎接近于0。在实验中在处理原数据集时忘记复制，结果把k-means预测的标签当作属性添加到原数据集中，偶然间得到了块对角矩阵，事后才发现是自己在画散点图时，没有采用深复制，不同变量名背后指向同一个DataFrame，但是我还是继续按照谱聚类的步骤走了一遍，加深了对谱聚类的影响。



(a) 高斯平滑　　(b) 自适应高斯平滑　　(c) 电流通路相似度

图 4: 图4(a)是按照欧式距离计算样本之间的距离矩阵，然后使用高斯核将距离矩阵转化成相似度矩阵。图4(b)是按照 [3]所提出的自适应选取σ所计算出的相似度矩阵，图4(c)是按照[3]所提出的电流通量计算的相似度矩阵。

在使用误添加k-means的聚类结果继续谱分解时，发现了k-means聚类一些错误的聚类结果，如图5所示，主要问题存在于第二个类别，k-means将第二个类别的样本过多误判成第一类别。这不是k-means算法自身的缺点，这是数据集自身的原因，如图1(a)（中图）所示，类别1和类别2存在较高的相似度，类别一和类别二在图1(a)（中图）中重叠的区域稍稍暗于类别一和类别三重叠的区域。



(a) 相似度矩阵　　(b) 特征值　　(c) 特征向量

图 5: 谱聚类的中间结果

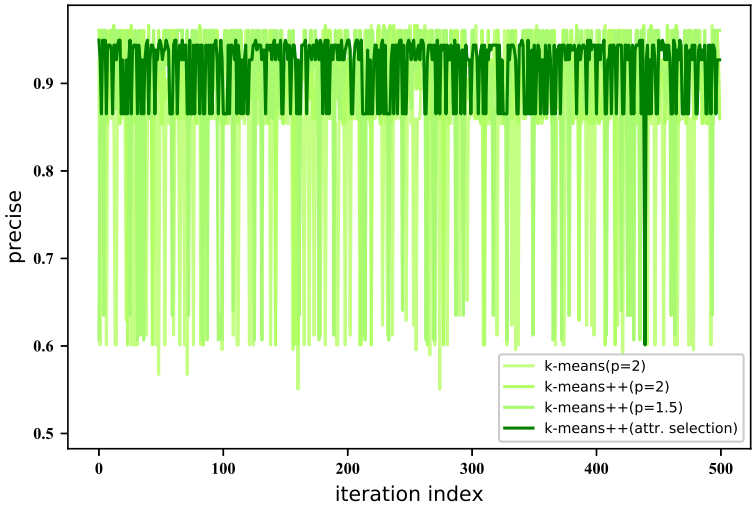


图 6: 这幅图展示了经过属性筛选和没有经过属性筛选对k-means算法稳定性的影响。横坐标为迭代次数，纵坐标为准确度。经过筛选后的属性集合为['OD280/OD315 of diluted wines', 'Total phenols', 'Flavanoids', 'Proline', 'Hue', 'Color intensity', 'Proanthocyanins', 'Alcohol', 'Magnesium']

5.4 聚类效果评价指标

在实验中，我发现Wine数据集和前面两个评价指标均存在不一致的情况，使用原数据集自带的类别标签按照式6计算时，得到22.9，使用k-means聚类结果计算SSC时，得到26.1，精度为0.96的SSC值不如精度为0.92的SSC值，这就导致我们的k-means算法明明得到更好的结果却浑然不知，返回一个较差的结果，另外一个指数DI也同样如此。

5.5 特征选择

最后，我使用[1]所提出的方法选择特征，在实现中遇到一些问题，在使用贪婪算法逐步扩大所选出的属性子集时，发现两个属性的聚类效果不如一个属性的聚类效果，这就导致算法在选择一个特征之后就停止了，我在特征选择算法中添加一个限制，即至少选出一半属性才能停止，结果确实选择了一半稍多一点的属性，得到的结果准确度可以超过0.9，之后没有继续探究再减少一些属性数目还不能保持这样的准确度。我还发现经过属性筛选后，不仅可以提高算法的运行速度，而且极大的提高了算法的稳定性，如图6所示，k-meas算法在经过属性筛选的数据集上准确度基本完全保持在0.9左右。

6 结论

聚类存在很多算法，k-means算法是其中一个高效且准确度高的算法，在解决聚类问题时，根据数据集样本的分布选择不同的聚类算法，对于Wine数据集来说，样本没有形成很明显的界限，簇之间的差距较小，这导致无论是经过高斯平滑还是使用电流通路计算两个样本之间的相似度，都不能形成理想的相似度块对角矩阵。影响聚类算法性能的一个重要因素在于相似度计算方式，本次实验发现使用p值约等于1.5左右的Minkowski距离公式可以达到很好的结果。最后属性的选择和预处理对结果也存在很大的影响，通过属性选择，出去一些干扰属性或冗余属性可以提高算法的性能和效率。

参考文献

[1] Jyoti Adhikary and M. Murty. Feature selection for unsupervised learning. volume 7665, pages 382–389, 11 2012.

[2] K.A.J. Doherty, R.G. Adams, and N Davey. Unsupervised learning with normalised data and non-euclidean norms. *Appl. Soft Comput.*, 7, 01 2007.

[3] Igor Fischer and Jan Poland. Amplifying the block matrix structure for spectral clustering. 03 2005.

[4] A.K. Jain, M. Murty, and P.J. Flynn. Data clustering: A review. *ACM Comput Surv*, 31:264–323, 01 1999.

[5] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 01 2004.