2023秋CS64038课程报告

姓名：傅彦璋
学号：23S003008
专业：计算机科学与技术
电子邮件：fuyanzhang@fyzdalao.xyz

一、课程报告

1. 频繁项集、序列和子图挖掘

1)三个问题的统一抽象描述

定义 1 频繁模式挖掘问题

集合$T = \{t_1, t_2, ...t_m\}$。映射$f(\cdot,\cdot) : T \times T \to \{0,1\}$称为包含函数。若$f(t_a, t_b) = 1$，我们称$t_a$包含$t_b$。多重集$S = \{s_1, s_2, ..., s_n\}$满足$s_i \in T, i = 1, 2, ..., n$。若某个$t \in T$满足$\sum_{i=1}^{n} f(s_i, t) \geq \delta$，则称$t$是$S$中的$\delta$-频繁模式。对于给定的$S$和常数$\delta$，频繁模式挖掘问题就是寻找$S$中$\delta$-频繁模式的问题。

频繁项集挖掘问题（定义2）、频繁序列挖掘问题（定义3）和频繁子图挖掘问题（定义4）均可以视为频繁模式挖掘问题的特例。

定义 2 频繁项集挖掘问题

频繁项集挖掘问题是一种频繁模式挖掘问题。在定义*1*中，若有项目集合I使得$\forall i \in [1, m], t_i \in 2^I$，且

$$f(t_a, t_b) = \begin{cases} 1 \ , \ t_b \subseteq t_a \\ 0 \ , \ else \end{cases}$$

则称该频繁模式挖掘问题是一个频繁项集挖掘问题。

定义 3 频繁序列挖掘问题

频繁序列挖掘问题是一种频繁模式挖掘问题。在定义*1*中，若$\forall i \in [1, m], t_i$是序列，且$f(t_a, t_b) = 1$当且仅当$t_b$是$t_a$的子序列，则称该频繁模式挖掘问题是一个频繁序列挖掘问题。

定义 4 频繁子图挖掘问题

频繁子图挖掘问题是一种频繁模式挖掘问题。在定义*1*中，若$\forall i \in [1,m], t_i$是图，且$f(t_a,t_b)=1$当且仅当t_b是t_a的子图，则称该频繁元素挖掘问题是一个频繁模式挖掘问题。

对于三个不同问题，一个通用且有用的性质是：如果$f(s,t_1)=0, f(t_2,t_1)=1$，则$f(s,t_2)=0$。即如果s不包含t_1，那么它一定也不包含t_2，所以我们就可以将t_2剪枝掉。这是三个问题很重要的共同点。

2)问题难度的递增

而如果我们已知一个模式t是频繁的，进而想构造包含t的极小的另一个模式s（即$f(s,t)=1$且s极小），可能会产生很多s，进而产生非常大的代价。

在频繁项集挖掘中，构造s相对容易，只需要取t的超集。

在频繁序列挖掘中，因为有次序的不同，由一个频繁的序列t构造极小的s可能会产生相当多的序列。在算法中我们可能需要验证这些序列是否已经被剪枝、是否频繁，代价比较大。

在频繁子图挖掘中，一个频繁子图t的极小母图s不仅数量多，而且难以表示。这让判断s是否已经被剪枝非常困难。更重要的是，子图匹配本身就是NP完全的，所以准确验证一个子图是否是频繁的这项工作是不可能的。

所以，频繁项集挖掘、频繁序列挖掘、频繁子图挖掘三个问题的难度是递增的。

3)减少结果数量

为了减少频繁模式挖掘的结果数量，可以提高支持度阈值，让更少的模式符合“频繁”的要求。可以限制模式的内容，比如在频繁项集挖掘中排除掉较小的项集。

2. A New Sparse Data Clustering Method Based on Frequent Items

1)该论文提出的算法与经典kMeans算法最大的差别

论文提出的算法与经典kMeans算法的最大区别在于计算聚类中心的方式不同。在经典kMeans和其衍生算法中，往往使用聚类中数据点的欧氏坐标的均值来表示聚类中心。该论文设计“FreqItem”，忽略低频出现的特征的同时，使用杰卡德距离代替欧几里得距离，来计算聚类中心。

2)经典kMeans算法不能很好解决改论文研究的聚类问题的原因

该论文的聚类问题针对高维度、分散的数据聚类，这样的数据特征给经典kMeans带来了困难。

第一，对于高维分散的数据，传统的计算聚类中心的方式并不好。一些噪音维度可能会给结果带来干扰。经典kMeans缺少滤除噪音、将注意力集中在重要特征上的机制。

第二，传统的初始聚类中心选取方法代价过高。对于大规模的数据，k-Means++等初始中心生成方式代价很高。而由于kMeans的只能收敛至局部最优解的特点，初始中心又相当重要。针对杰卡德距离，经典kMeans缺少更高效的初始中心生成方法。

3)聚类技术与频繁项集挖掘的融合

kMeans极其衍生算法的初始聚类中心选取相当重要。直观上，初始聚类中心应该由某些密集的数据点产生。该论文设计了SILK Overseeding算法来取得潜在的初始聚类中心。该算法利用若干独立的局部敏感哈希MinHash构造若干哈希表，得到的每个桶中的数据大概率是相近（杰卡德距离小）的。如果某些数据点组成的集合在相似的桶中频繁出现，则可认为这些数据点比较密集，应该用它们构造一个候选的初始聚类中心。这里，寻找频繁出现的数据点集合即可视为频繁项集的挖掘。

3. The Case for Learned Index Structures

1)将数据库索引设计问题转化为机器学习问题

该论文认为，像B树这类现有的索引结构都是一种通用的结构，它们不对数据分布模式有任何假设，也自然不会从数据分布模式中获益。如果我们能够习得数据库中的数据分布的模式特点，就能够针对性地设计高效的索引。

索引结构就是模型。不论是何种索引，皆可以看作一种模型——为其输入一个键值，其输出一个位置。机器学习方法可以习得一个模型，即索引结构，其反映并利用数据分布模式以提供高效的索引。所以索引结构设计问题被转化为机器学习问题。

2)基于机器学习的索引结构的好处

真实数据往往存在某种规律，基于机器学习的索引结构能够从中获益，极大提升时间和空间效率。

机器学习模型，尤其是神经网络很擅长处理复杂的关系，这使得基于机器学习的索引结构适合用在二维或多维索引上。

CPU的摩尔定律已死，但GPU仍在发展。基于机器学习的索引结构在未来可能会有更好的性能表现。

二、实验报告

1. 实验内容

我的课程实验将在经典的鸢尾花数据集上尝试kMeans、DBScan和OPTICS三种聚类算法，并分析和解释它们聚类结果的异同。选择这样的问题的动机有二。一是我在实践上完全不懂常用的数据处理手段，打算借此学习了解。二是对完全不同原理的聚类算法之间的差异比较感兴趣。

2. 算法

1)kMeans

kMeans算法是基于距离的聚类算法——它用距离衡量样本之间的相似程度。两个样本距离越近，则认为其越相似，越应该被划分在一个聚类中。算法在生成k个初始聚类后，不断地改变每个点的归属，令其加入距离其最近的聚类，直至稳定。显然，kMeans算法的一个麻烦之处在于常数k在算法开始前需要指定。如果对数据完全没有专家知识，这可能是棘手的。

2)DBScan

与kMeans不同，DBScan是基于密度的聚类算法。算法将密度高的样本区域划分为聚类。算法由某一样本点开始，探索其ϵ-邻域。若邻域中由足够多（多于$minPts$个）的样本点，则建立一个聚类，并添加所有能够ϵ-连通的样本点。因为DBScan算法基于密度，所以其能够滤掉噪点——如果某点的邻域内没有足够量的点，则认为其是噪点。

3)OPTICS

OPTICS是DBScan的改进算法，更为精细地定义了距离和可达的细节，使得算法对参数ϵ不那么敏感。这样便可以解决DBScan不能识别密度不同的聚类的问题。

3. 评价指标

我们使用聚类的完整性、同质性和V-measure来评价聚类结果。同质性$h = 1 - \frac{H(C|K)}{H(C)}$，完整性$c = 1 - \frac{H(K|C)}{H(K)}$.其中

$$H(C|K) = -\sum_{c,k} \frac{n_{ck}}{N} log\left(\frac{n_{ck}}{n_k}\right)$$

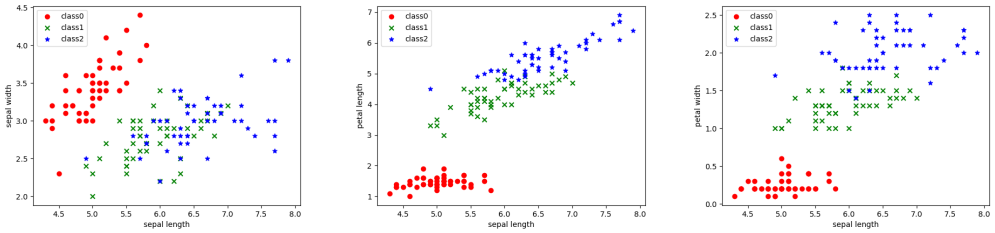
K表示算法给出的聚类，C表示真实的类别，n_{ck}是聚类K中类别为c的样本数量，n_k是聚类K中样本数量。

我认为获得每种聚类的关键特征比较重要，所以主要使用同质性来评价聚类结果。考虑到基于密度的算法DBScan和OPTICS会删除“噪点”，所以用完整性评价其结果很可能比较差。这看似不公平，但也有参考意义，因为正如我们在最后一节课看到的，OPTICS的的确确存在产生的聚类比真实聚类小的问题。V-measure是完整性和同质性的调和平均值。

4. 实验结果

1)真实类别

鸢尾花数据集的每个样本包含四个浮点数特征：SepalLength、SepalWidth、PetalLength和PetalWidth。*show_ground_truth.py*输出了数据在不同坐标系下的聚类分布（图1）。



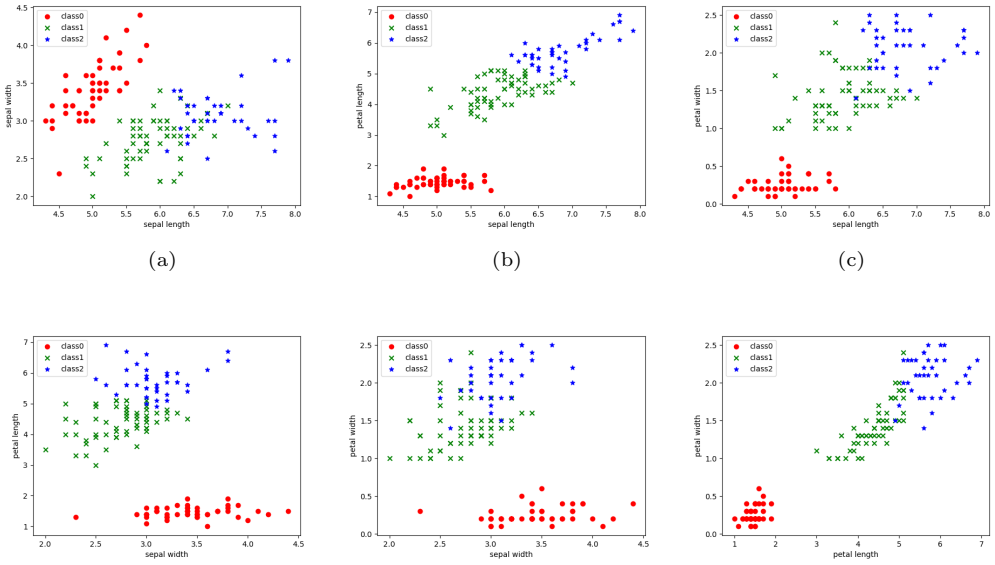
(a) (b) (c)

(d) (e) (f)

图 1: 真实类别分布

2)kMeans聚类效果

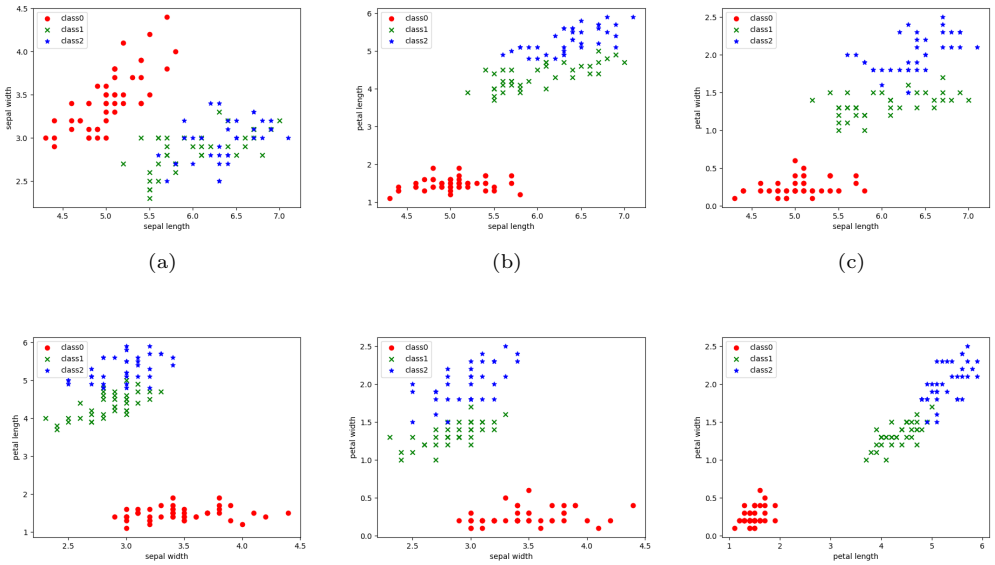
在给定k=3的条件后，使用scikit-learn的kMeans算法对数据集聚类，得到同质性=0.751，完整性=0.765，V-measure=0.758的指标。运行*kMeans.py*，可得到聚类的结果，如图2所示。



(a) (b) (c)

(d) (e) (f)

图 2: kMeans聚类结果



(a) (b) (c)

(d) (e) (f)

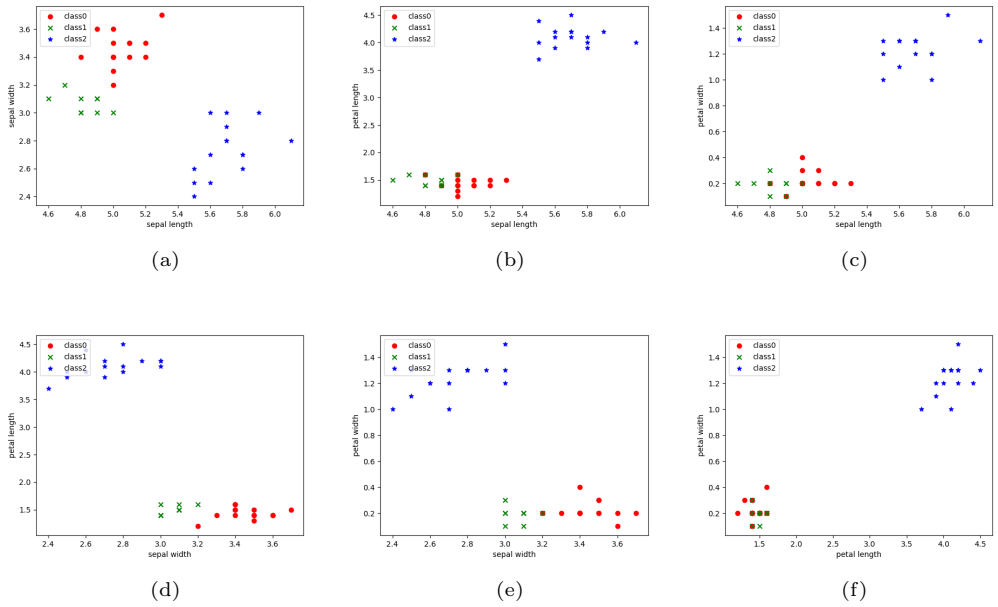
图 3: DBScan聚类结果

3)DBScan聚类效果

DBScan对参数非常敏感。比如，设置较大的eps值的时候，算法会将两个较近的聚类合成一类。而设置较小的eps或较大的$min_samples$的时候，算法会将许多样本当成噪点。在$DBScan.py$使用scikit-learn的DBScan算法并将参数调整合适后，可以运行得到如图3的聚类。这个结果的同质性=0.785，完整性=0.630，V-measure=0.699。从图像和完整性指标都能够看到，DBScan相比kMeans丢掉了许多样本。

4)OPTICS聚类效果

在鸢尾花数据集上，OPTICS的聚类效果明显很差。其指标同质性=0.510，完整性=0.341，V-measure=0.409.显然它删去了许多聚类边界的样本且误差很大。运行$OPTICS.py$可得图4的结果。



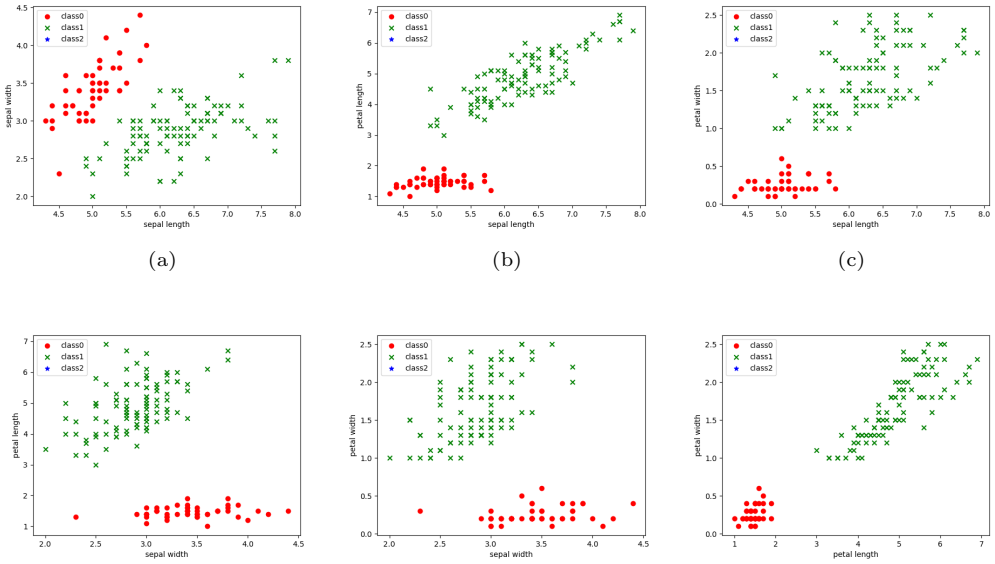
(a) (b) (c)

(d) (e) (f)

图 4: OPTICS聚类结果

表 1: 三种方法评价指标比较

method	homogeneity	completeness	V-measure
kMeans	0.751	0.765	0.758
DBScan	0.785	0.630	0.699
OPTICS	0.510	0.341	0.409



(a) (b) (c)

(d) (e) (f)

图 5: OPTICS聚类为2簇的结果

5. 结果分析

表1显示了三中算法的评价结果。从同质性和完整性的角度来看，在这样一个朴素的数据集上，kMeans的聚类效果不错。虽然设置合理的k值是一个麻烦，但是一旦有了好的k值，就相当于有了很有用的信息。kMeans在合理的k值下不会像两种基于密度的算法那样容易将相近的聚类合并在一起，这就很大程度上保证了同质性。

DBScan算法没有合理k值的指导，容易将相近的类别合并成一个聚类。而且聚类结果对参数非常敏感，想要避免错误的合并，同时又想要尽可能少地将样本判断为噪点，是很难甚至是不可能的。所以在鸢尾花数据集上，DBScan的效果比较平庸：同质性稍稍好于kMeans而完整性因为（可能是不合理的）删除噪点而劣于kMeans。

OPTICS算法本身有很大局限，可能将相当多的数据标记为噪音。在小规模的数据集上可以认为OPTICS算法是错误算法。

如果允许错误地将两个聚类合并为一个，那么OPTICS的结果会比较好（图5），这时的完整性可达1.0。然而这正显示了OPTICS在此数据集下正确区别两个簇的能力比较差。

6. 代码

在按照ReadMe.md在环境中安装所需python软件包后，可以通过运行$kMeans.py$、$DBScan.py$和$OPTICS.py$得到前文的聚类结果图像和评价指标。