李蓉,周维柏.基于集合论与图论的餐饮推荐聚类算法[J].湖南科技大学学报(自然科学版),2021,36(2):93-100. doi:10.13582/j.cnki.1672-9102.2021.02.014

LI R, ZHOU W B. A Clustering Algorithm Based on Set Theory and Graph Theory for Dining Recommendation [J]. Journal of Hunan University of Science and Technology (Natural Science Edition), 2021,36(2): 93-100. doi:10.13582/j.cnki.1672-9102.2021.02.014

基于集合论与图论的餐饮推荐聚类算法

李蓉*,周维柏

(广州商学院 信息技术与工程学院,广东 广州 511363)

摘　要:为了提高餐饮推荐系统的准确率,提出一种基于集合论和图论的餐饮高维非数值型数据聚类算法.首先将菜品数据进行预处理,找出需要的特征,删除不必要的特征,再将这些特征以集合的形式输入到系统中,使用改进的杰卡德相似系数对集合进行相似度计算,得到以集合表示方式的菜品数据间的相似度,接着将所有的数据转换为无向图,最后利用图形聚类算法进行聚类分析.实验结果表明:所提出算法的聚类过程不受噪声影响,具有很好的实用价值.

关键词:集合论;图论;聚类分析;图形聚类算法

中图分类号:TP311　　**文献标志码**:A　　**文章编号**:1672-9102(2021)02-0093-08

A Clustering Algorithm Based on Set Theory and Graph Theory for Dining Recommendation

LI Rong, ZHOU Weibai

(School of Information Technology & Engineering, Guangzhou College of Commerce, Guangzhou 511363, China)

Abstract: To improve the accuracy of dining recommendation system, a new clustering algorithm based on set theory and graph theory was proposed. First, the vegetable data was pre-processed to find out the features required and removed unwanted features. Then these features were input into the system as a set. Improved Jaccard similarity coefficient was used to calculate the similarity between two data items, to obtain the similarity between the dish data in the form of set representation, and all data items were transferred into an undirected graph. Finally, a graph-clustering algorithm was applied to conducting cluster analysis. The experimental results show that the algorithm has good practical value and clustering processes are not affected by noise.

Keywords: set theory; graph theory; cluster analysis; graph-clustering algorithm

餐饮推荐系统需将菜单上的所有菜品进行分类,传统数据分类方法将菜品的每一个关键字作为一个特征,需要使用数量相当庞大的特征来表示,很容易造成维度灾难,而且这种自然语言无法进行数值化处理,也难以实现高维度非数值化数据的聚类[1].传统数据分类方法使用决策树对菜品进行分类,决策树上每一个节点为各种菜的样式,以数据模式匹配来呈现数据,采用监督式分类方法,需事先对数据进行训练[2-3],若测试数据出现一条训练时未输入过的数据时,会导致分类错误甚至无法分类,另外菜品样式必

收稿日期:2019-12-12

基金项目:广东省普通高校特色创新项目资助(2018KTSCX264);广东省科技计划项目资助(2014B010102007)

* **通信作者**,E-mail:10139066@qq.com

须依靠人工处理,难免会出现人为因素产生的错误.

本文提出一种基于集合论与图论的高维度非数值型数据的聚类方法.先将数据进行预处理,根据菜品之间的关系建立关联图,找出最适合的数据特征并删除不必要的特征,再利用集合论的原理计算数据间的相似度,以图的形式呈现,最后利用图形聚类算法进行聚类分析.实验结果表明:算法的聚类过程不受噪声影响,具有很好的实用价值.

1 基于集合论与图论的高维度非数值数据的聚类算法

目前,聚类算法必须先将数据的特征数值化才能进行距离计算[3],但餐饮推荐系统中很多自然语言无法将其特征数值化,如菜品样式可能有中式、西式和日式等,如使用决策树将这些数据分类,在出现一款新菜品时,将导致分类错误甚至无法分类.

为解决上述问题,提出一种基于集合论和图论的高维度非数值型数据聚类算法,首先将菜品数据进行预处理,依照分析目的找出需要的特征并删除不必要的特征,再将这些特征以集合的形式输入到系统中,系统中的每一笔数据都为一个集合,接着将所有的数据以图形中的顶点表示,若2个顶点的相似度大于预设值,则2个顶点有一个无向边相邻,该图 $\boldsymbol{G}$ 为所有数据的关联图,最后利用图形聚类算法将图形进行切割,切割后每一个连通分量的所有顶点就是本文方法的聚类结果中的一个簇.系统流程如图1所示.

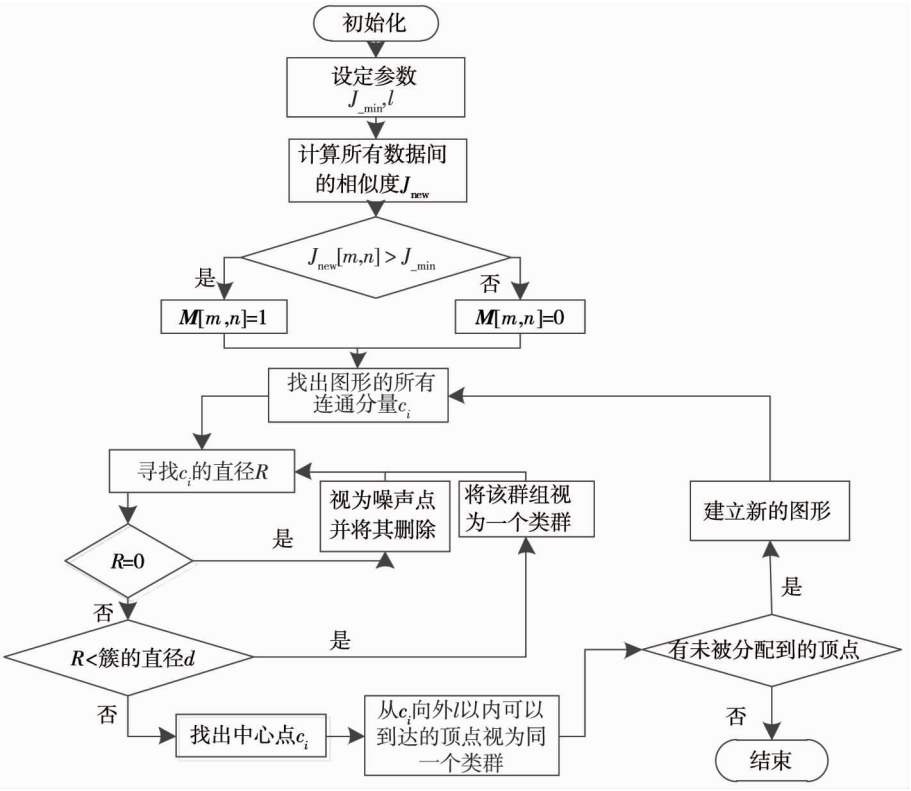

图1 本文算法系统流程

1.1 数据预处理

数据预处理主要为收集数据、数据整合、数据清理、数据转换和分析数据.若考虑有 n 笔数据集合 $\boldsymbol{S}=\{\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_n\}$,其中每一笔数据 $\boldsymbol{S}_i$ 是来自各种不同的菜品,这容易产生数据格式不同或重复的问题,因此,将数据进行整合,找出每一笔数据 $\boldsymbol{S}_i$ 的特征 F_{ij},其中 F_{ij} 为 $\boldsymbol{S}_i$ 的第 j 个特征,同时把不必要的特征删除.最后将特征以集合形式表示,如 $\boldsymbol{S}_i$ 的特征数量有 m 个,则可表示为 $\boldsymbol{S}_i=\{F_{i_1},F_{i_2},\cdots,F_{i_m}\}$,由于菜品的不同,因此,每一笔数据的特征数量不尽相同,但其特征都为同性质的特征.

1.2 建立图形

由于菜品较多,处理后的特征的数量相当庞大,使用传统的相似度计算方法效果差,如新增一数据

$\boldsymbol{S}_{new}=\{F_{new_1},F_{new_2}\}$，$\boldsymbol{S}_{new}$ 所含有的特征 $\{F_{new_1},F_{new_2}\}$ 尚未记录在数据库中，导致无法正确计算，因此，使用图论的图形表示，以解决此类数据的计算问题.

在数据预处理后就可以建立图形，图形中每一个顶点表示一笔数据，假如 2 个顶点间有一条边相邻，则表示 2 个顶点间的相似度大于系统的预设值，当一个顶点无任何边相邻时称为噪声点.因此，在建立图形前必须先计算数据间的相似度，以解决特征数量庞大及特征难以数值化的问题[4-5].

两集合相似度的计算方法有欧几里得距离、Ochiai 系数和杰卡德相似系数等[5].本文使用杰卡德相似系数，杰卡德相似系数是指 2 个集合 $\boldsymbol{S}_i$ 和 $\boldsymbol{S}_j$ 的交集元素在 $\boldsymbol{S}_i$ 和 $\boldsymbol{S}_j$ 的并集中所占的比例[5-6]，用符号 $J(\boldsymbol{S}_i,\boldsymbol{S}_j)$ 表示，如式(1)所示.

$$J(\boldsymbol{S}_i,\boldsymbol{S}_j)=\frac{|\boldsymbol{S}_i\cap\boldsymbol{S}_j|}{|\boldsymbol{S}_i\cup\boldsymbol{S}_j|}. \tag{1}$$

式中：$J(\boldsymbol{S}_i,\boldsymbol{S}_j)$ 范围为 0~1 之间，越接近 1 表示 $\boldsymbol{S}_i$ 集合内的元素和 $\boldsymbol{S}_j$ 集合内的元素的相似度越高[7].

当 $\boldsymbol{S}_i$ 是 $\boldsymbol{S}_j$ 的子集的情况下，使用杰卡德相似系数计算得到相似度的值比较低，对其进行改进，方法如式(2)所示.

$$J_{new}(\boldsymbol{S}_i,\boldsymbol{S}_j)=0.5\frac{|\boldsymbol{S}_i\cap\boldsymbol{S}_j|}{|\boldsymbol{S}_i|}+0.5\frac{|\boldsymbol{S}_i\cap\boldsymbol{S}_j|}{|\boldsymbol{S}_j|}. \tag{2}$$

改进的杰卡德相似系数在 $\boldsymbol{S}_i$ 是 $\boldsymbol{S}_j$ 子集的情况下，计算得到的相似度值高.

有了所有顶点间的相似度，接着还需在两点间建立一条边的最低标准值，以 $J_{\_min}$ 表示.设 2 个顶点 $\boldsymbol{S}_i$ 与 $\boldsymbol{S}_j$ 之间有一条边 e_{s_i,s_j} 相邻，则表示 $J_{new}(\boldsymbol{S}_i,\boldsymbol{S}_j)\geqslant J_{\_min}$，其中 $J_{\_min}$ 值必须介于 0~1，$J_{\_min}$ 值过高，表示建立一条边的门槛值很低，造成聚类结果正确率低，但 $J_{\_min}$ 值过低，图形的边数会很少，导致噪声点过多[8].

所建立的边为无向边，且所有边的权重值都相等.图形建立完成后，该图形即为所要分析的数据间的关联图.

1.3　图形聚类算法

关联图分析的算法很多，本文采用广度优先搜索算法寻找图的连通分量，使用 Dijkstra 算法计算起点的最短路径和 Floyd-Warshall 算法计算全局最短路径，最后利用图形聚类算法对图形进行聚类分析.

1.3.1　图形数据结构

图形数据结构分为邻接矩阵和邻接链表 2 种[9].邻接链表用链表的形式来表示图，图的每个顶点给定一个专属编号，将所有邻接于该顶点的顶点连成一个单链表，记录与该顶点相邻的顶点，当顶点数较多，边少时，矩阵会很稀疏，造成浪费空间[10]，另外对图的操作复杂度也会增加.邻接矩阵则是利用矩阵的方式记录所有顶点间相邻的边，先把图中每一个顶点给定一个专属编号，然后建立一个方形矩阵记录连接信息，方形矩阵中的每一元素皆表示 2 个顶点间的连接信息[10].图形邻接矩阵表示法如图 2 所示.如一个邻接矩阵有 5 个顶点，如图 2a 所示，那么可建立 5×5 的方形矩阵 $\boldsymbol{M}$，在矩阵 $\boldsymbol{M}$ 中记录相邻两顶点的对应位置，如图 2b 所示.计算完相似度后直接以相邻矩阵表示.由于本文所绘制的图为所有边的权重都相同的无向图，故使用相邻矩阵.以 0，1 表示有无相邻的边，且在无向图的相邻矩阵中 $\boldsymbol{M}(1,0)=\boldsymbol{M}(0,1)$，对角线上全部都为 0，故将重复值删除，转换为三角矩阵，以节省内存空间，如图 2c 所示.

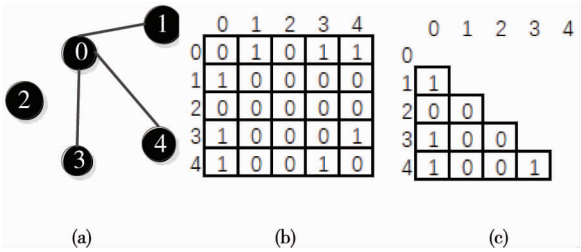

图 2　图形邻接矩阵表示法

1.3.2　图的遍历

图的遍历是从图的某一顶点出发，对图中的所有顶点访问一次，且仅访问一次.常用的方法有深度优先遍历和广度优先遍历：深度优先遍历是沿着树的深度遍历图的顶点，尽可能地往深处搜寻图，当顶点 a 所在的位置已经是最深处时，回到前一个顶点再一次搜寻其他点，这个过程重复进行，直至图中所有和 a 有路径相通的顶点都被访问到[9-11]；广度优先遍历刚好相反，广度优先遍历是从其中一个顶点开始，接着

访问此顶点的所有尚未访问过且相邻的顶点,由访问过的顶点继续进行先广度优先遍历后深度优先遍历的搜寻,直至图中所有与顶点 a 有路径相通的顶点都被访问到[12-13].

使用广度优先遍历来进行图的遍历,广度优先遍历的具体步骤如下:

步骤 1:把起点存入队列中.

步骤 2:如队列不为空,则重复下列步骤,即从队列中取出队列头的点,找出此点的各邻接点,如没遍历,则进行标记,全部存入队列中.

步骤 3:返回步骤 2,直到队列空为止.

1.3.3 起点的最短路径计算

起点的最短路径计算使用 Dijkstra 算法,Dijkstra 算法的流程图如图 3 所示.设 $\boldsymbol{V}$ 为图 $\boldsymbol{G}$ 所有的顶点集合,以 $\boldsymbol{V}[\boldsymbol{G}]$ 表示,Dijkstra 算法从原点 s 到点 u 的最短路径 $\boldsymbol{D}_s[u]$ 来计算下一个从原点 s 到点 v 的最短路径 $\boldsymbol{D}_s[v]$.初始化时,将原点到原点的距离 $\boldsymbol{D}_s[s]$ 设为 0,同时把所有其他(s 不能直接到达的)顶点的路径长度设为无穷大,$\boldsymbol{p}[v]$ 为一个保存已经找到了最短路径的顶点的集合,当算法结束时 $\boldsymbol{D}_s[v]$ 中存储的值将是原点 s 到点 v 的最短路径,若 $\boldsymbol{D}_s[v]$ 的值无穷大时,则表示原点 s 到点 v 中并不存在一条路径.

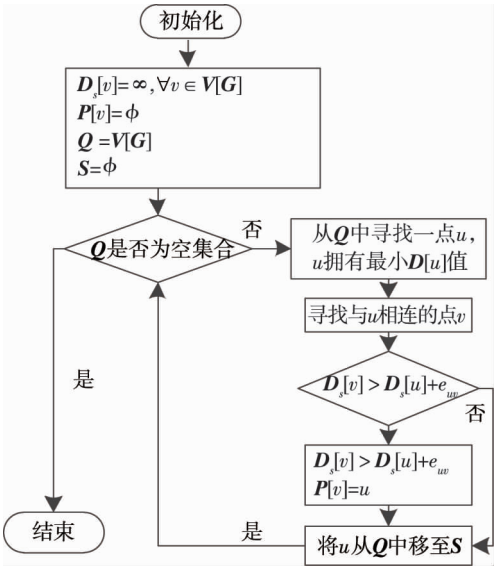

图 3 Dijkstra 算法流程

Dijkstra 算法是利用图形的边向外拓展为基础进行计算,如有一条从点 u 到点 v 的路径,长度为 e_{uv},那么可以通过此路径延伸到起始点来找到一条点 s 到点 v 的路径,其路径长度 $\boldsymbol{D}_s[v]=\boldsymbol{D}_s[u]+e_{uv}$,若 $\boldsymbol{D}_s[v]$ 的值比目前已知的还小,则此新的 $\boldsymbol{D}_s[v]$ 值将取代原来的值,存入 $\boldsymbol{V}[\boldsymbol{G}]$ 中,同时保留点 v 在此最短路径上的 $\boldsymbol{p}[v]$.

$\boldsymbol{S}$ 和 $\boldsymbol{Q}$ 为此算法的 2 个顶点集合,$\boldsymbol{S}$ 内存放所有已知最小 $\boldsymbol{D}_s[v]$ 值的顶点 v, $\boldsymbol{Q}$ 一开始保留所有图形的顶点,而后每一次都从 $\boldsymbol{Q}$ 中选一个点 u,此点 u 拥有最小 $\boldsymbol{D}_s[u]$ 的值,在对点 u 每条外接的边(u,v)向外拓展后,将点 u 从集合 $\boldsymbol{Q}$ 移动到集合 $\boldsymbol{S}$.

1.3.4 全局最短路径的计算

全局最短路径的计算采用 Floyd-Warshall 算法.Floyd-Warshall 算法采用动态规划方案来解决在一个图上每对顶点间的最短路径问题[13-14].令 $D_{i,j}^k$ 表示从点 i 到点 j 考虑经过点 k 的最短路径的长度,且满足所有中间点皆属于集合$\{1,2,\cdots,k\}$的一条最短路径的权值.如发生:

1)若最短路径经过点 k,则 $D_{i,j}^k = D_{i,k}^{k-1} + D_{k,j}^{k-1}$;

2)若最短路径不经过点 k,则 $D_{i,j}^k = D_{i,j}^{k-1}$.

即 $D_{i,j}^k = \min(D_{i,j}^{k-1}, D_{i,k}^{k-1} + D_{k,j}^{k-1})$, 其算法的复杂度为 $O(N^3)$.

定义图 $\boldsymbol{G}=(\boldsymbol{A},\boldsymbol{E})$,$\boldsymbol{A}$ 表示顶点(节点),$\boldsymbol{E}$ 表示边集,$|\boldsymbol{V}|=n$,并定义存在一个 $N\times N$ 的矩阵,其边的权重 $d_{i,j}$ 如式(3)所示.

$$d_{i,j}=\begin{cases}0, \text{ if } i=j;\\ d(i,j), \text{ if } i\neq j \text{ and } (i,j)\in\boldsymbol{E};\\ \infty, \text{ if } i\neq j \text{ and } (i,j)\notin\boldsymbol{E},\end{cases} \tag{3}$$

当 $d_{i,j}=0$ 时,表示没有经过任何的顶点,当 $d_{i,j}=d(i,j)$ 时,表示点 i 到点 j 之间是有直接相连,当 $d_{i,j}=\infty$ 时,表示点 i 到点 j 之间没有任何连接.

算法的具体步骤如下:

步骤 1:初始化,定义 $D[i,i]=0$,如果最短路径从点 i 到点 j 穿过中间节点 k,集合 $\boldsymbol{p}[i,j]=k$,相反,如果最短路径从点 i 到点 j 不穿过中间节点 k,集合 $\boldsymbol{p}[i,j]=\text{nil}$(nil 表示无法到达).

步骤 2:根据式(4)计算 D^k 的元素 $D^k_{i,j}$.

$$D^k_{i,j}=\min(D^{k-1}_{i,j},D^{k-1}_{i,k}+D^{k-1}_{k,j}). \tag{4}$$

步骤 3:当 $k=n$ 时,停止运算,否则令 $k=k+1$,返回到步骤 2.

步骤 4:重复以上步骤,直到寻找到终点为止,并结束.

1.3.5　图形聚类算法

图形聚类算法是一种基于图论的图形聚类分析方法[10],其原理为设法找到群的中心点,接着将中心点向外寻找路径为 l 以内的顶点,并将其归为同一个簇[13-14].

图形聚类算法首先利用广度优先搜寻方法找出图形的连通分量,如果图形中任意 2 个顶点之间都拥有一条路径,则称为连通图,反之称为非连通图,非连通图是由 2 个或 2 个以上连通图所组成,而这些连通图则为非连通图的连通分量,一张图 $\boldsymbol{G}$ 为 m 个连通图分量 c_i 所组成的,如式(5)所示.

$$\boldsymbol{G}=\{c_1,c_2,\cdots,c_m\}. \tag{5}$$

接着以每一个连通分量为单位分别计算,利用 Floyd-Warshall 算法在每一个连通分量的全局最短路径中,都找出一条最长的路径,称为直径[12-14].若连通分量的直径大于簇的直径 d,则在直径中每 d 个顶点归为一簇,其中 $d=2l$ 表示一个簇的直径,且位于中间的顶点则为该簇的中心点.然后利用 Dijkstra 算法将中心点向外搜寻 l 个顶点,这样便能顺利地将均匀图切割出一个长图,也可将长图聚类成多个簇.若连通分量的直径小于等于 r,则直接将该图形所有的顶点归为一个簇,若连通分量的直径等于 0,则表示该顶点并未与其他顶点相邻,即噪声点,将其删除.最后将噪声点除外所有尚未被分类的顶点及边所组成的图形重新进行图形聚类算法,直到所有顶点及边都被分类完为止.

2　实验结果与分析

2.1　实验环境

实验环境:Windows 10 X64、Intel(R) Core(TM) i7-9700F CPU @ 3.0 GHz、8 G RAM,开发语言环境 Python 3.7 (64-bit).

2.2　实验数据

实验数据是使用爬虫从美食杰网站(https://www.meishij.net/)爬取到 100 道菜品及其相对应的食材信息.每一道菜品的数据包括主料和辅料,本文只是对食材相近的食物进行聚类分析,所需特征仅仅是食材,因此,把每一道菜的调味料进行删除,并把食材相同但名称不同的数据进行整合(如牛腱、牛腩等都整合成牛),以降低数据维度,最后将预处理后的数据以 CSV 格式存储.数据具体形式如图 4 所示.

葱爆羊肉	羊里脊肉	葱	胡椒粉	孜然粉	淀粉	糖
红豆米糕	糯米粉	红糖	粘米粉			
酱香烤鱼	丁桂鱼	娃娃菜	金针菇	土豆	莲藕	莴笋
木须鸡肉	鸡蛋	黄瓜	木耳	鸡		
肉泥豆腐蒸蛋	猪	鸡蛋	豆腐	淀粉	葱	
四川烧白	猪	梅干菜	花椒粉			
酸菜烩牛杂	牛	酸菜	干辣椒	青花椒	葱	
番茄炖牛肉	牛	西红柿	洋葱	草果	八角	桂皮

图 4　本文数据格式

2.3　评价标准

聚类的有效性分析有 2 种评价标准:基于簇差异度和分散度的指标和基于人工判定的指标[14].由于本实验数据的特殊性,不适宜使用基于簇差异度和分散度的指标,因此,使用基于人工判定的指标.但人工判断不但耗时,而且因人的主观意识所导致的分类结果不尽相同,以美食杰网站的分类结果作为正确答案.假设本方法分类结果为 $\boldsymbol{C}$,美食杰网站的分类结果为 $\boldsymbol{W}$, 2 个数据 X_i 和 X_j 具有如下 4 种关系:

1)SS:X_i 和 X_j 在 $\boldsymbol{C}$ 和 $\boldsymbol{W}$ 中都属于同一个簇.

2)SD:X_i 和 X_j 在 $\boldsymbol{C}$ 中都属于同一个簇,在 $\boldsymbol{W}$ 就不是.

3)DS:X_i 和 X_j 在 $\boldsymbol{W}$ 中都属于同一个簇,在 $\boldsymbol{C}$ 就不是.

4) DD:X_i 和 X_j 在 $\boldsymbol{C}$ 和 $\boldsymbol{W}$ 中都不属于同一个簇.

若 n 为数据总数,a, b, c, d 分别代表 SS, SD, DS, DD 的数量,芮氏指标公式如式(6)所示.

$$P = \frac{a + d}{a + b + c + d}. \tag{6}$$

式中:P 为正确率; $a + b + c + d = C_2^n$.

聚类效果评估的芮氏指标取值范围均为[0,1],取值越靠近 1 说明聚类效果越好,反之,取值越靠近 0 说明聚类效果越差.

2.4 实验结果与分析

本文采用改进的杰卡德相似系数进行相似度计算,需调整 l 和 $J_{\_min}$ 以找到最佳聚类效果.l 从 1 开始,每次加 1,到结果不再改变为止,而 $J_{\_min}$ 数值为 0~1,从 0 开始每隔 0.1 计算一次,分别记录分类结果的饱和值、正确率、噪声点数量和类群数量,结果如图 5 所示.

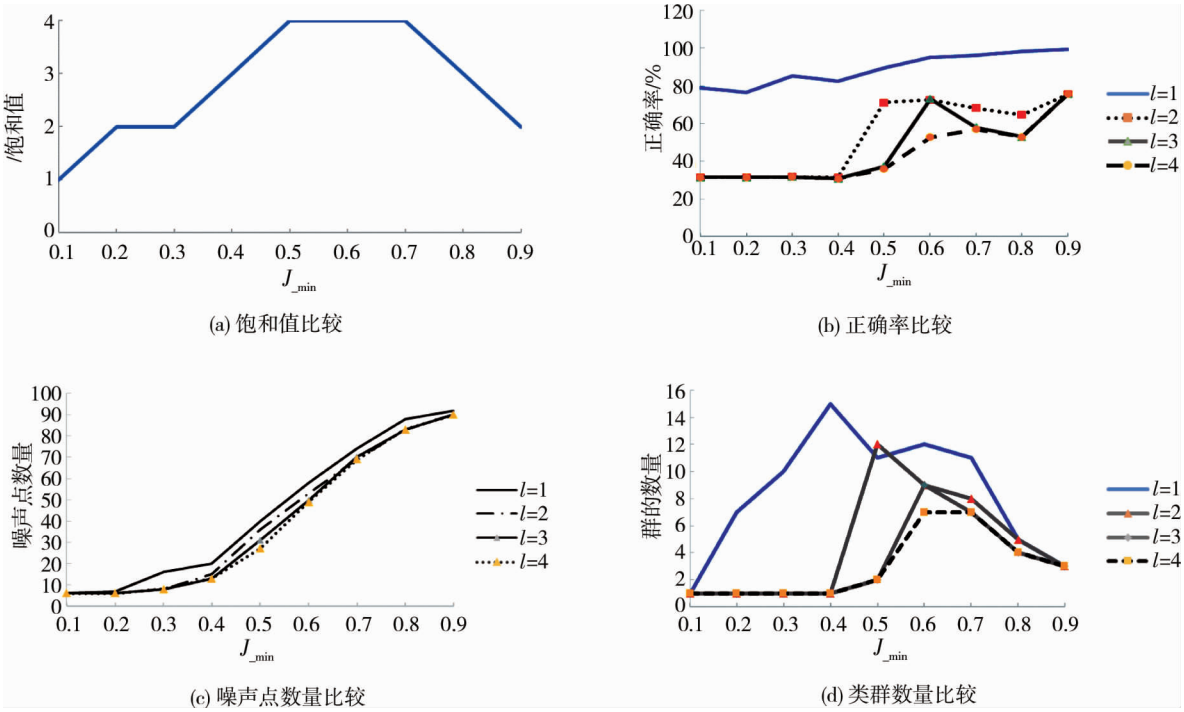

图 5 使用改进的杰卡德相似系数时,不同的 $J_{\_min}$ 与 l 条件下的效果比较

由图 5a 可以看出:l 的饱和值最高出现在中间,这是由于 $J_{\_min}$ 越低,表示建立一条边的门槛值极低,所有的顶点几乎都有一条边相邻,导致图形直径极小,故 l 饱和值也低,但若 $J_{\_min}$ 值越高,就会导致边太少,形成较多直径小的连通分量,l 饱和值也低.

由图 5b 和图 5c 可以看出:$J_{\_min}$ 值越高,最后输出结果的正确率也越高,但噪声点数量越多,有效被聚类的数据就相对变少.另外,l 值越高,向外延伸的顶点数越多,最后输出的结果正确率越低,但噪声点数量相对降低.

由图 5d 可以看出:$J_{\_min}$ 值越低,群的数量越少,这是由于 $J_{\_min}$ 值越低,导致边的数量越多,造成图形直径越小,因而造成簇的数量越少.但 $J_{\_min}$ 值越高,噪声点太多,簇的数量也少,同时 l 值越大,从中心点向外搜寻的点就越多,簇的数量自然减少.

为了验证本文算法的优越性,将本文算法与杰卡德相似系数、Ochiia 系数方法进行对比实验,不同算法性能的比较结果如图 6 所示.

由图 6 可以看出:与杰卡德相似系数和 Ochiia 系数方法相比,本文算法的正确率高,且噪声点数量大幅降低,有较好的使用价值,当 $J_{\_min}$ 超过 0.6 之后,Ochiia 系数方法的噪声数量接近 100,表示图中无任何边存在,无法计算正确率,使用性相对低.

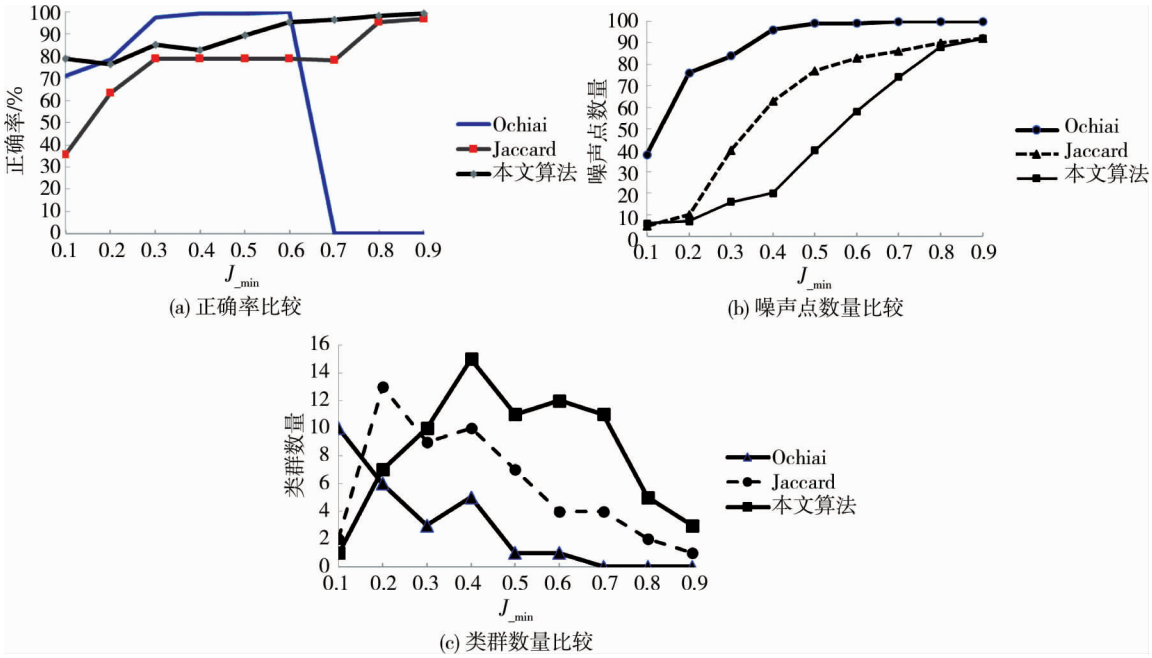

图 6　不同算法性能比较

将本文算法与传统数值量化后的聚类算法进行比较实验,除本文算法外,其他算法均使用杰卡德相似系数作为相似度,数据集的 90%作为训练集,耗时和 MAE 值均取 5 次实验结果的平均值,其中 KNN 算法的近邻大小为 5,SVD 使用 EM 算法,EM 算法的学习速度设为 0.005,过渡拟合值为 0.02,随机噪声值为 0.005,迭代次数为 20.实验结果如表 1 所示,表 1 中 ItemCF 为基于商品的协同过滤推荐算法,UserCF 为基于用户的协同过滤推荐算法,Slope One 算法为基于不同物品之间的评分差来预测用户对物品评分的个性化算法.

表 1　算法性能比较

算法	MAE	平均耗时/ms
ItemCF	0.863	11
UserCF	1.037	10
Slope One	0.723	30
KNN(k=5)	0.834	98
SVD	0.718	4
本文算法	0.652	10

由表 1 可以看出:本文算法、SVD 算法和 Slope One 算法的结果最为精确,且本文算法要优于 SVD 算法和 Slope One 算法,UserCF 算法的推荐结果最差.这主要是因为本文算法不必预先设定群的数量,能依照参数值自动分配,且能够非常有效地识别噪声点,聚类过程不会受到噪声的影响,所以推荐结果的准确度高.在速度方面,SVD 算法的速度最快,处理的平均时间为 4 ms,本文算法的平均时间在 10 ms 左右,ItemCF 算法和 UserCF 算法的平均处理时间都为 10 ms 左右,KNN 算法的速度最慢,平均处理时间为 98 ms.综合准确度和算法运行时间可以看出本文算法具有很好的实用性.

3　结论

1)本文算法正确率高且噪声点数量大幅降低,有较好的使用价值.

2)本文算法不必预先设定群的数量,依照参数值自动分配,能非常有效地识别噪声点,聚类过程不会受噪声点的影响.

3)本算法运算速度快,平均绝对误差 MAE 值小.

4)如今美食网站众多,每一个网站的用语不同,因此收集到的数据会更加复杂.未来希望能加入自然

语言分析,将相同物体、不同名称的数据自动识别出来,减少人工错误,有效地提升相似度的计算,更加准确地聚类分析结果.

参考文献:

[1] Dai X D, Gao L Q, Dong C R. Self-adaptive fuzzification in fuzzy decision tree induction[C]//2010 International Conference on Machine Learning and Cybernetics, July 11-14,2010, Qingdao,China, 2010:296-301.

[2] Pawel B, Eulalia S, Janusz K. Intuitionistic Fuzzy Decision Trees - A New Approach[C]//International Conference on Artificial Intelligence and Soft Computing. Poland: Zakopane, 2014:181-192.

[3] Piotr D, Eduard D A. A New Approach for Using the Fuzzy Decision Trees for the Detection of the Significant Operating Points in the Nonlinear Modeling[C]//International Conference on Artificial Intelligence and Soft Computing. Poland: Zakopane, 2016:279-292.

[4] David C, Christine L, Elod E Z, et al. Getting Clusters from Structure Data and Attribute Data[C]//Proceeding of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. Istanbul: Turkey, 2012.

[5] 吕晓波,马燕,张相芬,等.一种基于图金字塔的聚类算法[J].计算机应用与软件,2018,35(2):256-315.

[6] Comaniciu D. Image segmentation using clustering with saddle point detection[C]// Proceedings of International Conference on Image Processing. 2002:297-300.

[7] 陈梅.面向复杂数据的聚类算法研究[D].兰州:兰州大学,2016.

[8] Li X, Zhong X J. A Graph Clustering Algorithm for the Homology Detection[J]. Applied Mechanics and Materials, 2011(3): 1981-1986.

[9] 夏梦.应用于地理信息数据自动分类的高性能聚类算法[J].计算机应用与软件,2018,35(4):65-68.

[10] Hao Y, Austin R B, Jure L, et al. Local Higher-Order Graph Clustering[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2017:555-564.

[11]林俊山,温晓芳,陈梅.一种基于核心顶点的无参图聚类算法[J].计算机应用研究,2018,35(12):3598-3602.

[12] Luo W J, Zhang D F, Jiang H, et al. Local Community Detection with the Dynamic Membership Function[J]. IEEE Transactions on Fuzzy Systems,2018,26(5):3136-3150.

[13] Héctor M, David C. A Genetic Graph-Based Clustering Algorithm[C]//Proceedings of the 13th International Conference. Brazil: Natal,2012:216-225.

[14] Dubey A S. Comparison of Graph Clustering Algorithms[J]. International Journal of Computer Trends and Technology, 2013, 4(9):3230-3235.