**医保欺诈行为的主动发现**

**——基于引进指标权重的聚类分析算法**

**摘 要**

医疗保险欺诈行为的主动发现是维护医保基金安全的基础课题之一。本文将针对现有一些识别方法存在的问题进行改进，应用“基于引进指标权重的聚类分析算法”进行求解。

现有针对医保欺诈识别的模型，大多采用统计回归、神经网络等辅助学习（有监督）的方法。这就需要人工筛选出一定量的欺诈数据，因此存在较强的主观性、工作量较大等问题。

经典的非辅助学习（无监督）方法，如聚类分析将不同指标的重要程度视为等同的，忽略指标间权重的差异，会影响分类精度。

为解决上述问题，在没有先验知识的情况下，比较精确的识别出欺诈数据，本文引入基于指标权重的欧氏距离，用以刻画数据之间的相似程度，通过聚类算法将数据按照相似程度不同分成多类，分类中出现的孤立点即为疑似欺诈点。为了避免各个指标的权重给定中存在的主观性，本文引入只基于数据而非先验知识的权重指标评价函数CFuzziness(ω)，该函数刻画随着权重ω的改变，分类的模糊程度的变化。运用粒子群优化（MPSO）算法，极小化属性权重评价函数，得到每个指标的权重赋值。将得到的权重应用于聚类算法，即为本文所研究的基于引进指标权重的聚类分析算法。

通过以上求解思路进行求解，第一步可以得到执行科室、患者ID、单张账单总价、开嘱医生ID、患者拿药次数所对应的权重为：

w=[0.1576 0.4387 0.8407 0.1622 0.4314]

第二步可以按照账单号合并多条拿药记录，并以账单号为索引，通过聚类分析算法给出疑似欺诈账单记录213个。这样，“基于引进指标权重的聚类分析算法”就通过数据本身的差异，而非任何先验知识主动发现医保欺诈记录。

**关键词：指标权重 聚类分析 欺诈识别 粒子群优化算法**

1. 问题重述

随着社会经济发展，人民生活水平提高，越来越多的人在享受着医疗保险的福利。但于此同时，也存在参保人利用保险体系的漏洞进行医保欺诈行为，给医疗保险基金带来了巨大的损失。单张处方药费特别高，一张卡在一定时间内反复多次拿药等均为可能的医保欺诈情况。如何通过参保人的医保消费记录，主动发现此类欺诈行为，避免医疗保险基金的损失，成为迫在眉睫的问题。本文将就医保欺诈行为的主动发现这一课题进行研究。

1. 问题分析
2. **问题的难点**

1.没有已知的欺诈样本点，若采用人工选取样本点的方法会存在较强的主观性。

2.通常的聚类分析算法未考虑指标对于距离函数贡献程度的权重问题。

3.人工给出有效欺诈识别指标以及各指标相应权重也可能存在较强的主观性。

4.数据量较大，选择算法时需要考虑到算法的时间复杂度与空间复杂度。

1. **问题的总括**

本文研究的重要问题是在没有先验条件，即没有已知欺诈数据的情况下寻找出可能的欺诈记录。如果通过人工筛选给出部分欺诈数据作为学习材料，存在较强的主观性。为解决这样的问题，本文将采用非辅助学习（无监督学习）方法。通过比较分析，本文将采用聚类分析算法，对数据进行分类处理，认为出现的孤立点为疑似欺诈点。作为一个实际的聚类问题，各种指标对于距离函数贡献程度是不同的，但是传统的聚类分析算法中，无论是采用欧氏距离还是马氏距离，都将所有指标赋予相同的权重。本文考虑若能够给重要指标赋予较大权重，则可以很大程度提高聚类效果。本文通过定义的属性权重评价函数在无先验条件的情况下给出各参量的权重，再用基于属性权重的欧氏距离描述数据的相似程度。

1. 模型假设与符号说明
2. **条件假设**
3. 假设每个患者是否欺诈是相互独立的，不存在相互学习与模仿。
4. 假设医保欺诈只体现在医药费上，与住院费、治疗费无关（因为数据中只体现了中药、西药和中草药，未体现其他费用）。
5. 假设姓名、身份证号、出生日期、性别、电话号码等显著无效识别指标不会对是否欺诈造成影响，可以忽略。
6. 假设欺诈记录只出现在孤立点中（因为欺诈情况的单次金额及拿药次数会与正常情况偏离较大）。
7. **符号说明**

|  |  |
| --- | --- |
| 符号 | 符号解释 |
| X ={ X1，X2，…， Xn } | 待聚类的数据集X |
| Xij | 数据 Xi的第 j 维属性 |
| ω=（ω1，ω2，···，ωm） | 与输入属性相对应的权重矢量 |
| ωk | 第 k 维属性的权重 |
| dpq | 数据xp与数据xq的欧氏距离 |
| d | 数据xp与数据xq基于属性权重的欧氏距离 |
| Spq | 数据xp和xq在数据集X上的相似程度 |
| S | 数据xp和xq在数据集X上基于权重的相似程度 |

1. 模型建立

为了从大量的医保交易记录中准确的检测出可能的欺诈数据，本文采用基于改进的聚类分析算法建立模型，对标准化的数据进行聚类，由于欺诈数据（单张处方钱数较高、一定时间内开药次数较多）都会与正常数据之间有较大差异，因此本文以聚类过程中出现的孤立点为疑似欺诈点。而传统聚类方法忽略了各因素的权重差异，本文引入考虑各项指标权重的欧氏距离[1]作为衡量数据间差异的指标。为了得出一组只依赖基础交易数据而非专家经验的权重值，本文引入权重指标评价函数CFuzziness(ω)[2]，该函数刻画随着权重ω的改变，分类模糊程度的变化。通过调整ω的值，使分类的模糊程度尽量小，ω的最优解应使同一类的点尽量靠近，不同类的点尽量远离，即ω应该对应于CFuzziness(ω)取最小值时的最优解。要想求出ω的值，考虑到本题涉及到的数据量较大，常用的梯度下降等算法时间复杂度较大，且容易陷入局部最优解，因此本文采用粒子群优化算法[3]求解ω的最优解。基于以上分析，本节将按照以下四步建立模型并检验：

1. 筛选与处理数据，剔除显著无关的指标与无效交易记录；
2. 构造属性权重评价函数CFuzziness(ω)，用以描述聚类效果的不确定性，通过粒子群优化算法求得使属性权重评价函数取得最小值的最优解ω作为各项属性的权重；
3. 构造改进的欧氏距离函数d，用改进的欧式距离作为聚类函数的输入；
4. 应用聚类分析算法，无监督给出样本数据分类，以区分正常医保消费与欺诈记录；

将探索优化模型的思路用框图表示如下：

1. **数据筛选与处理**

4.1.1冗余项的忽略

根据生活经验，可以排除掉身高、性别等冗余项的干扰，而对于所在省份等不齐全的信息也不能考虑对于结果的影响。

4.1.2指标的确定

首先，欺诈行为的主体一定是有医保的患者。所以要根据表格2.1中的数据，筛选掉表格2.2中无医保的患者的取药记录。

其次，常见的两种欺诈类型表现为单张账单价格过高以及拿药次数过于频繁。因此单张账单的总价格和单个患者拿药次数应该分别作为判断是否为欺诈的第一项和第二项指标。

再次，根据生活经验，考虑部分科室、或者个别医生可能存在不合理地开医嘱作为患者取药凭证的现象，所以把执行科室以及开嘱医生ID作为影响判断是否欺诈的第三项和第四项指标。

最后，考虑到有一张欺诈类型的产生是刷他人的医保卡，可能存在部分人经常将自己的医保卡外借的情况，所以将患者ID作为影响判断是否为欺诈的结果的第五项指标。

4.1.3对原始数据的处理

从表格2.1中，导入患者ID和医保手册号——根据是否有医保手册判断患者是否有医保，根据判断结果忽略表格2.2中无医保患者的购药记录。

为了得到单张账单的总价以及患者的购药次数，从表格2.2中导入账单号、价格、患者ID——将账单号相同的记录合并，进而计算出单张单据的总价；根据患者ID来统计每位患者的购药次数。

考虑到影响判断是否欺诈的其他因素，还应该将表格2.2中的执行科室和开嘱医生ID导入Matlab中。

4.1.4数据处理后的形式

处理后的数据应该包括六项参数——单张账单总价、患者拿药次数、患者ID、执行科室、开嘱医生ID、账单号。其中前五项参数为影响判断结果是否是欺诈的判断指标，第六项数据账单号为索引值，唯一标识该条记录。用于根据判断的结果检索该条记录的其他信息。

1. **引进改进的欧氏距离**

设X ={ X1，X2，…， Xn } 为待聚类的医保消费记录数据集，每个数据Xi(1≤i≤n)由 m个指标组成，即维数为m，Xi=（Xi1,Xi2,···,Xim）,其中Xij是Xi的第j维属性。

为了描述样本点数据Xp和Xq在数据集X上的相似程度，通常使用的欧氏距离定义为

dpq= (4.1)

其中将所有指标赋予了相同的权重，未考虑不同指标对于在医保欺诈识别中贡献程度的差异，以此种欧氏距离为分类依据的聚类分析的结果可能与实际情况产生较大差异。

为改善聚类效果，本文引入基于属性权重的欧氏距离d，称之为改进的欧氏距离[1]，其定义为

d= (4.2)

ω=（ω1，ω2，···，ωm）是与输入属性相对应的权重矢量,其中ωk∈[0,1]为第k维属性在聚类中的重要性，ωk越大，第k维属性在相似性度量中的作用也越大。

1. **引进属性权重评价函数并求最优解**
2. 属性评价函数

为了确定4.2中ωk的值，本小节中将引入属性评价函数CFuzziness(w)[2]，通过求该函数在一定条件下的最优解，给出一组不依赖已有经验，只依据所给数据的权重值。

首先，先引入度量Xp和Xq相似程度的函数Spq，在普通的欧氏距离dpq下，相似性关系定义[2]为

Spq= (4.3)

采用改进的欧氏距离d作为度量指标后，在相似性关系不变 （即如果Spq>0.5,则S>0.5;如果Spq<0.5,则S<0.5）的前提下，相似性度量定义也相应变化，表示为

S= (4.4)

其中β∈[0，1]为一常数，通过调整β的值使得S近似按照正态分布的形式，均匀地分布[0,1]内。β的近似值可由下式给出[2]：

=0.5 (4.5)

显然，相似性函数S具有以下性质：

1. 与距离成反比，即两点间距离越大，两点相似程度越低，S→0；反之，两点之间的相似程度越大，S →1；
2. 当数据间的相似度在0.5附近时,模糊性较大;当数据间的相似度远离0.5,即S→0或1时模糊性较小。

为让聚类结果具有模糊性相对较小的性质，本文希望通过调整属性权重,使相似数据间的距离减小,不相似数据间的距离增大，即找到一个属性评价函数综合评价各个点之间的相似程度，使总体达到模糊性最小。优化相似矩阵，在相似性关系不变（即如果Spq>0.5,则S>0.5;如果Spq<0.5,则S<0.5）的前提下,使得相似度较大的数据xp与xq (Spq>0.5)，相似度变更大(S→1)；相似度较小的数据xp与xq (Spq<0.5),相似度变更小(S→0)。由于S→0或1 ，使得相似性矩阵的模糊性较小，易于决策，改进了聚类性能，提高聚类效果。

基于以上分析，引进属性权重评价函数，其定义[2]为：

CFuzziness(ω)=(S×logSpq+(1- S)×log (1-Spq))

该函数具有性质[2]

1. lim[S→0,S<0.5]or[S→1, S>0.5]CFuzziness(ω)=Min(CFuzziness(ω))
2. CFuzziness(ω)的值域为[0,1],当CFuzziness(ω)取最小值0时，模糊性最小。

故当CFuzziness(ω)取极小值时，满足相似的对象更相似，不相似的对象更不相似的条件。

1. 求最优解

为了最小化CFuzziness(ω)，采用粒子群优化算法[4]。

假设在一个维的目标搜索空间中，有个粒子组成一个群落，其中第个粒子表示为一个维的向量

， (4.6)

第个粒子的“飞行 ”速度也是一个维的向量，记为

， (4.7)

第**个粒子迄今为止搜索到的最优位置称为个体极值，记为

， (4.8)

整个粒子群落迄今为止搜索到的最优位置为全局极值，记为

 (4.9)

粒子在飞翔时不断追踪这两个值，并根据如下的公式(4.10)和(4.11)来更新自己的速度和位置[4]：

 (4.10)

 (4.11)

式(4.10)右边由三部分组成，第一部分反映了粒子的运动“习惯(habit)”，以参量表示，体现粒子维持自己先前速度的趋势；第二部分为“认知(cognition)”部分，代表粒子有向自身历史最佳位置逼近的趋势；第三部分为“社会(social)”部分，代表粒子有向群体或邻域历史最佳位置逼近的趋势，根据经验，通常。。是粒子的速度，，是常数，由用户设定用来限制粒子的速度。和是介于之间的随机数。

在本题中，有5个指标需要考虑，即D=5，种群数量N初始化为30，=1，

粒子群优化算法的流程如下：

① 初始化粒子群，包括群体规模，每个粒子的位置和速度

② 计算每个粒子的适应度值；

③ 对每个粒子，用它的适应度值和个体极值比较，如果 ，则用替换掉；

④ 对每个粒子，用它的适应度值和全局极值比较，如果则用替；

⑤ 根据公式(4.10) ,(4.11)更新粒子的速度和位置 ；

⑥ 如果满足结束条件(误差足够好或到达最大循环次数)退出，否则返回②。

通过这一方法可以较高效准确的给出权重ω的值。

1. **基于引进权重参量的聚类分析算法**

聚类分析是数据挖掘的重要方法之一，主要研究基于距离的聚类方法。通过合理的聚类，方便在没有先验知识的情况下认识事物内部的规律。其一般方法是将数据对象分成多个簇，在同一个簇中对象的相似度较高，不同簇中的对象相似度较低。

1. Matlab聚类分析的原理[5]

设待聚类样本空间X ={ x1，x2，…，xn},每个数据xi(1≤i≤n)由 m个指标组成,即用m维向量（xi1,xi2,···,xim）表示,X的样本聚类即是X的一个划分,满足

=X AiAj=Ø

将待处理数据看视作一个n维的数据空间,每个数据对象即为数据空间内的一个点,聚类分析的任务即为在此n维数据空间中基于距离（本文中基于改进的欧氏距离）寻找尽可能密集的点所构成的区域。

聚类分析方法很多, Matlab统计工具箱中的聚类分析应用的是系统聚类法。这种聚类法是聚类分析中应用最为广泛的一种方法,其基本原理是:

1. 将样本空间中的每个数据看成一类,这时有n类;
2. 计算各类之间的改进的欧氏距离,将距离最近的两类并成一类;
3. 计算新类与其余类的改进的欧氏距离,再将距离最近的两类合并;
4. 重复上述步骤,直至将所有的数据合并的类数等于决策者事先定出聚类的类别数。
5. 常用聚类距离及算法[5]

类与类之间的距离是一种衡量亲疏程度的指标，本文的距离计算方法采用前文所述改进的欧氏距离

d=

采用的聚类算法为平均连接算法：取类s和类t点之间的平均距离为类间距离。

d(s,t)=d

根据算法原理将类间距离最近的两类并为一类，直至类数等于预先给定的聚类类别数，聚类算法结束，得到聚类列及谱系图为问题最终解。

1. 模型求解
2. **数据导入**

bills表格为去除冗余影响因素后的购药记录（取自表格2.2费用明细表），其保留项为是否为欺诈记录的影响因素以及用于标识记录的索引数据。表格共包括5列数据，前四列为欺诈记录的影响因素——第一列执行科室、第二列病人ID、第三列价格、第四列开嘱医生ID，第五列为用于标识记录的索引数据（最后判断出欺诈记录后通过此项数据查找具体购药记录）——第五列账单号。

insurance表格去除冗余影响因素后的病人资料（取自表格2.1病人资料），表格仅包含两项，第一项为患者ID，第二项为医保手册号。医保手册号大部分为9位数字，还有一部分显示为1，表示该名患者没有医保手册，即这部分病人没有医保，所以这些患者一定不涉及医保欺诈，他们在bills中的购药记录可以删除不考虑。

将两个表格中的数据以矩阵形式导入。表格名即为对应的矩阵变量名。

1. **用于处理数据的脚本、函数**

5.2.1 DealPaientID（脚本）

该脚本的作用是删除insurance中没有医保（体现在表格insurance的第二列数据为1）的患者ID，运行过函数后，insurance中仅保存有医保的患者ID。（详细代码见附件DealPaientID.m）

5.2.2 SiftBillsNote（脚本）

该脚本的作用是删除bills中没有医保的患者的购药记录。通过查询bills表格中的患者ID在insurance中是否存在，删除在insurance中没有记录的患者的购药记录。经过此次调用，bills中仅保留有医保的患者的购药记录。（详细代码见附件SiftBillsNote.m）

5.2.3 CalcuCheck（脚本）

该脚本的作用是将bills中账单号相同（表示是同一次取药）的拿药记录的总价相加，合并成一条记录。合并后的新记录对应的第三列即为该张账单的总钱数。（详细代码见附件CalcuCheck.m）

5.2.4 CalcuFrequency（脚本）

该脚本的作用是统计每个人拿药的总次数，并在bills中增加第六列，表示该患者总共拿药的次数。（详细代码见附件CalcuFrequency.m）

5.2.5 standard（函数）

该函数的作用是将矩阵中数据标准化。

函数输入量为一个n行m列矩阵，输出为输入矩阵标准化后的矩阵。用线性标准化的方法，使矩阵中的每一个数据都处于0~1的范围内。（详细代码见附件standard.m）

5.2.6 InitSwarm（函数）

该函数的作用是初始化粒子群。

函数有4个输入参数和2个输出参数。输入参数包括种群大小（即同时有多少个个体一起向使适应度函数值最小的输入量进化）、粒子的维数（即为影响因素的个数）、粒子在各维中运动的范围（即各自变量的取值范围）、适应度函数（即为需要求最值的函数），输出参数包括初始化的粒子群（最初设定的种群中各个个体的值）、粒子当前最优解与全局最优解（目前为止各个位置的粒子所能取到的使适应度函数的值最小的自变量的解。（详细代码见附件InitSwarm.m）

5.2.7 BaseStepPso(函数)

该函数的作用是调整速度和位置，逼近最优解。

函数有8个输入参数和2个输出参数。输入参数包括粒子群矩阵（包含粒子的位置、速度和当前的目标函数值）、最优解粒子矩阵（包含每个粒子的历史最优解以及整个种群的历史最优解）、适应度函数（即为本题中的属性权重评价函数）、粒子在运动中各维的范围（自变量的取值范围）、粒子最大值和最小值（经验值）、总循环次数（迭代次数）以及当前次数。输出参数包括更新后的粒子群矩阵和最优解粒子矩阵。（详细代码见附件BaseStepPso.m）

5.2.8 PsoProcess(函数)

该函数将前两个函数（InitSwarm函数和BaseStepPso函数）组装起来，用于求解使适应度函数值最小对应的解。

该函数包括10个输入参数和1个输出参数。输入参数包括粒子群矩阵、粒子的维数、粒子在各维中运动的范围、粒子群的初始化函数（即为InitSwarm函数）、速度位置更新函数（即为BaseStepPso函数）、适应度函数（即为属性权重评价函数）、是否每次迭代暂停、是否图形化迭代过程、迭代次数。输出参数即为最后得到的使属性权重评价函数的值最小的自变量的值，即为最优解。

函数先调用InitSwarm函数来初始化粒子群，然后根据迭代的次数重复调用BaseStepPso函数，更新速度和位置，记录每一次迭代后的历史最优解。最后得到全局历史最优解。（详细代码见附件PsoProcess.m）

5.2.9 Griewank（函数）

该函数为属性权重评价函数，使该函数取得最小值的解即为最优解。

函数的输入值为一个m维矩阵（m为粒子的维度，即影响因素的个数）。输出值即为以输入值为权重算出的属性权重评价函数的值。（详细代码见附件Griewank.m）

5.2.10 CalcuFunBeta（函数）

该函数在Griewank函数中被调用，用于求解β的值。

在计算特定权重对应的属性权重评价函数的值之前，函数中包含一个未知量β（将待聚类的记录看成m维空间内的点，β与这些距离值应该满足某一特定函数关系）。此函数的输入量为β，输出不同β值对应的特定函数的值，以此来逼近β的实际值。（详细代码见附件CalcuFunBeta.m）

5.2.11 CalcuTrans\_DIS（函数）

该函数用于求加权后各点之间的距离。

该函数的输入量为各点坐标和各维度的权重值，输出量为任意不同两点之间的距离矩阵。（详细代码见附件CalcuTrans\_DIS.m）

5.2.12 CalcuClass（函数）

该函数用于计数聚类后，各类包括的点的个数。

该函数的输入量是一个2列的矩阵，第二列表示该点的所属类。输出量为一个2列的矩阵，第一列为各类的编号第二列为该种编号所包含的记录数。（详细代码见附件CalcuClass.m）

5.2.13 FindFraudNotes（函数）

该函数用于根据聚类结果找出疑似欺诈记录所对应的账单号。

该函数有两个输入量，无输出量，在函数中改变全局变量fraud（存放欺诈点对应的账单号）。输入量包括聚点类矩阵（矩阵有两列，第一列表示聚点的类别，第二列表示该类聚点的个数）和聚点所属类（一维列向量，表征改点所属的类别）。该函数还用到了原始DATA矩阵的备份——ORI\_DATA矩阵（定义为全局变量），通过查找点所属类别是否是聚类后的孤立点来确定该点是否是疑似欺诈点，并将疑似欺诈点的账单号导入到全局变量fraud中。（详细代码见附件FindFraudNotes.m）

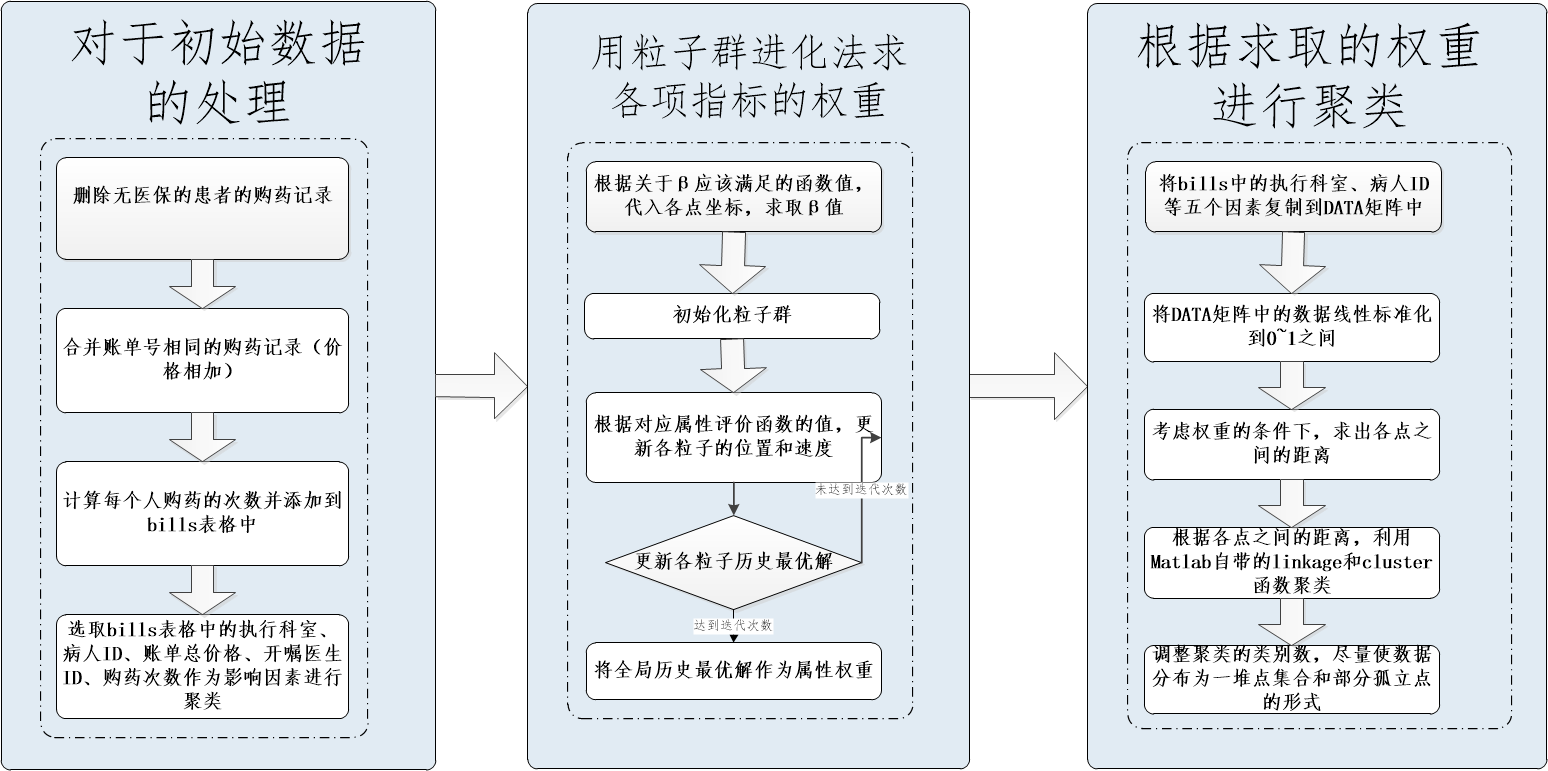
5.2.14 SearchNotes（函数）

该函数用于根据给定的账单号，查询该账单的全部信息。

函数有两个输入量和一个输出量。输入量包括fraud矩阵（只有一列，存放疑似欺诈点的账单号）和bills矩阵（包含所有账单的信息）。输出量为fraud矩阵（将fraud矩阵扩充为六列，显示该账单的全部信息）。

该函数以疑似欺诈点的账单号为索引，查询出该账单的其他信息。最终将所有疑似欺诈记录的信息记录在fraud矩阵中。（详细代码见附件SearchNotes.m）

**5.3程序求解框图**



**5.4 程序实现（命令行窗口内的语句）**

5.4.1 对于初始数据的处理

bills=xlsread(‘bills.xlsx’);

insurance=xlsread(‘insurance.xlsx’);

DealPatientID;

SiftBillsNote;

CalcuCheck;

CalcuFrequency;

%Excel表格bills.xlsx中有五列数据，分别是执行科室、患者ID、账单总价钱、开嘱医生ID、账单号（其中账单号作为索引数据，其他四列数据作为评判指标）。

%Excel表格insurance.xlsx中有两列数据，分别为患者ID和医保手册号，医保手册号为1的患者无医保。调用DealPtientID脚本将医保手册号为1的患者记录从insurance矩阵中删除。

%调用SiftBillsNotes脚本，将bills矩阵中无医保的患者的购药记录删除。

%调用CalcuCheck脚本，将bills中账单号相同的购药记录的总价相加，仅保留一条，其他同账单号的记录删除（最后可以根据账单号作索引可以找出所有的欺诈记录）。经过此项操作，bills中仅剩余85078条购药记录。

%调用CalcuFrequency脚本，计算每个患者的购药次数，在bills表格中增加第6列数据表示该患者总共的购药次数。

5.4.2 根据粒子群算法求取各项指标的权重

应用粒子群优化算法，求解目标函数最优解，得到各项指标的权重如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 指标 | 执行科室 | 患者ID | 单张账单总价 | 开嘱医生ID | 患者拿药次数 |
| 权重 | 0.1576 | 0.4387 | 0.8407 | 0.1622 | 0.4314 |

global DATA

global DIS;

global DDIS;

global ORI\_DATA;

global fraud;

fraud=[];

DATA=bills(1:10000,:);

ORI\_DATA=DATA;

DATA(:,5)=[];

DATA=standard(DATA);

DIS=pdist(DATA);

DDIS=squareform(DIS);

Scope=[0,1;0,1;0,1;0,1;0,1];

w=PsoProcess(30,5,Scope,@InitSwarm,@BaseStepPso,@Griewank,0,0,50);

%声明全局变量DATA（用于表示各项影响指标的值），DIS（以一维数组的形式存放各个点之间的距离），DDIS（以对称矩阵的形式表示各个点之间的距离）,ORI\_DATA（备份DATA中的数据，用于聚类后查询对应的账单号）,fraud(用于存放疑似欺诈点的账单号)。

%最初疑似欺诈点的个数为0，将fraud置零

%将bills中的前10000条数据导入DATA中（数据量过大，因此以10000个点为一组，分别进行聚类操作）。

%

%删除DATA矩阵的第五列数据，第五列代表账单号，不作为聚类的指标，仅作为用于查找购药记录的索引。

%将DATA中的数据线性单位化

%在DIS中以一维数组的形式存放DATA中一万个点各点之间的欧式距离

%在DDIS中以对称矩阵的形式，存放DATA中一万个点各点之间的欧式距离

%Scope表示各权重的取值范围均为0~1

%调用PsoProcess函数，求取各项指标的权重。30表示粒子群中有30个粒子；5表示点的维度为5（即共有5项影响因素）；Scope表示各点的取值范围；输入初始化粒子群、调节速度和位置、适应度函数的句柄；0表示迭代过程中不暂停并且不需要绘制图像；50表示迭代次数为50。输出w即为各部分的权重。执行后

%求得各部分的权重值w=[0.1576 0.4387 0.8407 0.1622 0.4314]。

5.4.3 根据所求的各项指标的权重进行聚类

trans\_DIS=CalcuTrans\_DIS(DATA,w);

z=linkage(trans\_DIS);

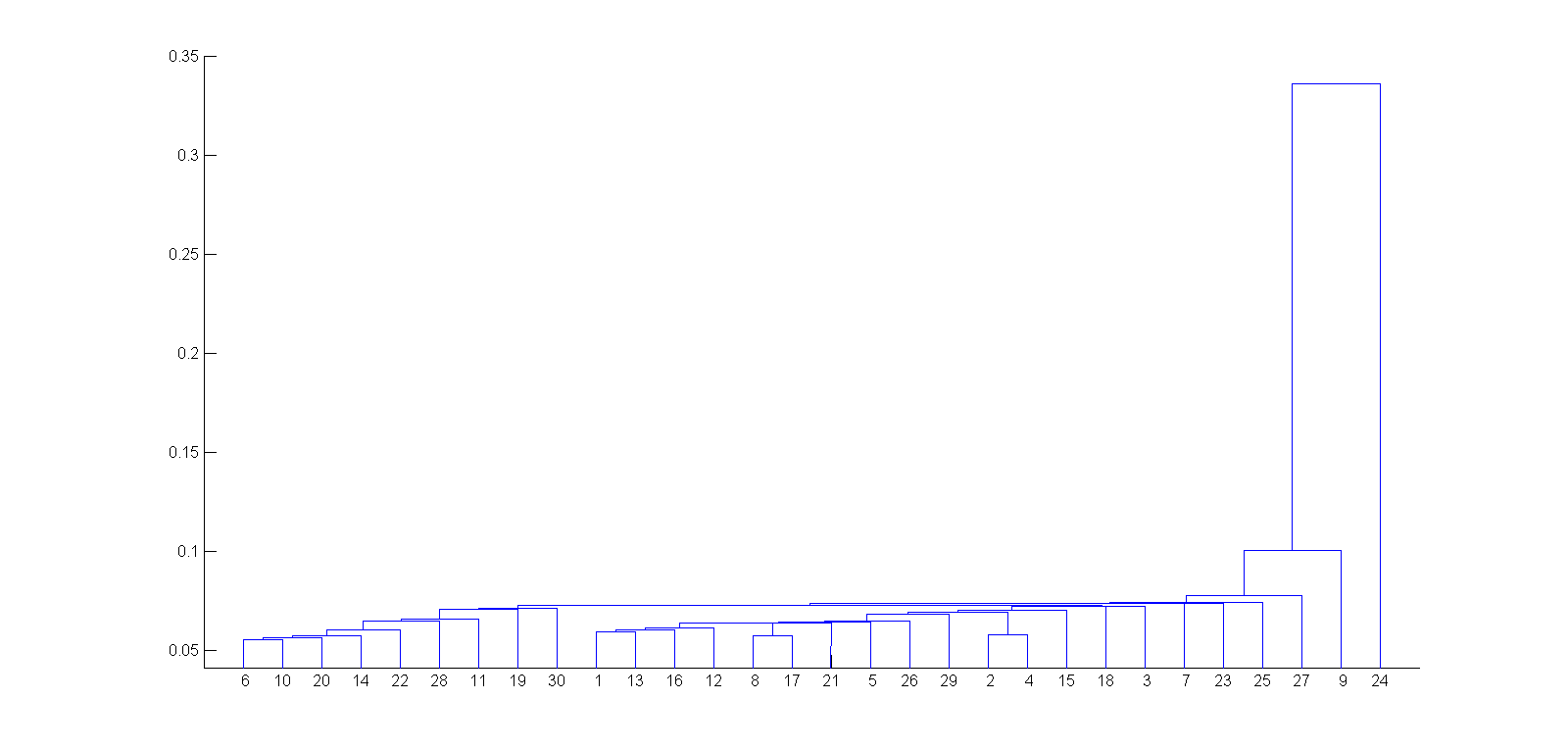
h=dendrogram(z);

t=cluster(z,3);

class=CalcuClass(t);

%根据各项指标的权重w求出前一万个点中各个点之间的距离并存放在一维数组trans\_DIS中。

%绘制输出聚类树形图的冰状图如下（由于绘图限制，只能显示前30个点的绘图情况）



%根据trans\_DIS的值进行聚类，聚成3类，并统计各类的点的个数，此时输出class为

|  |  |
| --- | --- |
| 1 | 2 |
| 2 | 9997 |
| 3 | 1 |

%表示第一类点有2个，第二类点有9997个，第三类点有1个。符合之前的假设（欺诈行为不存在模仿与学习，呈孤立点状分布）。

增加聚类种类，观察到以下结果：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | 2 | | 2 | 9995 | | 3 | 2 | | 4 | 1 |   聚成四类 | |  |  | | --- | --- | | 1 | 1 | | 2 | 9994 | | 3 | 2 | | 4 | 2 | | 5 | 1 |   聚成五类 | | |  |  | | --- | --- | | 1 | 1 | | 2 | 9993 | | 3 | 1 | | 4 | 2 | | 5 | 2 | | 6 | 1 |   聚成六类 |
| |  |  | | --- | --- | | 1 | 1 | | 2 | 9992 | | 3 | 1 | | 4 | 1 | | 5 | 2 | | 6 | 2 | | 7 | 1 |   聚成七类 | | |  |  | | --- | --- | | 1 | 3364 | | 2 | 6628 | | 3 | 1 | | 4 | 1 | | 5 | 1 | | 6 | 2 | | 7 | 2 | | 8 | 1 |   聚成八类 | | |

由以上表格可以看出，当将数据聚成八类时，会出现两大部分的集中点，不再满足欺诈点孤立的条件，因此此时选择将前一万条数据加权聚成七类。其中疑似的欺诈记录有8条。

5.4.4 根据聚类结果查找欺诈记录

FindFraudNotes( class,t );

……

%该函数无输出参数，直接在调用过程中，改变全局变量fraud的值，更新疑似欺诈点的账单号

5.4.5 用上述方法类推，分别对其他的记录进行聚类操作

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 原始数据记录 | 10001：20000 | | 聚类类别数 | 11 | | 疑似欺诈点个数 | 16 | | |  |  | | --- | --- | | 原始数据记录 | 20001：30000 | | 聚类类别数 | 11 | | 疑似欺诈点个数 | 15 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 原始数据记录 | 30001：40000 | | 聚类类别数 | 19 | | 疑似欺诈点个数 | 23 | | |  |  | | --- | --- | | 原始数据记录 | 40001：50000 | | 聚类类别数 | 18 | | 疑似欺诈点个数 | 21 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 原始数据记录 | 50001：60000 | | 聚类类别数 | 25 | | 疑似欺诈点个数 | 42 | | |  |  | | --- | --- | | 原始数据记录 | 60001：70000 | | 聚类类别数 | 19 | | 疑似欺诈点个数 | 23 | |
| |  |  | | --- | --- | | 原始数据记录 | 70001：80000 | | 聚类类别数 | 26 | | 疑似欺诈点个数 | 46 | | |  |  | | --- | --- | | 原始数据记录 | 80001：85078 | | 聚类类别数 | 13 | | 疑似欺诈点个数 | 19 | |

由以上数据可以知道疑似欺诈点一共有213个，即一共有213张账单疑似欺诈。账单号存放在全局变量一维数组fraud中。

5.4.6 检索所有疑似欺诈记录的全部账单信息

fraud=SearchNotes(bills, fraud);

xlswrite(‘FraudNotes.xlsx’,fraud);

%该函数以账单号为索引，检索账单号在bills中的记录，查询到所有疑似欺诈账单的信息。

%将所有疑似欺诈账单的记录写入excel表FraudNotes.xlsx（详见附件）中。

所有疑似欺诈的账单记录如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 账单号 | 病人ID | 执行科室 | 开嘱医生ID | 账单总价格 | 拿药次数 |
| 5041943 | 644604 | 2 | 142 | 1170.6 | 1 |
| 5042603 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5042604 | 163696 | 191 | 1180 | 16 | 25 |
| 5061954 | 649842 | 191 | 1029 | 1373.45 | 1 |
| 5062350 | 650134 | 191 | 1155 | 3645.2 | 1 |
| 5068511 | 97499 | 101 | 109 | 797.5 | 2 |
| 5076265 | 621485 | 191 | 1061 | 1783.06 | 3 |
| 5076301 | 650007 | 193 | 1327 | 1676.63 | 4 |
| 5088155 | 287278 | 2 | 145 | 1267.94 | 3 |
| 5089898 | 265627 | 425 | 1669 | 10 | 10 |
| 5091315 | 265627 | 425 | 1669 | 100.16 | 10 |
| 5094501 | 286123 | 191 | 794 | -1002.15 | 7 |
| 5095238 | 265627 | 425 | 1669 | 5.7 | 10 |
| 5101762 | 656502 | 423 | 2928 | -1315.21 | 4 |
| 5102295 | 265627 | 423 | 2921 | 140.98 | 10 |
| 5108963 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5108964 | 163696 | 191 | 1180 | 146.44 | 25 |
| 5110732 | 643639 | 2 | 150 | 652.33 | 7 |
| 5115475 | 612657 | 2 | 144 | 2354.68 | 1 |
| 5117703 | 537132 | 502 | 1028 | 1320.28 | 8 |
| 5118095 | 163696 | 191 | 1180 | 21.57 | 25 |
| 5118096 | 163696 | 191 | 1180 | 16 | 25 |
| 5120371 | 256400 | 502 | 1028 | 1290.07 | 3 |
| 5122971 | 339226 | 191 | 1060 | 1537.34 | 1 |
| 5130775 | 189268 | 502 | 1028 | 2489.07 | 1 |
| 5131392 | 325939 | 2 | 145 | 1201.94 | 1 |
| 5135118 | 639719 | 191 | 1326 | 2632.8 | 1 |
| 5140603 | 186919 | 191 | 1060 | 1930.25 | 7 |
| 5140624 | 186919 | 191 | 1060 | -1930.25 | 7 |
| 5140625 | 186919 | 191 | 1060 | 1930.25 | 7 |
| 5140636 | 186919 | 191 | 1060 | -1930.25 | 7 |
| 5140637 | 186919 | 191 | 1060 | 1930.25 | 7 |
| 5147758 | 183268 | 191 | 1062 | -860.5 | 5 |
| 5150058 | 659979 | 191 | 3388 | 10.23 | 11 |
| 5150059 | 659979 | 191 | 3388 | 12.98 | 11 |
| 5150614 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5150615 | 163696 | 191 | 1180 | 16 | 25 |
| 5157457 | 555728 | 191 | 926 | -1068.34 | 3 |
| 5158054 | 664092 | 191 | 763 | 1884.3 | 1 |
| 5170340 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5170341 | 163696 | 191 | 1180 | 16 | 25 |
| 5171756 | 336138 | 502 | 1028 | -347.02 | 5 |
| 5174584 | 164465 | 191 | 1060 | 1810.53 | 3 |
| 5175396 | 282260 | 2 | 530 | -432.83 | 3 |
| 5179156 | 398015 | 2 | 150 | 650 | 3 |
| 5180676 | 350618 | 423 | 2922 | 1142.61 | 1 |
| 5181058 | 423611 | 423 | 2921 | 907.84 | 1 |
| 5182620 | 98219 | 101 | 109 | 797.5 | 1 |
| 5182878 | 509502 | 423 | 2922 | 1309.94 | 1 |
| 5183460 | 253065 | 423 | 2945 | -364.94 | 5 |
| 5187399 | 572452 | 2 | 149 | 1163.77 | 1 |
| 5188217 | 667998 | 101 | 116 | -797.5 | 2 |
| 5196354 | 389496 | 423 | 2925 | -417.4 | 4 |
| 5205906 | 589176 | 191 | 1473 | 1884.3 | 1 |
| 5208577 | 159984 | 502 | 1028 | 148.68 | 13 |
| 5209304 | 320349 | 191 | 793 | -994.7 | 7 |
| 5209713 | 477945 | 2 | 110 | 1674.81 | 3 |
| 5210090 | 163696 | 191 | 1180 | 21.57 | 25 |
| 5210091 | 163696 | 191 | 1180 | 16 | 25 |
| 5211334 | 159984 | 502 | 1028 | 268.7 | 13 |
| 5212858 | 170256 | 502 | 1028 | -755.89 | 6 |
| 5213222 | 170256 | 502 | 1028 | -755.89 | 6 |
| 5217735 | 608684 | 2 | 108 | 1927.22 | 2 |
| 5222531 | 195858 | 502 | 1028 | -470.67 | 3 |
| 5222955 | 381773 | 502 | 1028 | 1826.89 | 1 |
| 5227115 | 354169 | 191 | 1060 | 1243.14 | 1 |
| 5230292 | 615989 | 2 | 149 | 1703.36 | 2 |
| 5232498 | 673481 | 423 | 2928 | -1266.52 | 3 |
| 5233390 | 174782 | 502 | 1028 | 1184.89 | 4 |
| 5234334 | 669778 | 423 | 2922 | 1349.29 | 7 |
| 5234347 | 454123 | 191 | 581 | 1250.19 | 1 |
| 5235432 | 669778 | 423 | 2922 | -1349.29 | 7 |
| 5235802 | 93845 | 2 | 1505 | -167.74 | 7 |
| 5235909 | 163696 | 191 | 1065 | 146.44 | 25 |
| 5235910 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5240636 | 500726 | 423 | 2922 | -601.66 | 3 |
| 5246021 | 203105 | 502 | 1028 | -782.18 | 3 |
| 5247235 | 190334 | 191 | 581 | 2219.4 | 3 |
| 5249245 | 405032 | 191 | 769 | 1884.3 | 1 |
| 5250009 | 361555 | 191 | 1062 | -697.29 | 6 |
| 5252243 | 239773 | 423 | 2922 | -367.57 | 6 |
| 5252258 | 239773 | 423 | 2922 | -367.57 | 6 |
| 5253872 | 217527 | 191 | 769 | 1884.3 | 2 |
| 5258446 | 537132 | 502 | 1028 | 1109.64 | 8 |
| 5259048 | 435919 | 502 | 1028 | -207.16 | 3 |
| 5259914 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5261516 | 198369 | 423 | 2922 | -413.68 | 6 |
| 5263541 | 201510 | 502 | 1028 | 1677.13 | 2 |
| 5263689 | 227205 | 502 | 1028 | 1215.7 | 1 |
| 5263808 | 539869 | 191 | 760 | 1884.3 | 1 |
| 5263905 | 677872 | 502 | 1028 | 257.76 | 12 |
| 5263914 | 677872 | 502 | 1028 | 257.76 | 12 |
| 5263928 | 677872 | 502 | 1028 | 257.76 | 12 |
| 5263941 | 677872 | 502 | 1028 | 85.92 | 12 |
| 5264134 | 677872 | 502 | 1028 | -257.76 | 12 |
| 5264137 | 677872 | 502 | 1028 | -257.76 | 12 |
| 5264138 | 677872 | 502 | 1028 | -257.76 | 12 |
| 5264140 | 677872 | 502 | 1028 | -85.92 | 12 |
| 5264245 | 402735 | 191 | 794 | -626.1 | 6 |
| 5264271 | 402735 | 191 | 794 | -626.1 | 6 |
| 5264392 | 402735 | 191 | 794 | -626.1 | 6 |
| 5264477 | 176464 | 502 | 1028 | 1361.56 | 4 |
| 5264527 | 677928 | 191 | 794 | -626.1 | 3 |
| 5264588 | 651604 | 191 | 1154 | -552.1 | 6 |
| 5267933 | 581961 | 2 | 1505 | 1286.49 | 1 |
| 5271510 | 171551 | 191 | 1326 | 1462.3 | 2 |
| 5271838 | 188293 | 191 | 1229 | -773.09 | 4 |
| 5276597 | 363050 | 191 | 1062 | 1304.2 | 4 |
| 5280045 | 579502 | 2 | 120 | 1703.36 | 1 |
| 5280795 | 387776 | 191 | 791 | 1776 | 4 |
| 5284448 | 455063 | 191 | 3279 | -248.89 | 5 |
| 5287078 | 603194 | 191 | 1060 | 1313.26 | 1 |
| 5292316 | 681358 | 2 | 1505 | 1170.73 | 1 |
| 5293640 | 619951 | 191 | 794 | -1037.4 | 6 |
| 5293647 | 191487 | 502 | 1028 | 1518.91 | 3 |
| 5294087 | 168453 | 502 | 1028 | 969.56 | 3 |
| 5294411 | 159988 | 502 | 1028 | 271.75 | 10 |
| 5294574 | 159988 | 502 | 1028 | 11 | 10 |
| 5295057 | 628287 | 2 | 113 | 1302.63 | 2 |
| 5295701 | 363026 | 423 | 2928 | 651.69 | 7 |
| 5296480 | 363026 | 423 | 2928 | -651.69 | 7 |
| 5296514 | 363026 | 423 | 2928 | 445.14 | 7 |
| 5297365 | 163696 | 191 | 1180 | 21.57 | 25 |
| 5297367 | 163696 | 191 | 1180 | 16 | 25 |
| 5300961 | 682730 | 423 | 2928 | 1819.83 | 1 |
| 5307060 | 537132 | 502 | 1028 | 317.33 | 8 |
| 5307492 | 405484 | 191 | 1161 | -646.46 | 7 |
| 5307668 | 537132 | 502 | 1028 | 97.17 | 8 |
| 5308033 | 463011 | 191 | 759 | 1884.3 | 1 |
| 5309723 | 405484 | 191 | 1161 | -646.46 | 7 |
| 5310005 | 159984 | 191 | 1180 | 349.75 | 13 |
| 5312365 | 222036 | 191 | 4134 | 561.86 | 1 |
| 5313259 | 279522 | 502 | 1028 | 1269.36 | 1 |
| 5314730 | 663556 | 191 | 4134 | 699.82 | 2 |
| 5314746 | 620312 | 191 | 4134 | 698.6 | 2 |
| 5320290 | 503627 | 191 | 920 | -733.74 | 4 |
| 5321937 | 190291 | 191 | 1060 | 1561.32 | 3 |
| 5322654 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5322655 | 163696 | 191 | 1180 | 16 | 25 |
| 5326902 | 660150 | 2 | 125 | 1703.36 | 1 |
| 5327215 | 685978 | 191 | 1029 | -740 | 5 |
| 5331872 | 191854 | 191 | 581 | 2234.92 | 6 |
| 5331891 | 191851 | 191 | 581 | 1829.17 | 1 |
| 5336214 | 686815 | 423 | 1554 | 2254.38 | 2 |
| 5337017 | 664984 | 424 | 2921 | -1103.35 | 4 |
| 5337509 | 159984 | 502 | 1028 | 223.02 | 13 |
| 5338441 | 687355 | 191 | 903 | 1650 | 1 |
| 5341771 | 366111 | 191 | 4134 | 531.11 | 1 |
| 5346113 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5346114 | 163696 | 191 | 1065 | 16 | 25 |
| 5349388 | 615716 | 101 | 109 | -860 | 5 |
| 5350169 | 474948 | 2 | 3312 | 394.88 | 4 |
| 5358548 | 560999 | 2 | 1505 | 1214.64 | 1 |
| 5361145 | 535515 | 101 | 112 | -860 | 2 |
| 5362034 | 690183 | 2 | 145 | 1363.53 | 1 |
| 5367209 | 537132 | 502 | 1028 | 1093.57 | 8 |
| 5367370 | 639856 | 101 | 3307 | 797.5 | 1 |
| 5367573 | 537132 | 502 | 1028 | -1093.57 | 8 |
| 5367575 | 537132 | 502 | 1028 | 817.13 | 8 |
| 5367584 | 537132 | 502 | 1028 | 258.8 | 8 |
| 5368638 | 423177 | 502 | 1028 | 1039.78 | 3 |
| 5371202 | 691512 | 502 | 1028 | 874.64 | 4 |
| 5371286 | 691512 | 502 | 1028 | -874.64 | 4 |
| 5372480 | 473418 | 191 | 4039 | 36.13 | 8 |
| 5372484 | 473418 | 191 | 4039 | -36.13 | 8 |
| 5372485 | 473418 | 191 | 4039 | 36.13 | 8 |
| 5372497 | 205486 | 191 | 581 | -1322.08 | 5 |
| 5372698 | 628287 | 2 | 109 | 1860.9 | 2 |
| 5374782 | 397488 | 191 | 759 | 1927.52 | 1 |
| 5375240 | 529410 | 2 | 145 | 1163.77 | 2 |
| 5375985 | 550049 | 502 | 1028 | 311.28 | 6 |
| 5377773 | 691903 | 423 | 2945 | -987.4 | 3 |
| 5379057 | 567473 | 101 | 112 | -797.5 | 2 |
| 5379165 | 233446 | 502 | 1028 | 1179.39 | 11 |
| 5379184 | 233446 | 502 | 1028 | -1179.39 | 11 |
| 5379185 | 233446 | 502 | 1028 | 1179.39 | 11 |
| 5379206 | 233446 | 502 | 1028 | -1179.39 | 11 |
| 5379207 | 233446 | 502 | 1028 | 1179.39 | 11 |
| 5379295 | 233446 | 502 | 1028 | -1179.39 | 11 |
| 5379305 | 233446 | 502 | 1028 | 747.25 | 11 |
| 5379307 | 233446 | 502 | 1028 | 432.14 | 11 |
| 5379539 | 233446 | 502 | 1028 | 18.17 | 11 |
| 5379939 | 233446 | 502 | 1028 | -18.17 | 11 |
| 5379940 | 233446 | 502 | 1028 | 16.63 | 11 |
| 5380253 | 183194 | 191 | 1059 | 1598.9 | 3 |
| 5380379 | 372669 | 193 | 1326 | 1522.42 | 2 |
| 5380481 | 163696 | 191 | 1180 | 21.57 | 25 |
| 5380482 | 163696 | 191 | 1180 | 280.52 | 25 |
| 5382553 | 167175 | 502 | 1028 | -689.6 | 5 |
| 5382753 | 266776 | 502 | 1028 | 17.69 | 9 |
| 5382835 | 266776 | 502 | 1028 | -17.69 | 9 |
| 5382837 | 266776 | 502 | 1028 | 16.93 | 9 |
| 5382911 | 692747 | 2 | 120 | 1170.6 | 1 |
| 5383150 | 661240 | 502 | 1028 | 950.88 | 1 |
| 5391693 | 79762 | 2 | 3409 | 3.09 | 1 |
| 5393531 | 363026 | 423 | 2921 | 286.88 | 7 |
| 5393976 | 142176 | 101 | 109 | 860 | 1 |
| 5395630 | 207554 | 423 | 2925 | 626.1 | 3 |
| 5395666 | 666034 | 423 | 2922 | -649.17 | 3 |
| 5397627 | 79068 | 2 | 3409 | 16.74 | 1 |
| 5398225 | 174479 | 423 | 2921 | 641.74 | 3 |
| 5399943 | 92297 | 2 | 3312 | 15.68 | 1 |
| 5400930 | 214237 | 423 | 2922 | 832.3 | 1 |
| 5401648 | 163696 | 191 | 1180 | 64.52 | 25 |
| 5401649 | 163696 | 191 | 1180 | 16 | 25 |
| 5403031 | 426398 | 423 | 2922 | 812.17 | 1 |
| 5403918 | 171323 | 423 | 2925 | 626.1 | 2 |
| 5405978 | 190176 | 502 | 1028 | 808.95 | 6 |
| 5406800 | 639799 | 2 | 140 | 1703.36 | 1 |
| 5407149 | 695632 | 191 | 1029 | 3700 | 1 |
| 5407195 | 36911 | 2 | 3312 | 43.28 | 2 |
| 5408696 | 92494 | 2 | 126 | 1163.77 | 7 |
| 5408947 | 478284 | 191 | 794 | 1037.4 | 5 |

1. 模型检验

将由模型得出的可能欺诈记录进行人工复查，结果发现所得出的213条数据较好的被归属为两种情况，一是单张处方药费特别高，二是一张医保卡在数据给出的一个月时间内反复多次拿药。与题目给出的可能的医保欺诈情况十分吻合，可以认为模型是正确有效的。

1. 模型评价与推广
2. 模型评价

前人针对保险欺诈的主动识别这一课题，多采用基于统计回归或神经网络思想的优化改进方法。而这两类方法都属于辅助学习（有监督）方法，需要预先给出一定量的已知欺诈数据，或拥有较为丰富的先验知识，以获取识别因子作为学习材料，用以主动识别其他大量数据中包含的可能欺诈数据。该方法的问题在于欺诈样本点的选取过于依赖人的主观性，且多为某一欺诈特征极为明显的数据，对于普遍意义上欺诈数据的识别不具备较强的参考价值。为避免人工筛选欺诈数据带来的主观误差，需要找到非辅助学习的方法。聚类分析法可以满足这一条件，但传统的聚类分析采用欧氏距离作为分类标准，不足之处在于将各种影响因素的重要程度视为相同的，会造成一些不重要的参量（如年龄）却和一些重要的参量（如病人一个月之内的开药次数）同等的影响着最后的分类。为解决该问题，本文将权重引进聚类分析，成为基于权重的聚类分析算法。

概括来说，本文所应用的“基于引进权重参量的聚类分析算法”主要有以下方面的优点：

(1)该模型不需要提供欺诈样本点，避免了主观评价对于分类的影响。

(2)引进权重参量，改进了聚类算法将所有影响因素视为同等重要的弊端，有效提高聚类效果，使得聚类问题有更优化的解决方案。

(3)权重的确定由模型给出，不依赖人为确定。

1. 模型推广

本文建立的医保欺诈识别模型不但对主动识别医疗保险欺诈记录具有较大帮助，且对其他异常检测问题的相关研究也具有一定的参考意义。

本文所应用的“基于指标权重的聚类分析”模型是对经典聚类分析方法的一种改进，有效克服了聚类分析由于结果相对粗糙而主要用于数据挖掘的预处理的问题，提升了聚类精度后，可以大大拓展聚类分析的应用领域。

参考文献

1. Wang Xi-zhao，Wang Ya-dong，Wang Li-juan． Improving fuzzy C-Means clustering based on feature-ωeight learning［J］． Pattern Recognition Letters，2004，10( 25) : 1123-1132
2. 王熙照,王丽娟,王利伟.传递闭包聚类中的模糊性分析.计算机工程与应用, 2003, 39(18): 92～ 94)
3. 武优西,侯丹丹,李建满,米少华.属性权重的聚类算法研究[J].小型微型计算机系统
4. 徐小平. 粒子群算法及其参数设置. 信息与计算科学,[本科生毕业论文].西安理工大学.2010
5. 李华强,费逸伟,姜旭峰,钟新辉. 基于Matlab聚类分析的磨粒分类识别研究[J]. 润滑与密封,2005,03:84-85+113.

附录

1. DealPaientID函数

%处理患者ID和对应的医保手册号，有医保手册的病人才有可能有欺诈行为

i=size(insurance,1);%表示一共有多少条记录

while i

if(insurance(i,2)==1)

insurance(i,:)=[];

end

i=i-1;

end

clear i%删除临时变量

2.SiftBillsNote函数

%bills矩阵删除无医保的病人的取药记录

%第1列是执行科室

%第2列是病人ID

%第3列是总价

%第4列是开嘱医生ID

%第5列是账单号

m=size(bills,1);%bills 中一共有m行

n=size(insurance,1);%insurance中一共有n行

maxID=insurance(n);%insurance中的最大值为MaxID

while m

if(bills(m,2)>maxID)%如果病人ID大于有医保的病人的最大ID，那他一定无医保

bills(m,:)=[];

m=m-1;

continue;

end

i=1;

while bills(m,2)<insurance(i,1);%找到第一个大于等于当前病人ID的号码

i=i+1;

end

if(~insurance(i,2)==bills(m,2))%说明该病人并无医保

bills(m,:)=[];

end

m=m-1;

end

%删除临时变量

clear maxID

clear m

clear n

clear i

3.CalcuCheck函数

%统计出每次拿药（账单号相同）所花的费用（储存在bills中）

%更新后的bills各列的意义如下

%第1列是执行科室

%第2列是病人ID

%第3列是该张账单的总价钱

%第4列是开嘱医生ID

%第5列是账单号

insurance(:,2)=0;%从0开始累加

m=size(bills,1);%表示有多少条记录

while (m-1)

if(bills(m,5)==bills(m-1,5))

bills(m-1,3)=bills(m-1,3)+bills(m,3);%叠加到相同账单中

bills(m,:)=[];%删除同一张账单中的不同药品

end

m=m-1;

end

clear m%清楚临时变量

4.CalcuFrequency函数

%计算每个人购药的次数,更新增加到bills的第6列中

m=size(bills,1);%bills中共有m条账单

%为bills添加一列，用于存放次数

temp=zeros(m,1);

bills=[bills,temp];

insurance(:,2)=[];%清除insurance原来的第二列

%为了查找方便（按坐标查找），直接建立一个更大的数组，病人ID作为脚标

n=max(insurance);

frequency=zeros(n,1);

while m

frequency(bills(m,2))=frequency(bills(m,2))+1;

m=m-1;

end

m=size(bills,1);

while m

bills(m,6)=frequency(bills(m,2));

m=m-1;

end

%清除临时变量

clear m

clear n

clear frequency

clear temp

5.standard函数

function [y]=standard( x )

%用于将x矩阵按列标准化到0~1之间

temp=[max(x);min(x)];

deno=temp(1,:)-temp(2,:);

[m,n]=size(x);

y=zeros(m,n);

while(m)

y(m,:)=(x(m,:)-temp(2,:))./deno;

m=m-1;

end

end

6．InitSwarm函数

function [ ParSwarm,OptSwarm ] = InitSwarm(SwarmSize,ParticleSize,ParticleScope,AdaptFunc)

%粒子群优化算法，期望通过此函数，算出各部分的权重

% 输入参数包括5个参数

%SwarmSize:种群大小的个数

%ParticleSize:一个粒子的维数，在本题中即为所有的影响因素（本题为5个影响因素）

%ParticleScope:一个粒子在运算中各维的范围

%ParticleScope的形式为矩阵形式，矩阵的行数即为账单的条目数，矩阵有两列，分别表示该账单对应的五个参数的最小值和最大值

%AdaptFunc:适应度函数

%输出参数包括2个参数

%ParSwarm初始化的粒子群

%OptSwarm粒子群当前最优解与全局最优解

%初始化粒子群矩阵，全部设置为【0~1】之间的随机数

ParSwarm=rand(SwarmSize,2\*ParticleSize+1);

for k=1:ParticleSize

ParSwarm(:,k)=ParSwarm(:,k)\*(ParticleScope(k,2)-ParticleScope(k,1))+ParticleScope(k,1);

%调节速度，使速度与位置的范围一致

ParSwarm(:,ParticleSize+k)=ParSwarm(:,ParticleSize+k)\*(ParticleScope(k,2)-ParticleScope(k,1))+ParticleScope(k,1);

end

%对每一个粒子计算其适应度函数的值

for k=1:SwarmSize

ParSwarm(k,2\*ParticleSize+1)=AdaptFunc(ParSwarm(k,1:ParticleSize));

end

%初始化粒子群最优解矩阵

OptSwarm=zeros(SwarmSize+1,ParticleSize);

%粒子群最优解矩阵全部设为0

[maxValue,row]=min(ParSwarm(:,2\*ParticleSize+1));

%寻找适应度函数最小的解在矩阵中的位置（行数）

OptSwarm=ParSwarm(1:SwarmSize,1:ParticleSize);

OptSwarm(SwarmSize+1,:)=ParSwarm(row,1:ParticleSize);

%在本题中，不考虑输入输出不合法的情况，保证输入四个参数，输出的两个参数都是合理的

End

7．BaseStepPso函数

function [ ParSwarm,OptSwarm ] = BaseStepPso(ParSwarm,OptSwarm,AdaptFunc,ParticleScope,MaxW,MinW,LoopCount,CurCount)

%功能描述：全局版本——基本粒子群算法的单步更新位置、速度的算法

%输入参数：

%ParSwarm:粒子群矩阵，包含粒子的位置，速度与当前的目标函数值

%OptSwarm：包含粒子群个体最优解与全局最优解的矩阵

%ParticleScope:一个粒子在运算中各维的范围

%AdaptFunc：适应度函数

%LoopCount:迭代的总次数

%CurCount:当前迭代的次数

%输出参数：同输入的同名参数

%线性递减策略

w=MaxW-CurCount\*((MaxW-MinW)/LoopCount);

%得到粒子群群体大小以及一个粒子维数的信息

[ParRow,ParCol]=size(ParSwarm);

%得到粒子的维数

ParCol=(ParCol-1)/2;

SubTract1=OptSwarm(1:ParRow,:)-ParSwarm(:,1:ParCol);

c1=2;

c2=2;

ParticleScope(:,1)=0;

ParticleScope(:,2)=1;

m=size(ParticleScope,1);

for row=1:ParRow

SubTract2=OptSwarm(ParRow+1,:)-ParSwarm(row,1:ParCol);

TempV=w.\*ParSwarm(row,ParCol+1:2\*ParCol)+2\*unifrnd(0,1).\*SubTract1(row,:)+2\*unifrnd(0,1).\*SubTract2;

%限制速度的代码

for h=1:ParCol

if TempV(:,h)>ParticleScope(h,2)

TempV(:,h)=unifrnd(-1,0);

end

if TempV(:,h)<-ParticleScope(h,2)

TempV(:,h)=unifrnd(0,1);

end

end

%更新速度

ParSwarm(row,ParCol+1:2\*ParCol)=TempV;

%确定约束因子

a=0.729;

%限制位置的范围

TempPos=ParSwarm(row,1:ParCol)+a\*TempV;

if TempPos(:,1)>ParticleScope(h,2)

TempPos(:,1)=ParticleScope(h,2);

end

if TempPos(:,h)<=ParticleScope(h,1)

TempPos(:,h)=ParticleScope(h,1)+1e-10;

end

for h=2:ParCol

for i=1:size(TempPos,1)

range=1;

for j=1:h-1

range=range-(TempPos(i,j));

end

if TempPos(i,h)>range

TempPos(i,h)=range;

end

if(TempPos(i,h)<ParticleScope(h,1))

TempPos(i,h)=TempPos(i,h)+1e-10;

end

end

end

ParSwarm(row,1:ParCol)=TempPos;

%计算每个粒子的新的适应度值

ParSwarm(row,2\*ParCol+1)=AdaptFunc(ParSwarm(row,1:ParCol));

if ParSwarm(row,2\*ParCol+1)<AdaptFunc(OptSwarm(row,1:ParCol))

OptSwarm(row,1:ParCol)=ParSwarm(row,1:ParCol);

end

end

%for循环结束

%寻找适应度函数值最小的解在矩阵中的位置(行数)，进行全局最优的改变

[minValue,row]=min(ParSwarm(:,2\*ParCol+1));

if AdaptFunc(ParSwarm(row,1:ParCol))<AdaptFunc(OptSwarm(ParRow+1,:))

OptSwarm(ParRow+1,:)=ParSwarm(row,1:ParCol);

end

8．PsoProcess函数

function [Result]=PsoProcess(SwarmSize,ParticleSize,ParticleScope,InitFunc,StepFindFunc,AdaptFunc,IsStep,IsDraw,LoopCount)

%功能描述：一个循环n次的PSO算法完整过程，返回这次运行的最小与最大的平均适应度以及在线性能与离线性能

%输入参数：

%SwarmSize:种群大小的个数

%ParticleSize：一个粒子的维数

%ParticleScope:一个粒子在运算中各维的范围（矩阵形式，第一列为最小值，第二列为最大值）；

%InitFunc:初始化粒子群函数

%StepFindFunc:单步更新速度，位置函数

%AdaptFunc：适应度函数

%IsStep：是否每次迭代暂停；IsStep＝0，不暂停，否则暂停。缺省不暂停

%IsDraw：是否图形化迭代过程；IsDraw＝0，不图形化迭代过程，否则，图形化表示。缺省不图形化表示

%LoopCount：迭代的次数；缺省迭代100次

%返回值参数

%Result为经过迭代后得到的最优解

%容错控制

if nargin<4

error('输入的参数个数错误。')

end

[row,colum]=size(ParticleSize);

if row>1||colum>1

error('输入的粒子的维数错误，是一个1行1列的数据。');

end

[row,colum]=size(ParticleScope);

if row~=ParticleSize||colum~=2

error('输入的粒子的维数范围错误。');

end

%设置缺省值

if nargin<6

LoopCount=100;

IsStep=0;

IsDraw=0;

end

if nargin<7

IsDraw=0;

LoopCount=100;

end

if nargin<8

LoopCount=100;

end

%控制是否显示2维以下粒子维数的寻找最优的过程

if IsDraw~=0

DrawObjGraphic(ParticleSize,ParticleScope,AdaptFunc);

end

%初始化种群

[ParSwarm,OptSwarm]=InitSwarm(SwarmSize,ParticleSize,ParticleScope,AdaptFunc)

%在测试函数图形上绘制初始化群的位置

if IsDraw~=0

if 1==ParticleSize

for ParSwarmRow=1:SwarmSize

plot([ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,1)],[ParSwarm(ParSwarmRow,3),0],'r\*-','markersize',8);

text(ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,3),num2str(ParSwarmRow));

end

end

if 2==ParticleSize

for ParSwarmRow=1:SwarmSize

stem3(ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,2),ParSwarm(ParSwarmRow,5),'r.','markersize',8);

end

end

end

%暂停让抓图

if IsStep~=0

disp('开始迭代，按任意键：')

pause

end

%开始更新算法的调用

for k=1:LoopCount

%显示迭代的次数：

disp('----------------------------------------------------------')

TempStr=sprintf('第 %g 次迭代',k);

disp(TempStr);

disp('----------------------------------------------------------')

%调用一步迭代的算法

[ParSwarm,OptSwarm]=StepFindFunc(ParSwarm,OptSwarm,AdaptFunc,ParticleScope,0.9,0.4,LoopCount,k)

%在目标函数的图形上绘制2维以下的粒子的新位置

if IsDraw~=0

if 1==ParticleSize

for ParSwarmRow=1:SwarmSize

plot([ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,1)],[ParSwarm(ParSwarmRow,3),0],'r\*-','markersize',8);

text(ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,3),num2str(ParSwarmRow));

end

end

if 2==ParticleSize

for ParSwarmRow=1:SwarmSize

stem3(ParSwarm(ParSwarmRow,1),ParSwarm(ParSwarmRow,2),ParSwarm(ParSwarmRow,5),'r.','markersize',8);

end

end

end

XResult=OptSwarm(SwarmSize+1,1:ParticleSize);

YResult=AdaptFunc(XResult);

if IsStep~=0

XResult=OptSwarm(SwarmSize+1,1:ParticleSize);

YResult=AdaptFunc(XResult);

str=sprintf('%g步迭代的最优目标函数值%g',k,YResult);

disp(str);

disp('下次迭代，按任意键继续');

pause

end

%记录每一步的平均适应度

MeanAdapt(1,k)=mean(ParSwarm(:,2\*ParticleSize+1));

end

%for循环结束标志

%记录最小与最大的平均适应度

MinMaxMeanAdapt=[min(MeanAdapt),max(MeanAdapt)];

%计算离线与在线性能

for k=1:LoopCount

OnLine(1,k)=sum(MeanAdapt(1,1:k))/k;

OffLine(1,k)=max(MeanAdapt(1,1:k));

end

for k=1:LoopCount

OffLine(1,k)=sum(OffLine(1,1:k))/k;

end

%记录本次迭代得到的最优结果

XResult=OptSwarm(SwarmSize+1,1:ParticleSize);

YResult=AdaptFunc(XResult);

Result=[XResult,YResult];

End

9．Griewank函数

function [ y ] = Griewank( x )

%输入参数为一个1\*m的矩阵，分别代表w1,w2,…，wm

%输出参数为适应度函数的值

global DIS %初始的距离值

global DATA %最原始数据

m=size(DIS,2);%表示距离的个数

[col,row]=size(DATA);%分别表示原始数据的行数和列数

%求解β

if(CalcuFunBeta(1)>=0)

beta=1;

else

l=0;

r=1;

i=10;

%循环10次，利用二分法求解β值，与实际β值的差不超过0.001

while i

temp=(l+r)/2;

f1=CalcuFunBeta(l);

fx=CalcuFunBeta(temp);

if(f1\*fx<=0)

r=temp;

else

l=temp;

end

i=i-1;

end

beta=(l+r)/2;%最后一次逼近β的值

%经过以上代码，求得β值

end

%以下代码为求加了权重之后的距离值

global trans\_DIS

trans\_DIS=zeros(1,m);

i=1;

s=1;

while i<col

j=i+1;

while j<=col

k=1;

while k<=row

trans\_DIS(s)=trans\_DIS(s)+(x(k))^2\*(DATA(i,k)-DATA(j,k))^2;

k=k+1;

end

trans\_DIS(s)=sqrt(trans\_DIS(s));

s=s+1;

j=j+1;

end;

i=i+1;

end

y=0;

ro=1./(1+beta\*trans\_DIS);

i=1;

while i<=m;

y=y+ro(i)\*log(DIS(i))+(1-ro(i))\*log(1-DIS(i));

i=i+1;

end

y=(-2)/(col\*(col-1))\*y;

end

10．CalcuFunBeta函数

function [ y ] = CalcuFunBeta( x )

%把给定的x值作为β，求对应的适应度函数的值

global DIS

global DDIS

m=size(DIS,2);%表示DIS的列数

n=size(DDIS,1);%表示点的个数

y=0;

while(m)

y=y+(1/(1+x\*DIS(m)));

m=m-1;

end

y=2/(n\*(n-1))\*y;

y=y-0.5;

end

11．CalcuTrans\_DIS函数

function [ trans\_DIS ] = CalcuTrans\_DIS( DATA,x )

%用来根据给定的w值算各个点之间的距离

[col,row]=size(DATA);

m=col\*(col-1)/2;

trans\_DIS=zeros(1,m);

s=1;

for i=1:(col-1)

for j=(i+1):col

for k=1:row

trans\_DIS(s)=trans\_DIS(s)+(x(k))^2\*(DATA(i,k)-DATA(j,k))^2;

end

trans\_DIS(s)=sqrt(trans\_DIS(s));

s=s+1;

end;

end

end

12．CalcuClass函数

function [ class ] = CalcuClass(x)

%用来统计各类包含的记录数

m=size(x);

max\_class=max(x);

class=zeros(max\_class,2);

for i=1:max\_class

class(i,1)=i;

end

for i=1:m

class(x(i),2)=class(x(i),2)+1;

end

end

13．FindFraudNotes函数

function [ ] = FindFraudNotes( class,t )

%用于找出欺诈点的账单号

%输入参数包括两个

%class为一个两列的矩阵，第一列表示第几类点，第二列表示该类点的总个数

%t为一列数组，表示该点对应的类别

%输出为欺诈点的账单号

global ORI\_DATA;%利用全局变量ORG\_DATA(表示DATA中最原始的数据，读取ORG\_DATA中存放的账单号)

global fraud;%利用全局变量fraud（存放所有欺诈记录的账单号）

[m,n]=max(class(:,2));

%m表示聚类点个数最多的类别中点的个数

%n表示聚类点最多对应的类别

%遍历class矩阵

for i=1:size(t,1)

if ~(t(i)==n)

fraud=[fraud,ORI\_DATA(i,5)];

end

end

end

14．SearchNotes函数

function [ fraud ] = SearchNotes( bills,fraud )

%此函数用于根据账单号查找购药记录

%函数有两个输入参数

%bills表示所有的购药记录，bills矩阵有六列

%第一列为执行科室

%第二列为病人ID

%第三列为该张账单的总价钱

%第四列为开嘱医生的ID

%第五列为账单号

%第六列为该患者的购药的次数

%输入fraud是一个一列的数组

%输入fraud表示疑似欺诈点的账单号

%输出参数为购药记录

%第一列为账单号

%第二列为病人ID

%第三列为执行科室

%第四列为开嘱医生ID

%第五列为该张账单的总价

%第六列为该患者一个月内的拿药次数

m=size(bills,1);%m表示bills的行数

n=size(fraud,1);%n表示fraud的行数

j=1;

for i=1:m

if bills(i,5)==fraud(j,1)

fraud(j,2)=bills(i,2);

fraud(j,3)=bills(i,1);

fraud(j,4)=bills(i,4);

fraud(j,5)=bills(i,3);

fraud(j,6)=bills(i,6);

if(j==n)

return;

end

j=j+1;

end

end