控制与决策
Control and Decision

基于网格自组织粒球模型的不平衡回归方法

胡峰, 周雨龙, 苏祖强, 代劲, 于洪

引用本文:
胡峰, 周雨龙, 苏祖强, 等. 基于网格自组织粒球模型的不平衡回归方法[J]. 控制与决策, 2025, 40(8): 2513-2524.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2024.1277

您可能感兴趣的其他文章

Articles you may be interested in

嵌入重采样技术的C4.5决策树集成分类算法的临床医学预测

Clinical prediction of C4.5 decision tree classification algorithm with embedded resampling technique

控制与决策. 2021, 36(6): 1342-1350 https://doi.org/10.13195/j.kzyjc.2019.1247

具有不确定丢包率和时变采样周期的Delta算子系统故障检测

Fault detection for delta operator systems with uncertain packet dropout rate and time-varying sampling periods

控制与决策. 2021, 36(5): 1101-1109 https://doi.org/10.13195/j.kzyjc.2019.1154

基于相互邻近度的密度峰值聚类算法

Density peaks clustering based on mutual neighbor degree

控制与决策. 2021, 36(3): 543-552 https://doi.org/10.13195/j.kzyjc.2019.0795

基于双权重多邻域保持嵌入的间歇过程故障检测

Fault detection of batch process based on double weight and multiple neighborhoods preserving embedding

控制与决策. 2021, 36(12): 3023-3030 https://doi.org/10.13195/j.kzyjc.2020.0659

基于改进多目标优化算法的分布式数据中心负载调度

Multi-objective optimization of energy and performance management in distributed data centers

控制与决策. 2021, 36(1): 159-165 https://doi.org/10.13195/j.kzyjc.2019.0702

基于网格自组织粒球模型的不平衡回归方法

胡 峰[1,2†]，周雨龙[1,2]，苏祖强[2]，代 劲[2]，于 洪[1,2]

(1. 重庆邮电大学 计算机科学与技术学院，重庆 400065；
2. 重庆邮电大学 计算智能重庆市重点实验室，重庆 400065)

摘 要: 现有的不平衡数据回归算法需要多次计算样本间的距离, 当样本数量较大时, 处理效率较低. 粒球模型可将样本集合迭代划分为多个粒球, 以降低样本规模. 但是, 当前的粒球模型依赖于样本类别标签, 不适合回归任务. 鉴于此, 首先, 利用粒球内样本区域的网格划分, 定义粒球的填充度, 提出一种网格自组织粒球模型 (GSOGB), 能够同时处理回归任务和分类任务; 然后, 在此基础上, 给出粒球内样本在邻域内的过采样策略, 提出基于网格自组织粒球模型的不平衡回归方法 (GSOGB-SMOTER). 实验结果表明: 所提出 GSOGB 在 12 个分类数据集上优于现有粒球模型; 所提出 GSOGB-SMOTER 在 10 个不平衡回归数据集连续目标值域的 5 个等长分区的 MSE 指标上略优于文献中的 7 种算法, 且具有鲁棒性和更高的运行效率, 能够快速处理较大规模数据的不平衡回归.

关键词: 不平衡数据；回归；粒球；填充度；网格；邻域

中图分类号: TP39 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2024.1277

引用格式: 胡峰, 周雨龙, 苏祖强, 等. 基于网格自组织粒球模型的不平衡回归方法 [J]. 控制与决策, 2025, 40(8): 2513-2524.

An imbalanced regression method based on grid self-organized granular ball model

HU Feng[1,2†]，*ZHOU Yu-long*[1,2]，*SU Zu-qiang*[2]，*DAI Jin*[2]，*YU Hong*[1,2]

(1. College of Computer Science and Technology， Chongqing University of Posts and Telecommunications，Chongqing 400065，China；2. Chongqing Key Laboratory of Computational Intelligence，Chongqing University of Posts and Telecommunications，Chongqing 400065，China)

Abstract: Existing imbalanced data regression algorithms require multiple calculations of distances between samples, which leads to low processing efficiency when the number of samples is large. The granular ball model can iteratively divide a sample set into multiple granular balls, reducing the sample scale. However, current granular ball models rely on sample category labels and are not suitable for regression tasks. This paper utilizes the grid division of the granular ball sample area, defines the filling degree of the granular ball, and proposes a grid self-organized granular ball model (GSOGB) that can handle both regression and classification tasks simultaneously. On this basis, an oversampling strategy for samples within the granular ball in the neighborhood is given, and an imbalanced regression method based on the grid self-organized granular ball model (GSOGB-SMOTER) is proposed. Experimental results show that the proposed GSOGB outperforms existing granular ball models on 12 classification datasets, and the proposed GSOGB-SMOTER slightly outperforms seven algorithms in the literature on the MSE metric of five equal-length target value domains of 10 imbalanced regression datasets, and has robustness and higher operational efficiency, capable of quickly processing large-scale imbalanced data for regression.

Keywords: imbalanced data；regression；granular ball；filling degree；grid；neighborhood

0 引 言

工业、金融、采掘、医疗等领域的数据往往具有不平衡特性, 即数据集中某些类的样本数量远远少于其他类, 但是却具有更高的价值, 其误分类的代价更高[1], 如医疗诊断、欺诈检测等. 根据样本标签的类型, 不平衡数据可分为类别标签数据和连续值标签数据, 与之对应的处理任务分别为不平衡数据分类和不平衡数据回归. 当前, 对于不平衡数据的研究

收稿日期: 2024-11-01；**录用日期:** 2025-03-19.
基金项目: 重庆市自然科学基金创新发展联合基金项目 (CSTB2023NSCQ-LZX0006)；国家自然科学基金重点项目 (62233018)；成都市重点研发计划项目 (2023-YF11-00059-HZ).
责任编委: 黄敏.
†通信作者. E-mail: hufeng@cqupt.edu.cn.

主要集中于不平衡数据分类, 而对于不平衡数据回归的研究较少. 但是, 现实世界中存在大量不平衡数据回归的处理需求[2], 如钢铁冶炼中的铁水质量预测、模具生产过程中的产品质量预测、矿井的瓦斯浓度预测等. 因此, 需要对不平衡回归算法开展相关研究.

数据采样主要是通过数据欠采样、过采样来降低数据的不平衡性. 简单的欠采样易导致数据信息丢失[3], 从而降低回归精度. 因此, 更多的研究聚焦于过采样方法. 过采样需要多次计算样本间距离, 当样本数量较大时, 会导致处理效率降低. 为此, 研究人员试图通过改进过采样策略来解决效率问题, 但是仍然存在一些局限性, 如: 1) 通过聚类降低样本规模, 但是此类方法依赖聚类参数, 易导致不精确的聚类结果, 进而影响模型的精度; 2) 简化样本邻居的搜索过程, 此类方法简单且效率高, 但是在噪声环境下易找到错误的邻居样本; 3) 重点关注决策边界的样本, 从而缩减邻居样本的搜索空间, 但是易忽视一些重要样本. 因此, 如何在降低样本规模、缩减样本邻居搜索空间的同时, 能够兼顾样本整体分布, 避免噪声样本的干扰, 将是一种解决不平衡回归的有效途径.

粒球模型[4] 能够将样本迭代粒化为若干个粒球, 后续采用分治思想在单个粒球上进行处理, 可降低样本规模, 提高处理效率. 然而, 现有的粒球模型依赖于样本的类别标签, 不适合处理连续标签的任务.

本文的主要内容如下: 1) 针对现有粒球模型不适合处理连续标签样本的不足, 结合粒球内样本区域的网格划分, 定义粒球填充度, 提出网格自组织粒球模型 GSOGB, 其不依赖于样本标签, 划分后的粒球可处理回归任务和分类任务; 2) 结合网格自组织粒球模型, 定义粒球稀有度和粒球内样本的邻域半径, 给出样本邻域范围内的过采样策略, 提出基于网格自组织粒球模型的不平衡回归方法 GSOGB-SMOTER, 能够处理较大规模的回归任务.

1 相关工作

1.1 不平衡回归的问题定义

机器学习中的回归任务旨在通过给定的回归训练数据集$D=\{(X_i,y_i)\}_{i=1}^{N}$, 拟合输入$X$与输出$y$间的函数关系$y=f(X)$, 并在预测阶段最小化预测值与真实值间的误差. 其中: X为数据样本的特征向量, y为数据样本的目标值. 不平衡回归任务是一类特殊的回归任务, 其特点在于: 1) 样本连续目标值的某些区域比其他区域更为重要; 2) 这些区域的样本在数据集中的数量较少[5]. 因此, 模型在建立和预测阶段, 为了降低预测值与真实值间的总体差异, 往往会倾向于关注那些样本数量较多的区域, 而忽视样本数量较少但是更为重要的区域, 从而影响模型对这些关键区域的预测准确性.

1.2 不平衡回归算法

不平衡回归问题的解决方案主要是从算法层面和数据层面提出的. 算法层面的方法专注于改变现有的学习算法, 从而增加其对于不平衡问题的敏感度, 如: Branco 等[6] 基于套袋策略提出了一种处理不平衡回归问题的集成方法 REBAGG; Steininger 等[7] 结合样本目标值的密度权重提出了一种不平衡回归样本的加权方法; Devi 等[8] 在 AdaBoost 模型中引入了基于误分类比率的成本函数, 提出了基于样本相关性和成本敏感的集成学习方法 CorrOV-CSEn. 算法层面的解决方法需要对学习算法进行修改或集成, 限制了这些方法的广泛应用. 数据层面的方法主要通过简单、快速的数据采样来实现, 通过在预处理阶段调整原始数据分布, 可降低数据的不平衡性[9], 使其能够适用于标准的学习算法. 目前, 很多学者针对不平衡回归的采样算法开展了相关研究, 主要可分为两类: 1) 将现有的不平衡分类的采样算法扩展为用于不平衡回归的采样算法. 此类算法基于 Ribeiro 等[2] 提出的连续目标值的相关性函数, 将回归任务中的样本划分为不同的类别, 如: 文献 [10] 和文献 [11] 分别在 SMOTE[12] 和 G-SMOTE[13] 的基础上, 提出了适用于回归任务的 SMOTER[10] 和 G-SMOTER[11]; Branco 等[5] 基于 SMOTER 和高斯噪声扰动 GN[14] 提出了 SMOGN; Song 等[15] 结合 SMOGN 和 Scalable Kmeans++ 聚类算法, 提出了一种分布式环境下处理不平衡回归任务的 DistSMOGN; Li 等[16] 提出了一种基于仿射组合的过采样算法 ACOS, 该算法利用仿射变换将新样本的合成区域控制在种子样本构成的凸包区域的内部或外部. 2) 直接用于不平衡回归的采样算法: Branco 等[14] 利用样本的相关性进行随机过采样和随机欠采样, 提出了基于样本相关性权重的 WERCS 采样算法; Camacho 等[17] 根据样本的密度权重选取种子样本及其邻居样本进行线性插值, 提出了加权过采样算法 WSMOTER.

过采样在处理大规模数据时效率不高, 经典的 SMOTER 和 SMOGN 在对每个少数类样本进行 k-最近邻搜索的过程非常耗时. 虽然随机欠采样或随机过采样的处理效率高, 但是, 它们只是进行简单的样本删除或复制, 易导致预测精度下降. 因此, 更多的研究聚焦于设计能够提高处理效率的 SMOTE 变体算法, 其处理策略可分为 3 种: 1) 通过聚类方法降低样本规模, 如 DBSMOTE[18]、Kmeans-SMOTE[19]、SOMO[20]、G-SOMO[21] 等, 此类方法利用样本的聚类

结果, 将少数类样本的采样控制在较小区域, 从而大大缩减了少数类样本的最近邻搜索空间, 但是其过多依赖于聚类算法, 当聚类参数选取不合理或样本分布不规则时, 易导致聚类效果不佳, 影响后续的采样过程; 2) 简化样本邻居的搜索过程, 如 WSMOTER 摒弃了通过计算样本间距离来寻找邻居的方法, 而是将目标值相似的样本视为邻居, 提升了处理效率, 但是, 当数据中存在较多目标值异常的噪声样本时, 易导致采样效果不佳; 3) 重点关注决策边界的样本, 从而减少邻居样本的搜索空间, 如 Borderline-SMOTE[22]、ADASYN[23]、Safe-Level-SMOTE[24]、SVM-SMOTE[25] 等, 这类算法只处理边界附近的少数类样本, 简化了在整个样本空间寻找最近邻样本的操作, 但是, 当样本的决策边界复杂或存在多个边界时, 可能会遗漏一些重要的样本.

1.3　粒球模型

粒球模型是由 Xia 等[4] 提出的一种用于粒计算的机器学习模型, 旨在通过将数据空间划分为多个粒球来进行数据的高效、可扩展和鲁棒的学习. 每个粒球可视为数据空间中的一个区域, 通过一系列粒球代替输入空间中的数据点, 以此达到覆盖原始输入空间的目的. 给定一个粒球GB的样本集合 $A=\{(X_i,y_i)\}_{i=1}^{n}$. 其中: $X_i=(x_{i1},x_{i2},\dots,x_{im})$为样本的$m$维特征向量, y_i为样本的类别标签. 这里给出处理分类任务的粒球模型的基本概念.

定义 1 (粒球中心、粒球半径)[4]　粒球GB的中心C和半径r的定义如下所示:

$$C=\frac{1}{n}\sum_{i=1}^{n}X_i, \tag{1}$$

$$r=\frac{1}{n}\sum_{i=1}^{n}\mathrm{dist}(X_i,C), \tag{2}$$

其中$\mathrm{dist}(X,C)$为样本(X,y)与中心C的距离.

定义 2 (粒球纯度)[4]　若GB内的样本有k个类别, P_i为相同类别的样本集合, $i=1,2,\dots,k$, $|P_i|$为P_i中的样本数量, 则粒球纯度T的定义为

$$T=\frac{\max(|P_i|)}{n},\ i=1,2,\dots,k. \tag{3}$$

结合定义 1 和定义 2, Xia 等[4] 提出了基础的粒球模型, 该模型结合样本类别标签、粒球纯度先验阈值, 通过执行Kmeans 聚类算法将样本集合迭代划分为多个粒球, 且每个粒球的纯度均达到指定的阈值. 在此基础上, Xia 等[4] 给出了改进的粒球模型: 加速粒球模型[26]、自适应粒球模型[26]. 然而, 这些模型仍然局限于处理分类数据, 不适合处理回归问题.

2　基于网格自组织粒球模型的不平衡回归方法 (GSOGB-SMOTER)

考虑到 SMOTER、SMOGN 等主流的不平衡回归采样算法在处理大规模数据时效率低下, 若采用粒球模型降低原始数据集的样本规模, 并在每个粒球中对样本进行过采样处理, 则可提高算法的处理效率. 然而, 现有粒球模型主要针对分类问题, 不适合回归任务. 若根据粒球内的样本分布驱动粒球的划分, 而不依赖于样本的标签值, 则能够设计出一种用于回归任务的粒球模型. 基于上述分析, 本文研究了一种新的粒球模型, 可用于处理回归任务, 并在此基础上开展不平衡回归算法的研究. 具体如下: 1) 定义粒球内样本区域的网格划分策略和粒球的填充度, 提出了网格自组织粒球模型 GSOGB; 2) 研究粒球的稀有度和粒球内样本的过采样策略, 提出了一种基于网格自组织粒球模型的不平衡回归方法 GSOGB-SMOTER. 整体流程如图 1 所示.

2.1　网格自组织粒球模型 (GSOGB)

粒球模型的目的是将多数相似的、均匀分布的样本用一个粒球来描述, 这里仅考虑样本的特征, 不考虑样本的标签值, 通过对样本区域进行网格划分来分析样本的分布情况, 定义粒球的填充度, 重新设计粒球的划分策略, 提出一种新的粒球模型. 给定一

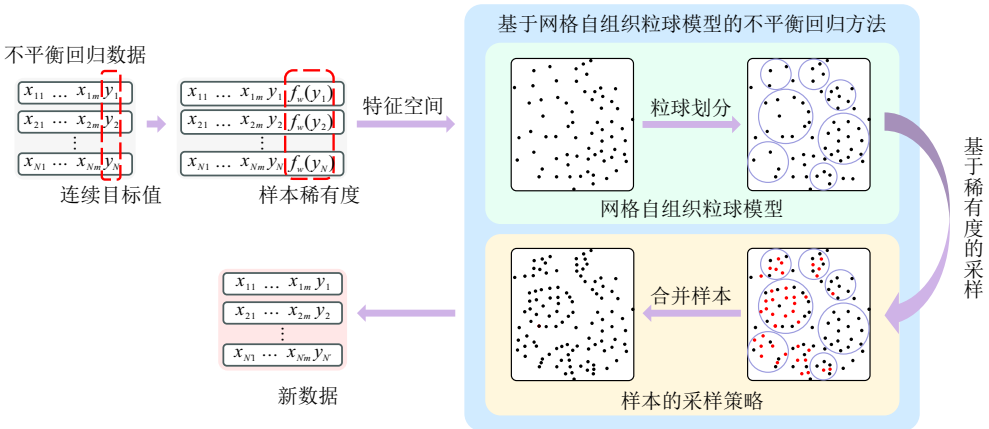


图1　GSOGB-SMOTER 整体流程

个粒球GB的样本集合$A=\{(X_i,y_i)\}_{i=1}^{n}$. 其中: $X_i=(x_{i1},x_{i2},\dots,x_{im})$为样本的$m$维特征向量, y_i为样本的连续标签, GB的中心C由式 (1) 得到. 则网格划分策略和粒球划分策略的相关定义如下.

2.1.1 网格划分策略

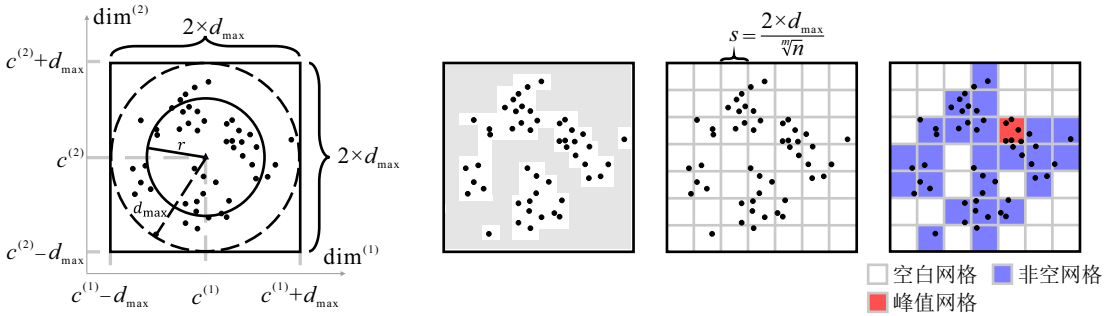
定义 3 (网格区域) 粒球GB的网格区域是由样本各维度的边界范围所构成的空间区域, 样本空间的第j维的边界$\text{Bound}^{(j)}$的定义如下所示:

$$\text{Bound}^{(j)}=[C^{(j)}-d_{\max},C^{(j)}+d_{\max}],\quad j=1,2,\dots,m. \tag{4}$$

其中: $C^{(j)}$为中心C在第j维度的值; $d_{\max}$为粒球内样本到中心C的最大距离, 其计算公式为

$$d_{\max}=\max(\text{dist}(X_i,C)),\ i=1,2,\dots,n. \tag{5}$$

根据定义 3, 网格区域在各维度覆盖的范围为$2\times d_{\max}$, 并以粒球的中心C作为网格区域的中心, 因此, 能够覆盖粒球内的所有样本. 图 2(a) 为粒球GB的网格区域示意图. 其中: $\dim^{(j)}$为样本空间的第j维度, "黑色的圆点"为样本点.



(a) 网格区域示意图 (b) 空白部分 (c) 网格划分 (d) 3类网格示意图

图2 网格划分策略

定义 4 (网格边长) 将网格区域细分为若干个小网格, 每个小网格的边长定义如下所示:

$$s=\frac{2\times d_{\max}}{\sqrt[m]{n}}. \tag{6}$$

根据定义 3 和定义 4, 网格区域能够细分的网格数量为$\text{grid_num}=(2\times d_{\max})^m/s^m=n$. 所提出策略能够使得细分的网格数量接近于样本的数量n, 当样本分散在不同的小网格中时, 样本的分布情况最为均匀.

图 2(b) 中的灰色区域可粗略地表示网格区域中不存在样本的空白部分, 网格划分策略能够利用小网格精确地表示样本在网格区域中的占比. 基于网格区域和网格边长的概念, 图 2(c) 展示了利用小网格对粒球GB的网格区域进行划分的结果.

定义 5 (空白网格、峰值网格、非空网格) 经网格划分后, 根据每个小网格内的样本数量, 将小网格分为以下 3 类: 1) 样本数量为 0 的网格为"空白网格"; 2) 样本数量最大的网格为"峰值网格"; 3) 其余的网格为"非空网格" (如图 2(d) 所示).

2.1.2 粒球划分策略

定义 6 (粒球填充度、最小填充度) 粒球网格区域进行网格划分后, 假设得到的空白网格数量为n', 则粒球填充度F和最小填充度$F_{\min}$的定义为

$$F=(n-n')/n, \tag{7}$$

$$F_{\min}=1/m. \tag{8}$$

粒球填充度通过计算存在样本的网格占比来衡量样本分布的密集程度, 填充度越高, 粒球越能够有效表示其覆盖的样本区域. 考虑到"维度灾难"的问题[27], 高维空间中样本点趋向于无穷远且聚集程度降低, 当样本数量不变时, 各维度上的样本数量与维度数量成反比. 因此, 为适应不同的数据维度, 本文用最小填充度动态来反映一个合格的粒球内大部分样本所占的区域比例, 从而减少粒球不必要的划分.

基于定义 6, 本文粒球模型划分策略如下.

step 1: 给定一个回归数据集$D=\{(X_i,y_i)\}_{i=1}^{N}$, 初始时将$D$作为初始粒球GB的样本集合, 采用网格划分策略将其网格区域进行网格化.

step 2: 计算粒球GB的填充度F, 若$F\geqslant F_{\min}$, 则GB终止划分, 作为最终的粒球; 若$F<F_{\min}$, 则GB需要继续划分.

step 3: 对于需要继续划分的粒球GB, 选取两个中心点 (粒球GB的中心C、峰值网格内样本的中心C_{peak}) 对粒球进行后续的划分: 将中心C和C_{peak}分别作为两个新粒球GB_1和GB_2的中心, 计算每个样本到两个中心的距离, 并将样本分配给最近中心所属的粒球, 得到两个新的粒球GB_1和GB_2.

step 4: 分别对GB_1和GB_2重复 step 2 ~ step 4 , 直至所有的粒球均不再继续划分.

所提出粒球模型基于网格划分策略量化样本的分布情况, 因此, 不再需要 Kmeans 方法通过多次迭代计算样本与中心的距离来确保聚类的稳定性, 而是仅计算一次样本与中心的距离, 并将样本分配给最近的中心, 从而能够提高粒球的划分速度.

2.1.3 算法描述

网格自组织粒球模型的伪代码如算法 1 所示.

算法 1　GSOGB 算法.

输入: 回归训练数据集$D=\{(X_i,y_i)\}_{i=1}^{N}$;

输出: 粒球集合 Balls.

1) 初始化Balls $=\varnothing$, Temp $=\varnothing$;

2) Temp $=\{D\}$ (D为初始粒球的样本集合);

3) 由式 (8), 计算最小填充度$F_{\min}$;

4) for each A in Temp do

5)　令A表示粒球GB的样本集, Temp=Temp$-A$;

6)　由式 (1)、(4)、(6), 计算出粒球的中心C、网格区域各维度的边界 Bound、小网格的边长s;

7)　将A中的样本分配到对应的小网格中, 并根据定义 5 将小网格分为“空白网格”“峰值网格”“非空网格”;

8)　由式 (7), 计算粒球GB的填充度F;

9)　if $F<F_{\min}$ then

10)　计算“峰值网格”内样本的中心C_{peak}, 将C和C_{peak}作为新粒球GB_1和GB_2的中心;

11)　计算GB中每个样本到C和C_{peak}的距离, 并将其分配给最近中心所属的粒球, 令A_1、A_2表示GB_1、GB_2的样本集合;

12)　Temp $=$ Temp$\bigcup A_1\bigcup A_2$;

13)　else

14)　Balls $=$ Balls$\bigcup A$;

15)　end if

16) end for

17) 返回 Balls.

算法的时间复杂度分析: 设回归数据集D的样本数量为N, 特征数为m, 假定最终生成的粒球数量为$a(a\ll N)$, 每个粒球的样本数量可视为N/a. 步骤 1) 和步骤 2) 初始化的操作以及步骤 3) 计算$F_{\min}$的时间复杂度皆为$O(1)$; 在步骤 4) 开始的循环中, 每次划分可得到两个新粒球, 故循环次数可视为$\log_2 a$; 步骤 6) 和步骤 7) 的时间复杂度皆为$O(N\times M/a)$, 由式 (8), 小网格中的样本数量可视为m, 则步骤 11) 的时间复杂度为$O(m^2)$, 步骤 12) 的时间复杂度为$O(N\times M/a)$. 因此, 步骤 4) $\sim$ 步骤 16) 的时间复杂度为$O(N\times M\times\log_2 a/a)$. 根据以上分析, 算法 1 的时间复杂度为$O(N\times M\times\log_2 a/a)$.

2.2 基于网格自组织粒球模型的过采样

2.2.1 粒球稀有度

用户在不平衡回归问题中更关注稀有样本, 因此, 度量样本稀有度至关重要. Steininger 等[7] 利用加权函数对样本密度进行权重分配, 得到了每个样本的密度权重, 用于反映样本的稀有度. 对于回归数据集$D=\{(X_i,y_i)\}_{i=1}^{N}$, 粒球稀有度的相关定义如下.

定义 7 (样本密度)[7]　样本(X_i,y_i)的密度为

$$p(y_i)=\frac{1}{Nh}\sum_{j=1}^{N}\text{KDE}\Big(\frac{y_i-y_j}{h}\Big). \tag{9}$$

其中: KDE$(*)$为核函数, 用于估计连续目标值的概率密度函数; h为带宽. 假设样本的密度集合为$\text{Density}^{(y)}=p(y_1),p(y_2),\ldots,p(y_N)$, 则将样本的密度进行 min-max 归一化后的样本密度$p'(y_i)$为

$$p'(y_i)=\frac{p'(y_i)-\min(\text{Density}^{(y)})}{\max(\text{Density}^{(y)})-\min(\text{Density}^{(y)})}. \tag{10}$$

定义 8 (样本稀有度)[7]　基于归一化后的样本密度, 样本(X_i,y_i)的稀有度$f_w(y_i)$的定义为

$$f_w(y_i)=\frac{\max(1-\alpha p'(y_i),\varepsilon)}{\dfrac{1}{N}\displaystyle\sum_{j=1}^{N}(\max(1-\alpha p'(y_j),\varepsilon))}. \tag{11}$$

其中: α为超参数, 决定对稀有样本的权重增加程度; ε为一个很小的正常数, 避免了权重为 0 或负的情况. 文献 [7] 表明, 样本稀有度的均值为 1.

定义 9 (粒球稀有度)　给定一个粒球GB的样本集合$A=\{(X_i,y_i)\}_{i=1}^{n}$, 则粒球稀有度定义为粒球内样本的平均稀有度, 其计算公式如下所示:

$$\text{rar}(\text{GB})=\frac{1}{n}\sum_{i=1}^{n}f_w(y_i). \tag{12}$$

2.2.2 粒球内样本的过采样策略

基于网格自组织粒球模型将样本表示为多个粒球后, 可认为粒球内的样本分布均匀且具有高度的相似性, 因此, 样本间也具有近似的稀有程度. 依据粒球稀有度能够快速找到稀有程度较高的样本, 并以粒球为单位对这些样本进行过采样处理. 过采样策略的相关定义如下.

定义 10 (样本邻域半径)　粒球GB的样本集合$A=\{(X_i,y_i)\}_{i=1}^{n}$, 则样本$(X_i,y_i)$的邻域半径$\delta$为

$$\delta = s \times f_w(y_i)/2. \tag{13}$$

基于邻域方法的相似性假设[28]以及噪声样本在数据集中的分布特性[29], 所提出过采样策略发生在相似样本间, 而噪声样本对于局部区域的影响会被稀释, 从而能够有效降低噪声对采样过程的干扰. 此外, 本文给出粒球的GB过采样数量的计算公式为

$$\text{samp\_num(GB)} = \begin{cases} \text{round}\Big((\text{rar(GB)} - 1) \times \sum\limits_{i=1}^{n} f_w(y_i)\Big), \\ \quad \text{rar(GB)} > 1; \\ 0,\ \text{rar(GB)} \leqslant 1. \end{cases} \tag{14}$$

其中round$(*)$为四舍五入函数. 根据定义 9 和定义 10, 粒球GB的过采样策略如下 (如图 3 所示).

step 1: 若rar(GB) >1, 则基于样本稀有度权重选择种子样本及其近邻样本, 稀有度越高, 被选中的概率越大, 从而减少低稀有度样本成为种子样本的可能;

step 2: 针对选定的种子样本, 如图 3 中的样本b, 在其邻域半径δ的范围内按照稀有度权重随机选取一个邻居样本b_1, 通过线性插值合成新样本b_{new};

step 3: 重复 step 2, 直至合成的新样本的数量达到samp_num(GB).

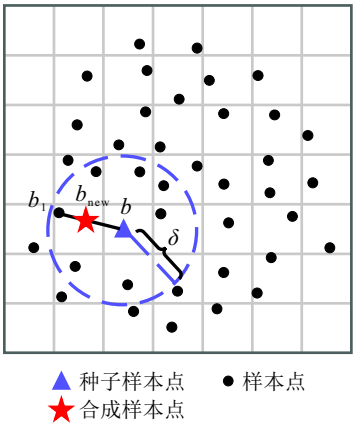


图3 样本邻域范围内的过采样示意图

2.2.3 算法描述

基于网格自组织粒球模型的不平衡回归方法的伪代码如算法 2 所示.

算法 2 GSOGB-SMOTER 算法.

输入: 回归训练数据集$D = \{(X_i, y_i)\}_{i=1}^{N}$;

输出: 新数据集$D_{\text{new}} = \{(X_i, y_i)\}_{i=1}^{N'}$.

1) 初始化新数据集$D_{\text{new}} = \varnothing$;

2) 由式 (11), 计算出每个样本的稀有度$f_w(y)$;

3) 调用算法 1, 将D中的样本粒化为a个粒球$\text{GB}_1, \text{GB}_2, \ldots, \text{GB}_a$;

4) 由式 (12) 和 (14), 分别计算每个粒球的稀有度rar(GB)和过采样数量samp_num(GB);

5) for each GB in $\text{GB}_1, \text{GB}_2, \ldots, \text{GB}_a$ do

6) 令$A = \{(X_i, y_i)\}_{i=1}^{n}$为GB的样本集合;

7) if rar(GB) >1 then

8) for $i = 1$ to do

9) 以样本的稀有度$f_w(y)$为权重, 选取种子样本$b = (X, y)$, 并计算b的邻域半径δ;

10) 在b邻域范围内按照稀有度随机选取一个邻居样本$b_1 = (X_1, y_1)$;

11) $b_{\text{new}} = (X_{\text{new}}, y_{\text{new}})$ ←种子样本b与邻居样本b_1线性插值;

12) $D_{\text{new}} = D_{\text{new}} \bigcup b_{\text{new}}$;

13) end for

14) end if

15) end for

16) $D_{\text{new}} = D_{\text{new}} \bigcup D$;

17) 返回D_{new}.

算法的时间复杂度分析: 设回归数据集D的样本数量为N, 特征数为m, 算法 1 生成的粒球数量为$a(a \ll N)$. 在算法 2 中: 步骤 1) 初始化操作的时间复杂度为$O(1)$; 步骤 2) 计算样本的稀有度中, 最耗时的是核密度估计 KDE 的计算, 其时间复杂度为$O(N \times \log_2 N)$; 步骤 3) 调用算法 1 将样本粒化为多个粒球, 其时间复杂度为$O(N \times m \times \log_2 a/a)$; 步骤 4) 计算rar(GB)和samp_num(GB)的时间复杂度分别为$O(N)$和$O(a)$; 在步骤 8) 开始的循环中, samp_num(GB)可视为与粒球内样本的数量N/a呈线性相关, 步骤 10) 寻找种子样本的邻居样本的时间复杂度为$O(N \times m/a)$, 步骤 11) 线性插值的时间复杂度为$O(m^2)$, 故步骤 5) ~ 步骤 15) 的时间复杂度为$O(N^2 \times m/a)$; 步骤 16) 的时间复杂度为$O(N)$. 故算法 2 的时间复杂度为$O(N^2 \times m/a)$.

3 实 验

针对所提出基于网格自组织粒球模型的不平衡回归方法, 本文将从粒球模型的对比实验和不平衡回归算法的对比实验两个方面进行评估验证. 实验的软硬件平台如下: 12th Gen Intel(R) Core(TM) i7-12700 CPU, 16 GB 内存, Windows 11 64 位操作系统, 使用的编程语言为 Python3.8 版本.

3.1 数 据 集

为了验证网格自组织粒球模型的有效性, 以及基于网格自组织粒球模型的不平衡回归方法的高效性和鲁棒性, 本文从 UCI 机器学习数据库和 KEEL 开源数据库中共选取了 12 个分类数据集和 10 个不平衡回归数据集进行实验. 表 1 和表 2 分别为分类

数据集和回归数据集的详情.

表1　分类数据集详情

数据集编号	数据集名称	样本数	特征数	类别数
DS_1	heart	270	13	2
DS_2	australian	690	14	2
DS_3	mammographic	830	5	2
DS_4	raisin	900	7	2
DS_5	contraceptive	1473	9	3
DS_6	titanic	2201	3	2
DS_7	rice	3810	7	2
DS_8	banana	5300	2	2
DS_9	bankruptcy	6819	95	2
DS_{10}	ai4i2020	10000	6	2
DS_{11}	codon	13026	64	11
DS_{12}	credit_card	30000	23	2

表2　不平衡回归数据集详情

数据集编号	数据集名称	样本数	特征数
DS_{13}	machineCPU	209	6
DS_{14}	boston	506	13
DS_{15}	mortgage	1049	15
DS_{16}	treasury	1049	15
DS_{17}	ele-2	1056	4
DS_{18}	availPwr	1802	15
DS_{19}	space_ga	3107	6
DS_{20}	cpuAct	8192	21
DS_{21}	house	22784	16
DS_{22}	gait	181800	6

3.2　实验设计与结果分析

3.2.1　粒球模型对比实验

为了考察网格自组织粒球模型 (GSOGB) 的性能, 本文将在表 1 中的分类数据集上对比文献 [4] 中纯度阈值分别为 90%、95%、100% 的基础粒球模型 (ORIGB), 文献 [26] 中纯度阈值分别为 90%、95%、100% 的加速粒球模型 (ACCGB) 以及自适应粒球模型 (ADPGB), 并根据以下指标来评估粒球模型: 分类准确率、生成的粒球数量、粒球覆盖度. 其中: 分类准确率根据文献 [4] 中基于粒球模型改进的 kNN 分类算法 GBkNN[4] 的分类准确率; 粒球覆盖度则根据文献 [26] 中的定义, 具体如下.

定义 11 (粒球覆盖度)[26]　给定一个数量为N的样本集合, 利用粒球模型将其粒化为$\mathrm{GB}_1, \mathrm{GB}_2, \ldots, \mathrm{GB}_a$, 每个粒球中的多数类样本的数量分别为$n_1^{(\mathrm{maj})}, n_2^{(\mathrm{maj})}, \ldots, n_a^{(\mathrm{maj})}$, 则粒球覆盖度的计算公式为

$$\mathrm{coverage}(\mathrm{GB}) = \frac{1}{N}\sum_{i=1}^{a} n_i^{(\mathrm{maj})}. \tag{15}$$

实验采用 10-折交叉验证的 GBkNN 算法.

表 3 为基于不同粒球模型的 GBkNN 算法的分类准确率. 由表 3 可见, GSOGB 在多数数据集上取得了较好的分类准确率, rank 平均排名小于其他粒球模型. 表 4 为不同粒球模型生成的粒球数量. 由表 4 可见, GSOGB 在多数数据集上能够生成比其他粒球模型更少数量的粒球. 图 4 表明, GSOGB 最终能够使得粒球的覆盖度达到 80% 以上, 并与生成的粒球数量呈现高度的正相关, 略优于基于粒球纯度信息划分的 ADPGB. 以上实验结果分析表明, GSOGB 在多数情况下能够利用较少数量的粒球, 覆盖数据集中的多数样本, 且从 GBkNN 算法的分类准确率可以看出, 生成的每个粒球质量较高.

3.2.2　不平衡回归算法对比实验

为了考察 GSOGB-SMOTER 的高效性和鲁棒性, 本文将在表 2 中的回归数据集上对比 RO[14]、WERCS[14]、SMOTER[10]、SMOGN[5]、G-SMOTER[11]、ACOS[16] 与 WSMOTER[17] 的预测精度和时间消耗. 此外, 本文将在部分数据集上引入不同类型 (特征噪

表3　基于不同粒球模型的 GBkNN 算法的分类准确率

粒球模型	ORIGB			ACCGB			ADPGB	GSOGB
纯度阈值	90%	95%	100%	90%	95%	100%	无	无
DS_1	0.759	0.778	0.774	0.770	0.774	0.737	0.793	**0.807**
DS_2	0.844	0.839	0.830	0.826	0.836	0.839	0.822	**0.858**
DS_3	0.757	0.757	0.747	0.776	0.745	0.751	0.781	**0.802**
DS_4	0.846	0.842	0.837	0.838	0.836	0.847	0.832	**0.856**
DS_5	0.436	0.439	0.441	0.439	0.423	0.424	0.439	**0.456**
DS_6	**0.791**	**0.791**	**0.791**	0.784	0.788	0.787	0.779	**0.791**
DS_7	0.914	0.914	0.909	0.916	**0.917**	0.909	0.913	0.916
DS_8	0.878	0.869	0.873	**0.893**	0.886	0.882	0.870	**0.893**
DS_9	**0.968**	**0.968**	0.967	**0.968**	0.965	0.966	0.965	**0.968**
DS_{10}	0.966	0.966	**0.971**	0.966	0.966	0.970	0.966	0.966
DS_{11}	0.965	0.974	0.978	0.977	0.983	0.985	**0.986**	0.978
DS_{12}	0.792	0.784	0.784	0.787	0.783	0.782	0.798	**0.799**
rank	3.917	3.833	4.500	4.000	5.083	5.083	4.750	**1.500**

表4 不同粒球模型生成的粒球数量

粒球模型	ORIGB			ACCGB			ADPGB	GSOGB
纯度阈值	90%	95%	100%	90%	95%	100%	无	无
DS_1	97	126	127	118	143	142	61	**35**
DS_2	108	246	271	153	277	296	142	**92**
DS_3	262	305	321	241	303	318	121	**111**
DS_4	209	282	334	231	322	349	220	**141**
DS_5	1 093	1 100	1 104	1 093	1 109	1 104	277	**174**
DS_6	13	13	13	12	13	13	**10**	14
DS_7	**2**	561	866	374	618	906	764	748
DS_8	827	1 071	1 306	852	1 083	1 326	**40**	341
DS_9	**2**	**2**	1 033	**2**	**2**	1 285	718	282
DS_{10}	**2**	**2**	977	**2**	19	1 142	562	782
DS_{11}	**65**	202	682	345	575	1 454	3 175	2 102
DS_{12}	10 867	14 907	15 849	11 491	15 238	16 042	3 843	**3 140**

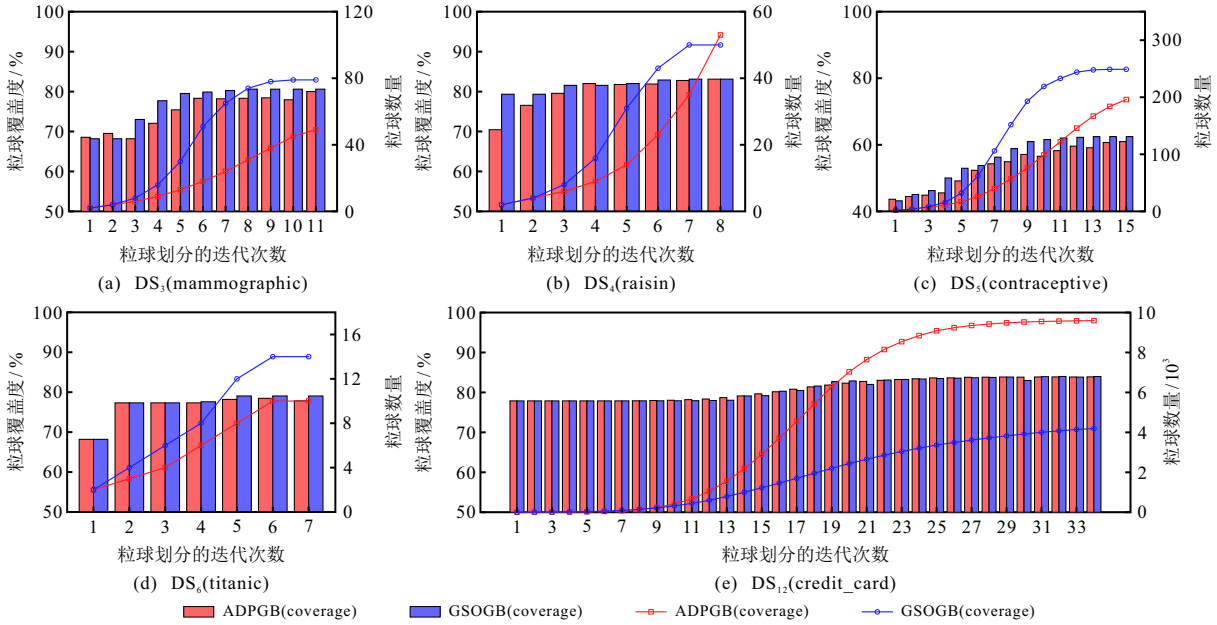


图4 ADPGB 和 GSOGB 粒球化过程中生成的粒球数量和粒球覆盖度的走势

声样本、重叠样本) 以及不同比例的噪声样本来考察 GSOGB-SMOTER 的鲁棒性. 其中: 对样本特征添加高斯扰动生成特征噪声样本, 改变相似特征的样本目标值生成重叠样本. 式 (11) 中的超参数α对于回归数据样本的稀有度至关重要, 因此, 实验首先将以数据集DS_{16} (treasury) 和DS_{20} (cpuAct) 为例, 考察不同取值的α对 GSOGB-SMOTER 的影响. 以上实验均采用 10-折交叉验证, 并使用 Python 机器学习库 Scikit-learn 的决策树回归模型 DTR 对算法的采样结果进行训练和预测.

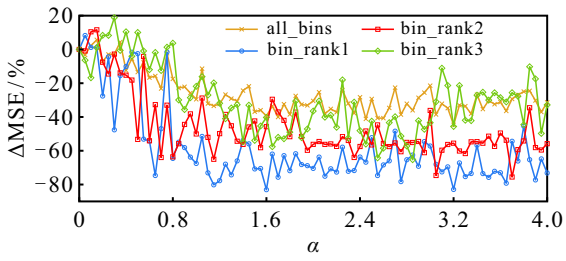
本实验根据以下指标评估算法: 回归数据集连续目标值域的 5 个等长分区的均方误差 MSE、算法的时间消耗. 考虑到不平衡数据中稀有样本与非稀有样本的分布差异, 标准的 MSE 等指标并不能够准确评估不平衡回归算法, 因此, 本文采用文献 [7] 中提出的评估目标值不同区域误差的方法来验证算法的可行性. 此方法的主要处理过程如下: 首先, 将连续目标值进行排序, 并将值域划分为 5 个分区, 每个分区包含目标值区间范围的 20%; 然后, 根据每个分区的样本数量为分区分配一个 1 ∼ 5 的排名, 样本数量越少的分区, 其排名越高; 最后, 评估不同分区的 MSE. 本文用$\text{bin}_1 \sim \text{bin}_5$表示 5 个分区, 用 bin_rank*表示排名为*的分区. 表 5 为回归数据集各分区的样本数量和排名, 其中$(*)$表示排名.

图 5(a) 和图 5(b) 展示了数据集DS_{16}(treasury) 和DS_{20}(cpuAct) 在不同α取值 (范围为0 ∼ 4.0, 步长为 0.05) 的 GSOGB-SMOTER 采样后不同分区相较于原始数据集的 MSE 变化率 (ΔMSE), 其中“all_bins”表示全部的分区 (由于数据集DS_{20}在排名第 1 的分区上不存在样本, 图 5(b) 中并未展示 bin_rank1 分区上的ΔMSE). 由图 5 可见: 当$\alpha < 1.0$时, 不同分区上的ΔMSE 不太稳定, 随着α不断增大,

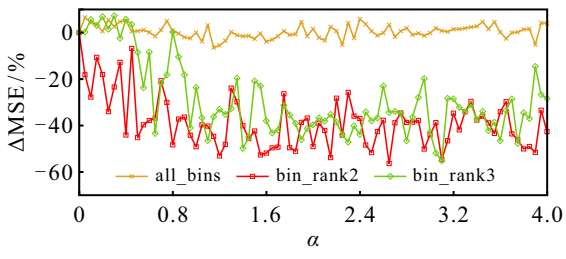
表5　回归数据集各分区的样本数量和排名

数据集编号	bin_1	bin_2	bin_3	bin_4	bin_5
DS_{13}	185 (5)	15 (4)	6 (3)	**1 (1)**	2 (2)
DS_{14}	77 (3)	240 (5)	127 (4)	38 (2)	**31 (1)**
DS_{15}	402 (5)	353 (4)	158 (3)	82 (2)	**54 (1)**
DS_{16}	514 (5)	379 (4)	73 (3)	53 (2)	**30 (1)**
DS_{17}	669 (5)	201 (4)	103 (3)	58 (2)	**25 (1)**
DS_{18}	1121 (5)	550 (4)	93 (3)	32 (2)	**6 (1)**
DS_{19}	1 (2)	**0 (1)**	14 (3)	1760 (5)	1332 (4)
DS_{20}	294 (3)	**0 (1)**	100 (2)	1299 (4)	6499 (5)
DS_{21}	20416 (5)	1819 (4)	369 (3)	98 (2)	**82 (1)**
DS_{22}	13333 (3)	109508 (5)	45994 (4)	6796 (2)	**6168 (1)**

各分区的 MSE 均呈现出下降趋势, 在$\alpha = 1.0$左右时开始趋于稳定. 可以看出, 排名越高的分区, 其 MSE 下降的比例越显著, 这是因为 GSOGB-SMOTER 能够在高稀有度样本的邻域范围内合成更多的样本, 且从实验结果可以分析出, 合成的样本能够有效降低模型在预测时的误差. 基于上述分析, 所提出 GSOGB-SMOTER 将采用$\alpha = 1.0$的取值进行不平衡回归算法的对比实验.



(a) DS_{16}(treasury)



(b) DS_{20}(cpuAct)

图5　超参数α不同取值下的各分区的 MSE 变化率

表 6 为 DTR 模型在各算法采样后对数据集不同分区进行回归预测的 MSE. 其中: “未采样”表示原始数据集, “–”表示当前分区不存在样本. 由表 6 可见, GSOGB-SMOTER 在 bin_rank1、bin_rank2 与 bin_rank3 上比其他算法有着更好的表现, 在 bin_rank4 上也略优于其他算法. 尽管在 bin_rank5 以及 all_bins 上, GSOGB-SMOTER 略逊色于原始数据集, 但是仍然优于其他算法. 这是因为回归模型在训练阶段为了降低总体误差, 往往会更注重样本数量较多的区间的预测准确性. 因此, 在未采样的情况下, 模型会特别关注 bin_rank5 (样本数量最多的分区); 而在 GSOGB-SMOTER 采样的情况下, 排名高的分区样本数量得到了有效补充, 使得模型不再仅仅专注于 bin_rank5 的预测准确性, 这导致了该分区的 MSE 略有上升. 由表 5 可见, bin_rank5 中的样本数量显著多于其他分区样本数量的总和. 因此, 在计算 all_bins 上的 MSE 时, bin_rank5 上的 MSE 对于总体 MSE 的影响更大, 也就导致了原始数据集在 all_bins 上略优于其他算法. 表 7 为各算法在不同数据集分区上的平均排名情况, 其中 SMOTER 和 SMOGN 算法在DS_{22}上取最低排名.

图 6 和图 7 分别为在训练集上引入不同比例 (范围为 0 ～ 0.5, 步长为 0.025) 的特征噪声样本和重叠样本时, 未采样 (None)、WSMOTER 和 GSOGB-SMOTER 在 bin_rank1 和 bin_rank2 (图中表示为 bin_rank1_2) 上使用 DTR 模型进行预测的 MSE. 由图 6 可见, 在引入特征噪声样本时, GSOGB-SMOTER 在 bin_rank1_2 的 MSE 明显低于 None 与 WSMOTER, 且更加稳定. 这是由于 GSOGB-SMOTER 基于样本在网格区域中的分布进行粒球划分, 每个小网格的边长由数据的整体分布计算得到, 因此, 特征噪声样本的引入不易改变各小网格的类别以及粒球填充度, 对粒球划分的整体过程影响较小. 由图 7 可见, 随着重叠样本比例的增加, 3 种方法在 bin_rank1_2 的 MSE 均呈现上升趋势. 然而, GSOGB-SMOTER 的 MSE 明显低于 None 与 WSMOTER. 这是由于当数据发生重叠时, 特征相似但是目标值不同的样本相互交叠, 固然会导致模型的性能降低. WSMOTER 基于样本目标值选取邻居, 当重叠样本较多时, 易导致选取特征差异较大的邻居进而合成噪声样本. 对于 GSOGB-SMOTER 而言, 当重叠区域内的常见样本与稀有样本目标值相近时, 粒球稀有度rar(GB)能够控制在 1.0 左右, 过采样的数量显著少于粒球内的样本数量, 数据重叠的影响得到了缓和; 当常见样本与稀有样本目标值相差较大时, 由于将过采样的范围控制在样本的邻域范围内, 并基于样本的稀有程度来选取种子样本及其邻居, 从而大幅降低了常见样本参与过采样的可能性, 在一定程度上限制了重叠样本的干扰. 因此, 在处理含特征噪声样本或重叠样本的数据集时, GSOGB-SMOTER 能够展现出更优越的稳定性和鲁棒性.

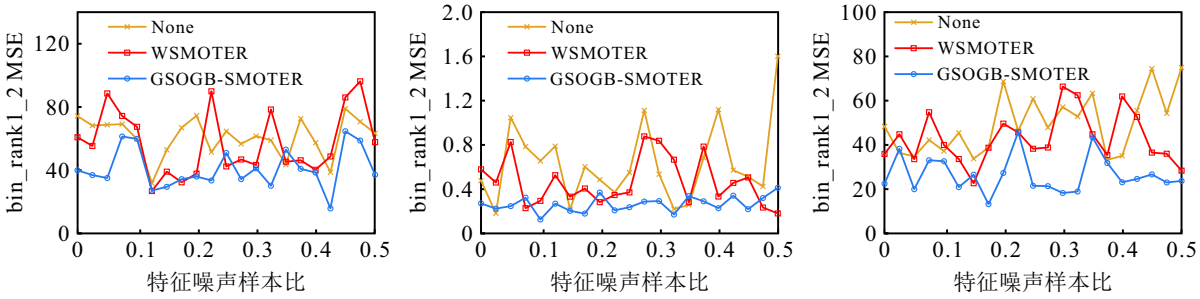
在本文对比实验中发现, SMOTER 和 SMOGN 在处理超过 1 万条样本规模数据的耗时较长, 特别是在处理高达 18 万条样本规模的DS_{22}(gait) 数据集时, 其运行时间超过 14 h, 不利于算法的实际应用, 这主要是由于它们需要在整个特征空间中计算样本间的距离. 相比之下, G-SMOTER、ACOS 与 GSOGB-SMOTER 只需要计算局部区域内的样本间距离,

表6 DTR 模型下不同排名分区的 MSE

数据集	分区	未采样	RO	WERCS	SMOTER	SMOGN	G-SMOTER	ACOS	WSMOTER	GSOGB-SMOTER
DS_{13}	bin_rank1	55225.000	55225.000	77841.000	24025.000	55225.000	55225.000	77841.000	55225.000	**17956.000**
	bin_rank2	156644.500	156644.500	156644.500	105230.500	**27630.500**	197557.000	151317.866	129050.000	37694.500
	bin_rank3	14956.000	15206.847	14956.000	17067.500	17018.500	25101.889	**5876.616**	12329.556	10220.778
	bin_rank4	6097.066	7355.485	6675.361	7504.563	8022.500	6787.656	**5762.861**	6384.566	6654.906
	bin_rank5	1257.313	1311.352	1404.617	1303.278	1908.871	1396.348	**1251.077**	1629.099	1388.302
	all_bins	2868.599	3887.568	4068.021	3355.944	3266.639	4662.606	3402.613	3704.518	**2431.907**
DS_{14}	bin_rank1	112.850	97.240	85.600	141.080	**66.550**	89.630	85.950	91.400	76.670
	bin_rank2	30.010	34.970	35.860	27.050	70.090	**19.640**	30.690	36.670	26.280
	bin_rank3	11.900	11.510	17.170	14.210	15.100	14.300	13.190	14.880	**7.500**
	bin_rank4	26.700	21.970	21.270	39.570	55.420	32.360	22.180	26.660	**20.830**
	bin_rank5	14.810	18.830	14.800	15.570	20.090	14.280	21.760	20.070	**14.190**
	all_bins	**20.010**	24.950	23.290	30.800	33.970	24.590	26.020	27.500	20.710
DS_{15}	bin_rank1	0.244	0.188	0.205	6.353	9.037	0.260	0.177	0.178	**0.164**
	bin_rank2	0.189	0.196	0.147	2.243	1.996	0.150	0.177	0.151	**0.119**
	bin_rank3	0.054	0.054	0.062	0.049	0.334	0.056	0.044	0.041	**0.040**
	bin_rank4	0.017	0.016	0.024	0.017	0.035	0.019	**0.015**	0.020	0.018
	bin_rank5	**0.010**	0.011	0.011	0.011	0.020	0.012	**0.010**	0.012	0.011
	all_bins	**0.034**	0.047	0.047	0.650	0.456	0.046	0.042	0.042	0.036
DS_{16}	bin_rank1	0.639	0.329	0.617	6.757	0.679	0.543	0.316	0.848	**0.312**
	bin_rank2	0.422	0.347	0.535	21.279	0.335	0.307	0.393	0.486	**0.248**
	bin_rank3	0.286	0.256	0.257	2.594	0.332	0.319	0.238	0.302	**0.203**
	bin_rank4	**0.080**	0.095	0.090	0.088	0.140	0.083	0.089	0.098	0.084
	bin_rank5	**0.028**	0.037	0.034	0.033	0.041	0.032	0.032	0.032	0.032
	all_bins	0.097	0.096	0.114	1.445	0.133	0.102	0.091	0.115	**0.082**
DS_{17}	bin_rank1	3091.330	3047.670	2883.590	3184.830	3184.830	3184.830	3184.830	2925.580	**2746.470**
	bin_rank2	3333.050	**3097.490**	3371.010	3366.820	3366.820	3366.820	3366.820	3332.680	3116.220
	bin_rank3	3648.930	3747.040	3643.240	4615.410	4918.720	4615.410	4615.410	3642.430	**3530.430**
	bin_rank4	3554.450	3583.030	6964.900	3277.390	110456.570	3994.780	6753.230	3562.560	**2655.530**
	bin_rank5	5758.040	5761.270	6648.430	6191.140	20944.810	5685.130	6190.310	5612.570	**5076.270**
	all_bins	**4033.020**	5034.080	6270.770	5320.540	35291.790	5103.470	5910.260	4937.970	4392.470
DS_{18}	bin_rank1	**522.833**	**522.833**	**522.833**	1677.833	4608.000	**522.833**	3114.833	**522.833**	**522.833**
	bin_rank2	726.563	737.563	588.788	785.025	753.725	650.625	620.038	786.625	**464.213**
	bin_rank3	287.407	270.888	429.594	204.448	162.571	164.721	**112.324**	316.201	166.330
	bin_rank4	36.061	39.216	32.000	38.566	307.988	32.082	**27.938**	39.059	30.860
	bin_rank5	11.159	8.435	12.887	13.568	58.340	11.298	9.365	9.322	**7.796**
	all_bins	**25.523**	40.792	43.555	45.649	164.733	34.155	38.203	43.845	27.310
DS_{19}	bin_rank1	–	–	–	–	–	–	–	–	–
	bin_rank2	4.679	4.678	**4.667**	5.953	4.906	5.221	4.987	**4.667**	**4.667**
	bin_rank3	0.388	0.388	**0.275**	0.442	0.352	0.368	0.353	0.395	0.320
	bin_rank4	0.018	0.025	0.019	0.031	0.035	**0.017**	**0.017**	**0.017**	**0.017**
	bin_rank5	0.023	0.023	0.024	0.026	0.046	**0.022**	0.024	0.023	0.024
	all_bins	0.024	0.027	0.025	0.032	0.044	**0.023**	0.024	0.024	0.024
DS_{20}	bin_rank1	–	–	–	–	–	–	–	–	–
	bin_rank2	47.267	43.331	37.834	75.222	79.692	30.298	28.740	36.474	**23.137**
	bin_rank3	0.085	0.079	20.578	20.569	0.071	0.089	18.629	19.573	**0.067**
	bin_rank4	25.015	25.563	24.361	89.039	94.221	23.193	23.096	24.293	**22.605**
	bin_rank5	10.055	**10.030**	10.342	14.375	18.939	10.308	10.348	10.077	10.198
	all_bins	**11.478**	12.542	13.306	27.362	31.445	12.222	12.961	13.052	11.979
DS_{21}	bin_rank1	856.369	942.839	**803.064**	851.500	867.016	876.912	809.776	862.190	844.030
	bin_rank2	348.877	384.414	348.043	400.885	332.448	374.745	367.007	408.767	**329.244**
	bin_rank3	157.198	154.743	142.565	138.854	135.142	159.751	142.933	150.898	**131.089**
	bin_rank4	53.950	63.212	63.475	51.037	**50.131**	55.506	61.871	61.232	53.268
	bin_rank5	10.242	11.276	12.471	11.774	16.540	10.692	11.788	11.559	**10.086**
	all_bins	**19.501**	22.706	22.904	21.761	25.560	21.435	22.364	22.533	19.888
DS_{22}	bin_rank1	5.893	5.053	5.290	运行超时	运行超时	265.702	**4.572**	207.843	5.711
	bin_rank2	14.553	13.673	14.800			107.809	13.122	91.287	**13.014**
	bin_rank3	2.361	2.294	2.436			39.688	2.442	88.041	**2.211**
	bin_rank4	3.108	3.230	3.339			62.950	**2.985**	29.267	3.003
	bin_rank5	**1.661**	1.773	2.077			36.673	1.731	28.020	1.668
	all_bins	2.715	2.689	2.959			53.901	**2.629**	41.217	3.178

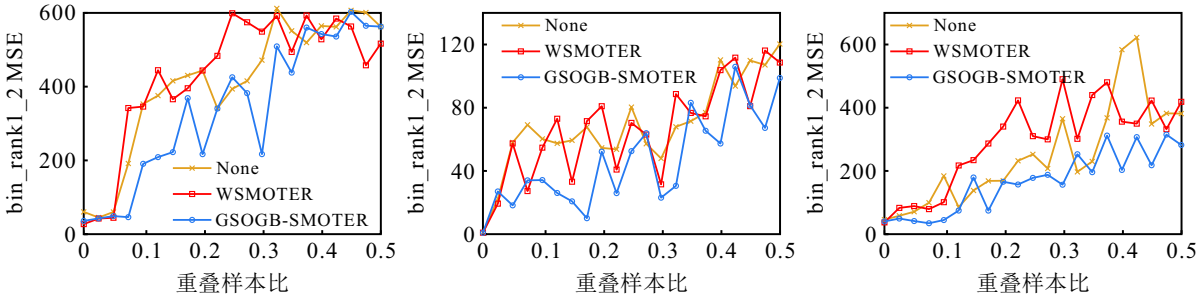
表7　不平衡回归算法在不同排名分区的 **Rank**

数据集分区	未采样	RO	WERCS	SMOTER	SMOGN	G-SMOTER	ACOS	WSMOTER	GSOGB-SMOTER
bin_rank1	4.875	4.125	3.500	6.625	6.250	5.125	4.125	4.500	**1.750**
bin_rank2	5.100	5.000	4.800	7.000	5.800	4.800	4.400	5.500	**1.300**
bin_rank3	4.900	4.500	5.500	6.500	5.800	6.300	3.700	5.600	**1.500**
bin_rank4	3.800	5.800	5.700	5.700	8.100	4.700	3.000	5.100	**2.300**
bin_rank5	**2.400**	3.900	5.900	6.200	8.800	4.200	4.500	4.500	2.500
all_bins	**1.700**	5.000	6.200	7.200	8.100	4.500	4.000	5.300	2.100



(a) DS_{14}(boston)　(b) DS_{16}(treasury)　(c) DS_{20}(cpuAct)

图6　引入特征噪声样本的训练集对于采样算法的影响



(a) DS_{14}(boston)　(b) DS_{16}(treasury)　(c) DS_{20}(cpuAct)

图7　引入重叠样本的训练集对于采样算法的影响

WSMOTER 则基于样本目标值寻找邻居, 从而有效降低了耗时. GSOGB-SMOTER 需要耗费一定时间进行网格自组织粒球化, 当样本规模不大时, 时间消耗主要来源于粒球化阶段, 因此, 耗时会略高于 G-SMOTER、ACOS 与 WSMOTER; 然而, 当样本规模达到 1 万以上时, GSOGB-SMOTER 可通过粒球化降低样本规模, 使其处理效率逐渐优于其他算法. 考虑到大规模数据集由于数据量大和来源多样而易引入噪声, 无论是处理大规模数据集的时间消耗, 还是在噪声环境下的鲁棒性, GSOGB-SMOTER 均更适合处理较大规模的不平衡回归数据.

4　结　论

不平衡数据在现实世界中广泛存在, 不平衡数据回归是不平衡数据处理中的一个难点. 本文针对不平衡回归采样算法在处理大规模数据集时效率不高这一问题, 通过改进原有的粒球模型, 提出了用于处理回归数据集的网格自组织粒球模型 GSOGB; 在此基础上, 提出了一种新的不平衡回归算法 GSOGB-SMOTER. 在 12 个分类数据集上验证了 GSOGB 的有效性, 在 10 个不平衡回归数据集上的实验结果表明, 相对于文献中的多种算法, GSOGB-SMOTER 具有较高的回归预测精度和鲁棒性, 且能够快速处理较大规模的不平衡数据的回归任务. 但是, 网格自组织粒球模型 GSOGB 在处理高维数据时, 易导致样本间距离度量不准确, 且存在较高的计算复杂性; 此外, 所提出 GSOGB-SMOTER 算法只能处理单目标的不平衡数据回归. 因此, 研究不平衡高维数据与多标签不平衡数据的回归算法将是下一步的研究工作.

参考文献 (References)

[1] 李艳霞, 柴毅, 胡友强, 等. 不平衡数据分类方法综述[J]. 控制与决策, 2019, 34(4): 673-688.
(Li Y X, Chai Y, Hu Y Q, et al. Review of imbalanced data classification methods[J]. Control and Decision, 2019, 34(4): 673-688.)

[2] Ribeiro R P, Moniz N. Imbalanced regression and extreme value prediction[J]. Machine Learning, 2020, 109(9): 1803-1835.

[3] Avelino J G, Cavalcanti G D C, Cruz R M O. Resampling strategies for imbalanced regression: A survey and empirical analysis[J]. Artificial Intelligence Review, 2024, 57(4): 82.

[4] Xia S Y, Liu Y S, Ding X, et al. Granular ball computing classifiers for efficient, scalable and robust learning[J].

Information Sciences, 2019, 483: 136-152.
[5] Branco P, Torgo L, Ribeiro R P. SMOGN: A pre-processing approach for imbalanced regression[C]. Proceedings of the 1st International Workshop on Learning with Imbalanced Domains: Theory and Applications. New York, 2017: 36-50.
[6] Branco P, Torgo L, Ribeiro R P. Rebagg: Resampled bagging for imbalanced regression[C]. Proceedings of the 2nd International Workshop on Learning with Imbalanced Domains: Theory and Applications. Dublin, 2018: 67-81.
[7] Steininger M, Kobs K, Davidson P, et al. Density-based weighting for imbalanced regression[J]. Machine Learning, 2021, 110(8): 2187-2211.
[8] Devi D, Biswas S K, Purkayastha B. Correlation-based oversampling aided cost sensitive ensemble learning technique for treatment of class imbalance[J]. Journal of Experimental & Theoretical Artificial Intelligence, 2022, 34(1): 143-174.
[9] 刘宁, 朱波, 阴艳超, 等. 一种混合CGAN与SMOTEENN的不平衡数据处理方法[J]. 控制与决策, 2023, 38(9): 2614-2621.
(Liu N, Zhu B, Yin Y C, et al. An imbalanced data processing method based on hybrid CGAN and SMOTEENN[J]. Control and Decision, 2023, 38(9): 2614-2621.)
[10] Torgo L, Ribeiro R P, Pfahringer B, et al. SMOTE for regression[C]. Portuguese Conference on Artificial Intelligence. Heidelberg, 2013: 378-389.
[11] Camacho L, Douzas G, Bacao F. Geometric SMOTE for regression[J]. Expert Systems with Applications, 2022, 193: 116387.
[12] Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: Synthetic minority over-sampling technique[J]. Journal of Artificial Intelligence Research, 2002, 16: 321-357.
[13] Douzas G, Bacao F. Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE[J]. Information Sciences, 2019, 501: 118-135.
[14] Branco P, Torgo L, Ribeiro R P. Pre-processing approaches for imbalanced distributions in regression[J]. Neurocomputing, 2019, 343: 76-99.
[15] Song X Y, Dao N, Branco P. Distsmogn: Distributed smogn for imbalanced regression problems[C]. Proceedings of the 4th International Workshop on Learning with Imbalanced Domains: Theory and Applications. New York, 2022: 38-52.
[16] Li Z Z, Huang N, Yi L Z, et al. Affine combination-based over-sampling for imbalanced regression[J]. Journal of Chemometrics, 2024, 38(3): e3537.
[17] Camacho L, Bacao F. WSMOTER: A novel approach for imbalanced regression[J]. Applied Intelligence, 2024, 54(19): 8789-8799.
[18] Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C. DBSMOTE: Density-based synthetic minority over-sampling technique[J]. Applied Intelligence, 2012, 36(3): 664-684.
[19] Douzas G, Bacao F, Last F. Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE[J]. Information Sciences, 2018, 465: 1-20.
[20] Douzas G, Bacao F. Self-organizing map oversampling for imbalanced data set learning[J]. Expert Systems with Applications, 2017, 82: 40-52.
[21] Douzas G, Rauch R, Bacao F. G-SOMO: An oversampling approach based on self-organized maps and geometric SMOTE[J]. Expert Systems with Applications, 2021, 183: 115230.
[22] Han H, Wang W Y, Mao B H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning[C]. International Conference on Intelligent Computing. Heidelberg, 2005: 878-887.
[23] He H B, Bai Y, Garcia E A, et al. ADASYN: Adaptive synthetic sampling approach for imbalanced learning[C]. IEEE International Joint Conference on Neural Networks. Hong Kong, 2008: 1322-1328.
[24] Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C. Safe-level-SMOTE: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem[C]. Advances in Knowledge Discovery and Data Mining. Heidelberg, 2009: 475-482.
[25] Nguyen H M, Cooper E W, Kamei K. Borderline over-sampling for imbalanced data classification[J]. International Journal of Knowledge Engineering and Soft Data Paradigms, 2011, 3(1): 4-21.
[26] Xia S Y, Dai X C, Wang G Y, et al. An efficient and adaptive granular-ball generation method in classification problem[J]. IEEE Transactions on Neural Networks and Learning Systems, 2024, 35(4): 5319-5331.
[27] Köppen M. The curse of dimensionality[C]. Proceedings of the 5th Online World Conference on Soft Computing in Industrial Applications. Espoo, 2000, 1: 4-8.
[28] Hu Q H, Yu D R, Xie Z X. Neighborhood classifiers[J]. Expert Systems with Applications, 2008, 34(2): 866-876.
[29] 陈庆强, 王文剑, 姜高霞. 基于数据分布的标签噪声过滤[J]. 清华大学学报: 自然科学版, 2019, 59(4): 262-269.
(Chen Q Q, Wang W J, Jiang G X. Label noise filtering based on the data distribution[J]. Journal of Tsinghua University: Science and Technology, 2019, 59(4): 262-269.)

作者简介

胡峰 (1978−), 男, 教授, 博士, 主要研究方向为粗糙集、机器学习、数据挖掘, E-mail: hufeng@cqupt.edu.cn;

周雨龙 (2000−), 男, 硕士生, 主要研究方向为机器学习、数据挖掘, E-mail: 2495503124@qq.com;

苏祖强 (1987−), 男, 副教授, 博士, 主要研究方向为机电设备的服务安全和寿命预测、机械信号处理, E-mail: suzq@cqupt.edu.cn;

代劲 (1978−), 男, 教授, 博士, 主要研究方向为粗糙集、云模型、数据挖掘, E-mail: daijin@cqupt.edu.cn;

于洪 (1972−), 女, 教授, 博士, 主要研究方向为数据挖掘、智能信息处理、三支决策, E-mail: yuhong@cqupt.edu.cn.