中国研究生创新实践系列大赛
“华为杯”第十七届中国研究生数学建模竞赛

学　　校	上海大学
参赛队号	**20102800087**
队员姓名	**1.** 张晨
	2. 陈晓常
	3. 耿文文

中国研究生创新实践系列大赛
“华为杯”第十七届中国研究生
数学建模竞赛

题 目 对汽油辛烷值的多变量预测和优化

摘 要：

近年来，小型汽车几乎成为了每家每户的“代步工具”，在享受便捷、舒适的交通方式的同时，越来越多的汽车尾气排放对大气环境产生了恶劣的影响，成为了我们亟需解决的问题。辛烷值是衡量和硫含量是衡量汽油燃料燃烧性和清洁性的两项重要指标。因此，比较分析、挖掘提取汽油相关指标的关系，寻求控制及优化汽油含硫量及辛烷值等的方法，对于理解汽油精致工艺，保护大气环境有着重要的意义。

基于此，本文在充分基于各类数据样本，并利用 Python、MATLAB 等分析工具，综合运用了各类数据处理分析方法、RandForest 随机森林模型、BP 神经网络模型、XGBoost 模型、PSO 粒子群等算法，旨在研究汽油中辛烷值和硫含量两个重要指标与其余操作位点变量之间关联性程度，并通过数据挖掘技术分析并构建该化工问题的建模，从而依据模型研究对汽油辛烷值和硫含量的主要操作变量优化方案。

针对问题一，需要对附件三中 285 号和 313 号原始样本数据依据附件二中“样本确定方法”进行数据预处理，并将处理后的数据分别加入到附件一中相应的样本号中。本文利用 Python 软件对该两个样本进行预处理。首先依据拉依达准则去除样本数据中被认为是含有粗大误差值的坏值，即异常值；其次依据原始样本集总结出来的数据变量的操作范围，对样本数据集中的不同位点数据进行范围限制，利用最大最小限幅方法对不在范围内的位点数据进行剔除；然后统计样本数据集中的空白值，对于少部分数据为空白的位点，计算其前后两个小时的均值作为填充值；最后对处理完成的 285 号和 313 号样本数据求其位点数据上的平均值作为最终处理结果，并分别替换到附件一对应编号的样本中。

针对问题二，需要对问题一得到的总计 363 个变量进行降维处理，筛选出具有代表性、独立性且数量为 30 以内的主要操作变量。本文首先对初始数据集采用附件二中给出的处理方法对其进行异常值剔除、空白值处理以及最大最小归一化；然后基于随机森林算法构建了特征提取模型，对其中参数进行调节优化，计算其各个特征的重要性系数，依据重要性系数降序排列，选择前 28 个操作变量作为最终筛选结果，其重要性比重占 325 个样本数据集将近一半，验证了其具有良好的代表性；最后利用最大互信息系数对该 28 个变量进行了相关性分析，验证了其具有较好的独立性。

针对问题三，需要利用问题二提取的主要变量采取数据挖掘的技术建立辛烷值（RON）损失预测模型，并对该模型进行验证分析。本文分别利用 BP 神经网络算法和 XGBoost 算法建立了辛烷值 RON 的损失预测模型：首先，设计了一个 5 层的神经网络结构模型，其中，输入层共有 28 个神经元，分别对应问题二所最终得到的 28 个主要操作变量，然后依次通过隐藏层和输出层，最终得到汽油辛烷值损失和硫含量两个预测变量；其次，为了可以进行结果的横向对比，建立了基于 XGBoost 算法的预测模型；然后为了防止模型出现过拟合的情况，本文采用了交叉验证的方法对两种模型分别进行训练和测试；最后依据 MSE，

RMSE 和 MAE 三种常用的检验回归拟合度的性能系数对两个模型的预测结果进行了比较分析，发现两种模型的拟合误差分别约为 0.048 和 0.041，拟合程度较好，且 XGBoost 的拟合程度优于 BP 神经网络模型

针对问题四，需要在保证硫含量不大于 5μg/g 的前提下，利用题三中模型得到辛烷值损失降幅大于 30%的样本对应的主要变量优化后的操作条件。本文利用粒子群（PSO）优化算法对 325 个不满足损失降幅大于 30%的样本进行优化，设置辛烷值损失为自适应函数，利用粒子群的粒子在解空间追随最优粒子进行搜索最优解，在自适应函数输出变量之一硫含量小于 5μg/g 的约束下，求得辛烷值损失的最小值，若满足损失降幅大于 30%，则保存最优粒子的空间位置坐标作为 28 个变量的值，用于后续分析和研究。

第五问要求实现模型的可视化操作，显示汽油辛烷值和硫含量在操作变量的优化过程中的变化轨迹。工业装置修改操作变量只能逐步调整，遵循每个主要变量的最大最小值和 133 号样本优化前后的取值范围，按照变量的步长依次改变主要变量的取值，带入预测模型，显示辛烷值和硫含量的变化轨迹。

本文所构建的模型均进行了交叉验证和评价，综合模型的应用结果，对于各个模型的优缺点进行评价，说明了现有模型的推广性和局限性，提出了部分问题的改进建议。

目录

一、问题背景与重述

1.1 问题背景

随着汽车工业的迅速发展以及节能减排理念的推广普及，汽油消费量在快速增长，对汽油质量的要求也越来越高。因此在保证一定汽油质量标准的前提下，提高汽油机的热效率，提高汽油的清洁性以及改善其经济性迫在眉睫[1]。

辛烷值（RON）是反映评价汽油燃烧性能的一个重要指标。首先，RON 与汽油燃料的抗爆震性能相关，是表示汽油燃料抗爆震性能的条件单位[2]，在一定范围内，RON 越高，汽油的抗爆震性能越好；其次，RON 也意味着燃料的经济效益，其每降低 1 个单位，相当于损失约 150 元/吨，所以降低 RON 损失将能显著提高化工企业的经济效益；最后，如图 1 所示，对比国内外汽油标准制定规范可以看到，近些年我国所制定的汽油标准，硫含量下降幅度较大，逐渐向清洁化趋势发展，但与国外先进水平相比仍有较大的差距，例如汽油辛烷值偏低等。

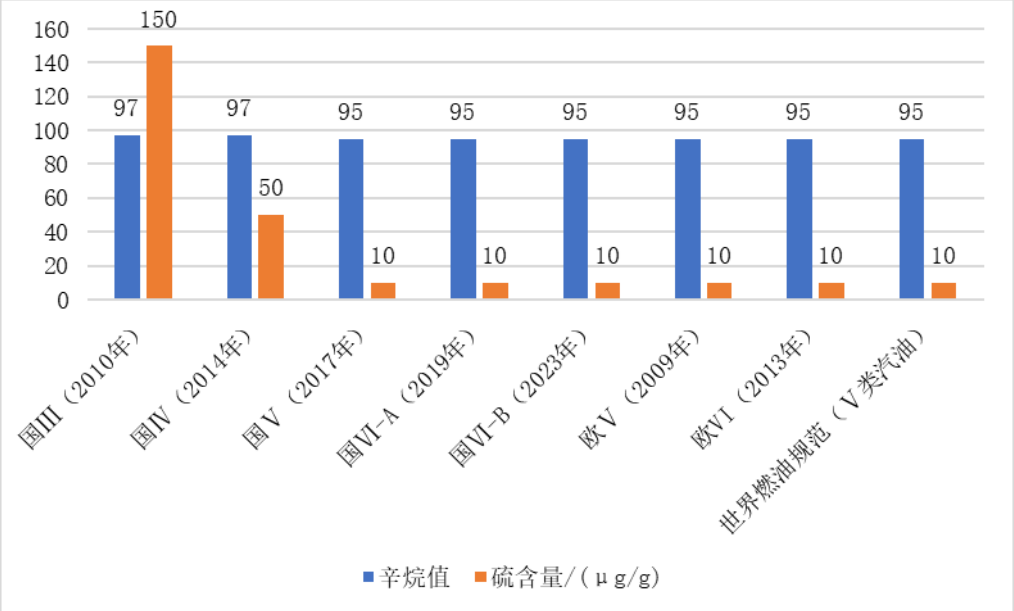

图 1 国内外汽油标准主要标准对比

然而在石油化工重油催化裂化的工艺过程中，对汽油进行脱硫和控制烯烃等举措的同时会普遍导致 RON 不同程度的损失，对汽油的抗爆性能以及石化企业的经济性能都造成了不利的影响。所以有必要利用数据挖掘等技术来对该化工过程进行建模优化，在保证脱硫效果的前提下，尽可能降低 RON 损失，旨在保护大气环境、减少排放污染的同时可以提高汽油的抗爆质量性能和化工企业的经济效益。

1.2 问题重述

依据从催化裂化汽油精制装置采集的数据样本，通过数据挖掘技术来建立汽油辛烷值（RON）损失的预测模型，并给出每个样本的优化操作条件，在保证汽油产品脱硫效果的前提下，尽量降低汽油辛烷值损失在 30%以上。

问题 1：数据处理。请你们利用题目所给的数据样本及方法对指定的数据样本进行预处理，并将处理后的数据加入到相应的样本号中。

问题 2：寻找建模主要变量。请你们根据提供的 325 个样本数据，通过降维的方法从 367 个操作变量中筛选出建模主要变量，使之尽可能具有代表性、独立性，并请详细说明

建模主要变量的筛选过程及其合理性。

问题 3：建立辛烷值损失模型。采用以上所处理好的样本和建模的主要变量，通过数据挖掘技术建立辛烷值 RON 损失预测模型，并进行模型验证。

问题 4：主要变量操作方案的优化。要求在保证产品硫含量不大于 5μg/g 的前提下，利用你们的模型获得 325 个数据样本中，辛烷值损失降幅大于 30%的样本对应的主要变量优化后的操作条件。

问题 5：模型的可视化展示。请你们对 133 号样本（原料性质、待生吸附剂和再生吸附剂的性质数据保持不变，以样本中的数据为准），以图形展示其主要操作变量优化调整过程中对应的汽油辛烷值和硫含量的变化轨迹。

二、基本假设与符号说明

2.1 基本假设

（1）假设题目附件中所有数据均是真实的；
（2）假设数据集中个别缺失数据对结果不会产生重大影响；
（3）假设操作变量的测量误差不对整体数据的评判产生影响；
（4）假设当前辛烷值不会对下一次辛烷值的测量产生影响；
（5）假设辛烷值的损失不受不在汽油精制装置历史数据中操作变量的影响。

2.2 符号说明

序号	符号	说明
1	w	网络权重
2	θ	阈值
3	n_estimators	树的棵树
4	min_samples_leaf	最小叶子权重和
5	Gini	基尼系数
6	Binary:logistic	逻辑回归
7	min_child_weight	最小叶子权重和
8	max_depth	树的最大深度
9	Gamma	节点分裂所需最小损失函数下降值
10	Subsample	每棵树随机采样比例
11	eta	学习率
12	$\vec{x}_i$	粒子在 D 维空间位置
13	$\vec{p}_i$	粒子在 D 维空间位置
14	$\vec{p}_g$	全局最优位置
15	I	主元变量
16	L	预测模型输出

三、全文技术路线图

本文技术路线如下：

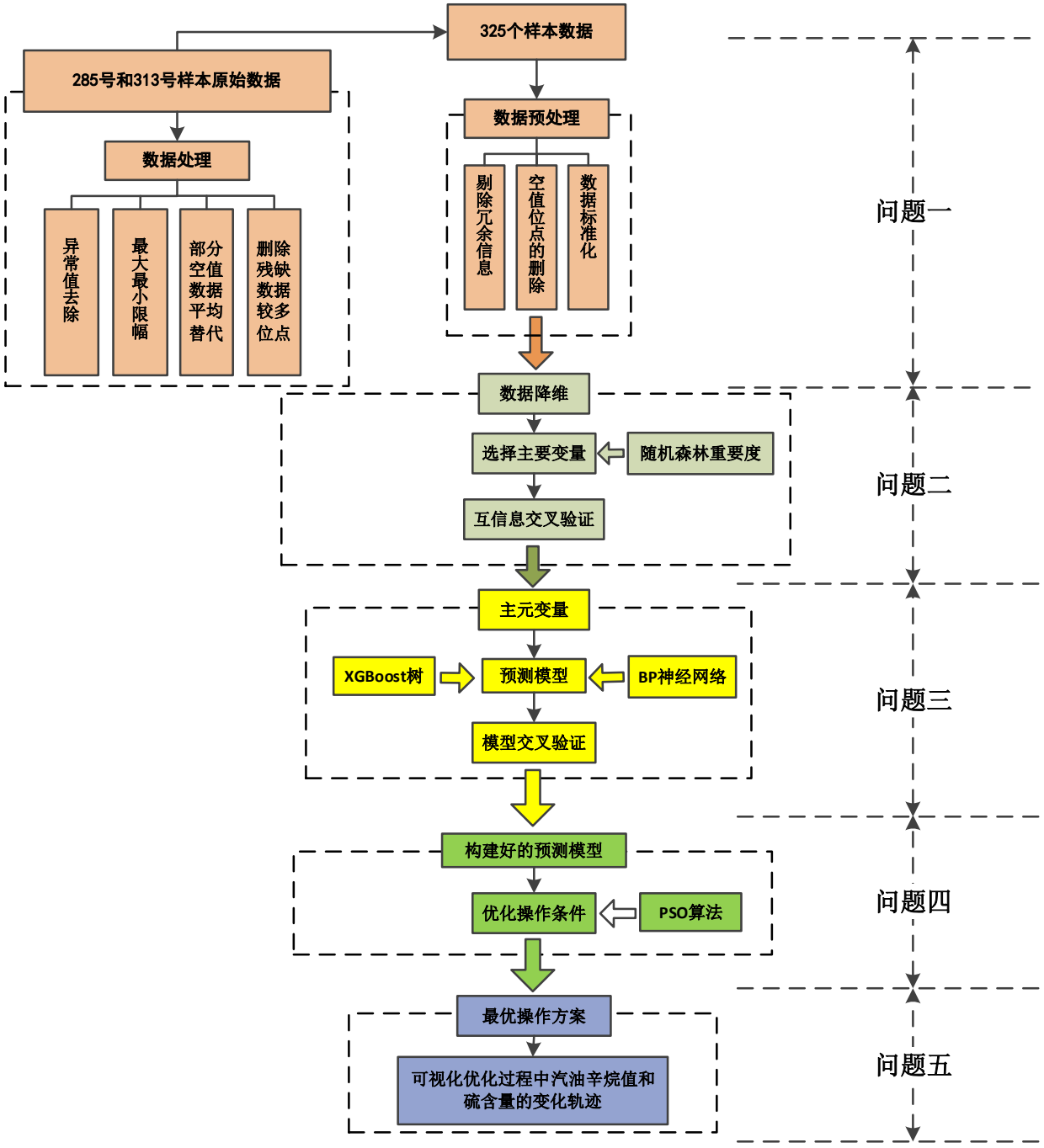

图 2 全文技术路线图

四、问题一的建模与求解

4.1 问题分析

在题目所提供的附件一数据集中，由于数据采集过程中存在传感器故障等外界干扰因素，该数据集存在部分位点数据缺失、异常等情况。此问题便是需要我们对附件一中的 285

号数据样本和 313 号数据样本依据附件二所提供的“样本确定方法”进行残缺数据、空值、异常数据等预处理，然后将处理好的两行样本数据分别加入到附件一对应的样本号中，此问题的思路流程图如图 3 所示：

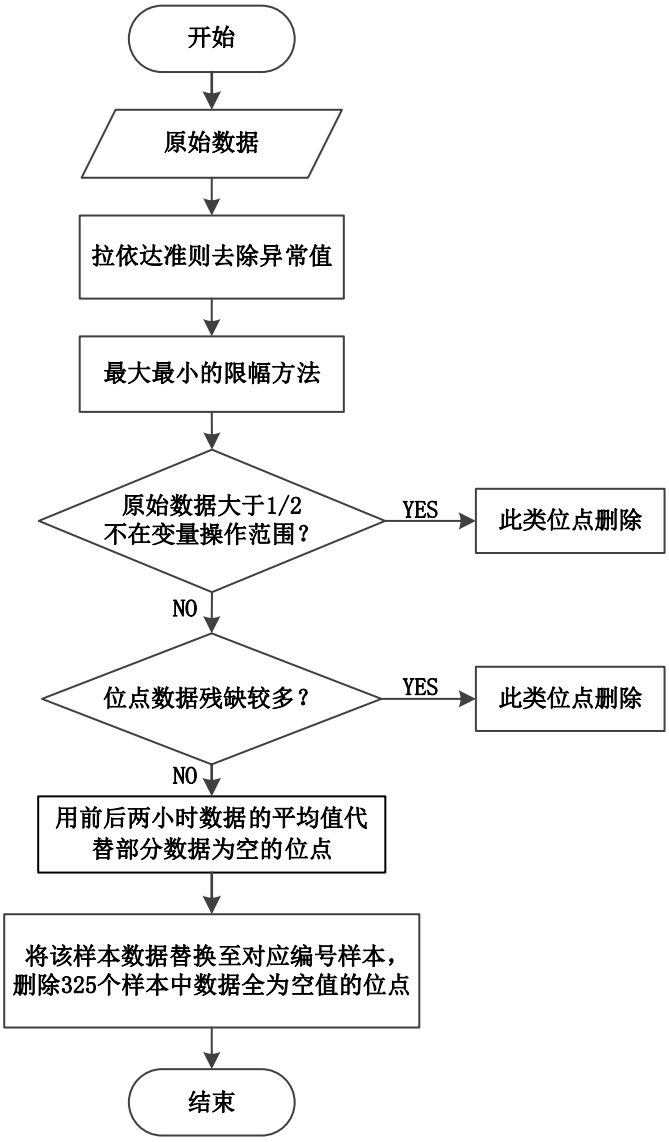

图 3 问题一数据处理流程图

4.2 原始数据预处理

4.2.1 无关信息剔除处理

数据挖掘需要我们按照既定的目标，通过对大量统计数据的探索，探究出数据间深层的规律，通过模型化的方法对数据进行建模。为了便于本问题的数据分析，针对附件三中提供的 285 号和 313 号样本原始数据进行预处理。本小节首先剔除了对本问题处理无价值意义的解释信息等栏目，其余数据作为有效信息值进行后续处理。

4.2.2 异常值处理

异常值是数据中存在的离群点，是数据集中不合理信息的表现。通常可通过均方差分析、回归分析、聚类分析方法或 3σ 检测法等方式挖掘出异常值，其常用的处理方式有采用平均值修正、删除异常值记录、将异常值表示为缺失值并按照缺失值处理等，对异常值

的有效剔除可提升后续数据处理的预测精度。

本文依据附件二中数据整定的相关要求，采用拉依达准则(3σ准则)对异常值进行去除。其基本原理是依据每个样本点与样本中心的距离来判断是否将可疑的数据从该组数据中剔除。当数据满足正态分布时，由图 3 所示，在围绕均值的 3σ范围内的，置信区间可达到 99.7%，基本囊括了所有的显著数据；对于非正态分布的情形，依据切比雪夫不等式的结果，3σ范围限制至少也将囊括 88.8%的数据，所以采用 3σ准则可去除远离均值点位的离群点数据，提升大样本条件下数据的可信度。

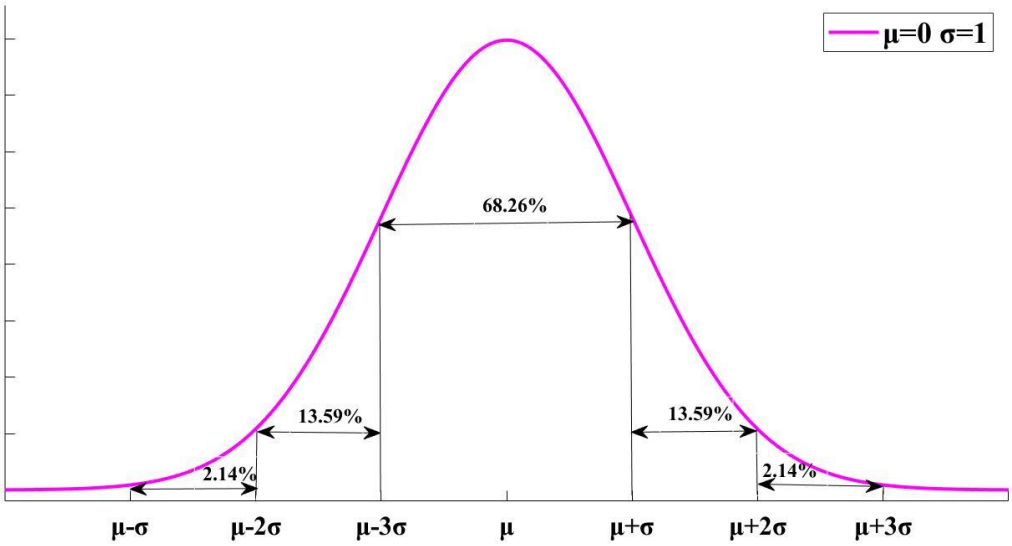

图 4 正态分布下 3σ准则

针对该问题，本文依据拉依达准则（3σ准则）剔除了被认为是含有粗大误差值的坏值，其余数据作为有效信息值进行后续处理。具体步骤为：

（1）依据下式计算各位点列数据的算术平均值和剩余误差；

$$x=\frac{x_1+x_2+\ldots+x_n}{n}$$

$$v_i=x_i-x$$

其中，$i=1,2,\ldots,n$。

（2）依据贝塞尔公式计算标准误差σ；

$$\sigma=\left[\frac{1}{n-1}\sum_{i=1}^{n}v_i^2\right]^{\frac{1}{2}}=\left\{\left[\sum_{i=1}^{n}x_i^2-\left(\sum_{i=1}^{n}x_i\right)^2/n\right]/(n-1)\right\}^{\frac{1}{2}}$$

（3）判断位点测量数据是否满足以下不等式，若满足，则认为该数据是含有粗大误差的坏值，即剔除该值，反之，则保留。

$$|v_b|=|x_b-x|>3\sigma \qquad (1\le b\le n)$$

据上述方法统计，313 号样本的 354 个位点数据中共有 30 个位点的数据中存在数据量为 1 的异常值，10 个位点的数据中存在数据量为 2 的异常值，1 个位点的数据中存在数据量为 3 的异常值，需将此类异常值数据加以剔除；285 号样本数据中不存在异常值，无需做数据剔除操作。对 285 号样本数据及 313 号样本原始数据异常值统计的结果如图 5 所示：

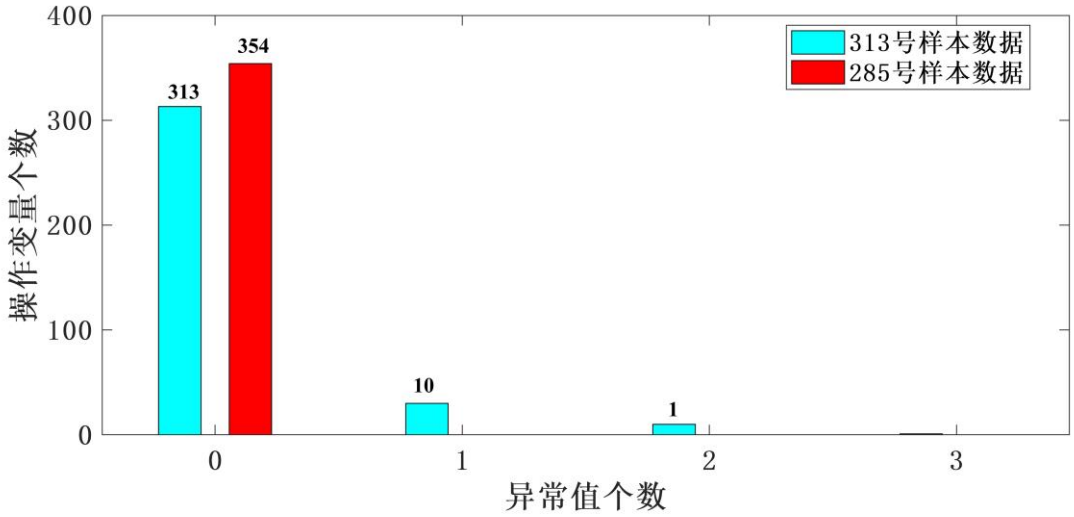

图 5　285 号样本和 313 号样本数据异常值统计

由图 6-7 所示为 313 号样本中数据位点 S-ZORB.LT_1501.DACA 的异常值处理前后的分布情况，图中蓝色和黄色线条分别代表$3\sigma+\mu$和$3\sigma-\mu$，依据3σ原则进行了该异常值的剔除。

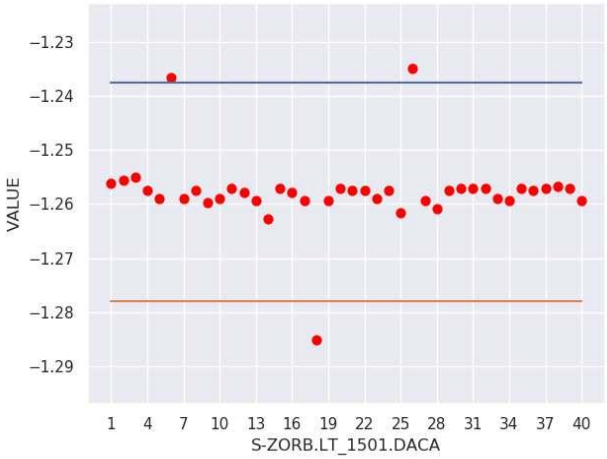

图 6　313 号样本数据异常值处理前

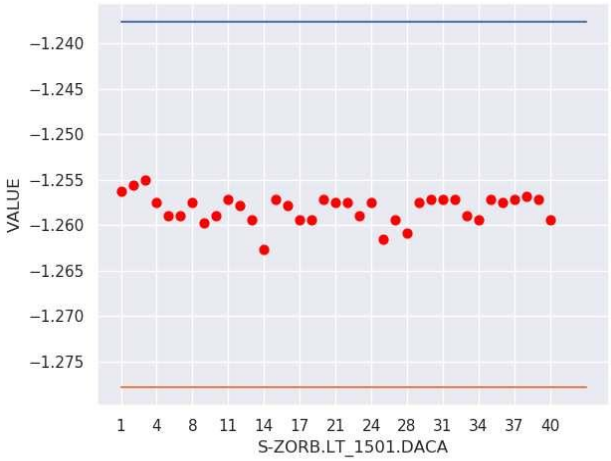

图 7　313 号样本数据异常值处理后

4.2.3　最大最小限幅处理

本文依据原始数据变量的操作范围对各个位点取值的限制，采用最大最小的限幅方法剔除不在操作范围内的样本数据值。针对某个具体的位点数据，将该列测量数据中不在范围中的数据按照 max－min 的思想，即将低于最小操作范围的数据限幅为最小操作范围值，将超过最大操作范围的测量数据限幅为最大允许值。需要注意的是，若某一操作变量的测量值中有大于二分之一的数据不在操作范围的范围限制内，则认为该测量数据不可信，将该位点剔除。

如表 1，2 分别为 285 号样本和 313 号样本中需要剔除的位点信息：

表 1　285 号样本剔除位点统计

编号	285 号样本位点	操作范围	位点剔除原因
1	S-ZORB.SIS_LT_1001.PV	40-80%	超出操作范围
2	S-ZORB.AI_2903.PV	0.5-3%	超出操作范围
3	S-ZORB.FT_1204.TOTAL	45000-2500000	超出操作范围

表 2　313 号样本剔除位点统计

编号	313 号样本位点	操作范围	位点剔除原因
1	SZORB.AT_5201.PV	0-5μg/g	超出操作范围
2	SZORB.SIS_LT_1001.PV	40-80%	超出操作范围
3	S-ZORB.AI_2903.PV	0.5-3%	超出操作范围
4	SZORB.FT_1204.TOTAL	45000-2500000	超出操作范围

由表 1 可知，对于 285 号样本，总共有 3 个位点需要进行剔除操作，位点的剔除均是由大量测量值不在操作范围内引起的，该类位点数据属于无效数据，符合数据剔除要求。对于 313 号样本中需要剔除的位点信息。由表 1 可知，总共有 4 个位点需要进行剔除操作，位点的剔除原因是测量数据超出 1/2 不在允许操作范围内和位点测量值均不在操作范围内。

4.2.4　数据空值及数据残缺位点的处理

在经过无用信息的剔除，利用 3σ 准则处理异常值以及对数据进行最大最小限幅后，本文接下来对部分数据为空值的位点进行了处理，其中，对于残缺位达到总数一半以上的位点进行剔除，对于仅有少部分数据为控制的位点，主要是通过计算前后两小时数据的平均值来替代。针对此问题，本文了调用了 Python 的 pandas 工具包对仅包含部分时间点的位点数据进行提取分析，剔除位点数据残缺较多的无效数据。

对 285 号样本及 313 号样本在不同位点下数据的缺失数量进行统计，发现对于数据空值位点，仅有小部分的数据需要进行平均值替代；并且在对 285 号以及 313 号样本数据相关位点数据进行残缺数据量统计时未发现有较多数据残缺的位点信息，因此无需依据此类限制进行位点删除。

4.2.5　处理结果分析

综上，依据 4.2.1 至 4.2.4 小节对 285 号样本和 313 号样本的数据处理结果如图 8、9 所示：

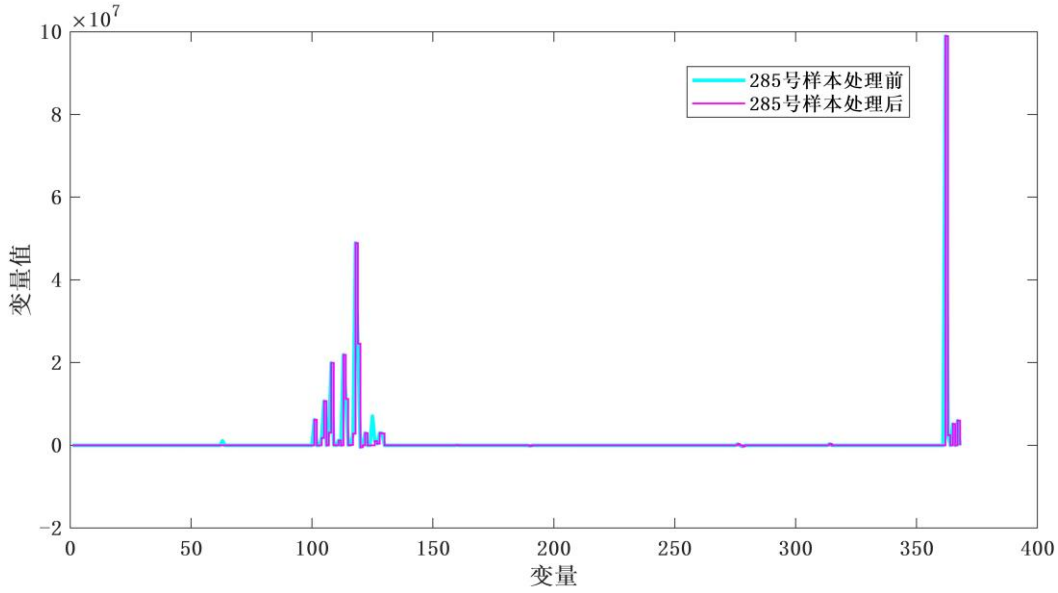

图 8　285 号样本数据处理前后对比

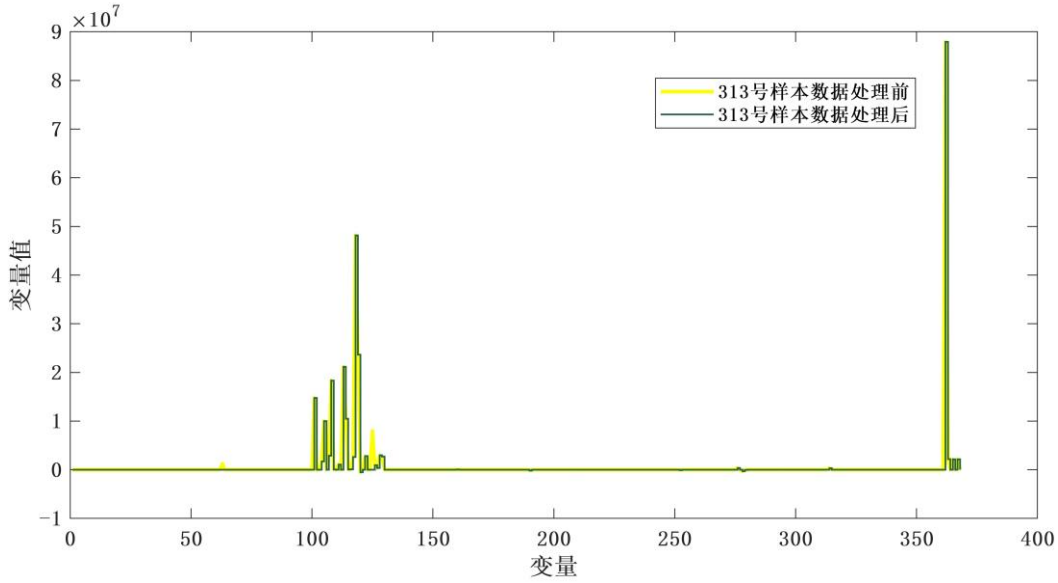

图 9　313 号样本数据处理前后对比

由图 8-9 可知，285 号样本数据和 313 号样本数据处理前后变量值改动不大，仅有小部分变量在处理前后其值发生了改变。

最后，依据题目要求，将处理后的两条样本数据按照取平均值的方法替换到“附件一：325 个样本数据.xlsx”中对应的编号数据中。

4.3 本章小结

针对本问题，本文首先针对附件三中的 285 号和 313 号样本原始数据剔除了对本问题处理无价值意义的解释信息等栏目；然后依据拉依达准则（3σ 准则）剔除了被认为是含有粗大误差值的坏值，即异常值；而后依据总结出来的原始数据变量的操作范围，对各个位点进行取值范围的限制，采用最大最小的限幅方法剔除掉了不在经验操作范围内的位点数据值；接着分别对 285 号和 313 号样本数据中的缺失值、空白值进行了统计，对于仅有少部分数据为空值的位点，利用其前后两个小时数据的均值进行填充，同时，由于未发现

残缺数据较多的位点变量，因此没有进行此类位点的剔除；最后，将经过以上数据预处理后的 285 号和 313 号样本数据在每个位点上求均值作为最后的结果值，并将其分别替换到了附件一对应编号的样本中。

五、问题二的建模与求解

5.1 问题分析

针对问题二，本文需要对问题一处理得到的 350 个操作变量以及 13 个性质变量，共计 363 个变量进行降维，从而筛选出具有代表性、独立性，且数量为 30 个以下的建模主要变量。为解决这个问题，本文认为首先应该对 325 个样本数据进行数据预处理，包括异常值剔除、空白缺失值处理以及进行最大最小归一化，处理方法本文仍采用附件二中提供的方法；然后本文建立基于随机森林算法建立特征提取模型，用于进行主要变量的筛选工作；最后对筛选出的主要变量进行重要性系数排序以及基于最大互信息系数方法对多变量进行相关性分析，衡量其代表性和独立性，进而对筛选出的主要变量进行性能评估。

5.2 模型建立

5.2.1 数据处理

本小节对问题一最终得到的 325 个样本数据集依次进行异常值剔除、缺失值填充以及最大最小归一化处理。其中，首先依据拉依达准则（3σ 准则）剔除了被认为是含有粗大误差值的异常值；其次对缺失值进行了平均值填充；最后依据下式进行最大最小归一化处理，将 325 个样本数据的值均转换到[0,1]的范围内。

$$x^{'}=\frac{x-\min(x)}{\max(x)-\min(x)}$$

5.2.2 随机森林主要变量提取模型

随机森林（Random Forest，简称 RF）算法是由 Breiman 在 Bagging 算法之后提出的另一种利用若干个决策树学习器进行分类预测以及回归预测的机器学习算法。随机森林算法构成的基本单元便是决策树，其中，每棵决策树的建立方法为：从样本集数据集中有放回地、随机地进行 m 个样本的采样选取，并在训练集中随机地选择 k 个属性，在各个节点处使用最优的属性进行树分割，从而便建立了一棵 CART 决策树。随机森林便是将若干个子模型的投票结果进行平均作为最终模型的输出，主要思想如图 5 所示：

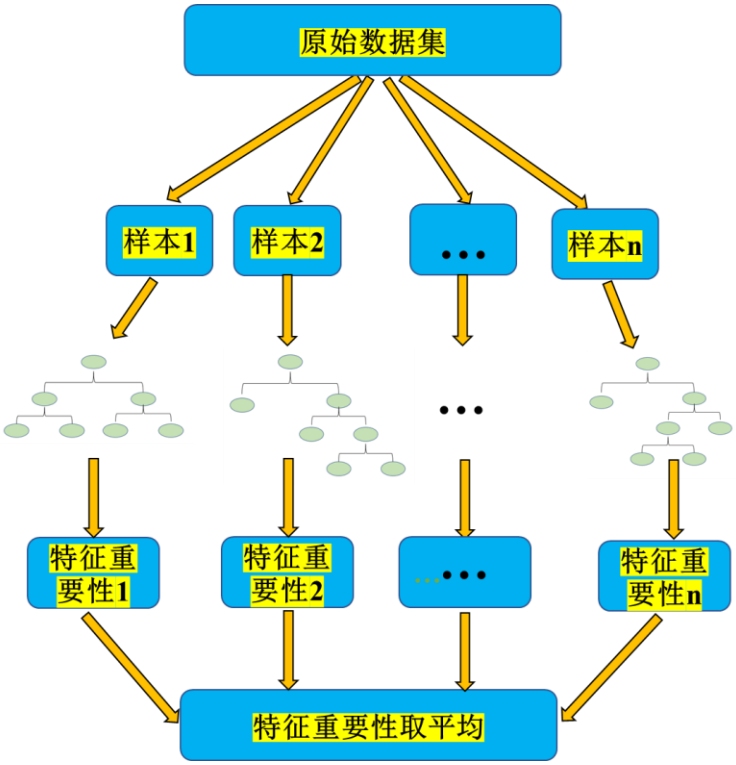

图 10 随机森林特征提取主要思想图

1）随机森林重要性系数计算

基于随机森林算法进行特征重要性评估通常是将所有决策树重要性求平均值作为最终结果，而在决策树中，本文采用以下步骤计算特征的重要性：

首先，计算某一节点 k 的重要性为：

$$n_k = \omega_k * G_K - \omega_{left} * G_{left} - \omega_{right} * G_{right}$$

其中，ω_k，ω_{left}，ω_{right} 分别为节点 k 以及其左右子节点中训练样本数目与总训练样本数目的比例；G_K，G_{left}，G_{right} 分别为节点 k 以及其左右子节点的不纯度。

其次，计算某一特征的重要性为：

$$f_i = \frac{\sum_{j \in nodes\_split\_on\_feature\_i} \eta_j}{\sum_{k \in all\_nodes} n_k}$$

最后，对每一特征的重要性进行标准化：

$$f_{ni} = \frac{f_i}{\sum_{j \in all\_features} f_j}$$

2）随机森林参数调优

本文所利用的随机森林算法中可调的有两个参数，分别为：min_samples_leaf 和 n_estimators，对其调节结果如图 8 所示：

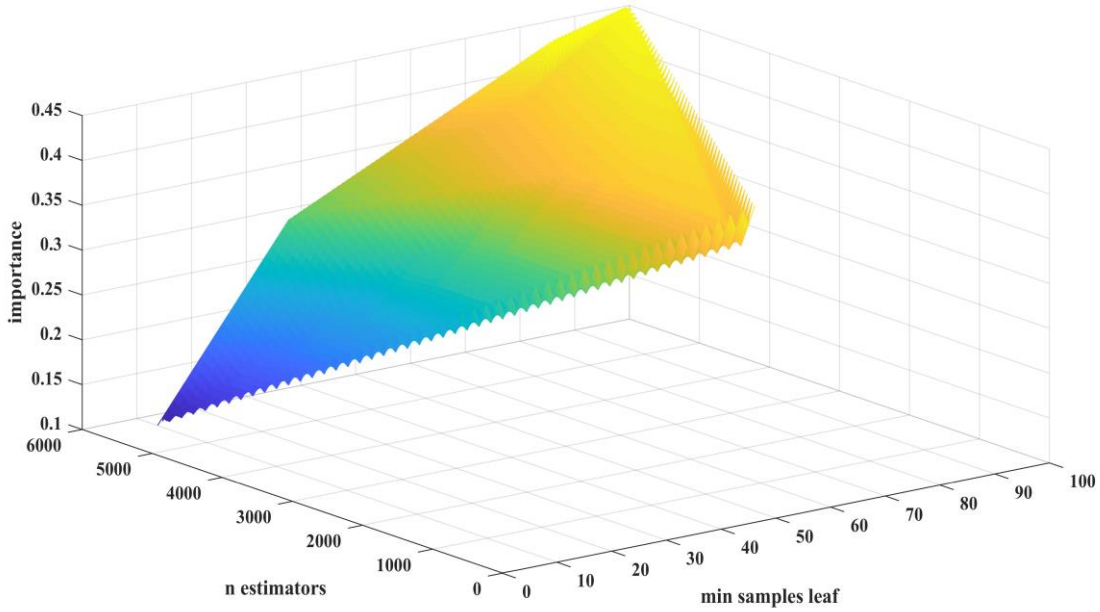

图 11 随机森林特征参数调优

由上图可以看出，当 min_samples_leaf 和 n_estimators 分别趋近于 80 和 5000 时，重要性系数 importance 不再变优，因此本文设置 min_samples_leaf 和 n_estimators 分别为 80 和 5000

综上，建立基于随机森林算法的特征变量提取模型流程图如图 12 所示：

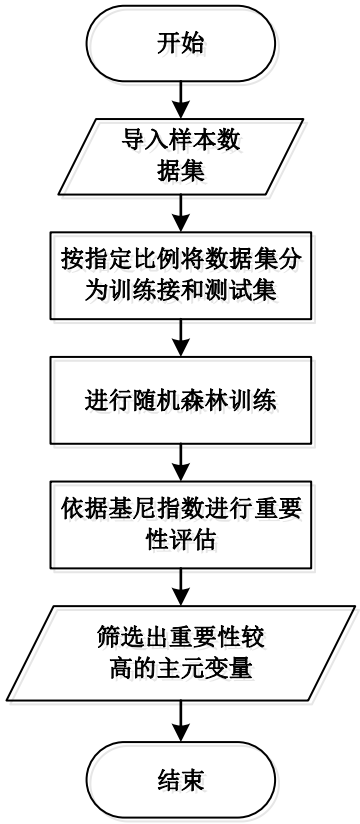

图 12 随机森林特征提取流程图

5.3 模型求解与结果分析

5.3.1 随机森林主要变量提取结果

将基于随机森林算法特征提取模型处理结果依据重要性系数从高到低进行排序，筛选出前 28 个变量作为主要变量，筛选结果如图 13 所示：

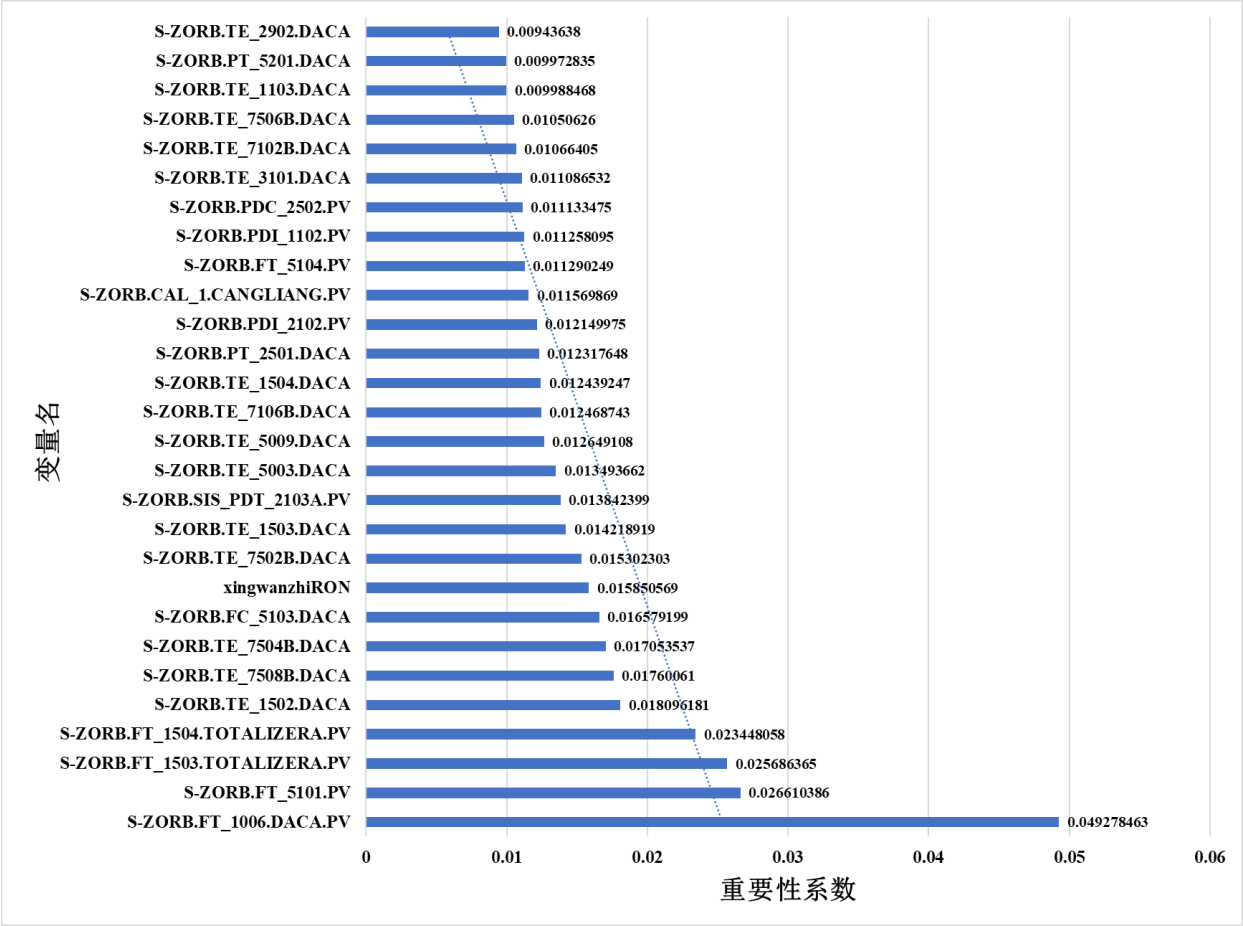

图 13 随机森林特征提取结果图

5.3.2 主要变量重要性分析

经统计，筛选出的 28 个主要变量重要性程度占全部 325 个变量的 45.39%，将近一半，因此本文认为所筛选出的 28 个主要变量有较好的代表性。

5.3.3 主要变量相关性分析

为了研究利用上述模型所提取的主要变量的独立性，本文采用最大互信息系数对该 28 个变量进行了相关性分析。其中，互信息是指两个变量之间的关联程度，相对于其它计算数据关联性的算法，互信息具有以下优点：

（1）适应性范围高，可以同时处理线性和非线性的数据；

（2）计算关联性不限于特定函数的类型，而可以广泛捕捉到整个数据样本的关联性；

（3）对于具有相同噪声水平的线性关系或是非线性关系，互信息具有相似的值，也就是说，互信息不仅可以纵向比较同一相关关联性，而且还可以横向比较不同相关关联性。

本文所采用的互信息主要计算步骤如下：

（1）对该样本数据集所画出的散点图进行网格化，并求出最大的互信息值；

（2）对最大的互信息值进行归一化；

（3）选择不同尺度下互信息的最大值作为互信息系数值。

利用以上步骤，经最大互信息关联性分析结果如图 14 所示：

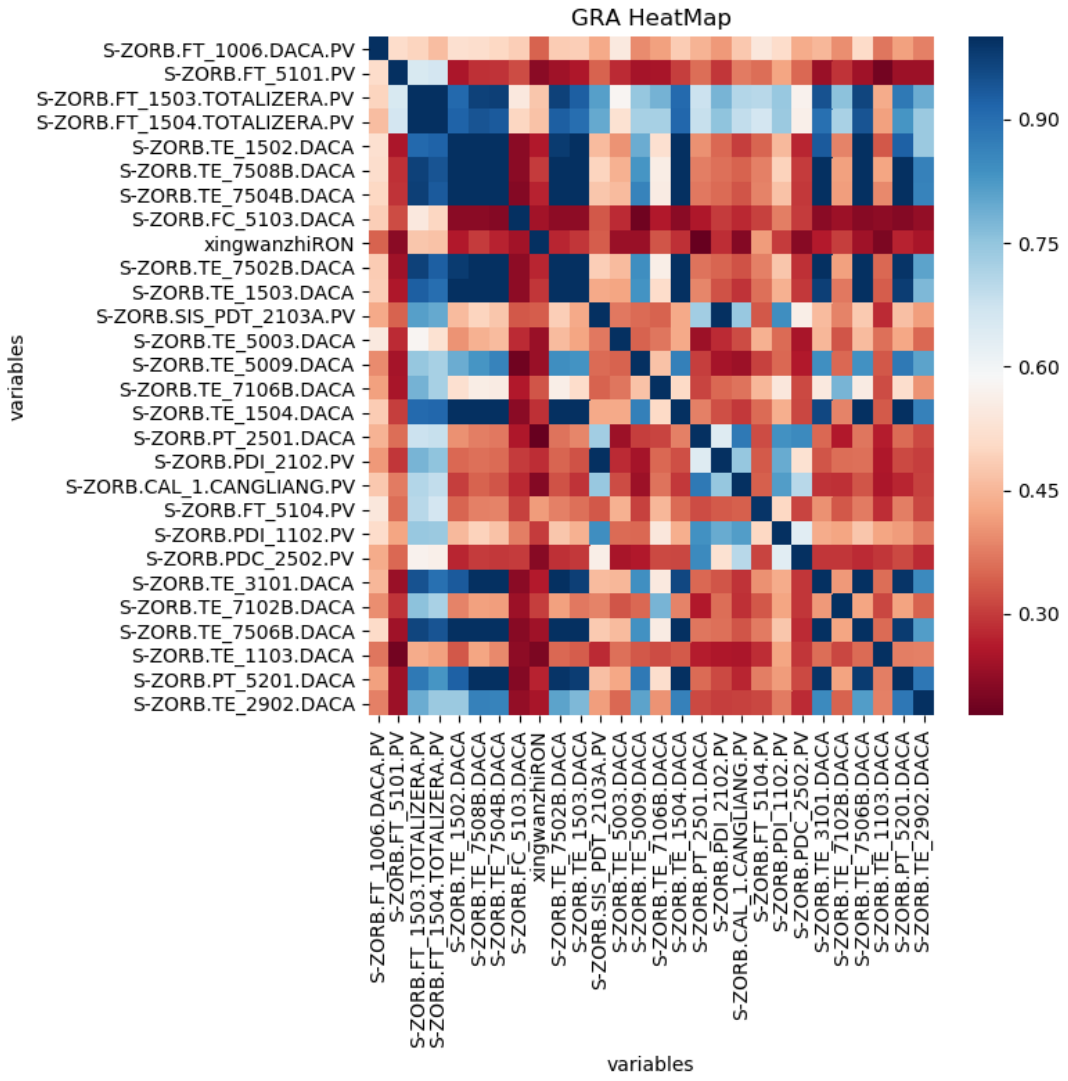

图 14 最大互信息关联性分析

由上图可以看出，在所筛选出的主要变量中，存在较少部分变量其相关系数高，考虑到绝大部分主要变量之间呈现出相关性程度低，彼此的独立性较好的结果，所以本文认为基于随机森林算法提取的主要变量具有较高的可信度。

5.4 本章小结

针对问题二，本章节首先对训练前的样本数据进行了包括异常值剔除、缺失值填充或删除以及进行最大最小归一化等数据预处理；其次搭建了基于随机森林的特征提取模型，并对其中可调的参数进行了调节优化，依据重要性系数，筛选出了 28 个主要变量，通过参数调优保证了模型筛选出的 28 个变量重要性占比达到将近一半；最后利用最大互信息系数方法对该模型训练的结果进行了独立性检验，发现该特征提取模型处理结果具有较高的可信度。

六、问题三的模型建立与求解

6.1 问题分析

根据题目要求，此问题需要利用问题二所提取的主要变量采用数据挖掘的技术建立辛烷值（RON）损失预测模型，并对该模型进行验证分析。通常现实系统中常采用定性和定量预测结合的方式进行分析预测，由于我们缺少辛烷值相关的工程经验，针对此问题，本文采用定量预测分析方法。首先对问题二提取出的主元变量进行可视化统计分析，确定可调控的操作变量和不可调控的非操作变量；然后对操作变量分别利用 BP 神经网络和 XGBoost 算法建立了辛烷值的损失预测模型，找出辛烷值损失与各主要变量之间的调控关系；最后采取交叉验证的方法分别对两个模型进行了性能验证。

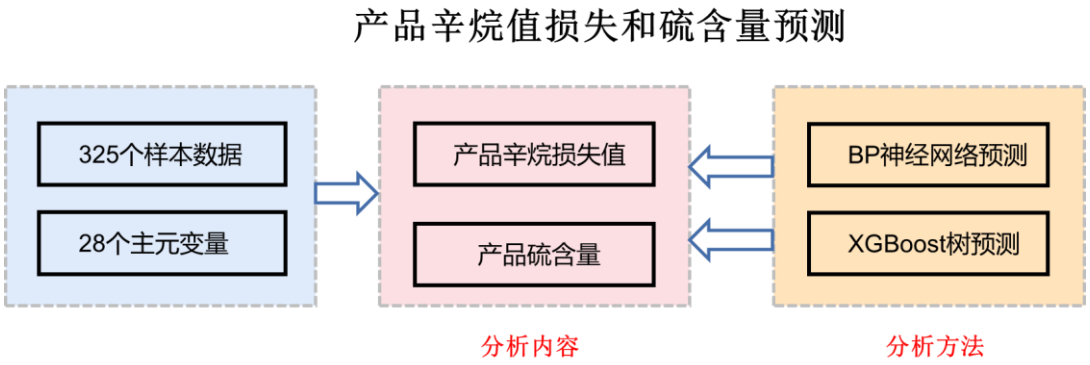

图 15 产品辛烷值损失和硫含量预测分析

6.2 模型建立

6.2.1 基于神经网络的预测模型建立

神经网络（Back Propagation，简称 BP）[6]是类比人体神经元结构建立的信息处理方法，采用大量神经元建立网络结构，实现对线性或是非线性函数的拟合、降噪以及预测、分类等多种功能。神经网络的网络结构一般包含输入层、输出层以及隐藏层，按照网络的连接方式，神经网络可分为前向网络和递归神经网络；对比不同的学习方式分为有导师学习方式和无导师学习方式；或者针对实现诉求的不同分为回归神经网络和分类神经神经网络。

本文是基于误差逆传播神经网络实现了对辛烷值损失预测的回归模型建立。其主要思想如下：

（1）首先将一组主元变量训练样本输入到网络结构中进行前向传播计算，输出层输出对应的辛烷值损失预测值；

（2）判断预测值与相应样本编号中实际辛烷含量的误差精度是否满足要求，若不满足，则通过输出层对该误差进行反向传播，并对网络中各层对应数量神经元的权值和阈值进行调整，从而保证网络的最终输出与实际样本数据的真实值间误差尽可能地减小，最终控制模型精度在允许的范围之内。

其中，本文通过梯度下降(Gradient descent)求解误差的最小值，求解思想如图 16 所示：

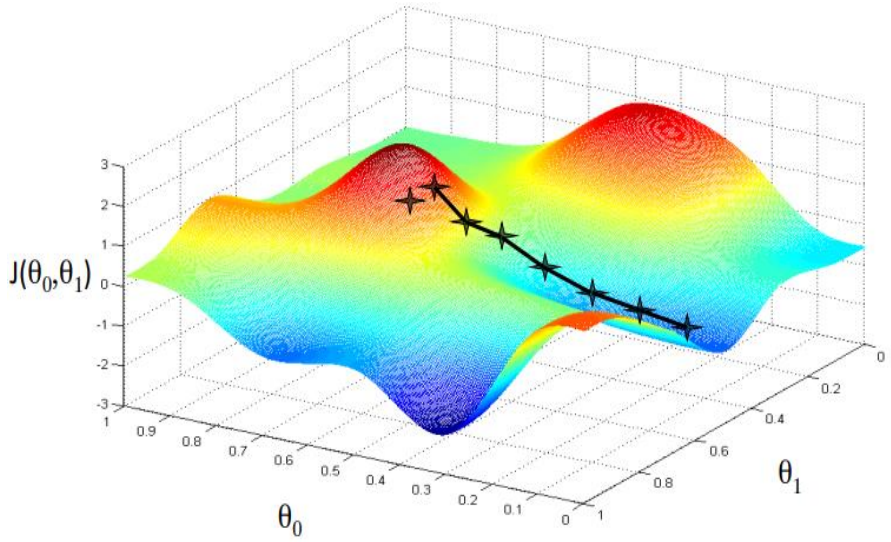

图 16 梯度下降法寻找误差最小值

基于以上思想，本文建立了包括 1 层输入层，3 层隐藏层以及 1 层输出层的 BP 网络结构，其中，输入神经元为 28 个，分别对应问题 2 所筛选出的 28 个主要变量；输出神经元为 2 个，分别对应汽油辛烷损失值和产品的硫含量，该 BP 结构图如图 17 所示：

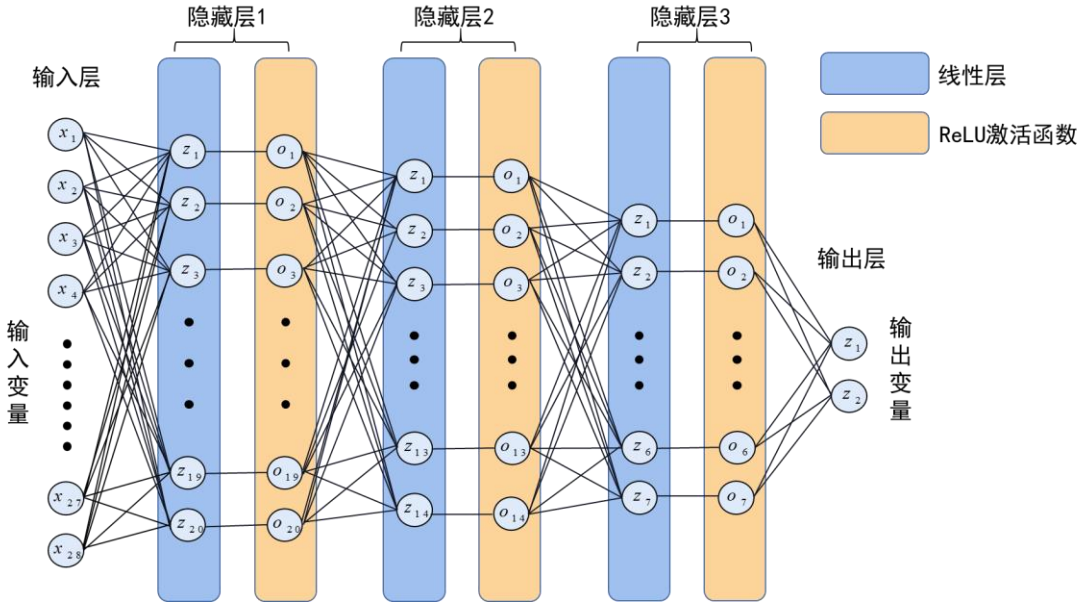

图 17 BP 神经网络结构图

假设该 BP 预测模型的输入、输出样本为：

$$x=[x_1,x_2,\cdots,x_m]\ (m=1,2,\cdots,28)$$

$$y=[y_1,y_2]$$

设置 3 层隐藏层神经元参数分别为：

$$O_{hid1}=[o_1,o_2,...,o_{20}]$$

$$O_{hid2}=[o_1,o_2,...,o_{14}]$$

$$O_{hid3}=[o_1,...,o_7]$$

设置输入层和隐藏层间的网络权重为W^1，隐藏层 1 与隐藏层 2 之间的网络权重为W^2，隐藏层 2 和隐藏层 3 间的网络权重为W^3，隐藏层 3 和输出神经元之间的网络权重为W^4，隐藏层与输出层间的阈值分别为θ^1、θ^2、θ^3和θ^4。

则通过网络权值和阈值分析，得到隐含层 1、2、3 的神经元的输出o_{hid1j}、o_{hid2s}、o_{hid3v}分别为：

$$o_{hid1j} = f(\sum_{i=1}^{28} w_{ji}^1 x_i - \theta_j^1) = f(net_j)$$

$$o_{hid2s} = u(\sum_{j=1}^{20} w_{sj}^2 o_{hid1j} - \theta_s^2) = u(net_s)$$

$$o_{hid3v} = u(\sum_{s=1}^{14} w_{vs}^3 o_{hid2s} - \theta_v^3) = u(net_v)$$

得到输出层神经元的输出z_k为：

$$z_k = g(\sum_{v=1}^{7} w_{kv}^4 o_{hid3v} - \theta_k^4) = g(net_k)$$

输出层神经元的输出z_k与期望输出y_k的输出误差为：

$$E = \frac{1}{2}\sum_{k=1}^{n}(y_k - z_k)^2 = \frac{1}{2}\sum_{k=1}^{n}\left\{y_k - g[\sum_{v=1}^{2} w_{kv}^4 o_{hid3v} - \theta_k^4]\right\}^2$$

代入中间隐藏层的输出o_{hid1j}、o_{hid2s}、o_{hid3v}可以最终得到输出误差E和输入的主元变量X之间的关系映射，对于输入层主元变量和输出层产品辛烷值损失、硫含量的映射可看作是非线性函数的拟合，只需要在每个隐藏层后增加非线性激活函数(本文采用 ReLU 函数)，便可实现最终的输出输出映射关系，从而建立辛烷值损失的预测模型。

本文 BP 算法总体流程如图 18 所示：

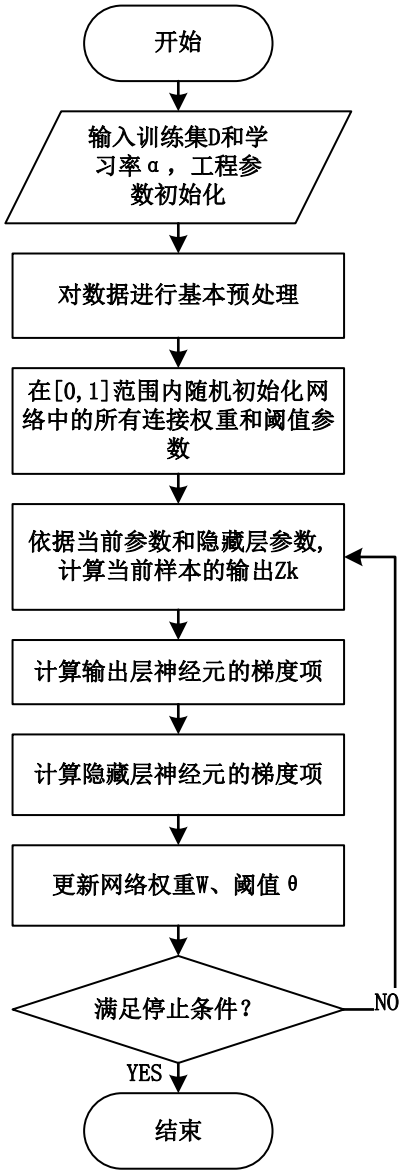

图 18　BP 预测模型主要流程图

由上图可知，首先输入训练样本数据 $D=\{(x_k,y_k)\}_{k=1}^{m=325}$；其次依据表 3 对网络中所有的连接权重和阈值参数进行标准随机初始化；最后计算输出样本的均方差损失并通过梯度下降的方法不断更新网络连接权重和阈值直到满足精度要求，并将满足停止条件的连接权重和阈值参数进行保存，作为预测模型的最终参数。

表 3　BP 预测模型参数配置

参数名称	参数值
输入层维数	28
隐藏层 1 维度	20
隐藏层 2 维度	14
隐藏层 3 维度	7
输出层维度数	2
最小训练速率	0.9
激活函数	ReLU
动态参数	0.8
迭代次数	5000
数据转换	z-score 标准化
优化器	Adam

6.2.2 基于 XGBoost 的预测模型建立

XGBoost 算法是由 Chen T[8]提出的一种基于决策树(Decision Tree)以及梯度提升(Gradient Boosting)的方法，具有 Boosting 族利用迭代算法将弱学习器提升为强学习器的算法属性，是对 GBDT 算法的进一步改良和优化。

XGBoost 算法的基本原理是将原始数据分割为多个子数据集，将各个子数据集随机分配给基分类器进行数据预测，通过弱分类器的结果按照一定的权重进行计算，一次预测最后的结果。XGBoost 算法实现目标函数在 $t=0$ 处的二阶泰勒展开，并通过正则项引入控制模型的复杂度，防止模型预测发生过拟合，使得模型求解最优解更加具有效率。XGBoost 目标函数的表达式为：

$$OBj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{i=1}^{t} \Omega(f_i)$$

其中：

$$\Omega(f_i) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \quad L_2$$

在上述式子中，目标函数右侧由损失函数和正则项构成。右侧第一项表示损失函数，其中 y_i 为真实值，$\hat{y}_i$ 为预测值；目标函数公式第二项定义了结构复杂度函数，其将 t 棵树的复杂度求和作为目标函数的正则项因子。叶子节点 T 构成复杂度，且随着叶子节点数量的增加模型的复杂性也在上升，因此通过引入惩罚项 γ 对叶子节点进行衰减，抑制叶子节点的数量生成(叶子结点的权重不宜取得过高)，实现对树的修剪。XGBoost 算法新生成的树需要拟合上一次预测的残差，当生成 t 棵树后，目标函数可以改变为：

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant$$

在该式中，$\Omega(f_t)$ 表示为正则项，其可进一步拆解成第 t 棵树和 $t-1$ 棵树的复杂度，基于已知的前 $t-1$ 树的复杂度，可以用常数 $constant$ 表示。依据上式分析，对目标函数 $L^{(t)}$ 进行二阶泰勒展开得：

$$L^{(t)} \cong \sum_{i=1}^{n}[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

对目标函数进行简化得：

$$\tilde{L}^{(t)} \cong \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

定义 $I_j = \{i \mid q(x_i) = j\}$ 作为叶子节点 j 的样本集合，将 OBj 代入 $L^{(t)}$，得到最终的目标函数公式：

$$\tilde{L}^{(t)} \cong \sum_{j=1}^{T}[(\sum_{i \subseteq I_j} g_i) w_j + \frac{1}{2}(\sum_{i \subseteq I_j} h_i + \lambda) w_j^2] + \lambda T$$

进一步地，依据表 4 对该预测模型中的相关参数进行初始化：

表 4　XGBoost 预测模型参数配置

参数名称	参数值
min_child_weight	1
max_depth	5
reg_lambda	1
subsample	0.7
colsample_bytree	0.8
max_delta_step	0.5
gamma	0
eta	0.1

6.3 模型求解与结果分析

在对两种预测模型进行模型训练和测试时，为了防止模型发生过拟合问题，本文选择交叉验证的方法进行样本数据集的切分。交叉验证会对数据集进行若干次的切分和整合，每次都会产生新的、不同的子数据集来作为此次学习的训练集和用作预测的测试集，具体实现流程为：

（1）对源数据集进行有放回地、随机地进行 K 次抽样；

（2）随机选取其中 1 份作为测试数据集，其余 K-1 份作为训练数据集；

（3）循环重复第二步，直至每个子数据集都有且只有一次作为测试数据集进行测试完毕；

（4）同时，也得到了 K 个训练集，分别对 K 个训练集训练后我们可以得到 K 个模型以及其各自不同的预测误差等性能评价指标。

本文将 K 值取 5，即对样本数据进行 5 次随机抽样，每次取 4 份也就是 80%的样本集作为训练集，取 1 份也就是 20%作为测试集。

6.3.1 基于神经网络的预测模型求解

依据上述交叉验证的思想，本文利用 BP 神经网络预测模型总共进行了五组实验，分别得到五组实验结果，其中一组如下图 19-21 所示：

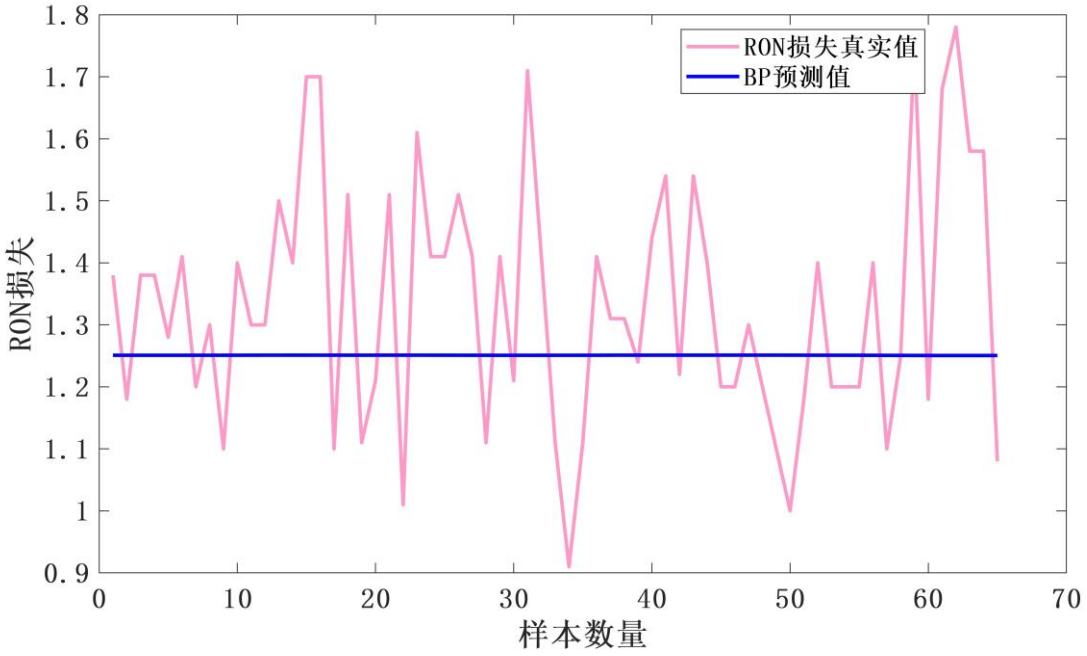

图 19　BP 模型预测 RON 辛烷值损失

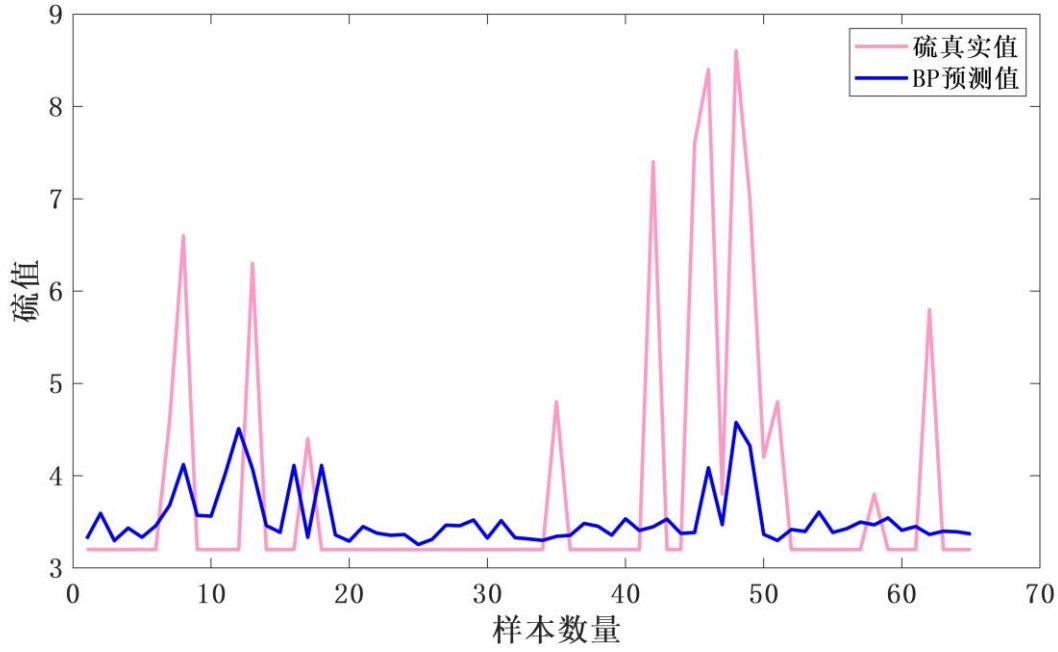

图 20　BP 模型预测硫含量

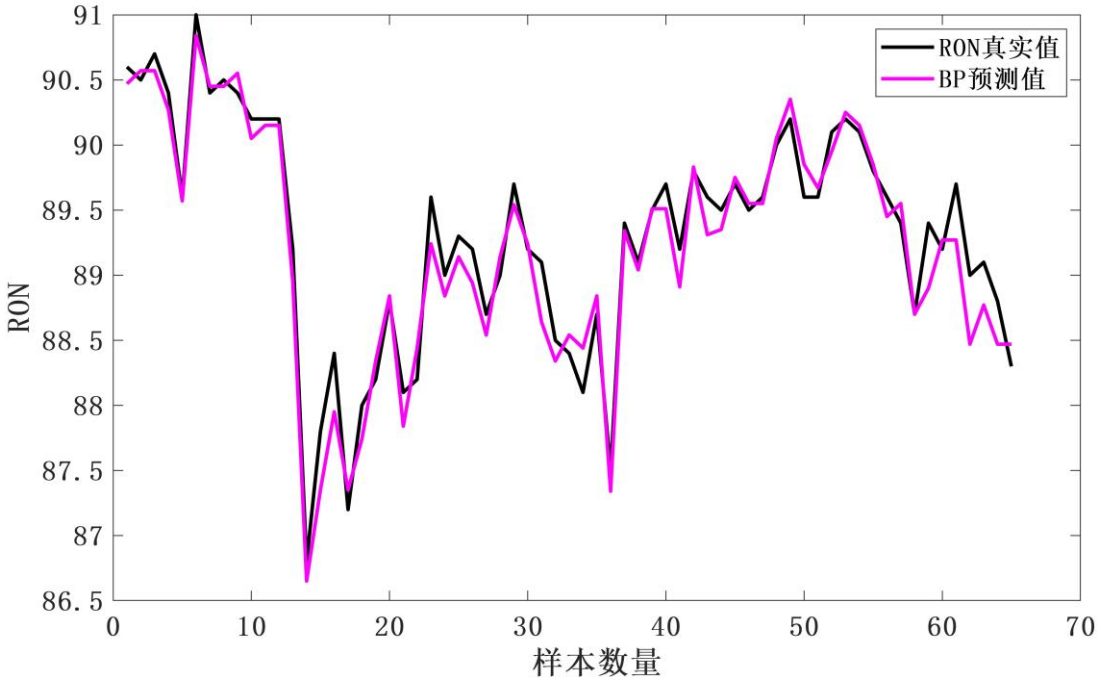

图 21　BP 模型预测辛烷值 RON

从图 20,21 可以看出，BP 神经网络预测模型训练结果对于真实值具有较好的跟随性，但与真实值的大小存在一定的差异，所以为了更准确地对该模型进行回归性能分析与评估，本文采用评估指标如下：

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}$$

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left|\left(y_i - \hat{y}_i\right)\right|$$

其中，y_i 表示预测值，$\hat{y}_i$ 表示真实值。以预测 RON 辛烷值损失为例，分别对五组结果计算统计如下表所示：

表 5　BP 预测模型性能评估参数

MSE	0.051284296	0.05097989	0.045572681	0.045691215	0.043974217
RMSE	0.226460363	0.225787268	0.21347759	0.213755035	0.209700302
MAE	0.176628901	0.176473844	0.17007325	0.170457735	0.169090673

6.3.2 基于 XGBoost 的预测模型求解

同样地，本文利用 XGBoost 预测模型也总共进行了五组实验，分别得到五组实验结果，其中一组如下图 17-19 所示：

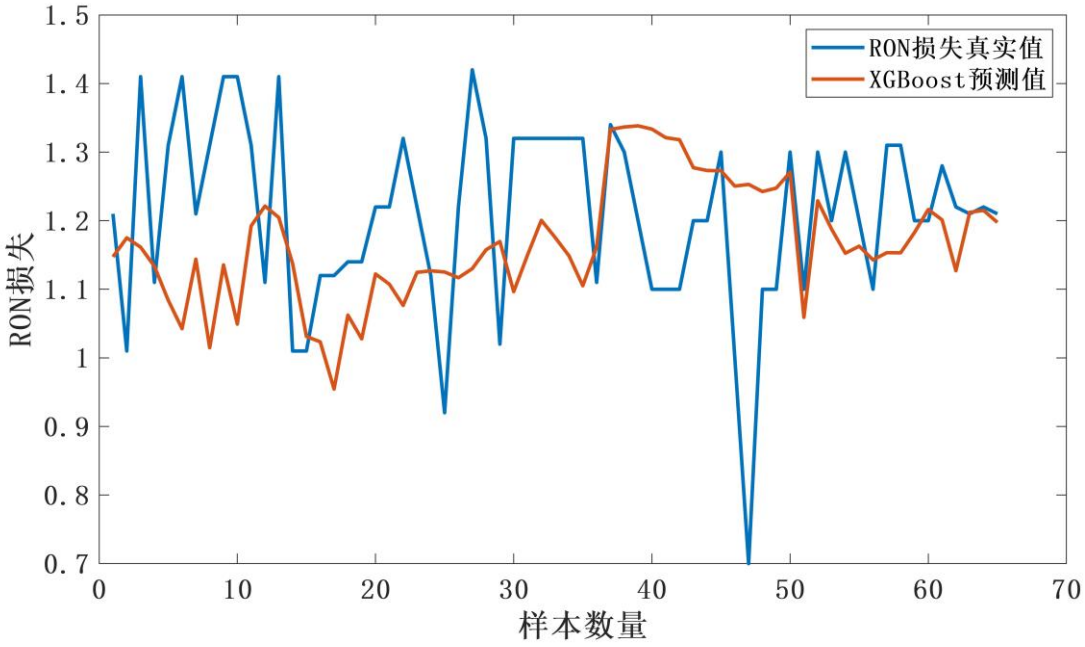

图 22　XGBoost 模型预测 RON 辛烷值损失

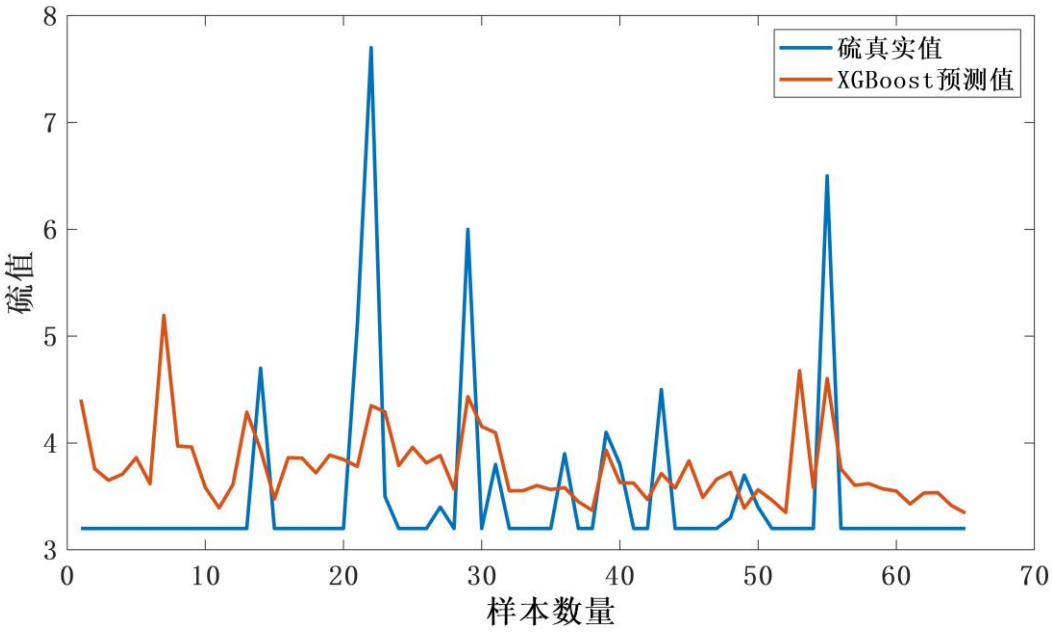

图 23　XGBoost 模型预测硫含量

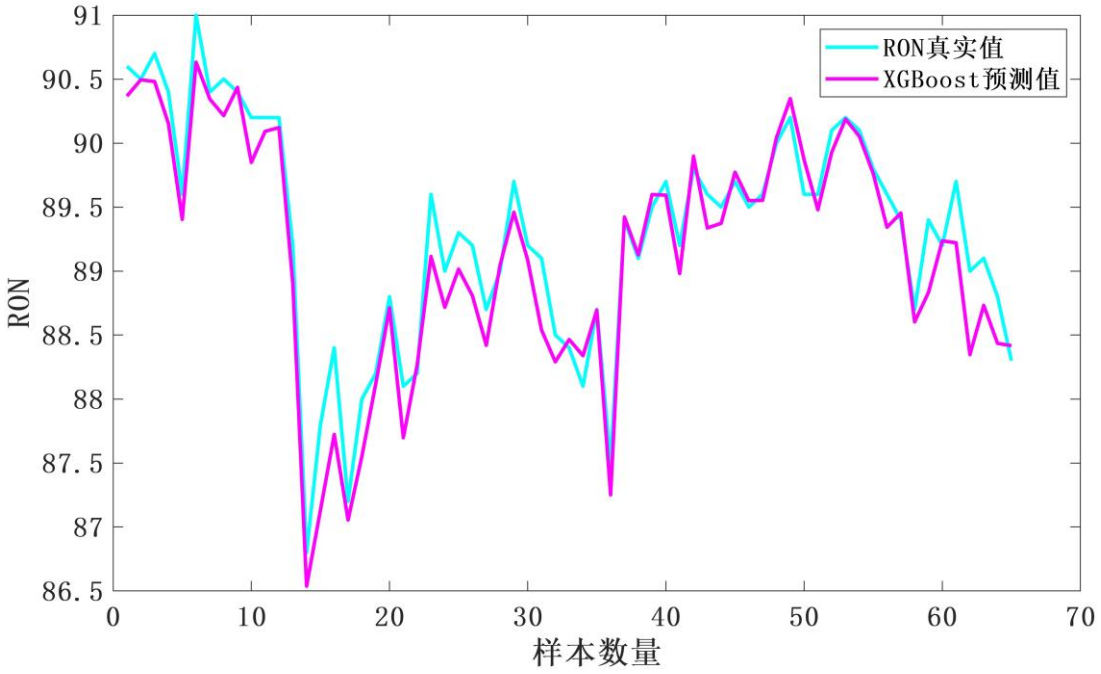

图 24　XGBoost 模型预测 RON 辛烷值

对比图 19-21，由图 22-24 可以很直观地看出，基于 XGBoost 模型的预测结果明显地优于 BP 神经网络预测模型，进一步地，同样以预测 RON 辛烷值损失为例，计算并统计 XGBoost 模型的回归性能评价参数，结果如下：

表 6　XGBoost 预测模型性能评估参数

MSE	0.050176833	0.051717006	0.028873784	0.024652604	0.051717006
RMSE	0.224001855	0.227413729	0.169922873	0.157011479	0.227413729
MAE	0.182141781	0.179344401	0.13432461	0.115967236	0.179344401

6.3.3 结果对比分析

上表的几个相应指标（MSE,RMSE,MAE）是综合考量回归算法优劣的三个常见评价指标，其值越接近 0 说明拟合准确度越高，观察上表和 BP 网络的指标，可知这两个网络都很好地拟合出了辛烷值损失这种数值波动较大的因变量。仔细分析，与 BP 神经网络比较，XGBoost 模型的 MAE 和 RMSE 更低，说明他能在波动较大的情况下也能很好的回归，可以判断出 XGBoost 模型的回归更好。

6.4 本章小结

针对问题三，本章节主要利用 BP 神经网络算法和 XGBoost 算法分别建立了辛烷值 RON 的损失预测模型：首先，本文设计了一个含有 3 层隐藏层，一层输入层以及一层输出层，共计 5 层的神经网络结构模型，其中，输入层共有 28 个神经元，分别对应问题二所最终得到的 28 个主要操作变量，然后依次通过隐藏层和输出层，最终得到汽油辛烷值损失和硫含量两个预测变量；其次，为了可以进行结果的横向对比，本文又建立了基于 XGBoost 算法的预测模型；接着为了防止模型出现过拟合的情况，本文采用了交叉验证的方法对两种模型分别进行训练和测试；最后依据 MSE，RMSE 和 MAE 三种常用的检验回

归拟合度的性能系数对两个模型的预测结果进行了比较分析，发现两种模型的拟合误差分别约为 0.048 和 0.041，拟合程度较好，且 XGBoost 的拟合程度优于 BP 神经网络模型。

七、主要变量操作方案的优化

7.1 问题分析

问题四建立在之前问题的基础上，在保证产品含硫量不大于 5ug/g 的限制条件下，将回归模型预测出来的汽油辛烷值损耗与实际数据进行损失降幅比较，提取其中损失降幅低于 30%的原始样本，对影响此类原始样本的主要变量进行优化，找到对主要变量最优的操作条件。该问题对应求解最优化中的组合优化问题，即在具有操作范围限制的给定主元变量的中寻找到使得辛烷值损失值最小的操作条件子集问题。

7.2 模型建立

多变量的组合优化问题可以通过对离散事件的排序、筛选或最优编排等方式找到最优解，常用的优化算法可分为精确算法、近似算法、启发式和元启发式的算法。在问题规模较小的时候，通过精确算法的求解可以在允许范围内找出最优解；动态规划、贪婪算法等近似算法涉及 NP-hard 复杂度，无法保证一定能在 P 问题的时间范围内精确地找到最优解；启发式算法的主要缺点是无法预计所得解与最优解的偏差程度；元启发式算法对启发式算法的改进，结合随机搜索与局部探索的特性，广泛地运用于组合优化以及函数的计算问题之中。本问题立足于工业生产实际中催化裂化汽油精制装置对原油的精制处理，提取的主元变量数量众多，问题规模很大，精确求解的计算代价非常昂贵，容易产生“组合爆炸”的问题，近似算法难以实现复杂的现实问题。因此本文拟采用元启发算法中的粒子群优化算法对主元变量的操作条件进行最优求解。

7.2.1 粒子群算法优化主元操作条件

粒子群算法(Partial Swarm Optimization)是一种基于迭代的优化算法，系统通过初始化一组随机解，粒子在解空间追随最优粒子进行搜索，通过不断迭代搜寻最优解。粒子群算法可以描述为由 m 个粒子组成的群体在 D 维空间中以特定的速度运行，各个粒子进行搜索时需要考虑自身搜索到的历史最好点以及群体内其他粒子的历史最好点，由此进行位置(解变量)的变化。其基本公式可表述为：

$$\vec{x}_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$$

$$\vec{v}_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$$

$$\vec{p}_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$$

$$\vec{p}_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$$

其中粒子在 D 维空间中的位置表示为 $\vec{x}_i$，飞行速度为 $\vec{v}_i$，自身粒子的历史最优位置表示为 $\vec{p}_i$，目前为止整个群体中粒子发现的最优位置用 $\vec{p}_g$ 表示。粒子的更新公式如下所示：

$$v_{id}^{k+1} = wv_{id}^{k} + c_1\xi(p_{id}^{k} - x_{id}^{k}) + c_2\eta(p_{gd}^{k} - x_{id}^{k})$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1}$$

其中，

$$\xi, \eta \in U[0,1]$$

其中 v_{id}^{k}，x_{id}^{k} 分别是粒子的位置和速度；ξ, η 是介于(0,1)之间的随机数；c_1，c_2 表示学习因子；w 是粒子更新的权重系数(取值通常为 0.1 - 0.9)，粒子的每一维度的最大速率收到最大限幅 $V_{\max}$ 。

求解 PSO 问题解决的优化问题包含问题的编码以及适应度函数两个重要步骤。针对本问题，设需要求解的 28 个主要变量为 $I = [i_1, i_2, i_3, ..., i_{28}]$，优化目标汽油辛烷损失值包含在预测模型的输出 $L = [l_1, l_2]$ 中，则一个粒子可直接编码为 $(i_1, i_2, i_3, ..., i_{28})$，适应度函数则为预测模型的输出 L 。

7.2.2 PSO 中算法参数以及经验设置

（1）粒子数，一般取 5-40. 其实对于大部分的问题 10 个粒子已经足够可以取得好的结果;

（2）粒子的长度（维度），这是由优化问题决定, 就是问题解的长度;

（3）粒子的范围，由优化问题决定,每一维可以设定不同的范围;

（4）最大速度 Vmax，决定粒子在一个循环中最大的移动距离,通常设定为粒子的范围宽度

（5）学习因子，c1 和 c2 通常等于 2;

（6）中止条件，最大循环数以及最小错误要求。

表 7 粒子群算法主要参数

参数名称	参数值
粒子数	40
最大迭代次数 max_iter	50
最大速度 Vmax	0.05
维度 dim	28

上表是实际求最优解过程中使用的参数，迭代次数较小的原因是样本个数太多加上时间有限，只能有所割舍。

7.3 模型求解与结果分析

首先将第三个问题建立的模型导入到粒子群算法中，利用附件 4：354 个操作变量信息中提取到的主要变量的取值范围约束粒子的移动空间。由于拟合的回归模型准确度非常高，所以 354 个样本的辛烷值损失预测值绝大部分都不满足降低 30%，必须需要去迭代进行优化。经过 354*50 次的粒子群迭代寻找到每个样本的最优解。输出变量硫含量作为约束，必须小于 5μg/g，另一个输出变量辛烷值损失作为自适应函数进行优化。下图就是经过 PSO 优化后的辛烷值损失的对比图。

下图是优化后对应的硫含量的对比图，所有样本都满足 5μg/g 的约束。

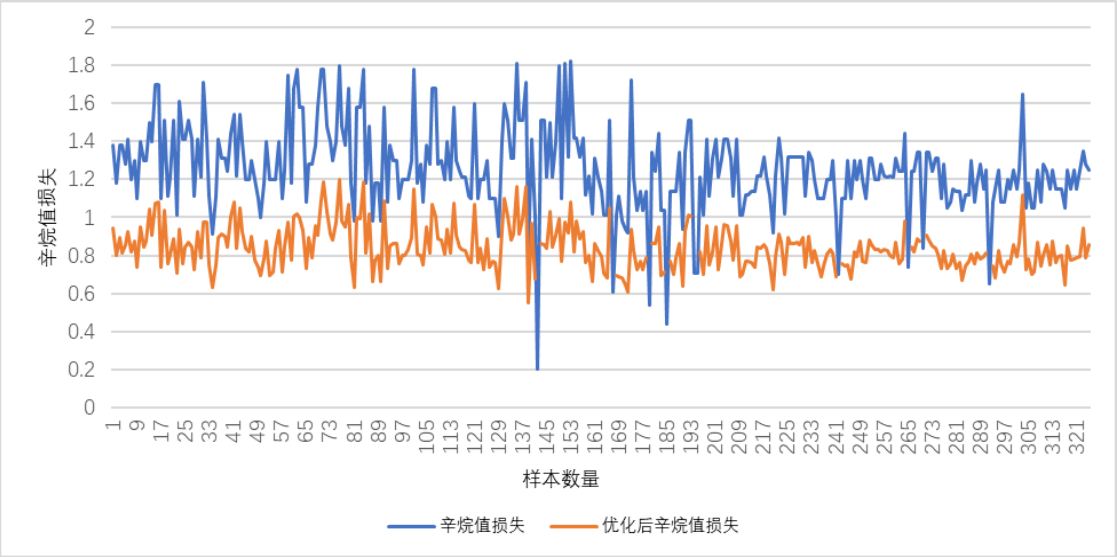

图 25　PSO 优化的辛烷值损失

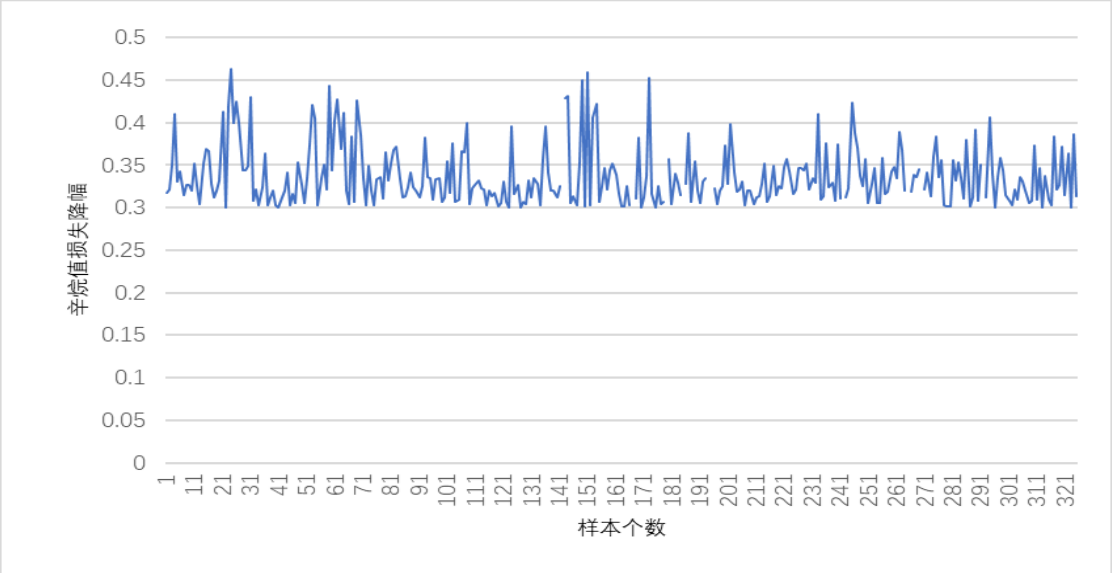

图 26　PSO 优化的辛烷损失降幅

观察上图可知，PSO 优化后辛烷值降幅非常大，部分样本降幅甚至接近 50%，如此大的辛烷值损失降幅必定后导致硫含量发生较大变化。下图进行分析所有样本内硫含量的变化，很显然，硫含量大幅提升，这说明汽油产品的辛烷值和硫含量是呈负相关的。辛烷值决定汽油的安全性，硫含量高又会不符合标准且污染空气，必须要有所取舍。若仔细观察图，可以发现折线有断点，这是因为个别样本进行预测或者是找不到满足条件的最优解，经过统计，这类样本数目 10 个，占总体样本极少，不影响整体模型的效果

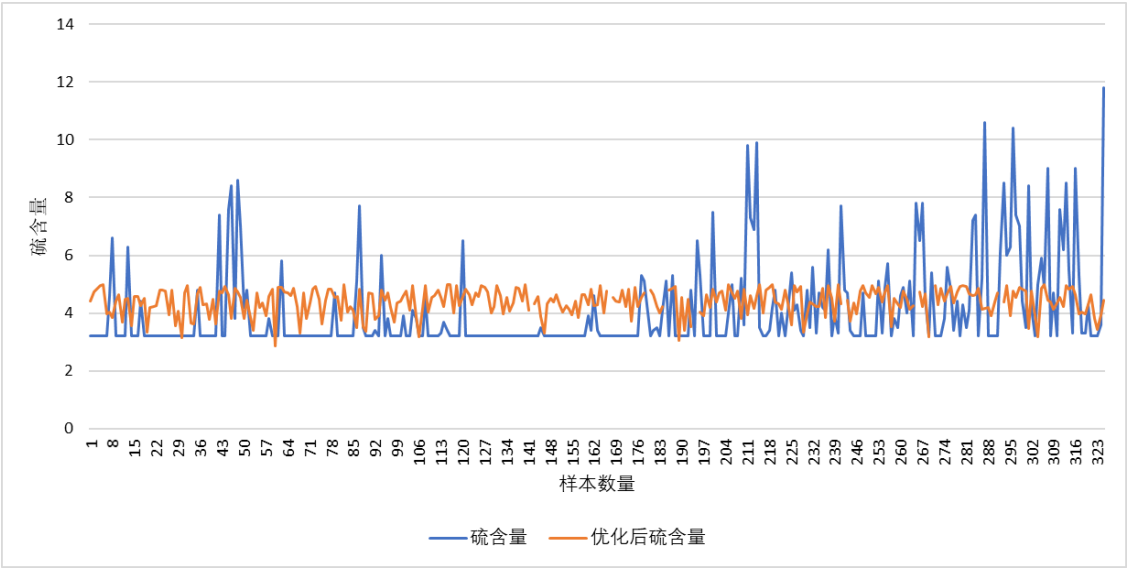

图 27　PSO 优化的硫含量

7.4 本章小结

针对问题四，本章首先介绍了如何挑选合理的优化算法去进行多变量调优以及粒子群算法的简单建模过程，简单介绍了较大影响粒子群算法效果的参数和普遍值。以辛烷值损失作为自适应函数，每个样本经过 50 次迭代，对 325 个样本的操作变量进行调优，经过对比后发现，有 315 个样本未找到满足条件的最优解，且绝大部分降幅超过了 30%，效果很好，可是另一方面，硫含量也显著增加。这说明粒子群优化算法在汽油催化裂化应用下是有价值的，最优操作变量为后续工厂调参提供经验。

八、　模型的可视化展示

8.1 问题分析

本题取一个 133 号样本去展示操作变量优化调整过程中对应的汽油辛烷值和硫含量的变化轨迹。我依据附件四:354 个操作变量信息获取对应的 28 个主要变量的最大值，最小值以及Δ值。从上个问题求解到的优化后的 133 号变量的操作条件，再提取原始数据文件附件一对应的 133 号的 28 个主要变量的值。由优化前后的变量值，以及变量调整步长Δ，画出辛烷值和硫含量在优化调整过程中的变化轨迹。

8.2 模型求解及结果分析

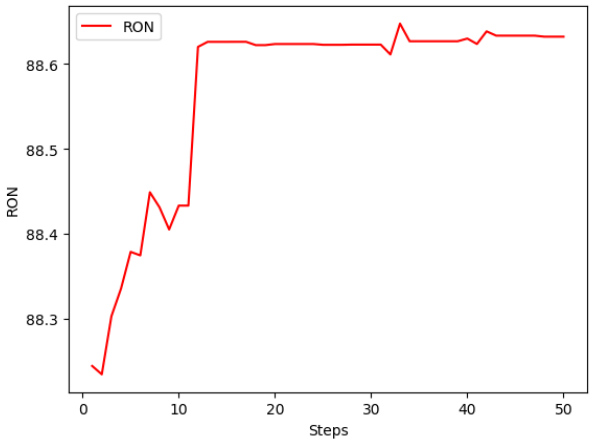

图 28 辛烷值步长轨迹图

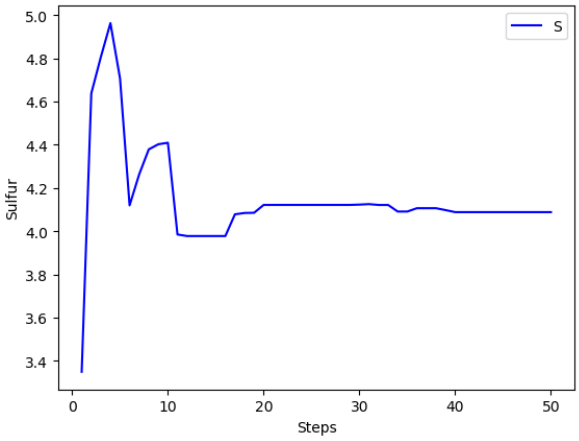

图 29 硫含量步长轨迹图

观察上图可知，硫含量在前期迭代接近 5μg/g， 不满足第四题的条件，所以直接调整参数降低硫含量，紧接着辛烷值也开始增大，两者变化关系存在一定负相关性，且存在间隔可以让催化裂化工厂改变工业装置参数得以施加影响，最终辛烷值和硫含量都趋于平稳。如果能仔细分析每个主要变量的变化，或许可以利用数据挖掘技术去发掘变量调优过程的变化规律，给工业装置调参提供经验。

九、 模型评价与改进

问题一是对附件三“285 号和 313 号样本原始数据.xlsx”进行数据处理。因此，本文根据题目要求，依据附件二“样本确定方法”中所提供的确定数据的整定，样本筛选逻辑方法依次对部分存在问题的数据进行了数据处理。首先，依据拉依达准则对被认为是含有粗大误差值的异常值进行剔除，消除了这些可能是因位点存在问题而测量到的异常值对后续数据挖掘工作的影响；其次依据数据变量的操作范围，利用最大最小的限幅方法对不在范围内的部分数据进行了剔除；接着对于部分数据为控制的位点利用其前后两个小时的平均代替；最后将处理完成后的 285 号和 313 号样本数据替换至附件一中对应编号的样本。

问题二是寻找建模主要变量，首先得经过数据预处理，然后通过随机森林进行特征提取获取影响辛烷值损失的主要操作变量，然后计算主要变量之间的最大互信息系数矩阵，分析主变量之间的相关性，相关性特别大的主变量可以筛除，将其它相关性小的变量带入作为主要变量。

问题三是利用问题二中经过数据提取降维后提取的主要变量对辛烷值的损失建立损失预测模型。本文同时采用 BP 神经网络和 XGBoost 树方法对主要变量进行回归拟合，并通过模型交叉验证，检验了预测模型输出的合理性和可靠性。采用具有 3 隐藏层的 BP 神经网络对输入的 28 个主要变量进行训练，输出层输出包含汽油辛烷损失以及产品硫含量的 2 维变量。XGBoost 在代价函数里加入了正则项, 从 Bias-variance tradeoff 角度来讲，模型的 variance 由正则项降低了，使学习出来的模型更加简单，防止过拟合。比较两种预测模型的回归指标，判断出 XGBoost 的预测更好。

问题四是对主要变量操作条件的优化，在保证产品硫含量在限定范围内的条件下，使用预测模型预测 325 个样本的辛烷损失值估计，并提取样本中辛烷值损失降幅达到 30%的样本来优化主要变量的操作条件。将该问题建立为组合优化问题，通过元启发式优化算法

PSO 对最优解进行逼近。PSO 算法将 28 个主要变量建模为单个粒子的编码变量，将辛烷值损失值作为适应度函数，最优 315 个样本的操作变量优化后满足辛烷值损失降幅大于 30%。多样本多维度的优化问题计算复杂度高，很难应用于实际工业操作环境中，后续有待研究更加简单有效的优化算法来实现工业应用，提高经济效益。

问题五是展示参数调优过程中辛烷值和硫含量的变化轨迹，后续可以进一步分析每个主要变量的变化轨迹，在高维空间里研究如何给工业装置调参。

参考文献

[1] 杨苗, 长庆石化汽油辛烷值的数学建模法计算研究[D].西安石油大学,2015.

[2] Ivanchina E D, Kirgina M V, Chekantsev N V, et al. Complex modeling system for optimization of compounding process in gasoline pool to produce high-octane finished gasoline fuel[J]. Chemical Engineering Journal, 2015, 282: 194-205.

[3] Kirgina M , Gyngazova M , Ivanchina E . Mathematical modeling of high-octane gasoline blending[C]// International Forum on Strategic Technology. IEEE, 2012.

[4] 张皖,张志芳,李玲,唐林.国内外车用汽油标准探析[J].交通节能与环保,10(01):18-20,2014.

[5] S. J. Alexandrovna and Duong Chi Tuyen, "A new approach to gasoline octane modeling," Proceedings of 2011 6th International Forum on Strategic Technology, Harbin, Heilongjiang, ,pp. 758-760,2011.

[6] 莎罗树下飞逝， 6-有导师学习神经网络的回归拟合，https://wk.baidu.com/view/83c5f8e09e314332396893ce?fromShare=1，2020.09.20.

[7] 毕青松,梁雪春,陈舒期.基于 mRMR-RF 特征选择和 XGBoost 模型的钓鱼网站检测[J].计算机应用与软件,37(09):296-301，2020.

[8] Chen T，Guestrin C． XGBoost: ascalable tree boosting system［C］/ /22nd ACM SIGKDD International Conference on Knowl-edge Discovery ＆ Data Mining，2016: 785-794．

[9] 李爱国,覃征,鲍复民,贺升平.粒子群优化算法[J].计算机工程与应用,2002(21):1-3.

[10] 宫华,李作华,刘洪涛,郝永平.基于改进 PSO-BP 神经网络的贮存可靠性预测[J].运筹与管理,2020,29(08):105-111.

[11] 1550280265, 组合优化问题求解算法思路的整理（VRP、SDVRP、container loading）,https://blog.csdn.net/qq_32732581/article/details/84442393,2020.09.19.

附录

代码 1：数据预处理

```
import pandas as pd
import numpy as np

##异常值用0， 空值用nan。
def read_file():
    data=pd.read_excel("/home/zc/Desktop/zc/data/附件三：285 号和 313 号样本原始数
据.xlsx",skiprows=1,sheet_name='操作变量')
    #读取 285 和313 的数据
    data=data.drop(index=0,axis=0)
    data_5=data.iloc[41:81,:]
    #统计这两组数据每列存在的0 值数量
    data_5.set_index(["时间"], inplace=True)
    #将0 置换成null
    df_5=data_5.replace(0,np.nan)
    print("read file finished")
    print(df_5.head())
    return df_5

def average_fill(df_5):
   #计算均值
    #df_5.loc['mean']=df_5.mean(axis=0)
    #将某列数据有缺值的用均值补全
    list_index=list(df_5.index)
    for index, row in df_5.iteritems():
        #print(sum(i > 0 for i in np.isnan(list(row))))
        if (sum(i>0 for i in np.isnan(list(row)))<40) &(sum(i>0 for i in
np.isnan(list(row)))>0):
            mean_lst=[]
            for i in row:
                if(i>0):
                    mean_lst.append(i)
            mean=np.mean(mean_lst)
            ##把均值带到Nan 里
            for i in range(len(row)):
               if np.isnan(row[i])>0:
                   df_5.loc[list_index[i],index]=mean
    df_5.loc['zero'] = df_5.isnull().sum(axis=0)
    null_lst=[]
    for index, row in df_5.iteritems():
        if row['zero'] == 40:
```

```
            null_lst.append(index)
    print(len(null_lst))
    print(null_lst)
    df_5 = df_5.drop(index=["zero"], axis=0)
    df_5.loc['mean']=df_5.mean(axis=0)
    df_5=df_5.fillna(0)
    df_5.to_csv("/home/zc/Desktop/zc/data/deal2/data_313_ave_fill.csv")
    print("average finished")

def meet_range(df_5):
    var_range_data=pd.read_excel("/home/zc/Desktop/zc/data/变量范围
/range_max_min.xlsx",usecols=['no',"min","max"])
    print(var_range_data)
    print(df_5.head())
    print(df_5.shape)
    df_5=df_5.fillna(0)
    n=0
    for index, row in df_5.iteritems():
        #print(var_range_data.iat[n,1],var_range_data.iat[n,2])
        #判断在幅值外的样本数量
        if sum(i < var_range_data.iat[n,1] for i in row)+sum(i > var_range_data.iat[n,2]
for i in row)>20:
            df_5[index]=np.nan
        else:
            df_5.loc[df_5[index]<var_range_data.iat[n,1],index]=var_range_data.iat[n,1]
#min_val
            df_5.loc[df_5[index]>var_range_data.iat[n,2],index]=var_range_data.iat[n,2]
#max_val
        n=n+1
    df_5.to_csv("/home/zc/Desktop/zc/data/deal2/data_313_meet_range.csv")
    print("meet_range finished")
    return df_5

def sigma_3(df_5_met):
    df_5_met.loc['mean']=df_5_met.mean(axis=0)
    for index, row in df_5_met.iteritems():
        std = pd.Series.std(row[:-1])
        for i in range(len(row) - 1):
            if (abs(row[i] - row['mean']) > 3 * std):
                row[i] = 0
    df_5_met=df_5_met.drop(index=["mean"],axis=0)
    #df_5_met.loc['mean']=df_5_met.mean(axis=0)
    df_5_met.to_csv("/home/zc/Desktop/zc/data/deal2/data_313_sigma.csv")
    print("3 sigma finished")
```

```
    return df_5_met

def main():
    df_5=read_file()
    df_5_met=sigma_3(df_5)
    df_5_met=meet_range(df_5_met)
    average_fill(df_5_met)
main()
```

代码 2：随机森林特征提取

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

path="/home/zc/Desktop/zc/data/dimension/df_wuguiyihua.csv"
df=pd.read_csv(path,sep='\t',dtype=float)
df.set_index(["no"],inplace=True)
print(df.head())

#x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state
= 0)
x_train, y_train = df.iloc[:, 1:].values, (df.iloc[:, 0].values)*100
feat_labels = df.columns[1:]
forest = RandomForestClassifier(n_estimators=5000, random_state=0,
n_jobs=-1,min_samples_leaf=1)
forest.fit(x_train, y_train.astype('int'))
importances = forest.feature_importances_
indices = np.argsort(importances)[::-1]
sum=0
outpath="/home/zc/Desktop/zc/data/dimension/max_28_wu.csv"
outfile=open(outpath,'w+')

for f in range(x_train.shape[1]):
    print("%2d) %-*s %f" % (f + 1, 30, feat_labels[indices[f]], importances[indices[f]]))
    if(f<28):
        sum=sum+importances[indices[f]]

string=str(int(f+1))+'\t'+str(feat_labels[indices[f]])+'\t'+str(importances[indices[f
]])+'\n'
        outfile.write(string)
outfile.close()
print(sum)
print("finished")
```

代码 3：最大互信息系数

```
import numpy as np
import pandas as pd
from minepy import MINE
import seaborn as sns
import matplotlib.pyplot as plt

# 读取数据进入内存
wine = pd.read_csv("/home/zc/Desktop/zc/data/fit/main28_data_wu.csv", sep='\t')
wine.set_index("no",inplace=True)
wine=wine.drop(columns=["RON_loss"])
print(wine.head())
def MIC_matirx(dataframe, mine):
    index_lst=list(dataframe)
    print(index_lst)
    data = np.array(dataframe)
    n = len(data[0, :])
    result = np.zeros([n, n])
    for i in range(n):
        for j in range(n):
            mine.compute_score(data[:, i], data[:, j])
            result[i, j] = mine.mic()
            result[j, i] = mine.mic()
    RT = pd.DataFrame(result)
    RT.columns = index_lst
    RT.index = index_lst
    return RT


def ShowHeatMap(DataFrame):
    colormap = plt.cm.RdBu
    ylabels = DataFrame.columns.values.tolist()
    f, ax = plt.subplots(figsize=(8, 8))
    ax.set_title('GRA HeatMap')
    sns.heatmap(DataFrame.astype(float),
                cmap=colormap,
                ax=ax,
                annot=None,
                yticklabels=ylabels,
                xticklabels=ylabels)
    ax.set_xlabel("variables",fontsize=10)
    ax.set_ylabel("variables",fontsize=10)
    plt.show()
```

```
mine = MINE(alpha=0.6, c=15)
data_wine_mic = MIC_matirx(wine, mine)
print(data_wine_mic)
ShowHeatMap(data_wine_mic)
print("finished")
```

代码 4：XGBoost 回归

```
# ================基于Scikit-Learn 接口的回归================
import xgboost as xgb
from keras import metrics
from xgboost import plot_importance

##cross
MSE_list = []
RMSE_list = []
MAE_list = []
df_all=pd.DataFrame(columns=['A'])
cross_file = pd.read_csv("/home/zc/Desktop/zc/data/fit/cross_data.csv", sep=',')
dataset = pd.read_csv('/home/zc/Desktop/zc/data/fit/main28_data_wu_fit.csv', sep=',')
dataset.set_index(["no"], inplace=True)
for i in range(5):
    train_index = cross_file[("train" + str(i + 1))]
    test_index = cross_file[("test" + str(i + 1))].fillna(-1).astype('int')
    test_index2 = []
    for k in list(test_index):
        if k >= 0:
            test_index2.append(k)

    print(dataset.head())
    y_train = dataset.iloc[train_index,-2:]
    print(y_train.shape)
    x_train = dataset.iloc[train_index,0:-2]
    print(x_train.shape)
    y_test = dataset.iloc[test_index2,-2:]
    print(y_test.shape)
    x_test = dataset.iloc[test_index2,0:-2]
    print(x_test.shape)
    model =
MultiOutputRegressor(xgb.XGBRegressor(objective='reg:squarederror',**other_params))#*
*other_params
    model.fit(x_train, y_train)
```

```
    # 对测试集进行预测
    ans = model.predict(x_test)
    MSE_loss = metrics.mean_squared_error(ans[:,0], y_test.iloc[:,0])
    RMSE_loss = np.sqrt(metrics.mean_squared_error(ans[:,0], y_test.iloc[:,0]))
    MAE_loss = metrics.mean_absolute_error(ans[:,0], y_test.iloc[:,0])
    MSE_s = metrics.mean_squared_error(ans[:, 1], y_test.iloc[:, 1])
    RMSE_s = np.sqrt(metrics.mean_squared_error(ans[:, 1], y_test.iloc[:, 1]))
    MAE_s = metrics.mean_absolute_error(ans[:, 1], y_test.iloc[:, 1])
    if (i == 0):
        all_list = [ list(y_test.iloc[:,0]),list(ans[:,0]), list(y_test.iloc[:,
1]),list(ans[:, 1])]
        df = pd.DataFrame(all_list)
        df = df.T
        df.rename(columns={0: 'real_loss1', 1: 'pre_loss1', 2: 'real_s1', 3: 'pre_s1'},
inplace=True)
        df_all = df
        # print(df)
    else:
        df_all.insert(4 * i, "real_loss" + str(i + 1), list(y_test.iloc[:,0]))
        df_all.insert(4 * i + 1, "pre_loss" + str(i + 1), ans[:,0])
        df_all.insert(4 * i + 2, "real_s" + str(i + 1), list(y_test.iloc[:, 1]))
        df_all.insert(4 * i + 3, "pre_s" + str(i + 1), ans[:, 1])

    MSE_list.append(MSE_loss.numpy()), MSE_list.append(MSE_s.numpy())
    RMSE_list.append(RMSE_loss), RMSE_list.append(RMSE_s)
    MAE_list.append(MAE_loss.numpy()), MAE_list.append(MAE_s.numpy())
df_all.loc["MSE"]=MSE_list*2
df_all.loc["RMSE"]=RMSE_list*2
df_all.loc["MAE"]=MAE_list*2
df_all.to_csv("/home/zc/Desktop/zc/data/fit/bp_xgboost_cross_6.csv", sep='\t')
print("finished")
```

代码 5：BP 神经网络

```
def Train_Model(data_train,kk):
    modelfile = './modelweight'
    y_mean_std = "./y_mean_std.txt"
    data_train = np.matrix(data_train)
    data_mean = np.mean(data_train, axis=0)
    data_std = np.std(data_train, axis=0)
    data_train = (data_train - data_mean) / data_std
    x_train = data_train[:, 0:(data_train.shape[1] - 2)]
    y_train = data_train[:, data_train.shape[1] - 2:]
    model = Sequential()
    #model.add(Dense(x_train.shape[1], input_dim=x_train.shape[1],
```

```
kernel_initializer="uniform"))
    model.add(Dense(20, input_dim=x_train.shape[1], kernel_initializer="uniform"))
    model.add(Activation('relu'))
    model.add(Dense(14, input_dim=x_train.shape[1], kernel_initializer="uniform"))
    model.add(Activation('relu'))
    model.add(Dense(7, input_dim=x_train.shape[1], kernel_initializer="uniform"))
    model.add(Activation('relu'))
    model.add(Dense(2, input_dim=x_train.shape[1]))
    model.compile(loss='mean_squared_error', optimizer='adam')
    model.fit(x_train, y_train, epochs=2000, batch_size=x_train.shape[0])
    #model.save_weights("./modelweight"+str(kk)+".h5")
    #model.save("./model"+str(kk)+".h5")
    #print("weights")
    weights = np.array(model.get_weights())
    print(weights[0].shape)
    print(weights)
    y_mean = data_mean[:, data_train.shape[1] - 2:]
    y_std = data_std[:, data_train.shape[1] - 2:]
    print("训练完毕")
```

代码 6：粒子群优化算法（PSO）

```
# from sko.GA import GA, GA_TSP
from keras import metrics
from sklearn.metrics import r2_score:
import pandas as pd
import numpy as np
import torch

def get_range():
    var_range_data = pd.read_excel("/home/zc/Desktop/zc/data/变量范围
/range_max_min.xlsx",
                                   usecols=['no', "位号", "min", "max"])
    var_range_data.set_index(["no"], inplace=True)
    weidian_28 = pd.read_csv("/home/zc/Desktop/zc/data/fit/main28_data_wu_fit.csv")
    weidian_28 = weidian_28.iloc[:, 1:-2]
    weidian_28 = weidian_28.T
    weidian_28 = weidian_28.reset_index()
    df_merge = pd.merge(weidian_28, var_range_data, left_on='index', right_on="位号",
how="left")
    df_merge.set_index(["index"], inplace=True)
    df_merge = df_merge.iloc[:, -2:]
    index_lst = list(df_merge.index)
    ron_index = index_lst.index("xingwanzhiRON")
    ron_value = 0#x_pso.iloc[0, ron_index]
```

```
    df_merge.loc["xingwanzhiRON", :] = 999999
    range_arr = np.array(df_merge)
    return range_arr

#优化目标函数
N, in_dim, H, H1, out_dim = 325, 39, 1000, 1000, 2

dataset = pd.read_csv('/home/zc/Desktop/zc/data/fit/main28_data_wu_fit.csv',sep=',')
dataset.set_index(["no"],inplace=True)
#dataset=dataset.iloc[:3,:]
print(dataset.head())
y = dataset.iloc[:,-2:]
print(y.shape)
X = dataset.iloc[:,0:-2]
print(X.shape)
# XGBoost 训练过程
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
other_params = {'learning_rate': 0.1, 'n_estimators': 4000, 'max_depth': 5,
'min_child_weight': 1, 'seed': 0, 'subsample': 0.8, 'colsample_bytree': 0.8, 'gamma': 0,
'reg_alpha': 0, 'reg_lambda': 1}
model =
MultiOutputRegressor(xgb.XGBRegressor(objective='reg:squarederror',**other_params))#*
*other_params
model.fit(X_train, y_train)
# 对测试集进行预测
ans = model.predict(X_test)
'''MSE_loss = metrics.mean_squared_error(ans[:,0], y_test.iloc[:,0])
RMSE_loss = np.sqrt(metrics.mean_squared_error(ans[:,0], y_test.iloc[:,0]))
MAE_loss = metrics.mean_absolute_error(ans[:,0], y_test.iloc[:,0])
print(MSE_loss.numpy(),RMSE_loss,MAE_loss.numpy())'''
#上下限
max_min_arr = get_range()
uplim=list(max_min_arr[:,1])
lwlim=list(max_min_arr[:,0])
def goal(ron_r, model):
    def f(x_pre):
        x_pso = pd.DataFrame([x_pre], columns=range(28))
        y_pred=model.predict(x_pso)
        ron = y_pred[0,0]
        s = y_pred[0,1]
        loss = 0
        if s < 0:
            s = 0
        if s > 5:
```

```
            loss += (s - 5) ** 2
        if ron < 0:
            ron = 0
        if ron > ron_r * 0.7:
            loss += (ron - ron_r * 0.7) ** 2
        return loss
    return f
new_loss_lst=[]
err_list=[]
new_s_lst=[]
outpath="/home/zc/Desktop/zc/data/youhua/var5.csv"
out=open(outpath,'w+')#len(y.index)
for i in range(325):
    x_pre=X.iloc[i,:]
    y_test = y.iloc[i,0]
    x_pre=list(x_pre)
    lwlim[8]=x_pre[8]
    uplim[8] = x_pre[8]+0.01
    #print(uplim)
    ron_r = float(y.iloc[i, 0])
    pso = PSO(func = goal(ron_r, model), dim=28, pop=40, max_iter=50, lb=lwlim, ub=uplim)
    pso.run()
    if pso.gbest_y == 0:
        print('save: '+str(i))
        r = list(pso.gbest_x)
        x_pso = pd.DataFrame([r], columns=range(28))
        y_pred = model.predict(x_pso)
        new_loss=y_pred[0,0]
        percent=(y_test-y_pred[0,0])/y_test
        new_loss_lst.append(new_loss)
        err_list.append(percent)
        new_s_lst.append(y_pred[0,1])
        print(new_loss)
        print(percent)
        out.write("\t".join(str(i) for i in r) + '\n')
    else:
        new_loss_lst.append(np.nan)
        err_list.append(np.nan)
        new_s_lst.append(np.nan)
        out.write('\n')
    print(i)
dataset["alter_loss"]=new_loss_lst
dataset["alter_s"]=new_s_lst
dataset["alter_err"]=err_list
```

```
dataset.to_csv("/home/zc/Desktop/zc/data/youhua/youhua5.csv",sep="\t")
print("finished")
out.close()
```

代码 7：辛烷值和硫的轨迹

```
import pandas as pd
import matplotlib.pyplot as plt
import xgboost as xgb
from xgboost import plot_importance
from matplotlib import pyplot as plt
dataset =
pd.read_csv('C://Users\86188\Desktop\zcc2\zc\data//fit//main28_data_wu_fit.csv',sep='
,')
dataset.set_index(["no"],inplace=True)
print(dataset.head())
y = dataset.iloc[:,-2:]
print(y.shape)
X = dataset.iloc[:,0:-2]
print(X.shape)
other_params = {'learning_rate': 0.1, 'n_estimators': 5000, 'max_depth': 5,
'min_child_weight': 1,
                'seed': 0, 'subsample': 0.8, 'colsample_bytree': 0.8, 'gamma': 0,
'reg_alpha': 0, 'reg_lambda': 1}
model =
MultiOutputRegressor(xgb.XGBRegressor(objective='reg:squarederror',**other_params))#*
*other_params
model.fit(X, y)

# 对测试集进行预测

#x1=pd.DataFrame(X_train, columns=range(28))
ans = model.predict(X)
#ans = model.predict(X.iloc[1:2,:])
print(ans)
print(type(ans))
y_real=y.iloc[1:]
print(type(y_real))
print(y_real)
print("loss R2 得分:", r2_score(ans[:,0], y.iloc[:,0]))
print("S  R2 得分:", r2_score(ans[:,1], y.iloc[:,1]))
var_path="C://Users\86188\Desktop\zcc2\zc\data/youhua/var2.csv"
var=open(var_path,"r")
k=0
new_list=[]
```

```
for line in var:
    if(k==3):
        line=line.strip()
        new_list=line.split("\t")
        break
    k=k+1
print(" new file: ")
new_list = [float(x) for x in new_list]
print(new_list)
print(len(new_list))
print("raw file: ")
raw=pd.read_csv("C://Users\86188\Desktop\zcc2\zc\data//fit/main28_data_wu_fit.csv",se
p=',')
raw.set_index("no",inplace=True)
raw=raw.iloc[:,0:-2]
index_list=list(raw)
#print(index_list)
raw_list=list(raw.iloc[112,:])
raw_list = [float(x) for x in raw_list]
print(raw_list)

print("deta file: ")
var_range_data = pd.read_excel("C://Users\86188\Desktop\zcc2\zc\data/变量范围
/range_max_min_deta.xlsx",usecols=[ "位号","deta"])
var_range_data.set_index("位号",inplace=True)
deta_list=[]
for index in index_list[:]:
    if(index!="xingwanzhiRON"):
        deta_list.append(abs(var_range_data.loc[index,"deta"]))
    else:
        deta_list.append(0)
deta_list = [float(x) for x in deta_list]
print(deta_list)
print("---------------------------")
print(new_list)
print(deta_list)
print(raw_list)
steps=0
beishu_list=[]
loss_list=[]
s_list=[]#(chazhi.count(0)!=len(chazhi)
chazhi = []
for i in range(len(new_list)):
    chazhi.append(float(new_list[i]) - float(raw_list[i]))
```

```
print(chazhi)
for i in range(len(chazhi)):
    if(deta_list[i]==0):
        beishu_list.append(0)
    else:
        beishu_list.append(int(chazhi[i]/deta_list[i]))
def f(i):
    if(i>0):
        return 1
    if(i<0):
        return -1

while(beishu_list.count(0)!=28):
    this_list=[0]*28
    for i in range(len(beishu_list)):
        if(beishu_list[i]==0):
            raw_list[i]=new_list[i]
        else:
            this_list[i]=deta_list[i]*f(beishu_list[i])
    #更新操作变量
    steps=steps+1
    for i in range(len(raw_list)):
        raw_list[i]=raw_list[i] + this_list[i]

    for i in range(len(beishu_list)):
        if (beishu_list[i]>0):
            beishu_list[i]=beishu_list[i]-1
        if (beishu_list[i] < 0):
            beishu_list[i] = beishu_list[i]+1
    print(new_list)
    print(raw_list)
    print(deta_list)
    print(beishu_list)
    print("======================")
    x_pre = pd.DataFrame([raw_list], columns=range(28))
    y_pred = model.predict(x_pre)
    loss_list.append([steps,89.4-y_pred[0,0]])
    s_list.append([steps, y_pred[0, 1]])
x_pre = pd.DataFrame([new_list], columns=range(28))
y_pred = model.predict(x_pre)
loss_list.append([steps,89.4-y_pred[0,0]])
s_list.append([steps, y_pred[0, 1]])
arr_loss=np.array(loss_list)
arr_s=np.array(s_list)
```

```
'''plt.plot(arr_loss[:,0],arr_loss[:,1],color='r',label="RON")
plt.xlabel('Steps')
plt.ylabel('RON')
plt.legend()
plt.show()'''
plt.plot(arr_s[:,0][:50],arr_s[:,1][:50],color='b',label="S")
plt.xlabel('Steps')
plt.ylabel('Sulfur')
plt.legend()
plt.show()
```