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	东北大学
参赛队号	**20101450030**
队员姓名	**1.** 王哲
	2. 卢鸿浩
	3. 徐博天

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 降低汽油辛烷值损失模型的研究
——使用Lasoo回归模型对汽油辛烷值的建模与优化

摘 要：

辛烷值是反映汽油燃烧性能的最重要指标，有效降低汽油中辛烷值的损失具有重要的意义。本文先对原始数据进行预处理，再采用相关系数法对 325 个样本数据进行特征选择，并通过 Lasso 回归进一步筛选特征，并得到线性回归模型来对辛烷值的损失进行预测，之后根据启发式思路，对参数多轮次分步优化，从而给出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，并以图形的形式展示了 133 号样本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。

针对问题一，需要对 285 和 313 样本的原始数据进行预处理，根据样本确定方法，对 325 个样本中数据全部为空值的位点进行删除操作，针对空值可使用均值插值法进行填充处理，针对异常值首先使用拉依达准则挑选出异常值，最后再计算出每个特征的均值填充到附件一中。

在问题二中，为了筛选出具有独立性和代表性的变量，采用了相关系数法来进行特征选择，通过调整确定相关系数阈值为 0.5 得到相关性较高的 56 个特征，并结合方差选择法最终选取出 16 个特征与 11 个不包括产品性质的固定变量共同作为建模的主要变量。

问题三主要建立了 Lasso 回归预测模型，同时进行进一步的特征选择，根据回归系数向量中 0 的个数，将特征数量由 16 个减小为 12 个。并用 50%的样本数据对模型进行训练，其余 50%用作测试。根据模型中得到的信息准则 AIC 与调整参数λ的关系，从而得到预测效果最好的λ值来对辛烷值的损失进行预测，引入确定系数 R-squared 和均方差 MSE 来对模型进行评价。最终得到测试集预测效果 R-squared=0.9599，MSE=0.0700，说明建立的线性回归模型对辛烷值损失的预测具有很好的效果。

对于问题四，先获取了产品硫含量的 Lasso 回归预测模型，把问题抽象为一个离散优化问题。然后采用启发式的思路，对参数多轮次分步优化，对 325 个样本，进行 100 次迭代，辛烷值的损失值降幅平均可达 36.4%，最后给出了辛烷值损失降低大于 30%的样本的变量调整情况和优化前后辛烷值的损失值及损失降幅。

问题五仍按照问题四中的方法，多轮次分步优化，对 133 号样本进行 100 轮优化调整，绘制了产品辛烷值和硫含量随迭代轮次的变化轨迹，并给出了每轮优化中的变量调整情况。优化调整后， 辛烷值的损失由 1.31 降低到 0.63，辛烷值损失的降幅达到 52.2%。硫含量由 1.96μg/g 降低为 3.2μg/g。

关键词：特征选择、相关系数法、Lasso 回归、预测

目 录

一、问题重述 3
1.1 问题背景 3
1.2 问题提出 3
1.2.1 目标 3
1.2.2 问题 3
二、问题分析 4
三、模型假设 5
四、符号说明 5
五、模型的建立与求解 5
5.1 问题一数据的整定 5
5.2 问题二模型的建立与求解 6
5.2.1 特征选择模型的建立 6
5.2.2 特征选择模型的求解 7
5.3 问题三模型的建立与求解 7
5.3.1 Lasso 回归预测模型的建立 7
5.3.2 Lasso 回归预测模型的求解 9
5.3.3 Lasso 回归预测模型的验证 10
5.4 问题四模型的建立与求解 11
5.4.1 产品硫含量的预测模型的建立 11
5.4.2 优化辛烷值损失的模型 11
5.4.3 模型的求解 11
5.5 问题五求解 13
六、模型的评价与推广 14
6.1 模型的评价 14
6.1.1 模型的优点 14
6.1.2 模型的缺点 14
6.2 模型的推广 15
参考文献 15
附录 15
附录 1：问题二的程序 15
附录 2：问题三的程序 18
附录 3：问题四的程序 21
附录 4：问题五的程序 24

一、问题重述

1.1 问题背景

随着小型车辆数目的日益增加，对汽油的需求量也大大增加，而汽油燃烧产生的尾气排放对大气环境有重要影响。为此，世界各国都制定了日益严格的汽油质量标准。汽油清洁化重点是降低汽油中的硫、烯烃含量，同时尽量保持其辛烷值。

由于我国原油对外界依存度高，且大部分是中东地区的含硫和高硫原油。这部分重油含硫量高，难以直接利用。因此，我国在大力发展以催化裂化为核心的重油轻质化工艺技术，将重油转化为汽油、柴油和低碳烯烃，超过 70%的汽油是由催化裂化生产得到，因此成品汽油中 95%以上的硫和烯烃来自催化裂化汽油。故必须对催化裂化汽油进行精制处理，以满足对汽油质量要求。化工过程的建模一般是通过数据关联或机理建模的方法来实现的，取得了一定的成果。但是由于炼油工艺过程的复杂性以及设备的多样性，它们的操作变量之间具有高度非线性和相互强耦联的关系，而且传统的数据关联模型中变量相对较少、机理建模对原料的分析要求较高，对过程优化的响应不及时，所以效果并不理想。

辛烷值（以 RON 表示）是反映汽油燃烧性能的最重要指标，现有技术在对催化裂化汽油进行脱硫和降烯烃过程中，普遍降低了汽油辛烷值。辛烷值每降低 1 个单位，将造成很大的经济损失，如果能降低辛烷值的损失，则能够带来很大的经济效益。因此需要通过某些技术来降低某些石化企业的辛烷值损失。

1.2 问题提出

1.2.1 目标

依据从催化裂化汽油精制装置采集的 325 个数据样本（每个数据样本都有 354 个操作变量），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（欧六和国六标准均为不大于 10μg/g，但为了给企业装置操作留有空间，本次建模要求产品硫含量不大于 5μg/g）的前提下，尽量降低汽油辛烷值损失在 30%以上。

1.2.2 问题

问题一：数据处理：参考近 4 年的工业数据的预处理结果，依附件二中的“样本确定方法”对 285 号和 313 号数据样本进行预处理，并将处理后的数据分别加入到附件一中相应的样本号中，供下面研究使用。

问题二：筛选建模主要变量：根据附件一中 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，使之尽可能具有独立性、代表性，降维后的主要变量尽量在 30 个以下，并详细说明建模主要变量的筛选过程及其合理性。

问题三：采用上述样本和建模主要变量，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。

问题四：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，利用模型获得 325 个数据样本中，辛烷值（RON）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，其中，优化过程中原料、待生吸附剂、再生吸附剂的性质保持不变，以它们在样本中的数据为准。

问题五：模型的可视化展示：工业装置为了平稳生产，优化后的主要操作变量往往只能逐步调整到位，对 133 号样本保持原料性质、待生吸附剂和再生吸附剂的性质数据不变，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。

综合分析上述 5 个问题，给出本题的解题流程图如下所示：

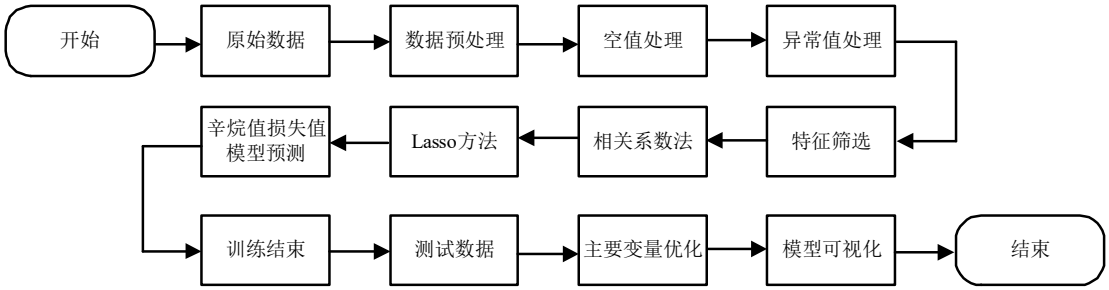

图 1 解题流程图

二、问题分析

问题一的分析：根据题目要求需要对 285 和 313 样本的原始数据进行预处理，根据样本确定方法，对 325 个样本中数据全部为空值的位点进行删除操作，针对空值可使用均值插值法进行填充处理，针对异常值，首先使用拉依达准则挑选出异常值，最后再计算出每个特征的均值填充保存到附件一中。

问题二的分析：为了筛选出具有独立性和代表性的变量，需要对 325 个特征进行提取。可选用方差选择法和相关系数法进行特征选择，结合两种方案来对 325 个特征进行筛选和提取，通过调整其相关参数，选出符合条件的操作变量。

问题三的分析：问题三主要采用附件一中提供的 325 个样本数据，需要采用问题二特征选择选出的操作变量来对辛烷值的损失预测。将 325 个样本数据划分为两部分，一半做训练集，一半做测试集。采用 Lasso 回归模型进一步筛选问题二中的操作变量，来获取辛烷值损失的预测模型。通过模型可得到 AIC 和调整参数的关系图，找到预测最好的模型调整参数来对辛烷值的损失进行预测，并计算确定系数(R-squared)和均方差(MSE)来表示预测结果和真实值的相关性，最后根据预测结果给出模型验证。

问题四的分析：本题需要用问题三的辛烷值损失预测模型对辛烷值损失进行优化，优化目标是需要保证产品硫含量低于 5μg/g 的情况下，尽量降低辛烷值损失。因为在优化过程中，需要保证产品硫含量保持在较低水平，一个产品硫含量的预测模型被提出，同样采用 Lasso 回归获取产品硫含量的预测模型。考虑到参数调整需遵循 Δ 值，并满足上下限，该问题可以抽象一个离散优化问题。因为已得出的硫含量的预测模型和辛烷值损失模型都是线性模型，而且问题规模不大，可采用启发式的思路，通过多轮次分步优化，每轮每个变量最多调整一个 Δ 值。之后对 325 个样本，进行多次迭代，给出辛烷值损失降低大于 30% 的样本的变量调整情况和优化前后辛烷值的损失值及损失降幅。

问题五的分析：问题五要求对 133 号样本的主要操作变量进行调整，各操作变量每次允许调整幅度值为固定值，且变量值不能超出所要求的范围。可同样按照问题四中的方法，多轮次分步优化，记录每轮调整的变量调整情况，并得到产品辛烷值和硫含量随迭代轮次的的变化轨迹。

三、模型假设

1.假设特征之间具有一定的多重共线性的性质。
2.假设删除异常数据或对部分残缺数据进行填补不会对结果造成影响。
3.假设选择出的主要特征能够反映原数据的特征。
4.假设优化后的操作变量能够逐步调整到位。
5.假设查找到的相关资料和模型参数具有一定的可信性与合理性。

四、符号说明

序号	符号	符号说明
1	var	方差
2	μ	均值
3	$corr$	相关系数
4	$P(\beta)$	惩罚函数
5	λ	调整参数
6	β	回归系数
7	$L(\beta)$	$Lasso$回归模型的损失函数
8	$X(n \times p)$	预测变量矩阵
9	$Y(n \times 1)$	变量向量即标签
10	$\beta(p \times 1)$	回归系数向量
11	RSS	残差平方和
12	df	自由度
13	$\hat{y}_i$	预测数据
14	$\overline{y_i}$	原始数据的均值
15	y_i	原始数据
16	M	样本总数
17	T	除产品性质外所有变量的下标集合
18	T_1	辛烷预测模型中操作变量的下标集合
19	T_2	是 354 个操作变量的下标集和
20	β_2	硫含量的预测模型参数
21	x_Min	第i个变量的取值下界
22	x_Max	第i个变量的取值上界
23	SSE	原始数据对应点的误差的平方
24	SSR	预测数据与原始数据均值之差的平方和
25	SST	原始数据和均值之差的平方和

五、模型的建立与求解

5.1 问题一数据的整定

问题一要求对 285 和 313 样本的原数据进行预处理，并将结果加入到 325 个样本数据

中对应的样本编号中。根据附件二给出的样本确定方法和原数据集可知，不良数据总共分为 2 类，针对不同类型的不良数据需要进行不同的操作，和操作的先后顺序进行预处理。预处理过程如下：

步骤一：空值处理。附件 3 中包含 285 号样本和 313 号样本的共 40 行数据，通过插值法对空值进行处理，计算每列除去空值后样本的平均值（因为原始数据为 2 小时内的采集数据，满足附件三前后两小时数据均值的填充），将均值填入空值的位置完成空值处理。因数据中异常空值均为 0，不是 nan，用 python 的 pandas 包处理前，我们需要将 0 全部替换为 nan，再调用 fillna()函数来将均值进行填充即可。

步骤二：去除异常值。由附件二给出的要求，用到拉依达准则（3σ准则），也称标准差法，适用于有较多组数据的时候。具体过程为：设对被测量变量进行等精度测量，得到$x_1, x_2, , \ldots\ldots, x_n$,算出其算术平均值$x$及剩余误差$v_i = x_i - x\ (i = 1,2, \ldots\ldots n)$，按贝塞尔公式算出标准误差$\sigma$，若某个测量值$x_b$的剩余误差$v_b(1 \le b \le n)$，满足$|v_b| = |x_b - x| > 3\sigma$，则认为$x_b$是含有粗大误差值的坏值，应予剔除。贝塞尔公式如下：

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

编写 python 代码实现该方法的接口，并应用在数据集中即完成了数据集异常值的处理。代码详见附录 1。

步骤三：预处理结果插入附件一对应位置。将附件三：285 和 313 号样本的原始数据的空值和异常值分别处理完后，取每列数据的均值分别替换附件一：325 个样本数据对应的样本号中，以便进行接下来问题的处理。

步骤四：第三步，预处理结果插入附件一对应位置。将附件三：285 和 313 号样本的原始数据的空值和异常值分别处理完后，取每列数据的均值分别替换附件一：325 个样本数据对应的样本号中，以便进行接下来问题的处理。

5.2 问题二模型的建立与求解

为了建立降低辛烷值损失模型涉及包括 7 个原料性质、2 个待生吸附剂性质、2 个再生吸附剂性质、2 个产品性质等变量以及另外 354 个操作变量（共计 367 个变量），工程技术应用中经常使用先降维后建模的方法，这有利于忽略次要因素，发现并分析影响模型的主要变量与因素。因此问题二需要我们根据附件一：325 个样本数据，通过降维的方法筛选出尽可能具有代表性、独立性 30 个特征变量，来作为辛烷值损失预测模型的输入。

5.2.1 特征选择模型的建立

对于问题二，我们选择方差选择法和相关系数法两种方法来做特征选择[1]，并将两种方法进行比对，选择较优的方法来进行以达到降维的效果。325 个样本数据处理好之后，为了减轻维数灾难问题和降低学习任务的难度，我们需要选择有意义的特征输入机器学习的模型进行训练，通常来说要从两个方面考虑来选择特征，如下：

a.特征是否发散

如果一个特征不发散，例如方差接近于 0，也就是说样本在这个特征上基本上没有差异，这个特征对于样本的区分并没有太大的作用。

b.特征与目标的相关性

显而易见，与目标相关性高的特征应当优先选择。特征与特征之间相关性高的，应当优先去除掉其中一个特征。

（1）方差选择法：通过编写 Python 程序来实现方差选择法，应用于 325 个样本数据集。第一次将方差阈值设为 0.6 时，筛选出 256 个特征，较难精确筛选出 30 个以内的特征，并调高方差阈值，经过调试，方差值在 100 时仍然有 156 个特征大于阈值，即满足筛

选条件的有 156 个特征。此方法对于此数据集特征选择效果较差，因此需结合其他方法再次做特征选择来作比较。

（2）相关系数法：由于方差系数法较难选出 30 个以内的特征，选用相关系数法[2]方法来进行特征选择。首先将相关系数阈值设置为 0.9，经过层层筛选得到了 205 个特征，较难选出 30 个以内的特征，于是修改相关系数为 0.85，0.8，0.6，0.5，最后当相关系数为 0.5 时筛选出了 56 个特征，并结合方差选择法最终选出 16 个操作变量，来作为辛烷值损失预测模型的输入。选出的 16 个操作变量如表 1 所示。

5.2.2 特征选择模型的求解

通过相关系数法再 354 个操作变量中选出的 16 个操作变量如下表 1 所示：

表 1 操作变量筛选结果

特征代号	特征中文名称
S-ZORB.CAL_H2.PV	氢油比
S-ZORB.PDI_2102.PV	反应过滤器压差
S-ZORB.PT_2801.PV	还原气压力
S-ZORB.TE_2103.PV	反应器上部温度
S-ZORB.TC_2101.PV	反吹氢气温度
S-ZORB.FC_2301.PV	D105 流化氢气流量
S-ZORB.TC_5005.PV	稳定塔下部温度
S-ZORB.TE_5102.PV	干气出装置温度
S-ZORB.TE_5202.PV	精制汽油出装置温度
S-ZORB.PT_9301.PV	蒸汽进装置压力
S-ZORB.FT_9301.PV	蒸汽进装置流量
S-ZORB.FT_1501.PV	新氢进装置流量
S-ZORB.FT_5104.PV	轻烃出装置流量
S-ZORB.FT_5101.PV	干气出装置流量
S-ZORB.FT_9101.PV	污油出装置
S-ZORB.TE_9001.PV	燃料气进装置温度

5.3 问题三模型的建立与求解

问题三主要采用附件一中提供的 325 个样本数据，以及上述问题二通过降维的方法的到的 27 个建模主要变量来建立辛烷值（RON）损失预测模型。本题主要通过编写 python 程序，建立 Lasso(Least absolute shrinkage and selection operator)回归模型来对辛烷值的损失进行预测。为对结果进行模型验证，我们在所有 325 个样本的数据集中随机选取 50%的样本作为训练集，其余的 50%样本作为测试集进行预测结果比对。

5.3.1 Lasso 回归预测模型的建立

(1)Lasso 回归介绍

目前比较成熟的集中数据挖掘技术有[3]：决策树判定法、支持向量机法、回归、关联规则分析法以及遗传算法等。每种方法都各有千秋，针对本题而言，我们选用了回归当中的 Lasso 回归方法来做预测。Lasso 方法[4]作为正则化方法的一种，从本质上讲是一种回归模型，在求解时加入 L1 范式，以回归系数的绝对值之和作为惩罚函数来压缩回归系数[5]，即：

$$P(\beta) = \lambda \sum_{j=1}^{p} |\beta_j| \quad (2)$$

在参数估计中，考虑到绝对值符号参与计算较为复杂，因此可将转为$\pm\beta_j$，其中的正负号与β_j保持一致。Lasso 损失函数为如下公式：

$$L(\beta) = \|Y - X\beta\|^2 + P(\beta) \quad (3)$$

与其他的正则化方法相比较，如岭回归则采用回归系数的平方和来作为惩罚函数，难以将不重要的预测特征的系数压缩为 0，同时也容易将较为重要的回归系数进行过度压缩，因此本题采用了 Lasso 回归预测模型，针对预测问题来讲，通常使用 Lasso 回归从众多特征中选取出主要的特征，解决特征之间的多重共线性的问题，这不仅保证了较高的预测精度，而且降低了模型训练过程中的计算量，同时简化了模型的复杂度，对于高维度的数据中具有大量的应用。Lasso 回归的损失函数不是连续可导的，求解需要数值算法，如坐标轴下降法或最小角回归法[6]。文中采用的是最小角回归法。

在问题二的基础上，由相关系数法得到的 16 个操作变量，Lasso 模型的程序对相关系数进行二次压缩，剔除掉系数为 0 的特征列，最终将 16 个操作变量压缩为 12 个操作变量。此时最终筛选出来的主要变量为 23 个，包括 7 个原料性质、2 个待生吸附剂性质、2 个再生吸附剂性质和 354 个操作变量提取的 12 个操作变量。从而更具有代表性和独立性。

（2）参数λ的选择

参数λ决定了回归系数被压缩的程度，不同的λ可能对结果产生不同的影响。目前主要有机器学习领域的交叉验证方法，以及信息标准的方法。考虑到复杂性，本题λ的取值依据信息准则。即针对不同的λ值，均采用 Lasso 回归模型计算得到信息标准的值，即 AIC(Akaike Information Criterion)和 BIC(Bayesian Information Criterion)。其具体计算公式分别如下所示：

$$AIC = nlog(RSS) + 2df \quad (4)$$

$$BIC = nlog(RSS) + dflog(n) \quad (5)$$

由于 AIC 是从预测角度，选择一个好的模型用来预测，因此本题只给出 AIC 和λ的关系图。如图 1 所示，通常我们会选择产生整体最小的信息标准时参数λ的值，由图可以看出整体产生最小的信息标准时λ取值在10^{-3}与10^{-2}之间，因此在程序中，通过几次调整，最终λ的值设定为 0.005。

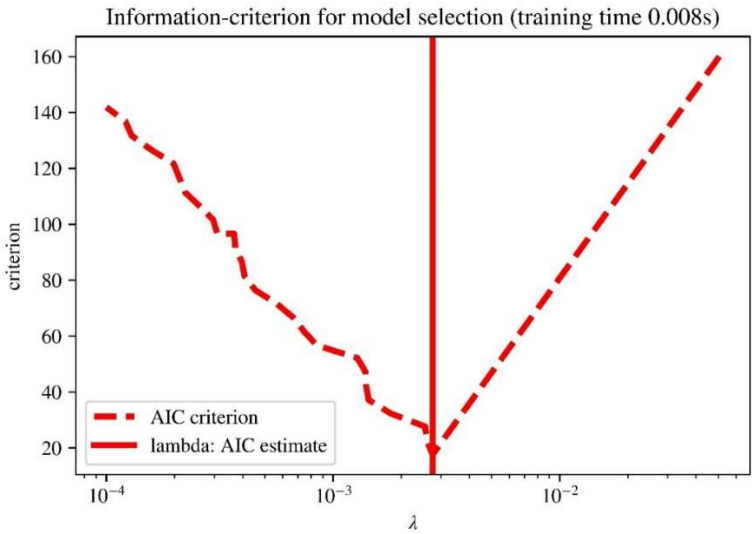

图 1　AIC 和λ的关系图

5.3.2 Lasso 回归预测模型的求解

通过编写 python 代码，导入处理好的数据，调用 sklearn 中的 Lasso 及其他相关的库。以处理好的 325 个样本，23 个建模主要操作变量作为整体的数据集，以产品性质中的辛烷值作为标签列，来对辛烷值进行预测，将预测结果与题目所给附件一原料性质中的辛烷值做差值，从而得到最终的辛烷值损失，并与产品性质中的原损失进行比对。

将处理好的 325 个样本和 23 个建模主要操作变量作为整体的数据集，经过 Lasso 回归模型的预测，经 Python 处理得到产品性质中的真实辛烷值和预测得到的辛烷值对比图，如图 2 所示。

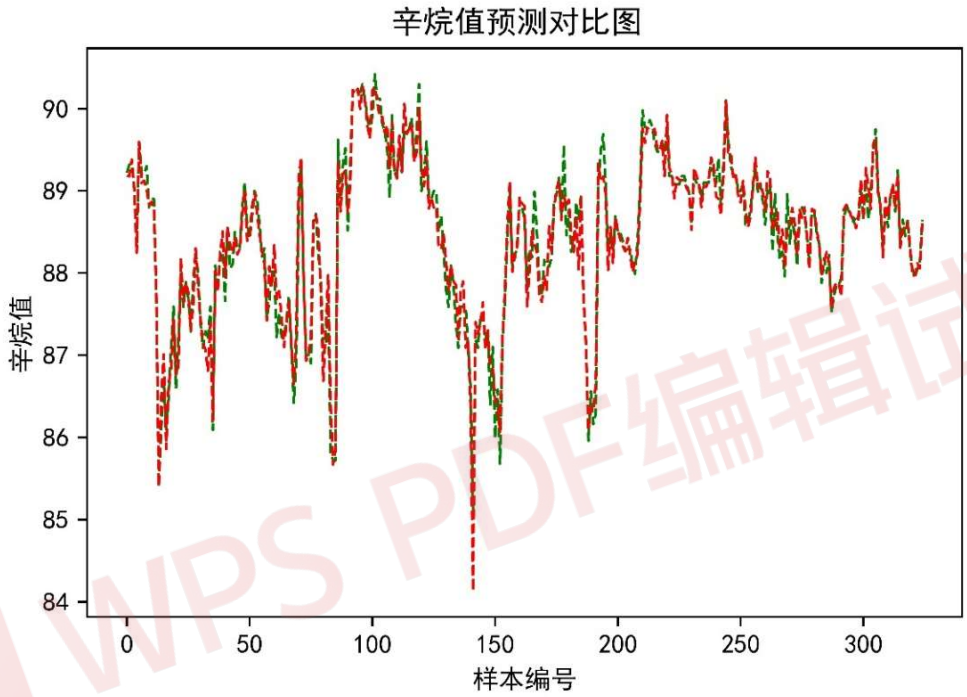

图 2 产品真实辛烷值与预测值对比图

图中绿色虚线表示真实的产品辛烷值，红色虚线表示经 Lasso 预测模型得到的预测辛烷值。由图可以看出 Lasso 回归预测模型得到的产品辛烷值预测与真实的产品辛烷值重合性很高，几乎一致。说明预测模型得到了较好的效果。

将原料辛烷值与模型得到的预测产品辛烷值做差，从而得到预测的额辛烷值损失，将真实的辛烷值损失与预测后的辛烷值损失进行比对，如下图 3 所示。

通过图 3 可以看出，原辛烷值损失波动较大，得到的预测后的辛烷值损失较小，说明 Lasso 回归预测模型能够降低噪声，具有较好的鲁棒性。

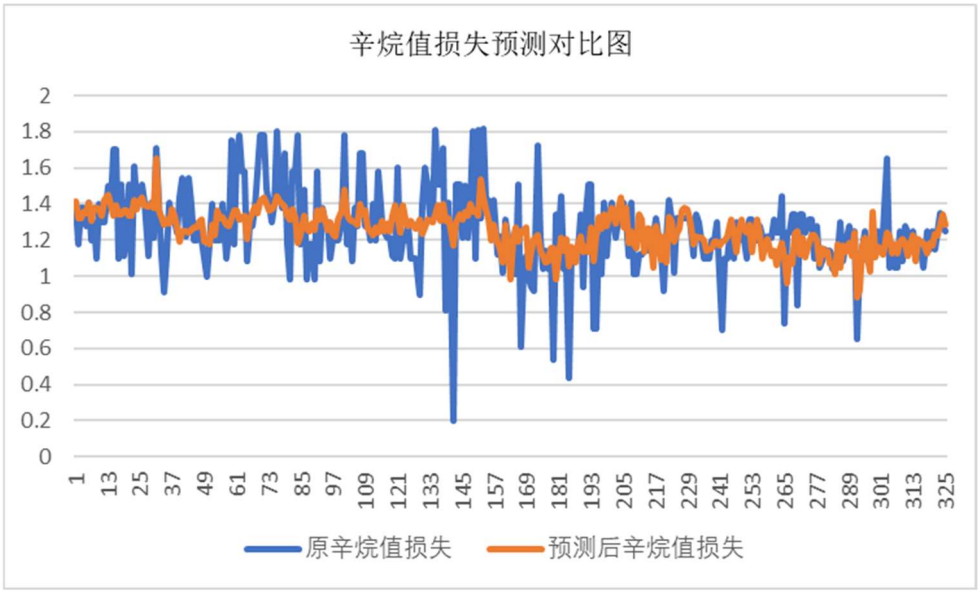

图 3 辛烷值损失预测对比图

5.3.3 Lasso 回归预测模型的验证

为准确评判该 Lasso 回归预测模型的预测效果，并对模型进行验证[6]，我们引入了$R-squared$ 和 MSE 这两个评价参数。

$R-squared$表示确定系数，越接近 1 说明相关性越强，主要由SSE、SSR和 SST以及拟合数据和决定,SSE，SSR和SST的计算公式分别如下：

$$SSE=\sum_{i=1}^{n}(y_i-\hat{y}_i)^2 \tag{6}$$

$$SSR=\sum_{i=1}^{n}(\hat{y}_i-\bar{y}_i)^2 \tag{7}$$

$$SST=\sum_{i=1}^{n}(y_i-\bar{y}_i)^2 \tag{8}$$

由$SST=SSE+SSR$，可得$R-squared$ 计算公式为：

$$R-squared=1-SSE/SST \tag{9}$$

MSE则表示均方误差，即真实值与预测值之差平方的期望，MSE的值越小，则说明预测效果越好，MSE具体公式为：

$$MSE=\frac{1}{M}\sum_{i=1}^{M}(y_i-\hat{y}_i)^2 \tag{10}$$

下面使用两种数据集划分方式获取两组模型参数，并给出预测效果。

1. 根据 Lasso 回归预测模型，得到以 325 个样本作为训练得到一组模型参数，以该模型参数对 325 个样本做预测，可得$R-squared$和MSE分别为：

$$R-squared=0.9599$$

$$MSE=0.03875$$

根据$R-squared$和MSE的值可以看出 Lasso 回归预测模型具有良好的预测效果。问题四和问题五中将采用这组模型参数。

2. 为了更好的验证 Lasso 回归预测模型，我们把整个数据集划分为两部分，用其中的 50%作训练集，剩下的 50%作测试集，对训练集和测试集进行预测。此时得到训练集的$R-squared=0.9369$，$MSE=0.02789$；测试集的$R-squared=0.9485$，$MSE=0.0700$。从而更说明了 Lasso 回归预测模型具有良好的预测效果。

5.4 问题四模型的建立与求解

5.4.1 产品硫含量的预测模型的建立

以 325 个数据样中产品性质中的硫含量对应列的数据作为标签Y_2，以 365 个变量（包括 7 个原料性质、2 个待生吸附剂性质、2 个再生吸附剂性质以及另外 354 个操作变量）作为特征矩阵X，采取问题 3 中用到 lasso 回归，对产品硫含量的预测模型进行拟合，拟合出的模型参数记为β_2。

$$\mathrm{X}\beta_2 \approx Y_2 \tag{11}$$

得到的模型中筛选出 198 特征，β_2中为 0 的项对应的特征是剔除掉的特征。

5.4.2 优化辛烷值损失的模型

根据产品硫含量的预测模型和产品辛烷值值的预测模型。考虑到参数变量的须按步长调整，和变量参数 x 要满足范围要求，可以得到模型：

$$\min \quad \sum_i \beta^i x_i + \sum_j \beta^j x_j, x_i\text{是变量}, i \in T_1, j \in T_2 \tag{12}$$

$$s.t. \quad x\_Min_i \le x_i \le x\_Max_i, i \in T_1 \tag{13}$$

$$\sum_i \beta_2^i x_i \le 5, i \in T \tag{14}$$

$$x_i \in \{x_i + \mathrm{t}\Delta_i | \mathrm{t} \in \mathrm{Z}, \mathrm{Z}\text{ 是整数集}\}, i \in T_1 \tag{15}$$

5.4.3 模型的求解

此问题是一个离散优化问题，可以采用的方法有智能优化方法（如模拟退火算法[7]，遗传算法[8]等），或者采用分支定界法[9]等精确求解算法。考虑到求出的硫含量的预测模型和辛烷值损失模型都是线性模型，问题规模不大，且为了方便实际生产的应用，此处采用启发式的思路，采用多轮次分步优化，在每一轮中，对 x_i（$i \in T_1$）变量进行遍历，并调整变量x_i（增大或减小一个Δ_i），使其满足变量参数 x 的范围要求的情况下，尽量提高产品辛烷值含量并降低产品硫含量。伪代码如图 4 所示，具体步骤的流程图如图 5 所示。

```
for iter in range(Max_iter):
    for i in T1:
        step = 0
        if (β1^i>0) and (β2^i<0):
            step = 1
        else if (β1^i<0) and (β2^i>0):
            step = -1
        x_i = x_i + step*Δ_i
        if (x_i< x_Min_i) or (x_i> x_Max_i)
            x_i= x_i - step*Δ_i
```

图 4 伪代码

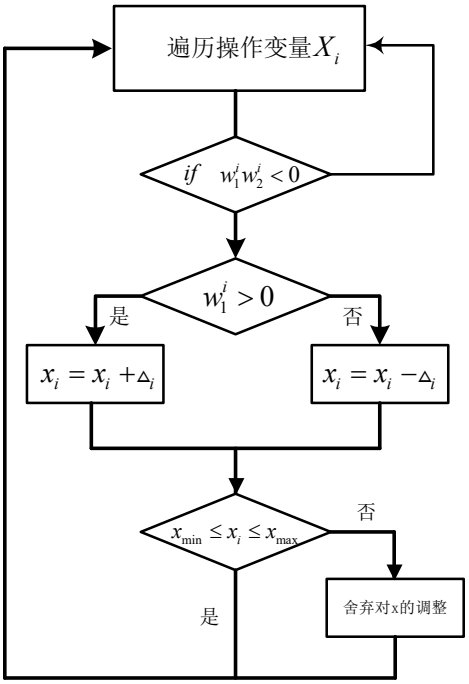

图 5 流程图

对 325 个样本，进行 100 次迭代后，可得到硫含量低于 5μg/g，且 RON 值损失降低 30%的样本有 225 个。优化前，325 个样本的 RON 的损失值的百分比总和为 407.8，优化后 RON 的损失值的百分比总和为 236.4。总体上，RON 的平均损失值降幅达 36.4%。325 个样本的优化情况见附录 2（“附件 2：优化过程中变量调整情况.xlsx”），附录 2 列出了所有样本的主要变量的累计调整情况。降幅大于 30%的数据样本的主要变量的累计调整情况见附件 3（“附件 3：RON 损失降幅大于 30%的样本的变量调整况.xlsx”）。优化前后的 RON 损失值和 RON 损失降幅情况见附件 1（“附件 1：优化前后 RON 的损失值及损失降幅.xlsx”）。

从附件 3 可以分析得到：使得 RON 损失大幅降低所调整的特征列有 7 个：S-ZORB.PDI_2102.PV，S-ZORB.TE_2103.PV，S-ZORB.FC_2301.PV，S-ZORB.TE_5102.PV，S-ZORB.TE_5202.PV，S-ZORB.FT_1501.PV，S-ZORB.FT_5104.PV，S-ZORB.FT_9101.PV。调整的平均幅度如表 2 所示。

表 2 使得 RON 损失大幅降低所调整的特征列

主要变量	平均调整幅度（调整单位是 Δ 值）	Δ 值
S-ZORB.PDI_2102.PV	-11.928	1
S-ZORB.TE_2103.PV	-13.919	1
S-ZORB.FC_2301.PV	6.0267	50
S-ZORB.TE_5102.PV	-11.196	1
S-ZORB.TE_5202.PV	-4.589	1
S-ZORB.FT_1501.PV	-2.696	50
S-ZORB.FT_5104.PV	53	50
S-ZORB.FT_9101.PV	-0.43303	5

5.5 问题五求解

问题要求对 133 号样本的主要操作变量（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）进行调整，各操作变量每次允许调整幅度值为固定值，且变量值不能超出所要求的范围。

仍按照问题四中的方法，多轮次分步优化，在每一轮中，对模型 2 中变量进行遍历，每一轮调整中，每个变量最多变化一个 Δ 值。对 133 号样本进行 100 轮调整，附件 4（“附件 4：133 号样本的变量调整情况.xlsx”）给出了每轮调整的变量调整情况，图 6 给出了中产品辛烷值随迭代轮次的变化轨迹，图 7 给出了硫含量随迭代轮次的变化轨迹。图中可以看出当调整轮数到达 80 次时，产品硫含量和产品 RON 含量趋于稳定，其中 RON 含量趋于一个较高的值，更接近于原料的 RON，RON 的损失由 1.31 降低到 0.63，优化后 RON 损失的降幅达到 52.2%。硫含量趋于一个较低的值，约为 1.96，低于 5μg/g，也低于优化前的原料硫含量——3.2μg/g。

表 3 优化前后 RON 的变化情况

原料 RON	优化前的产品 RON	优化后的产品 RON	优化前的 RON 损失	优化后的 RON 损失	优化后 RON 损失的降幅
89.4	88.09	88.77	1.31	0.63	0.522

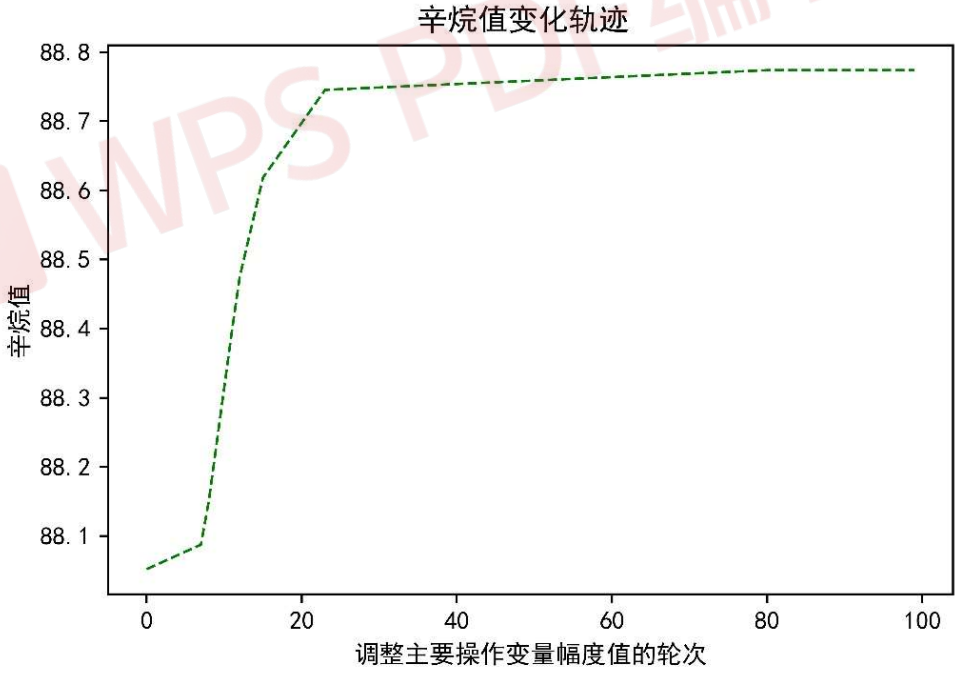

图 6 辛烷值变化轨迹

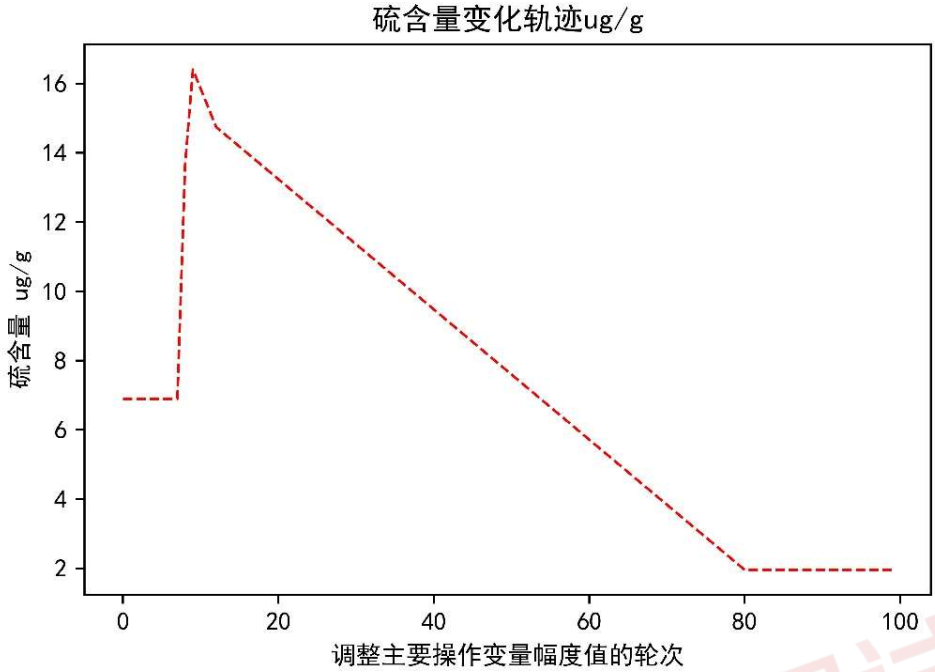

图 7 硫含量变化轨迹

六、模型的评价与推广

6.1 模型的评价

6.1.1 模型的优点

1．对于问题二，计算样本数据各个特征之间的相关系数，减少了样本特征之间的多重共线性关系。

2．对于问题三，我们采用了 Lasso 回归预测模型来对辛烷值的损失进行预测，同时针对预测结果分别计算出了相关系数和均方误差，并且都得到了较好的值，模型简单易懂，能够保证预测结果的准确性。

3．对于问题三，将预测得到辛烷值的损失与真实的辛烷值损失放到同一个图中进行对比，发现预测后的辛烷值损失变化起伏要小于真实的辛烷值损失变化，说明该模型具有较好的鲁棒性。

4．对于问题四，可以看到该优化模型是对保证硫含量低于 5μg/g 的情况下降低 RON 损失有很好的效果，能带来很好的经济效益，并且遵循了每个变量每次调整一个 Δ 值的要求，很符合工业生产的要求。

6.1.2 模型的缺点

1．相关系数法对大功率噪声不鲁棒，只对线性相关敏感，如果是非线性相关，比如平方关系，相关系数可能很小。

2．问题四采用启发式的思路，所求出的解不一定是最优解。虽然整体效果很好，仍有极个别的样本效果不佳。

3．Lasso 回归预测模型得到的不是精确解。

6.2 模型的推广

相关系数法能够通过设置相关系数阈值来对多维特征的数据进行特征选择，并能达到较优的结果，可应用在多个行业的数据清洗过程；Lasso 模型对特征非常多的数据，需要进行特征压缩，以及对数据进行预测时，Lasso 都能起到较好的效果；问题四中的启发式思路针对本问题能达到较好的优化，在未来面向汽油生产类似的多步骤大规模的工业行业，如乙烯的加工，煤的加工，钢铁的生产等，都可以采用该模型进行优化调整，提升产品性能，获取更高的经济效益。

参考文献

[1]马利星,胡敏.特征工程研究领域发展趋势的可视化分析[J].北京信息科技大学学报(自然科学版),2020,35(04):32-37.

[2]冀树德,张兴刚,孙学东,闫瑞琦,高海涛.基于相关系数法的柴油机动态特征提取[J].汽车技术,2014(12):18-22.

[3]王雪辉. 基于数据挖掘技术的股票预测研究与应用[D].海南大学,2020.

[4]张沥今,魏夏琰,陆嘉琦,潘俊豪.Lasso 回归：从解释到预测[J/OL].心理科学进展:1-15[2020-09-20].

[5]Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. J. Roy. Statist. Soc. Ser. B. 58 267–288. MR1379242

[6]Efron B , Hastie T , Tibshirani J R . Least Angle Regression[J]. Annals of Stats, 2004, 32(2):407-451.

[7] Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983). Optimization by simulated annealing. Science, 220: 671-680.

[8]Goldberg D E . Genetic Algorithms In Search, Optimization, and Machine Learning[J]. Ethnographic Praxis in Industry Conference Proceedings, 1988, 9(2).

[9]Padberg M , Rinaldi G . A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Traveling Salesman Problems[J]. SIAM Review, 1991, 33(1).

附录

附录 1：问题二的程序

```
# 首先加载数据分析常用库
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
from sklearn import tree
from sklearn.ensemble import ExtraTreesRegressor, GradientBoostingRegressor

'''读取文件'''
```

```
df = pd.read_excel('附件三：285 号和 313 号样本原始数据.xlsx', sheet_name='操作变量')
df285 = df.iloc[2:, 1:] #样本 285 的 40 组原始数据
df313 = df.iloc[43:83, 1:]   #样本 313 的 40 组原始数据
df = pd.read_excel('附件一：325 个样本数据.xlsx').iloc[1:,16:]       #所有控制变量
df1 = pd.read_excel('附件一：325 个样本数据.xlsx').iloc[1:,2:16]     #主要变量(不变)

'''数据清洗'''
def clean(df):
    #  看看数据情况
    # df.info()
    #  查看数据前 5 行
    # print(df.head())
    #  查看数据集详细描述值（最大值，均值，方差等）
    # print(df.describe())

    #  数据清洗
    #  探索是否有缺失值
    # print('\n 缺失值处理前>>>', df.isnull().sum(), '\n>>>', df.shape)
    #  方式一：均值填充
    df = df.replace(0, np.nan)
    df = df.fillna(df.mean())
    return df

'''标准差法'''
def std(df):
    a = df.values
    m = 0
    N = a.shape[0]
    for i in range(a.shape[1]):
        for j in range(a.shape[0]):
            m += a[j][i]
        M = m / N
        for j in range(a.shape[0]):
            S = np.sqrt(np.sum((a[j][i] - M) ** 2) / N)
        L, R = M - 3 * S, M + 3 * S #阈值

        #判断当前列的每个值是否在阈值范围内，不在则替换为均值
        for j in range(a.shape[0]):
            if a[j][i] < L or a[j][i] > R:
                a[j][i] = M
    return a

'''方差选择法'''
def VarianceThreshold(df, threshold):
```

```
    dfc = df.copy()
    # print('>>>特征名：\n', dfc.columns.tolist())
    # 1 求方差
    var = np.sum(np.power(np.matrix(dfc.values) - np.matrix(dfc.mean()), 2), axis=0) /
(dfc.shape[0] - 1)
    T = []
    # 2 筛选大于阈值的特征
    for index, v in enumerate(var.reshape(-1, 1)):
        if v > threshold:
            T.append(index)
    dfc = dfc.iloc[:, T]
    return var, dfc

'''相关系数法'''
def corr_selector(df):
    dfc = df.copy()
    CORR = np.zeros((dfc.shape[1], dfc.shape[1]))
    delete, save = [], []
    for i in range(dfc.shape[1]):
        if dfc.columns.tolist()[i] not in delete:
            save.append(dfc.columns.tolist()[i])
        for j in range(i+1, dfc.shape[1]):
            # 计算特征与特征之间的相关系数
            cov = np.sum((dfc.iloc[:,i]-dfc.iloc[:,i].mean()) * (df.iloc[:,j]-df.iloc[:,j].mean()))
            std = np.sqrt(np.sum((df.iloc[:,i]-df.iloc[:,i].mean())**2)) *
np.sqrt(np.sum((df.iloc[:,j]-df.iloc[:,j].mean())**2))
            corr = cov/std
            CORR[i][j] = corr
            # 筛选掉高线性相关两两特征中的某一个特征
            if (np.abs(corr) > 0.5) and (dfc.columns.tolist()[j] not in delete):
                delete.append(dfc.columns.tolist()[j])
    dfc_ = dfc[save].copy()
    return CORR, dfc_

if __name__ == "__main__":
    df = clean(df)  #附件一空值处理
    #相关系数法进行特征选择（降维）
    corr, dfc = corr_selector(df)
    # print(corr)
    # print(dfc)
    #将主要变量和筛选出来的操作变量合并成一个数据集
    dfc = dfc.iloc[:, 0:16]  # 选择前 17 个特征
    res = pd.concat([df1, dfc], axis=1, ignore_index=True)
    print(res)
```

```
# 定义路径 IO 并写入结果集，以便接下来作为损失预测模型的输入
IO = "数据清洗结果.xlsx"
res.to_excel(IO, sheet_name="筛选特征后的数据")
```

附录 2：问题三的程序

（1）预测

```
import time
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
from sklearn.linear_model import LassoCV, LassoLarsCV, LassoLarsIC
# from sklearn.linear_model import Ridge, RidgeCV     # Ridge 岭回归,RidgeCV 带有广义交叉验证的岭回归
import matplotlib.pyplot as plt    # 可视化绘制
import matplotlib as mpl
from sklearn import linear_model
from sklearn.metrics import mean_squared_error #均方差
from sklearn.metrics import mean_absolute_error #平方绝对误差
from sklearn.metrics import r2_score#R square    R_square 一般用于回归中评估模型的好坏程度，其值越接近 1，代表模型性能越好
plt.rcParams['font.sans-serif']=['SimHei']
plt.rcParams['axes.unicode_minus']=False
mpl.rcParams.update(
{
'text.usetex': False,
'font.family': 'stixgeneral',
'mathtext.fontset': 'stix' ,
}
)
# from pylab import *
# mpl.rcParams['font.sans-serif']=['SimHei']
X1 = pd.read_excel("data.xlsx",header = None) #读取 csv 文件  所有的特征
#X = pd.read_csv("data.csv",header = None) #读取 csv 文件
#y = pd.read_csv("y.csv",header = None) #读取 csv 文件  训练集
y = pd.read_csv("y.csv",header = None) #读取 csv 文件  所有的标签
# #切割数据样本集合测试集生成 X 和 y 矩阵
X1 = np.array(X1)
X_test = np.array(X1[0:165,:])#训练集  0-226
X_train = np.array(X1[165:,:])#测试集  227-325
y = np.array(y)
y_test = np.array(y[0:165,:]) #训练集标签
y_train = np.array(y[165:,:]) #训练集标签
```

```
reg = linear_model.Lasso(alpha = 0.002)#辛烷的 alpha
#reg = linear_model.Lasso(alpha = 0.002)#S 的 alpha
reg.fit(X_train,y_train) #X_train  是训练集
# reg.fit(X,y) #X_train  是训练集
W = reg.coef_
W = np.array(W)
w0 = reg.intercept_

print('w0 = ',w0)
#np.savetxt("W.csv", W, delimiter=',')
#np.savetxt("S_W.csv", W, delimiter=',')
sum = 0
EPSILON = 1e-4
for i in range(27):
    if(W[i]==0):
        sum = sum+1
        print(i)
    # if(W[i]!=0):
    #     print(i)
#print(reg.coef_)
print("------0 的个数-----",sum)
predicted = reg.predict(X_train)
# predicted = reg.predict(X)
predicted = np.array(predicted)
np.savetxt("predicted_y.csv", predicted, delimiter=',')
model_bic = LassoLarsIC(criterion='bic')
t1 = time.time()
# model_bic.fit(X, y)
model_bic.fit(X_train, y_train)
t_bic = time.time() - t1
alpha_bic_ = model_bic.alpha_

model_aic = LassoLarsIC(criterion='aic')
model_aic.fit(X_train, y_train)
# model_aic.fit(X, y)
alpha_aic_ = model_aic.alpha_

def plot_ic_criterion(model, name, color):
    criterion_ = model.criterion_
    plt.semilogx(model.alphas_ + EPSILON, criterion_, '--', color=color,
                 linewidth=3, label='%s criterion' % name)
    plt.axvline(model.alpha_ + EPSILON, color=color, linewidth=3,
                label='lambda: %s estimate' % name)
```

```
    plt.xlabel(r'$\lambda$')
    plt.ylabel('criterion')


plt.figure()
# plot_ic_criterion(model_aic, 'AIC', 'b')
plot_ic_criterion(model_bic, 'AIC', 'r')
plt.legend()
plt.title('Information-criterion for model selection (training time %.3fs)'
              % t_bic)
plt.savefig('lambda.jpg',dpi = 900)

#调用  训练集
mse = mean_squared_error(y_train,predicted)
r2 = r2_score(y_train,predicted)
# mse = mean_squared_error(y,predicted)
# r2 = r2_score(y,predicted)
print('r2 = ',r2)
print('mse = ',mse)

y_test_predicted = w0 + np.dot(X_test,W)
mse = mean_squared_error(y_test,y_test_predicted)
r2 = r2_score(y_test,y_test_predicted)
print('r2 = ',r2)
print('mse = ',mse)
```

（2）画图

```
import matplotlib.pyplot as plt   #  可视化绘制
import numpy as np
import pandas as pd
df = pd.read_csv("result_S_RON.csv",header = None) #读取训练集 csv 文件,新的 y_max
arr = np.array(df) #

plt.rcParams['font.sans-serif']=['SimHei']
plt.rcParams['axes.unicode_minus']=False
x = range(100)
plt.plot(x,arr[:,0],color='g',linewidth=1,linestyle='--')
plt.xlabel('调整主要操作变量幅度值的轮次') #X 轴标签
plt.ylabel("辛烷值") #Y 轴标签
plt.title("辛烷值变化轨迹  ") #标题
plt.savefig('辛烷值变化轨迹.jpg',dpi = 900)
plt.figure()
x1= range(100)
plt.plot(x1,arr[:,1],color='red',linewidth=1,linestyle='--')
```

```
plt.xlabel('调整主要操作变量幅度值的轮次') #X 轴标签
plt.ylabel("硫含量  ug/g") #Y 轴标签
plt.title("硫含量变化轨迹 ug/g") #标题
plt.savefig('硫含量变化轨迹.jpg',dpi = 900)
```

附录 3：问题四的程序

```
import numpy as np
import pandas as pd
import random

A = pd.read_csv("RON_A_325.csv", header=None)
A = np.array(A)
print("A_shape = ", A.shape)
w = pd.read_csv("W.csv", header=None)
w = np.array(w)
y_max = pd.read_csv("RON_y_input_325.csv", header=None)
y_max = np.array(y_max)
a_windows = pd.read_csv("特征范围.csv", header=None)
a_windows = np.array(a_windows)
y_label = pd.read_csv("RON_y_label_325.csv", header=None)
y_label = np.array(y_label)
steps = pd.read_csv("steps.csv", header = None)
steps = np.array(steps)

w0 = 9.75944051

ws = pd.read_csv("W_s.csv", header=None)
ws = np.array(ws)
As = pd.read_csv("select.csv", header=None)
As = np.array(As)
ys_label = pd.read_csv("yS_labels.csv", header=None)
ys_label = np.array(ys_label)
ws0 = -3.50607604

m, n = A.shape
print(m, n)
print(a_windows.shape)

A_adj = A
As_adj = As
y_adj = np.zeros([m, 1])
ys_adj = ys_label
print("(y_max[0]")
```

```
print(y_max[0])

location = [13, 16, 21, 24, 27, 30, 31, 35, 36, 37, 38, 39]

num_unchange = 11
num_feature = 23
record_feature_change = np.zeros([m, num_feature-num_unchange])

for roundd in range(100):
    j = 0   #  特征索引
    for row in range(m):
        a = A_adj[row][:]
        a_temp = a
        for i in range(num_feature - num_unchange):       # i 是变化的特征的索引
            if random.random() < 1:
                a_temp = a
                j = i + num_unchange       # j 是特征索引
                a_win = a_windows[:][i]
                if (w[j][0] > 0) and (a[j]-steps[i] < a_win[1]) and (ws[location[i]][0] <= 0):
                    a[j] += steps[i]       # int((a_win[1]-a[j])/steps[i]) *
                    As_adj[row][location[i]] = a[j]
                    record_feature_change[row][i] += 1
                elif (w[j][0] < 0) and (a[j]-steps[i] > a_win[0]) and (ws[location[i]][0] >= 0):
                    a[j] -= steps[i]       # int((a[j]-a_win[1])/steps[i]) *
                    As_adj[row][location[i]] = a[j]
                    record_feature_change[row][i] -= 1
                if np.dot(a, w)+w0 >= y_max[row][0] or np.dot(a, w)+w0 < y_label[row][0]:
                    a = a_temp
                    #print("j = ")
                    #print(j)
                    break
            else:
                break
        y_adj[row][0] = np.dot(a, w) + w0
        #print("y_adj", y_adj[row][0])
        #print("y_label", y_label[row][0])
        ys_adj[row][0] = np.dot(As_adj[row][:], ws) + ws0
        A_adj[row][:] = a

loss_orgin = y_max - y_label
loss_adj = y_max - y_adj
reduce_loss = loss_orgin-loss_adj
reduce_loss_rate = np.zeros([m, 1])
positive_rlr = 0
```

```
negetive_rlr = 0

print("m = ", m)
for i in range(m):
    reduce_loss_rate[i][0] = (loss_orgin[i]-loss_adj[i]) / loss_orgin[i]
    if reduce_loss_rate[i][0] > 0.3:
        positive_rlr += 1
    else:
        negetive_rlr += 1
sum_loss_orgin = sum(loss_orgin)
sum_loss_adj = sum(loss_adj)

print("m = ", m)
sum_n_Slow_RoNgood = 0
index_Slow_RoNgood = np.zeros([m, 1])
for row in range(m):
    if (ys_adj[row][0] <= 5) and (reduce_loss_rate[row][0] > 0.3):
        index_Slow_RoNgood[row][0] = 1
        sum_n_Slow_RoNgood += 1

print("sum_n_Slow_RoNgood = ", sum_n_Slow_RoNgood)

print("(y_max[0]")
print(y_max[0])
print("positive_rlr = ", positive_rlr)
print("negetive_rlr = ", negetive_rlr)
print("sum_loss_orgin = ", sum_loss_orgin)
print("sum_loss_adj = ", sum_loss_adj)
print("reduce of sum_loss = ", (sum_loss_orgin-sum_loss_adj)/sum_loss_orgin)

ys = np.hstack((y_label, y_adj))
ys = np.hstack((ys, y_max))
ys = np.hstack((ys, reduce_loss_rate))

np.savetxt("result.csv", A_adj, delimiter=',')

np.savetxt("ys.csv", ys, delimiter=',')

record_final_feature = A_adj

np.savetxt("record_feature_change.csv", record_feature_change, delimiter=',')
np.savetxt("record_final_feature.csv", record_final_feature, delimiter=',')

np.savetxt("ys_adj.csv", ys_adj, delimiter=',')
```

```
np.savetxt("index_Slow_RoNgood.csv", index_Slow_RoNgood, delimiter=',')

record_Slow_RoNgood = np.zeros([225, num_feature-num_unchange])
k = 0
for row in range(m):
    if (ys_adj[row][0] <= 5) and (reduce_loss_rate[row][0] > 0.3):
        record_Slow_RoNgood[k][:] = record_feature_change[row][:]
        k += 1
np.savetxt("record_Slow_RoNgood.csv", record_Slow_RoNgood, delimiter=',')
```

附录 4：问题五的程序

```
import numpy as np
import pandas as pd
import random

A = pd.read_csv("RON_133.csv", header=None)
A = np.array(A)
w = pd.read_csv("W.csv", header=None)
w = np.array(w)
y_max = 89.4
y_label = 88.09
w0 = 9.75944051

a_windows = pd.read_csv("特征范围.csv", header=None)
a_windows = np.array(a_windows)
steps = pd.read_csv("steps.csv", header = None)
steps = np.array(steps)

location = [13, 16, 21, 24, 27, 30, 31, 35, 36, 37, 38, 39]

ws = pd.read_csv("W_s.csv", header=None)
ws = np.array(ws)
As = pd.read_csv("S_W.csv", header=None)
As = np.array(As)
ys_label = 3.2
ws0 = -3.50607604

aw, bw = ws.shape
wz = 0
for i in range(aw):
    if ws[i][:] == 0 and i > 10:
        wz += 1
```

```
print("wz = ", wz)
aw, bw = ws.shape
wz2 = 0
for i in location:
    if ws[i][:] == 0 and i > 10:
        print(i)
        wz2 += 1
print("wz = ", wz2)

m, n = A.shape
print(m, n)
print(a_windows.shape)

A_adj = A
y_adj = np.zeros([m, 1])
ys_adj = ys_label
print("(y_max[0]")
print(y_max)

print("steps.shape = ")
print(steps.shape)
print("As.shape = ")
print(As.shape)

num_unchange = 11
num_feature = 23

roundd_MAX = 100
result_S_RON = np.zeros([roundd_MAX, 2])  # 记录硫和辛烷值
record_detail_feature_change = np.zeros([roundd_MAX, num_feature - num_unchange])  # 记录每轮的特征调整的情况
record_feature_change = np.zeros([1, num_feature-num_unchange])  # 记录每个特征调整的步数和
for roundd in range(roundd_MAX):
    j = 0  # 特征索引
    a = A_adj
    a_temp = a
    for i in range(num_feature - num_unchange):      # i 是变化的特征的索引
        if random.random() < 1:
            a_temp = a
            j = i + num_unchange      # j 是特征索引
            a_win = a_windows[:][i]
            print("j = ")
            print(j)
```

```
                print("i = ")
                print(i)
                if (w[j][0] > 0) and (a[0][j]-steps[i] < a_win[1]) and (ws[location[i]][0] <= 0):
                    a[0][j] += steps[i]        # int((a_win[1]-a[j])/steps[i]) *
                    As[location[i]][0] = a[0][j]
                    record_feature_change[0][i] += 1
                    record_detail_feature_change[roundd][i] += 1
                elif (w[j][0] < 0) and (a[0][j]-steps[i] > a_win[0]) and (ws[location[i]][0] >= 0):
                    a[0][j] -= steps[i]        # int((a[j]-a_win[1])/steps[i]) *
                    As[location[i]][0] = a[0][j]
                    record_feature_change[0][i] -= 1
                    record_detail_feature_change[roundd][i] -= 1
                if np.dot(a, w)+w0 >= y_max or np.dot(a, w)+w0 < y_label or np.dot(As.T, ws) +
ws0 > ys_adj:
                    a = a_temp
                    print("j = ")
                    print(j)
                    break
            else:
                break
        y_adj = np.dot(a, w) + w0
        print("start:")
        print(As.shape)
        print(ws.shape)
        ys_adj = np.dot(As.T, ws) + ws0
        print("y_adj", y_adj)
        print("y_label", y_label)
        A_adj = a
        #  记录数据
        result_S_RON[roundd][:] = [y_adj, ys_adj]



loss_orgin = y_max - y_label
loss_adj = y_max - y_adj
reduce_loss = loss_orgin-loss_adj
reduce_loss_rate = np.zeros([m, 1])
positive_rlr = 0
negetive_rlr = 0

reduce_loss_rate = (loss_orgin-loss_adj) / loss_orgin
if reduce_loss_rate>0:
    positive_rlr += 1
else:
```

```
        negetive_rlr += 1

print("(y_max[0]")
print(y_max)
print("positive_rlr = ", positive_rlr)
print("negetive_rlr = ", negetive_rlr)

print("reduce of sum_loss = ", (loss_orgin-loss_adj)/loss_orgin)

print("y_label = ", y_label)
print("y_adj", y_adj)
print("y_max", y_max)

print("ys_label = ", ys_label)
print("ys_adj", ys_adj)

np.savetxt("result_S_RON.csv", result_S_RON, delimiter=',')


record_feature_change_133 = np.vstack((record_feature_change,
a[0][num_unchange:num_feature]))

np.savetxt("record_feature_change_133.csv", record_feature_change_133, delimiter=',')

np.savetxt("record_detail_feature_change_133.csv", record_detail_feature_change, delimiter=',')
```