浙江工商大学第一次研究生
数学建模竞赛模拟

赛　　题	B 题
队伍编号	90
队员姓名	1.周洋
	2.胡佳伟
	3.吴金波

题目　降低汽油精制过程中的辛烷值损失模型

摘　　要：

现有技术在对汽油进行催化裂化过程中，普遍降低了汽油辛烷值，故对汽油精制过程进行有效建模，具有重要的研究价值。本文从理想到实际，逐步按需建立了降低汽油精制过程中的**辛烷值损失模型**，对题目所给出的问题进行快速有效求解。

针对问题一，对附件三中的 285 号和 313 号样本进行数据预处理，其中非操作变量认为短时间内保持不变，故不进行处理，主要针对操作变量采取如下处理方式：变量缺失值采取剔除和平均数插补的方式进行处理，最终剔除的变量数为 11 个和 8 个，插补的变量数为 0 个和 5 个；根据附件四的变量范围，变量值超出上下限的采取剔除的方式进行处理，最终剔除的样本变量数据分别为 560 个和 616 个，这里不排除变量范围存在误差的原因；根据 **3σ** 原则，变量值在 3σ 区间外的予以剔除，最终剔除的样本变量数据分别为 85 个和 181 个。

针对问题二，筛选出辛烷值损失模型中的主要建模变量。首先，根据变量类型不同对操作变量和非操作变量进行差异化筛选建模:对于操作变量，建立基于**随机森林**的特征重要度筛选模型;对于非操作变量，建立基于**皮尔逊相关性分析**的非操作变量特征筛选模型。然后,对模型进行求解，从所有变量中筛选出 25 个建模主要变量，其中包括 17 个主要操作变量和 8 个主要非操作变量。

针对问题三，要求通过从问题 1 和问题 2 中得到的样本数据和主要变量，采用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。在催化裂化汽油精制过程中，我们了解到：辛烷值损失= 原材料辛烷值 – 产品辛烷值。基于这一信息，接下来的模型建立以产品辛烷值作为目标变量，并同时构建三种预测模型：**多元线性回归模型**、**支持向量回归模型**和**随机森林模型**，通过对三种模型的对比分析，比较三个模型的评价指标，选取其中最合适的产品辛烷值预测损失模型。最终选取了基于支持向量回归模型的产品辛烷值预测模型，其中各项评价指标：R2=0.958，MSE=0.041，MAE=0.149。

针对问题四，以辛烷值损失降幅最大为优化目标，选取的 17 个操作变量为决策变量，产品含硫量不超过 5μg/g 和操作变量不超出上下限区间为约束条件，建立**优化模型**，寻找经过优化后辛烷值损失降幅超过 30%的样本，并对操作变量的优化过程进行分析。基于**粒子群算法**，对该优化目标求解，首先从 325 个样本中剔除硫含量大于 5μg/g 的样本，留下 266 个样本进行建模，最终明显发现经过粒子群算法的优化，辛烷值损失降幅超过 30%的样本达到了 60.53%，可以认为**粒子群算法**对该优化模型具有良好的优化能力。

针对问题五，要求对 133 号样本其主要操作变量优化调整过程中对应的产品汽油辛烷值和产品硫含量的变化轨迹进行**可视化**展示。本题的求解主要是根据前文所建立的优化模型，对 133 号样本的主要操作变量进行迭代优化，最后将迭代优化过程以图形的方式展示，绘制产品辛烷值和硫含量的变化轨迹。

最后，对问题的模型与算法中的优点和不足进行了总结。

关键词：辛烷值损失模型；随机森林；皮尔逊相关性分析；支持向量回归；优化模型；粒子群算法；可视化

目录

1 问题重述

1.1 背景知识

伴随着国家经济和科技的快速发展，人民的物质生活也不断提升，私家车的数量得到了非常广的普及。但是由于汽车尾气的排放造成的大气污染从而引发的环境问题也日益严峻，因此，通过降低汽油硫，烯烃含量来减少汽车污染排放物的手段越来越有效且作用明显。

辛烷值（以 RON 表示）是反映汽油燃烧性能的最重要指标。虽然现有技术在对催化裂化汽油进行脱硫和降烯烃过程中，能够有效降低汽油硫和烯烃的含量，但与此同时也降低了的汽油的辛烷值。辛烷值每降低 1 个单位，相当于损失约 150 元/吨。以一个 100 万吨/年催化裂化汽油精制装置为例，若能降低 RON 损失 0.3 个单位，其经济效益将达到四千五百万元。因此，降低汽油辛烷值损失不仅对能源的高效利用有着重要作用，对经济效益也有着巨大的推动。

为了对汽油辛烷值损失有着更好的预测和把控，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（控制变量）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。

近年来，随着数据挖掘的技术的兴起和日趋成熟，受到了业界极大的关注。其主要原因是数据挖掘能够从海量高维的数据中发现数据之间潜在的价值信息，更利于人们做出正确的决策。因此，采用数据挖掘技术预测辛烷值损失能够有效避免数据关联和机理建模引发的传统问题，从而有效降低辛烷值损失，提高经济效益和能源利用。

1.2 要解决的问题

现需根据题目所给背景知识和数据，解决下面五个问题：

问题一：数据预处理。请参考近 4 年的工业数据(见附件一“325 个数据样本数据.xlsx”)的预处理结果，依“样本确定方法”（附件二）对 285 号和 313 号数据样本进行预处理（原始数据见附件三“285 号和 313 号样本原始数据.xlsx”）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：寻找降低辛烷值损失模型的主要变量。由于催化裂化汽油精制过程是连续的，虽然操作变量每 3 分钟就采样一次，但辛烷值（因变量）的测量比较麻烦，一周仅 2 次无法对应。但根据实际情况可以认为辛烷值的测量值是测量时刻前两小时内操作变量的综合效果，因此预处理中取操作变量两小时内的平均值与辛烷值的测量值对应。这样产生了 325 个样本（见附件一）。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，请你们根据提供的 325 个样本数据（见附件一），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（为了工程应用方便，建议降维后的主要变量在 30 个以下），并请详细说明建模主要变量的筛选过程及其合理性。（提示：请考虑将原料的辛烷值作为建模变量之一）。

问题三：建立辛烷值（RON）损失预测模型。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，利用你们的模型获得 325 个数据样本(见附件四“325 个数据样本数据.xlsx”)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（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（即：问题 2 中的主要变量）往往只能逐步调整到位，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。（各主要操作变量每次允许调整幅度值 Δ 见附件四“354 个操作变量信息.xlsx”）。

2 模型假设和符号说明

2.1 模型假设

本文作出如下假定：

（1）假设题目所采集数据可以良好反映工厂实际化工过程即汽油精制过程中的一般运行情况；

（2）假设汽油精制过程中都属于正常运行状态，没有异常情况出现;

（3）假设通过数据找出的主要操作变量可以反映出辛烷值损失和硫含量变化的一般规律;

（4）假设每改变--次主要操作变量的值，产品汽油的辛烷值和硫含量都会相应及时的发生变化。

2.2 符号说明

序号	变量名称	变量描述
1	r	相关系数
2	x_{ij}	样本变量值
3	v_i	剩余误差
4	σ	标准误差
5	$\overline{x_i}$	样本均值
6	RON_{loss}	辛烷值损失
7	z_i	决策变量
8	S	硫含量
9	$x_{i,n}^{j}$	粒子的位置
10	$v_{i,n}$	粒子的速度
11	R^2	相关系数
12	MSE	均方误差
13	MAE	绝对平均误差

3 问题一：样本及原始数据预处理

3.1 问题一分析

本文数据由两部分组成，分别为操作变量与非操作变量。其中操作变量来自于中石化高桥石化实时数据库（霍尼韦尔 PHD），采集时间为 2017 年 4 月至 2020 年 5 月，采集了共 354 个操作变量。非操作变量来自 LIMS 实验数据库，采集时与操作变量一致，采集了共 13 个非操作变量。变量数据的采集频次与年份相关。而问题一的要求是根据附件二关于样本数据的处理方法，针对附件三的 2 个样本原始数据进行数据预处理，并加入到附件一的原始数据中，从而为解决后面的问题提供数据支撑。

对于实时数据库采集的操作变量数据，部分变量由于人为或非人为的原因会造成变量数据存在不完整、异常等显著误差，包括记录错误、操作错误、测量错误等，致使变量数据与实际数据无法始终保持一致，因此需要对这类变量进行数据预处理，从而为模型的建立提供干净的数据。而对于实验数据库采集的非操作变量，包括原料、待生吸附剂、再生吸附剂的相关变量，由于采集频次为每周 2 次，可以认为该类变量的动态变化范围较小，不需要进行数据预处理，且实验中假设该类变量的性质始终保持不变，因此问题一只需要对操作变量的数据进行处理。

在本文提供的原始数据中，大部分变量属于正常范围，但记录数据的装置在部分位点（变量）上存在问题，这会导致部分变量的数据在部分时间段为空值，或者部分样本的数据超出该变量的上下限以及存在异常。本文根据文献资料以及操作经验，针对样本中出现的问题，对这两个样本的数据通过以下流程进行处理：

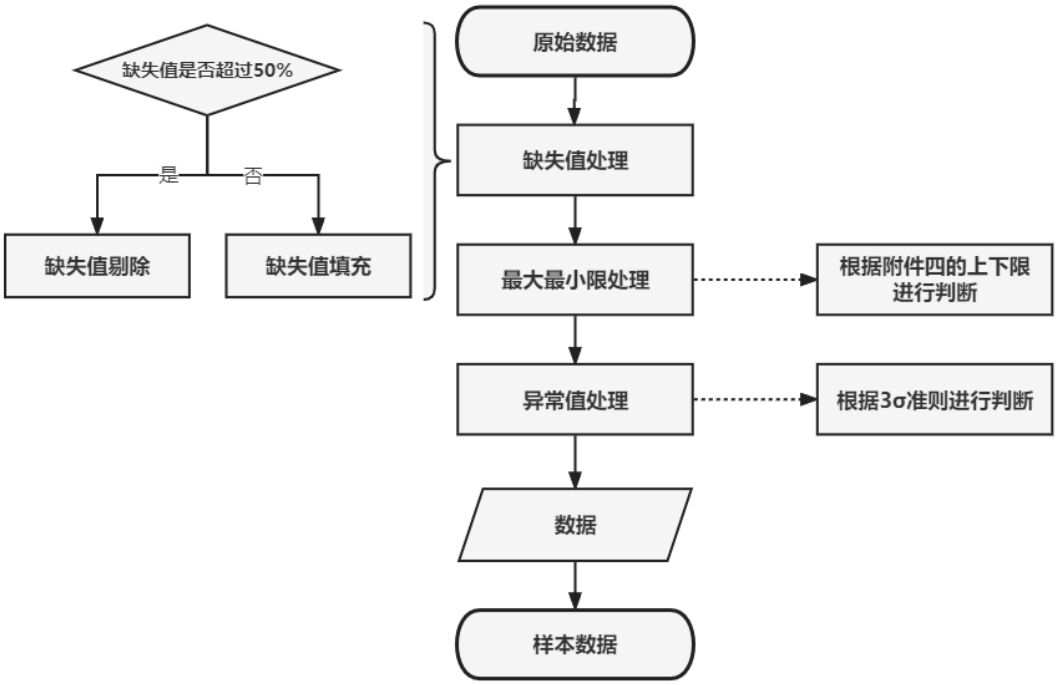


图 3-1 问题一思路流程图

3.2 缺失值处理

如果某个变量仅含有部分时间点上的数据时，我们可以认为该变量存在缺失值，因此需要对其进行缺失值处理。缺失值处理一般有两种处理方式，分别为剔除和插补，其中插补的形式较多，包括均值插补、随即插补、回归插补和随机回

归插补等，题目中指定用前后两小时的平均数进行插补，但在样本中仅提供了两个小时的样本数据，故本文选择用该变量在这两个小时的有效数据的平均值进行填充。

针对不同的情况需要采取不同的方式进行处理。在本文中，我们人为设定：当变量出现空缺的数量超过总样本数的 50%时，需要对该变量进行剔除；当变量出现空缺的数量未超过总样本数的 50%时，则可以对该变量进行插补。具体操作形式见下图：

$$a=\sum_{j=0}^{40}(x_{ij}=0) \tag{3-1}$$

$$x_{i(j=(x_{ij}=0))}=\begin{cases}0 & a>20\\ \frac{1}{a}\sum_{j=1}^{40}x_{ij} & a\le 20\end{cases} \tag{3-2}$$

其中样本数共有 40 个，a 表示某个变量空缺值的个数，x_{ij} 表示样本变量值。

我们分别对 285 样本和 313 样本进行缺失值处理，得到以下结果。

变量剔除数	变量插补数
11	0

变量剔除数	变量插补数
8	5

根据处理结果，可以发现 285 样本中仅出现了变量列全部为空值的情况，而 313 样本两种情况都有出现。

3.3 最大最小限处理

根据附件 4，可以得知各个操作变量的取值范围，因此通过 python 进行数据处理，从而得到样本变量是否有超过最大最小限的情况，如果存在，则对超出最大最小限的变量赋值为 0，代表对该样本变量值进行剔除。

285 样本数据剔除数	313 样本数据剔除数
560	616

在进行这一步操作中，可以清楚发现，大量变量并不在变量的取值范围中，通过查看附件一的原始数据，仍然可以发现这些变量不仅在这两个样本中存在这种超限的异常情况，在其它样本中仍然存在，因此，这里我们推测部分异常情况是由于变量的单位未对应所造成，比如“%”，未将绝对数换算成相对数，从而导致变量数值上与最大最小限的数值存在较大的差异。

3.4 异常值处理

3.4.1 达拉依准则

达拉依准则又称 3σ 准则，在统计学中较为流行，常用于判断数值是否出现较大的异常。在统计学中，我们认为某个数值的取值基本上集中于该数值平均数上下浮动 3σ 个单位的范围内。

在正态分布中，μ 表示均值，σ 表示标准差，3σ 准则就是指，数值分布在 $(\mu-3\sigma,\mu+3\sigma)$ 的概率为 99.74%，因此，我们可以认为数值的取值基本集中在这

个区间范围，当出现数值不在这个区间的情况时，则认为该数值为异常值，应当予以剔除。

在本文中异常值处理主要是根据拉依达准则（3σ准则）来剔除，即取空值。

$$\overline{x_i} = \frac{1}{40}\sum_{i=0}^{40} x_{ij}$$

$$v_i = x_{ij} - \overline{x_i}$$

$$\sigma = \sqrt{\frac{1}{39}\sum_{i=0}^{39}(x_{ij-}\overline{x_i})^2}$$

$$x_{ij} = 0 \qquad (if \ \left|v_i - x_{ij}\right| > 3\sigma)$$

3.4.2 处理结果

通过异常值处理，我们在 285 样本中共发现了 85 个异常值，在 313 样本中共发现了 181 个异常值。

3.5 数据处理结果和结论

通过缺失值处理、最大最小限幅处理与异常值处理，当前的数据已经符合样本数据的标准，我们对各样本的变量数据值取平均数，作为该样本这两个小时内的变量数据，以对应辛烷值的操作变量数据，并将其加入到附件一中。具体处理结果可以看附件三的

通过前后对比附件一 285 样本和 313 样本的数据变量，可以发现有几个变量值存在轻微浮动的情况，基本上只有小数点后几位的变动，由于计算上存在差异，可以认为这属于正常的浮动范围，因此我们认为这些变量数据能够进行后面几个问题的运算过程中去。

4 问题二：通过降维筛选辛烷值损失模型的主要变量

4.1 问题二分析

针对问题二，需要筛选出辛烷值损失模型中的主要建模变量。由于非操作变量与操作变量之间含义与性质相差较大，故可以分开讨论。对于 354 个操作变量，题目表明其具有高度非线性和相互强相关，所以传统线性降维方法（如主成分分析和 LCA 等）不合适，本文使用基于皮尔逊相关系数和随机森林的特征重要度筛选方法，对 354 个操作变量进行 2 次筛选和降维；对于非操作变量，由于其具有潜在的化学相关性，本文对其进行相关性分析，使用皮尔逊相关系数进行筛选。图 4-1 给出该问题的思路流程图。

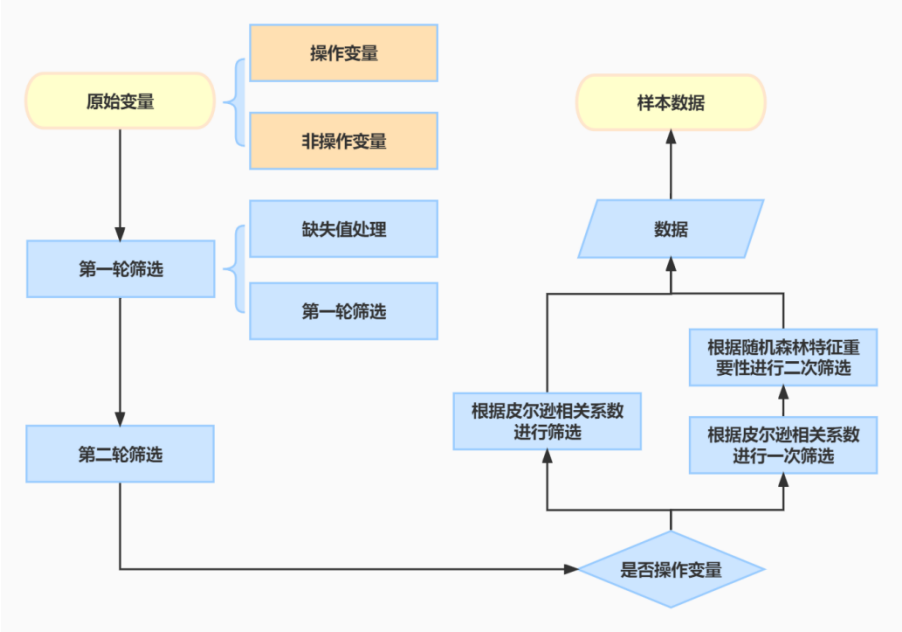


图 4-1 问题二思路流程图

4.2 问题二模型建立

4.2.1 基于随机森林特征重要性的操作变量筛选模型

随机森林（Random Forest，简称 RF）是一种新兴起、高度灵活的机器学习算法，拥有广泛的应用前景，在大量分类以及回归问题中具有极好的准确率。并且，随机森林算法自带特征筛选机制，即随机森林能够评估各个特征在相应问题上的重要性。。此处，辛烷值损失模型的操作变量具有数量多、高度非线性、相互强赧联的特点，与 RF 的应用条件契合，所以本文考虑建立基于随机森林的操作变量特征筛选模型，对进行一次皮尔逊相关系数筛选处理后的操作变量进行二次有效筛选。基于随机森林的操作变量特征筛选模型中随机森林训练过程包含以下步骤：

Stepl：原始训练集为 N，应用 bootstrap（有放回抽样）法有放回地随机抽取 k 个新的自助样本集，并由此构建 k 棵分类树，每次术被抽到的样本组成了 k 个袋外数据：

Step2：假如特征空间共有口个特征，则在每一轮生成决策树的过程中，从 D 个特征中随机选择 d 个特征（d<D）组成一个新的特征集，通过使用新的特征集来生成决策树，在 k 轮中共生成 k 个相互独立的决策树：

Step3：将生成的多棵树组成随机森林，相互独立的若干棵决策树的重要性是相等的，无需考虑它们的权值。随机森林算法的流程图如图 4-2 所示。

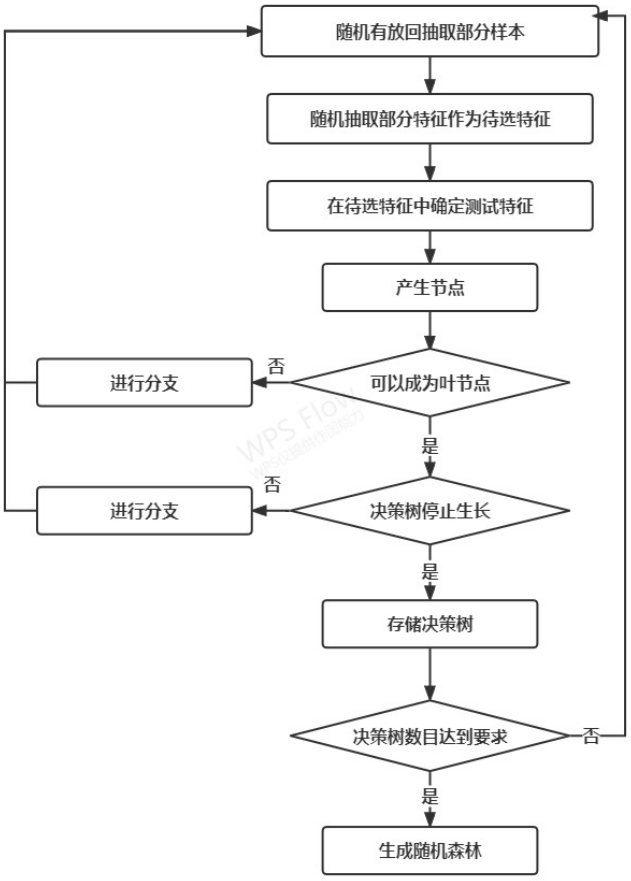


图 4-2 随机森林实现流程图

4.2.2 基于皮尔逊相关系数的非操作变量特征筛选模型

皮尔逊相关系数，又称为皮尔逊积矩相关系数，是用于度量两个变量 X 和 Y 之间的相关性，其值介于-1 与 1 之间。一般用于分析两个连续变量之间的关系，是一种线性相关系数，公式为：

$$r=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{i=1}^{n}(y_i-\bar{y})^2}} \tag{4-1}$$

其中 r 是两个变量 X 和 Y 之间的皮尔逊相关系数，(i=1,2,⋯,n)是向量 X 的元素，（i=1,2,⋯,n)是向量 Y 的元素。相关系数 r 的取值范围为-1≤r≤1。r>0，为正相关，r<0 为负相关,r=0，表示不存在线性相关。$|r=1|$表示完全线性相关。

当$0<|r|<1$时，表示两个变量之间存在不同程度的线性相关：

① $0.8<|r|<1$，表示两个变量是极强相关：

② $0.6<|r|<0.8$，表示两个变量是强相关：

③$0.4<|r|<0.6$，表示两个变量是中等程度相关：

④ $0.2<|r|<0.4$，表示两个变量是弱相关：

⑤$0<|r|<0.2$ ，表示两个变量是极弱相关或无相关。

当两个变量的标准差都不为零时，相关系数才有意义，皮尔逊相关系数适用于：1.两个变量之间是线性关系，都是连续数据。2.两个变量的总体是正态分布，或接近正态的单峰分布。3.两个变量的观测值是成对的，每对观测值之间相互独

立。

4.3 问题二模型求解

4.3.1 操作变量特征筛选模型求解

利用 python 对上述基于随机森林的操作变量特征筛选模型进行求解得到操作变量的的特征权值。其次，为了有效剔除次要变量，筛选主要变量，本文界定对产品中辛烷值(RON)的贡献度不大于 1%（特征权值不大于 0.01）的变量在误差允许的范围内为建模中的“次要变量”。以此，对操作变量中的“无关变量”和“次要变量”进行剔除，得到操作变量中的 17 个建模主要变量。如表 4-1 所示（变量按特征权值降序排列）.

表 4-1 建模主要操作变量表

变量编号	1	2	3	4	5	6
特征权值	0.070362	0.067359	0.065035	0.052205	0.039383	0.038735
变量编号	7	8	9	10	11	12
特征权值	0.037991	0.031554	0.030259	0.029854	0.028660	0.024377
变量编号	13	14	15	16	17	
特征权值	0.024143	0.022858	0.022756	0.021946	0.021864	

变量编号对应的变量名称依次是 8.0MPa 氢气至反吹氢压缩机出口，原料换热器管程进出口压差，1.1 步骤 PIC2401B.OP，反应过滤器压差，精制汽油去进料缓冲罐流量，反应过滤器压差，加热炉主火嘴瓦斯入口压力，D-105 下锥体松动风流量，进装置原料硫含量，闭锁料斗氧含量　EH-102 出口空气总管温度，过滤器 ME-101 压差，紧急氢气总管，还原器温度　E-101A 壳程出口管温度，K-103B 进气压力，D-110 底压力。

4.3.2 非操作变量特征筛选模型求解

按照皮尔逊相关系数计算方法计算 13 个非操作变量两两之间的皮尔逊相关系数。得到图 4-3。

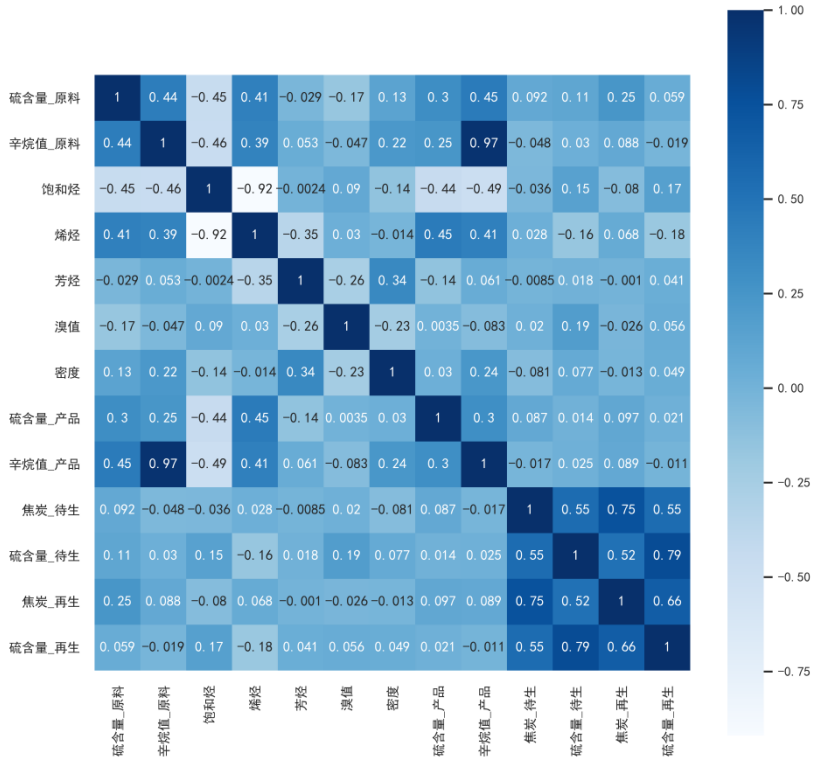


图 4-3 非操作变量相关系数热力图

根据热力图，选取了辛烷值产品，辛烷值原料，硫含量原料和硫含量产品四个典型变量，并根据相关系数的大小对十三个变量分别在四个典型变量下做了排序，排序结果如表 4-2 所示。

表 4-2 相关系数排序表

排序	辛烷值_产品	辛烷值_原料	硫含量_原料	硫含量_产品
1	辛烷值_产品	辛烷值_原料	硫含量_原料	硫含量_产品
2	辛烷值_原料	辛烷值_产品	辛烷值_产品	烯烃
3	饱和烃	饱和烃	饱和烃	饱和烃
4	硫含量_原料	硫含量_原料	辛烷值_原料	硫含量_原料
5	烯烃	烯烃	烯烃	辛烷值_产品
6	硫含量_产品	硫含量_产品	硫含量_产品	辛烷值_原料
7	密度	密度	焦炭_再生	芳烃
8	焦炭_再生	焦炭_再生	溴值	焦炭_再生
9	溴值	芳烃	密度	焦炭_待生
10	芳烃	焦炭_待生	硫含量_待生	密度
11	硫含量_待生	溴值	焦炭_待生	硫含量_再生
12	焦炭_待生	硫含量_待生	硫含量_再生	硫含量_待生
13	硫含量_再生	硫含量_再生	芳烃	溴值

然后按排序的次序依次赋分，排序为一赋一分，依次到排序十三赋十三分。得到十三个变量的赋分排名，辛烷值_产品得分 10 分，排名第一；饱和烃得分 12 分，排名第二；辛烷值_原料和硫含量_原料得分都为 13 分，并列第三；烯烃得分 17 分，排名第五；硫含量_产品得分 19 分，排名第六；焦炭_再生得分 31，排名第 7；密度得分 33 分，排名第 8；芳烃得分 39 分，排名第 9；溴值得分 41 分，排名第 10；焦炭_待生得分 42 分，排名第 11；硫含量_待生得分 45 分，排名第 12；硫含量_再生得分 49 分，排名第 13。取前 8 名为非操作变量中有关损失辛烷值建模的主要变量。

4.3.3 主要变量筛选结果

综合上述，本文对建模主要变量的筛选过程及结果总结如下：

（1）采用基于随机森林的操作变量特征筛选模型对问题一预处理后的操作变量进行筛选，保留 17 个建模主要操作变量。

(2）采用基于皮尔逊相关性分析的非操作变量特征筛选模型对 13 个非操作变量进行筛选，保留 8 个建模主要非操作变量。

（3）总共筛选出 25 个建模主要变量。

5 问题三：辛烷值损失预测方法研究

5.1 问题三分析

在上述问题一和问题二中，我们已经对样本与变量进行了分析和筛选。本节则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。在催化裂化汽油精制过程中，我们了解到：辛烷值损失= 原材料辛烷值 － 产成品辛烷值。基于这一信息，接下来的模型建立以产品辛烷值作为目标变量。本文同时构建三种预测模型：多元线性回归模型、支持向量回归模型和随机森林模型，通过对三种模型的对比分析，选出其中效果最好的模型。模型构建流程图 5-1 如下：

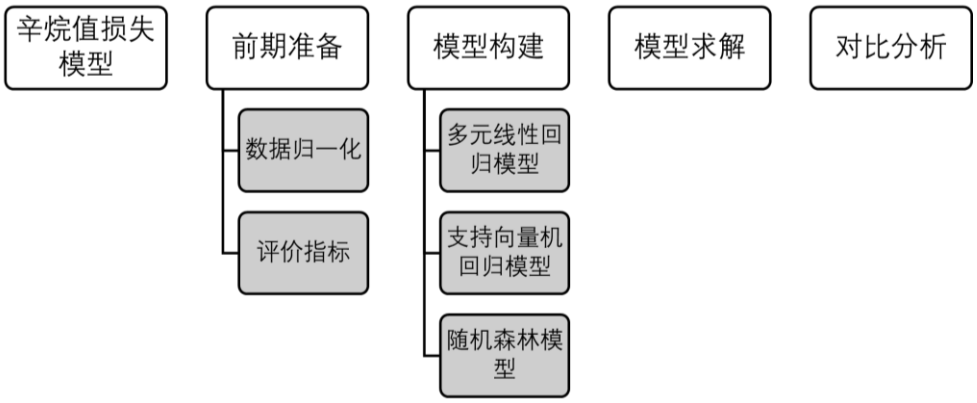


图 5-1 问题三模型构建流程图

5.2 辛烷值损失模型的构建

5.2.1 数据归一化处理

由于主要变量量纲有所不同，本文将对数据进行归一化处理来消除不同量纲给数据带来的影响。归一化主要有两种方式：极差变换法，正态标准化。极差变换法通过利用变量取值的最大值和最小值（或者最大值）将原始数据转换为界于某一特定范围的数据，从而消除不同量纲的影响；正态标准化通过对数据的特征属性减去均值，除以方差将数据分布转化为均值为 0，方差为 1 的标准正态分布。本文将采用正态标准化对数据做预处理。公式如下：

$$x^{*}=\frac{x-\mu}{\sigma} \tag{5-1}$$

5.2.2 评价指标

在建立模型的过程中，我们需要对模型的拟合效果进行检验来判断模型是否适用。对于辛烷值损失模型，本文采用以下三种评价指标。如表 1 所示：

表 5-1 评价指标

评价指标	含义
R2	相关系数
MSE	均方误差
MAE	绝对平均误差

相关系数是反映变量之间相关关系密切程度的统计指标，可以反映出试验数据与拟合模型之间的吻合程度。

均方误差（mean-square error, MSE）是反映估计量与被估计量之间差异程度的一种度量。

平均绝对误差是观测值与算术平均值的偏差的绝对值的平均。可以准确反映实际预测误差的大小。

5.2.3 基于多元线性回归的产品辛烷值预测模型求解

多元线性回归模型是用来进行回归分析的数学模型，由于客观事物内部规律的复杂性及人们认识程度的限制，无法分析实际对象内在的因果关系，建立合乎机理规律的数学模型。

多元线性回归模型的基本假设：

$$\begin{cases} E\left(\varepsilon_i\right)=0, \quad i=1,2,\cdots,n \\ \operatorname{cov}\left(\varepsilon_i,\varepsilon_j\right)=\begin{cases}\sigma^2, i=j \\ 0, i\neq j\end{cases} \quad i,j=1,2,\cdots,n \end{cases}$$

$$\begin{cases} \varepsilon_i \sim N\left(0,\sigma^2\right), \quad i=1,2,\cdots,n \\ \varepsilon_1,\varepsilon_2,\cdots,\varepsilon_n \text{ 相互独立} \end{cases}$$

多元线性回归模型的一般形式：

$$y=\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p+\varepsilon \qquad (5\text{-}2)$$

在实际过程中，根据获得的数据，多元线性回归模型可表示为式（5-3）。

$$\begin{cases} y_1=\beta_0+\beta_1 x_{11}+\beta_2 x_{12}+\cdots+\beta_p x_{1p}+\varepsilon_1 \\ y_2=\beta_0+\beta_1 x_{21}+\beta_2 x_{22}+\cdots+\beta_p x_{2p}+\varepsilon_2 \\ \quad \vdots \\ y_n=\beta_0+\beta_1 x_{n1}+\beta_2 x_{n2}+\cdots+\beta_p x_{np}+\varepsilon_n \end{cases} \qquad (5\text{-}3)$$

我们将处理好的变量数据代入式（5-3），通过最小二乘法得到一个估计的辛烷值多元线性回归模型。通过 python 实现模型预测值和真实值对比结果可视化。如图 5-2 所示：

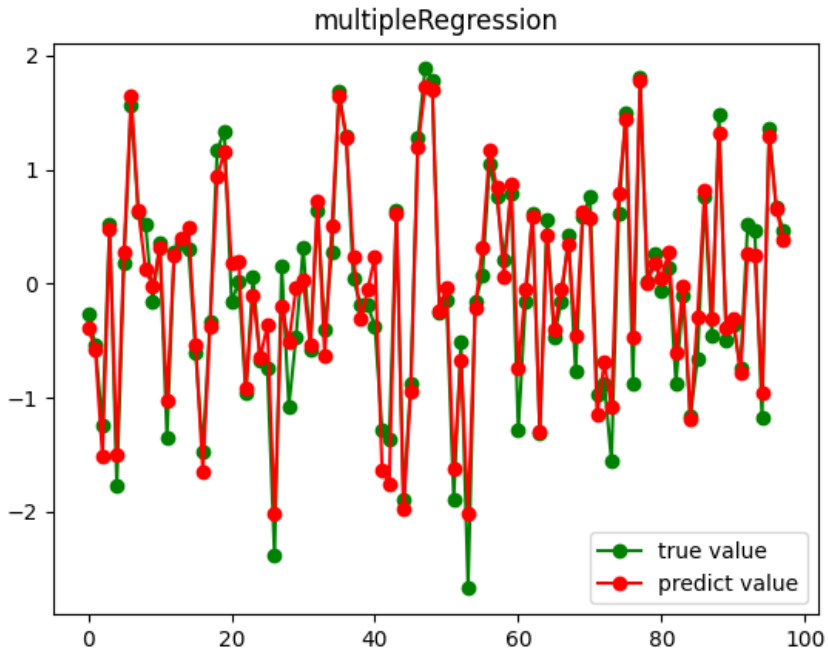


图 5-2 基于多元回归的产品辛烷值预测结果

5.2.4 基于随机森林的产品辛烷值预测模型求解

首先我们需要了解决策树的概念，分类决策树模型是一种描述对实例进行分类的树形结构，决策树由结点和有向边组成。结点有两种类型:内部结点和叶结点，内部结点表示一个特征或属性，叶结点表示个类。

图 5-3 为决策树示意图，其中圆形代表内部节点，三角形代表叶节点。

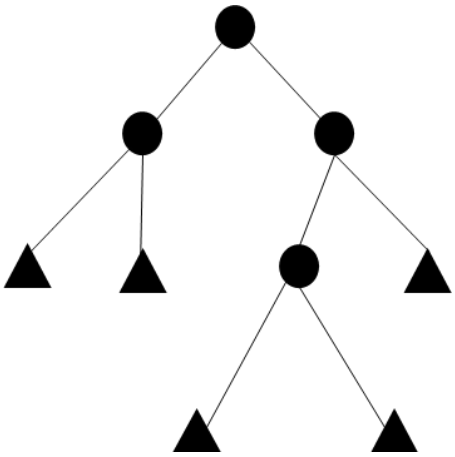


图 5-3 决策树示意图

随机森林（Random Forest，简称 RF）是通过集成学习的 Bagging 思想将多棵树集成的一种算法：它的基本单元就是决策树。是 Bagging 方法的一种具体实现。它在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入随机性。通俗的说法是 RF 会训练多棵决策树，然后将这些结果融合在一起就是最终的结果。随机森林可以用于分类，也可以用于回归。主要在于决策树类型的选取，根据具体的任务选择具体类别的决策树。

随机森林是由许多决策树组成的整体模型。通过对每个决策树的预测平均来进行预测。就像森林是树木的集合一样，随机森林模型也是决策树模型的集合。这使随机森林成为一种强大的建模技术，它比单个决策树要强大。

随机森林中的每棵树都在对数据子集进行训练。其背后的基本思想是在确定最终输出时组合多个决策树，而不是依赖于各个决策树。每个决策树都有很高的方差，但是当我们将所有决策树并行组合在一起时，由于每个决策树都针对特定样本数据进行了完美的训练，因此结果方差很低，因此输出不依赖于一个决策树而是多个决策树木。对于回归问题，最终输出是所有决策树输出的平均值。随机森林建立过程如下

1.从数据集中随机选择 N 个样本子集。

2.基于这 N 个样本子集构建决策树。

3.选择算法中所需树的棵数，然后重复步骤 1 和 2。

4.取森林中所有决策树预测值的平均值。

通过 python 实现模型预测值和真实值对比结果可视化。如图 5-4 所示：

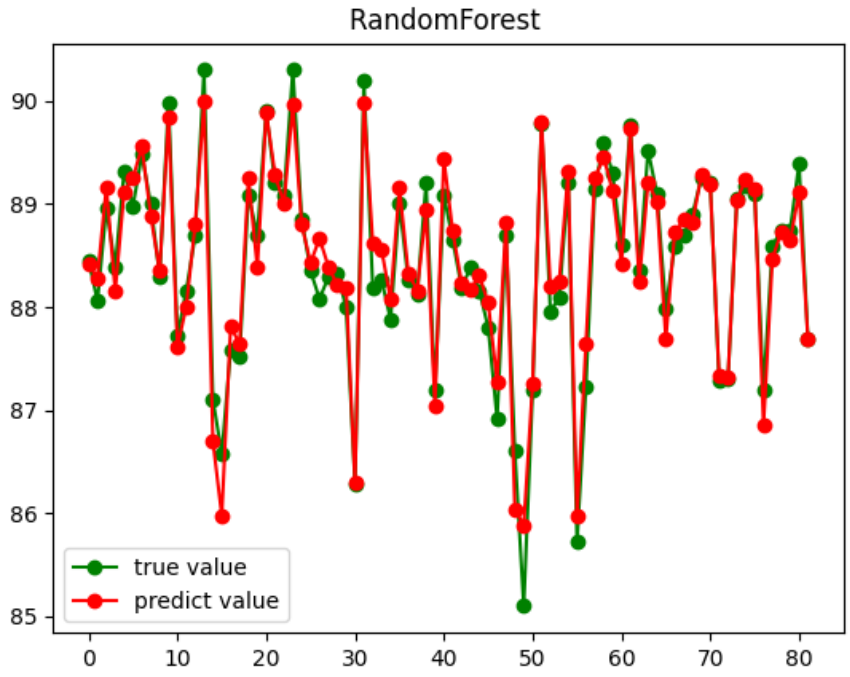


图 5-4 基于随机森林的产品辛烷值预测结果

5.2.5 基于支持向量机的产品辛烷值预测模型求解

给定训练样本级 $D=\{(x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m)\}, y_i\in\{-1,+1\}$，分类学习最基本的想法就是基于训练集 D 在样本空间中找到一个划分超平面，将不同类别的样本分开，但能将训练样本分开的划分超平面可能有很多，如图 5-5 所示。

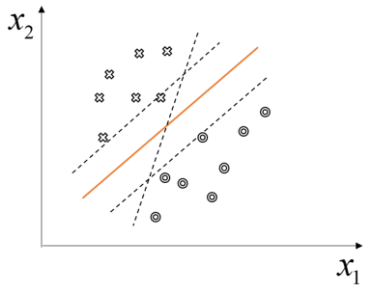


图 5-5 超平面样本划分

图 5 中红色的直线划分超平面所产生的分类结果是最好的，泛化能力最强。在样本空间中,划分超平面可通过如下线性方程来描述:

$$\omega^T x+b=0 \tag{5-4}$$

其中 $\omega=(w_1;w_2;\ldots;w_d)$ 为法向量，决定了超平面的方向；b 为位移项，决定了超平面与原点之间的距离。超平面被 ω 和 b 确定，并记作 (ω,b)。样本空间任意点 x 到超平面 (ω,b) 的距离为

$$\gamma=\frac{2}{\|\omega\|} \tag{5-5}$$

若超平面 (ω,b) 能对训练样本正确分类，则超平面 (ω,b) 满足一下条件：

$$\begin{cases}\omega^{\mathrm{T}}x_i+b\geqslant+1, & y_i=+1\\ \omega^{\mathrm{T}}x_i+b\leqslant-1, & y_i=-1\end{cases} \tag{5-6}$$

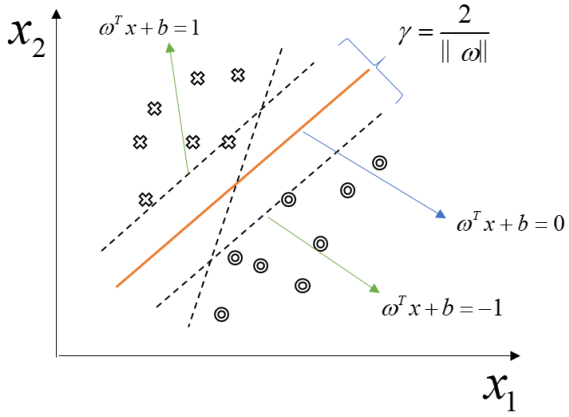


图 5-6 支持向量与间隔

如图 5-6 所示，距离超平面最近的这几个训练样本点使式(1.6)的等号成立，它们被称为“支持向量”(support vector),两个异类支持向量到超平面的距离之和为

$$\gamma=\frac{2}{\|\omega\|}$$

想要要找到距离最大的划分超平面，需要满足一下条件：

$$\begin{aligned} &\min_{\omega,b} \quad \frac{1}{2}\|\omega\|^2 \\ &\text{s.t.} \quad\quad y_i\left(\omega^{\mathrm{T}}x_i+b\right)\geqslant 1, \quad i=1,2,\ldots,m. \end{aligned} \tag{5-7}$$

这就是支持向量机的基本型。

本文利用支持向量机模型并引入线性核对产品辛烷值进行回归预测。通过 python 实现模型预测值和真实值对比结果可视化。如图 5-7 所示：

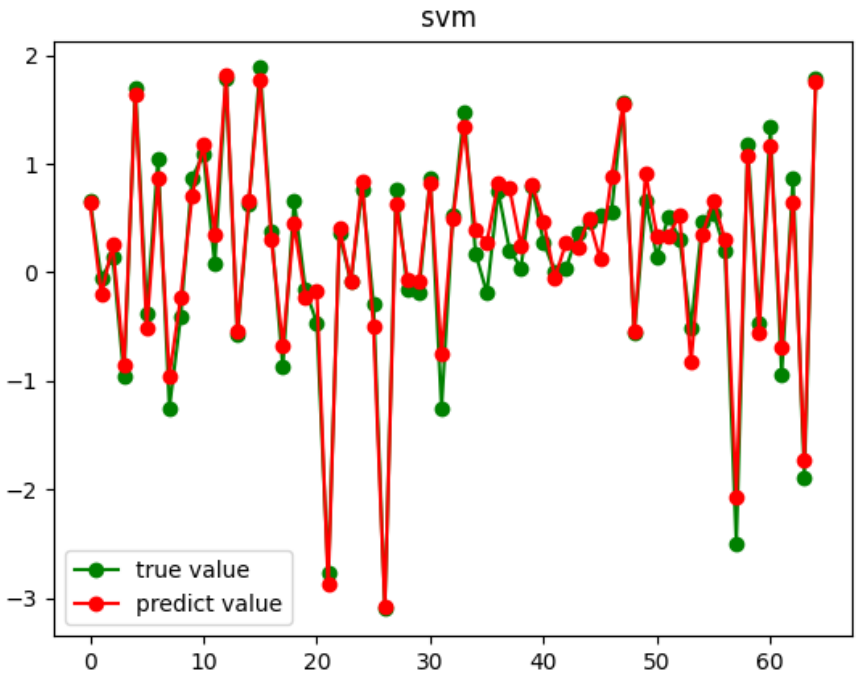


图 5-7 基于支持向量机的产品辛烷值预测结果

5.3 模型对比分析与结果

本文采用三种预测模型：多元线性回归模型、随机森林模型、支持向量回归模型。它们分别有以下优缺点：

多元线性回归模型

- 优点：回归分析法在分析多因素模型时，更加简单和方便；运用回归模型，只要采用的模型和数据相同，通过标准的统计方法可以计算出唯一的结果。
- 缺点：泛化能力低，不宜推广。

随机森林模型

- 优点：避免产生过拟合的模型；具有很好的抗噪能力。
- 缺点：由于随机森林的两个随机性，导致运行结果不稳定。

支持向量回归模型

- 优点：有大量的核函数可以使用，从而可以很灵活的来解决各种非线性的分类回归问题；样本量不是海量数据的时候，分类准确率高，泛化能力强。
- 缺点：非线性问题的核函数的选择没有通用标准，难以选择一个合适的核函数。

本文通过 python 对三种产品辛烷值回归预测模型进行求解，分别得到三种

模型的评价指标。如表 5-2 所示：

表 5-2 模型评价指标

模型	R^2	MSE	MAE
多元线性回归	0.957	0.042	0.153
随机森林	0.945	0.053	0.172
支持向量回归	0.958	0.041	0.149

不难看出，随机森林模型拟合程度较差，效果不好；支持向量机模型预测效果最好。综合分析，基于支持向量回归的产品辛烷值模型更加合适。

6 问题四：主要变量操作方案优化方法设计

6.1 问题四分析

针对问题四，其要求是在保证产品硫含量不大于 5μg/g 的前提下，对主要变量中的操作变量进行优化，以达到辛烷值损失尽可能小的结果，并且在优化后筛选出辛烷值损失降幅大于等于 30%的样本。

问题四实际上是一个多约束的最优化问题。因此我们需要针对该问题要求，确定决策变量、优化目标和优化条件，从而建立一个优化模型，并选择一个合适的算法对模型进行求解，最后对模型结果作一些定性、定量的分析和必要的解释。

根据题意，我们可以得知，本题的所涉及的决策变量为问题二所选取的操作变量，共 17 个操作变量。本题的优化目标为，通过对决策变量的调整使辛烷值损失降幅最大，也就是辛烷值损失通过优化后的数值尽可能小，其中我们需要通过问题三所筛选出的最优秀的预测模型对辛烷值进行预测，同时，我们还要保证控制硫含量尽可能小，因此同样需要使用预测模型对硫含量进行预测。本题的约束条件为两类。第一类为对决策变量的约束条件，决策变量不能在调整过程中出现超出附件四中所规定的操作变量范围，一共有 17 个约束条件。第二类为对硫含量的约束条件，产品的硫含量最终应不大于 5μg/g。

在本题中，选取了优化算法中的粒子群算法对该模型进行求解。粒子群算法属于启发式算法，能有效处理变量组合优化的问题，粒子群算法的适用范围较广，优化效果好，且算法较为简单，容易理解。因此，本文使用粒子群算法来求解模型。

6.2 基于粒子群算法的优化模型

由于本题中需要保证硫含量不大于 5μg/g，因此在进行模型求解前，我们需要对样本中硫含量已经大于 5μg/g 的样本进行剔除，最后共留下了 266 个样本用来建立优化模型。

6.2.1 优化模型建立

（1） 建立决策变量

将问题二中选取的操作变量设为决策变量。即{ S-ZORB.FT_1504.TOTALIZERA.PV、S-ZORB.PDI_1102.PV、S-ZORB.PC 2401.PIDA.OP、…、S-ZORB.PT 7508.DACA、S-ZORB.SIS PT_2703}设为$\{z_1, z_2, ..., z_{17}\}$，记为 Z=$\{z_1, z_2, ..., z_{17}\}$

（2） 建立目标函数

我们记最开始的辛烷值损失为RON_{loss1}，优化结束后，辛烷值损失为RON_{loss2}，因此为了得到辛烷值损失降幅最大的目标，我们设定目标函数为：

$$\max f = \frac{RON_{loss1} - RON_{loss2}}{RON_{loss1}}$$

由于在操作变量中，我们选取了原料中的辛烷值作为操作变量，而没有选择产品中的辛烷值作为操作变量，同时，根据题意，优化过程中原料、待生吸附剂、再生吸附剂的性质保持不变因此我们可以将此目标函数改为：

$$\min f = RON_{loss2}$$

这在实现粒子群算法的优化中可以在一定程度上提高算法的效率

（3） 约束条件

1、决策变量的约束条件：

决策变量 $z_1, z_2, ..., z_{17}$ 应当满足以下约束条件：

$$z_{\min i} \le z_i \le z_{\max i}$$

其中 $z_{\min i}$ 和 $z_{\max i}$ 为主要操作变量的上下限。

2、产品硫含量的约束条件：

产品中的硫含量 s 应当满足以下条件：

$$0 \le s \le 5$$

6.2.2 粒子群算法

粒子群算法（PSO）是针对群体的演化算法，其主要思想来源于鸟类的飞行研究。鸟类在进行群体飞行时，其具体移动方位并不是固定的，每只鸟都会根据群体的总体移动方向以及自身范围内的总体移动方向来进行移动，就好像，鸟的飞行过程不仅仅受到单一因素的影响，而是不同因素叠加的影响。这里以鸟类觅食为例，当鸟群去觅食时，对其中一个鸟来说，就它会在一定范围内随机搜寻食物，当它发现一个食物时，但它只知道这个食物与它的距离，但不知道方向，因此对这只鸟而言，找到食物最好的方法就是去往与这个食物最近的鸟的方向进行飞行，这样每只鸟的距离越来越近，直到最终找到食物。

在 PSO 算法中，将每只鸟定义为一个粒子，这种鸟的搜索空间定义为一个解空间，当鸟在飞行过程中，鸟与食物的距离定位为粒子的适应度，而在实际应用过程中，粒子的适应度值由目标函数进行提供，一般来说，当求解最小值问题时，适应度值越小越好。而鸟总是往食物方向飞行，因此适应度值也会慢慢减小。PSO 算法认为，每个粒子有两个特征，分别为速度和位置，速度主要由本身经验和群体经验决定，既可以随即得到，也可以通过赋予权重获得最好参数。而位置的定义就较为复杂，由于粒子本身的运动不仅受自身的惯性影响，它还受到全局最优和个体最优的影响，就好比鸟在飞行过程中，不仅受到食物的影响，也会受到离食物最近的鸟的影响。同时，每个粒子必须保证始终个体最优，所以每个粒子都需要记住自己历史适应度值，从而在每一次迭代后进行判断，保持个体最优。

PSO 算法将速度公式定义如下：

$$v_{i,n+1}^{j} = v_{i,n}^{j} + c_1 r_{i,n}^{j}(pbest_{i,n}^{j} - x_{i,n}^{j}) + c_2 R_{i,n}^{j}(gbest_{n}^{j} - x_{i,n}^{j})$$

$$x_{i,n+1}^{j} = x_{i,n}^{j} + v_{i,n+1}^{j}$$

其中，我们假定有 N 个粒子，每个粒子在 M 维空间求解，则在第 n 次迭代过程中，第 i 个粒子的位置为 $x_{i,n} = (x_{i,n}^{1}, x_{i,n}^{2}, ..., x_{i,n}^{M})$，第 i 个粒子的速度为 $v_{i,n} = (v_{i,n}^{1}, v_{i,n}^{2}, ..., v_{i,n}^{M})$，此时个体最优为 $pbest_{i,n}^{j}$，群体最优为 $gbest_{n}^{j}$。

PSO 算法的流程图大致如下：

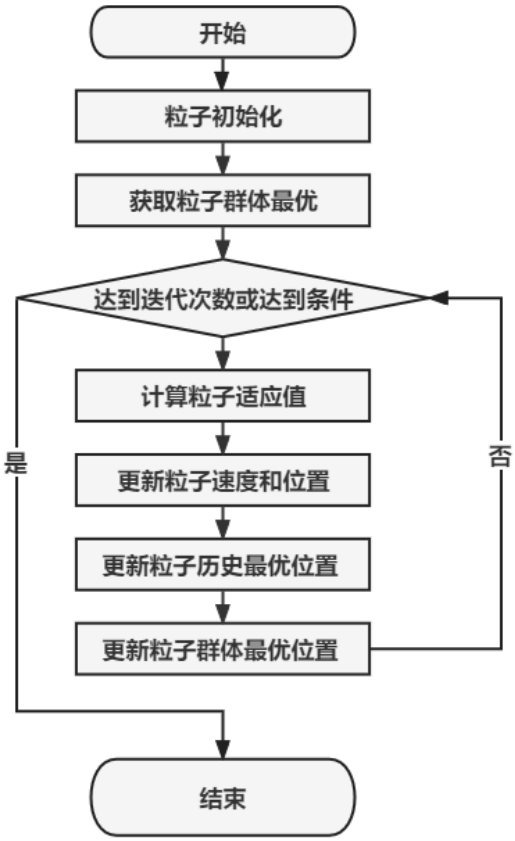


图 6-1 PSO 算法流程图

6.2.3 实证研究

在本题中，我们将 266 个样本，通过粒子群算法来进行优化求解，其中我们对模型的参数设定如下表：

表 6-1 模型参数设定表

参数	含义	值
w	惯性权重	0.5+0.4*(i/I)
C1	学习常数	2
C2	学习常数	2

其中对于惯性权重 w，将其值设定为 0.5+0.4*(i/I)，其中 i 为第 i 次迭代，I 为总迭代数，这种设定是为了使惯性权重从 0.5 逐步递增到 0.9，使该模型在前期有较好的全局搜索能力，后期有较好的局部搜索能力。

在 266 个样本中，有 168 个样本经过优化算法后，辛烷值损失降幅超过了 30%。统计各个辛烷值（RON）损失降幅的频数和频率，如下所示：

表 6-2 辛烷值损失降幅的频数和频率表

降幅范围(%)	频数	频率
(80, 100]	0	0.00%
(60, 80]	14	5.26%
(50, 60]	54	20.30%
(40, 50]	50	18.80%
(30, 40]	43	16.17%

(20, 30]	35	13.16%
(10, 20]	11	4.14%
(0, 10]	6	2.26%
异常样本	0	0.00%
总计	266	1

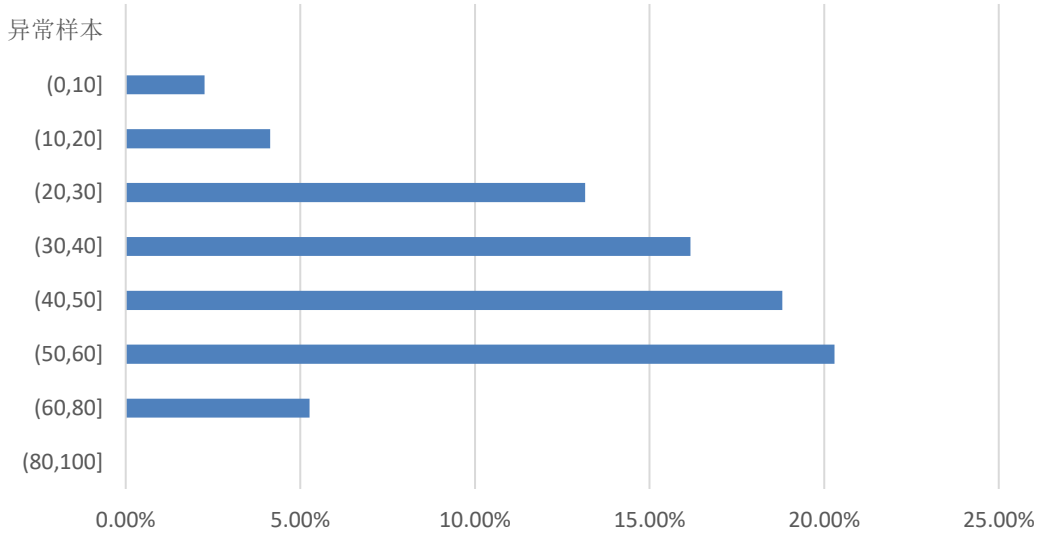
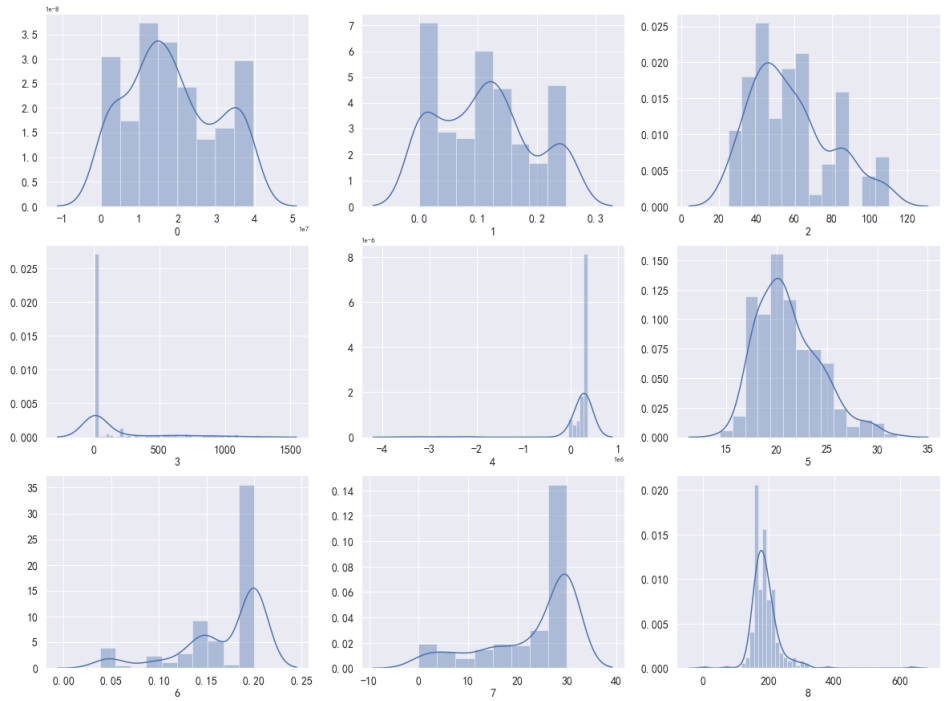
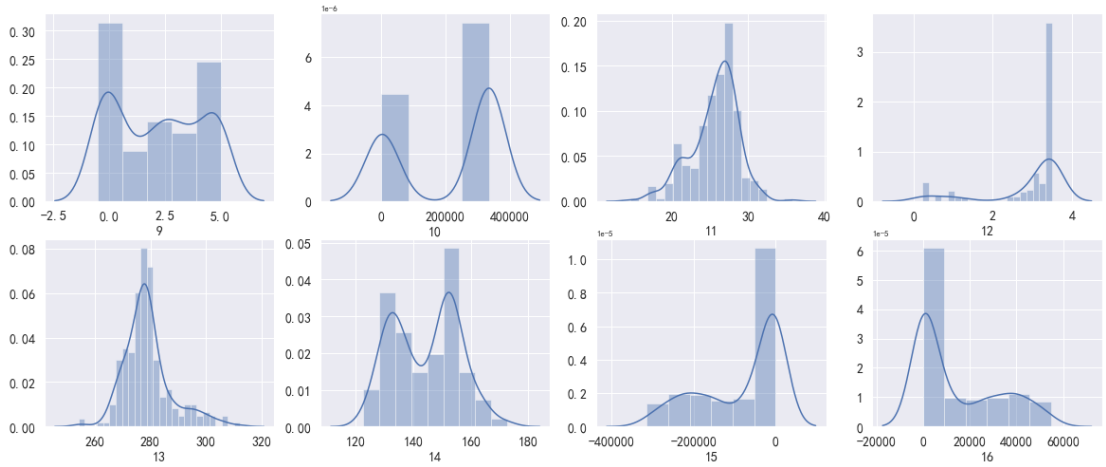


图 6-2 辛烷值损失样本柱状图

表中的异常样本表示该样本经过优化后硫含量超标，大于 5μg/g。通过柱状图，可以明显发现辛烷值损失降幅超过 30%，达到 60.53%，只有少数样本最终损失降幅未能达标。其中，辛烷值损失降幅超过 60%的达到了 25.56%，接近样本的 1/4，这是由于其原始辛烷值损失本身就比较高，导致其优化过程中，效果会比较好。





根据操作变量的分布图，可以发现部分变量呈正态分布，集中在均值分布。

样本编号	1	2	3	4	5	6
均值	18565612	0.111719	58.15046	148.8417	168656.2	21.17981
标准差	11538537	0.080178	21.27562	289.7407	539616.7	3.12508
最小值	80000	0	25	8	-3664580	14.35065
最大值	39558757	0.25	110	1261.995	350000	31.96805
样本编号	7	8	9	10	11	12
均值	0.167761	22.98737	189.6031	2.170147	210488.1	25.60416
标准差	0.045396	9.366758	46.45738	1.971345	161316	3.103966
最小值	0.03822	0	1.582905	-0.5	75.1712	14.47561
最大值	0.2	30	640.5102	5	335357.6	35.80647
样本编号	13	14	15	16	17	
均值	2.919271	278.9364	144.2994	-83100.6	14513.36	
标准差	1.011305	8.655094	11.46113	98873.73	17754.71	
最小值	0.209332	254.0271	122.5975	-314601	0.15	
最大值	3.5	312.4448	172.6794	75	54809.38	

通过对操作变量的描述性分析，可以观察到各个操作变量优化后的取值范围，可以为未来操作变量的调整提供参考建议。

6.3 结果结论

通过建立基于粒子群算法的组合优化模型，我们成功优化了 266 个样本的辛烷值损失降幅，并分析了经过优化后，操作变量的分布情况。

7 问题五：模型可视化展示

7.1 问题五分析

为了催化裂化汽油精制过程的平稳生产，需要逐步调整变量。本题要求对133 号样本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。本题的求解主要是根据前文所建立的优化模型，对 133 号样本的主要操作变量进行迭代优化，最后将迭代优化过程以图形的方式展示，绘制产品辛烷值和硫含量的变化轨迹。

7.2 结果可视化

针对 133 号样本，我们需要根据附件四中各个操作变量的 Δ 值进行逐步调整，直到所有样本均优化结束。而在粒子群算法中，对操作变量进行逐步调整，实际上就是对各个粒子的速度进行约束，将其每一步变化后的速度改为 Δ 值，约束方案设定如下：

$$v_{i,n+1}^{j}=\begin{cases}\Delta & v_{i,n+1}^{j}>0\\ 0 & v_{i,n+1}^{j}=0\\ -\Delta & v_{i,n+1}^{j}<0\end{cases}$$

在迭代过程中，我们设定最大迭代次数为 50 次，这是根据迭代过程的群体最优所需的迭代次数所判断的结果，此次还可以进行改进。

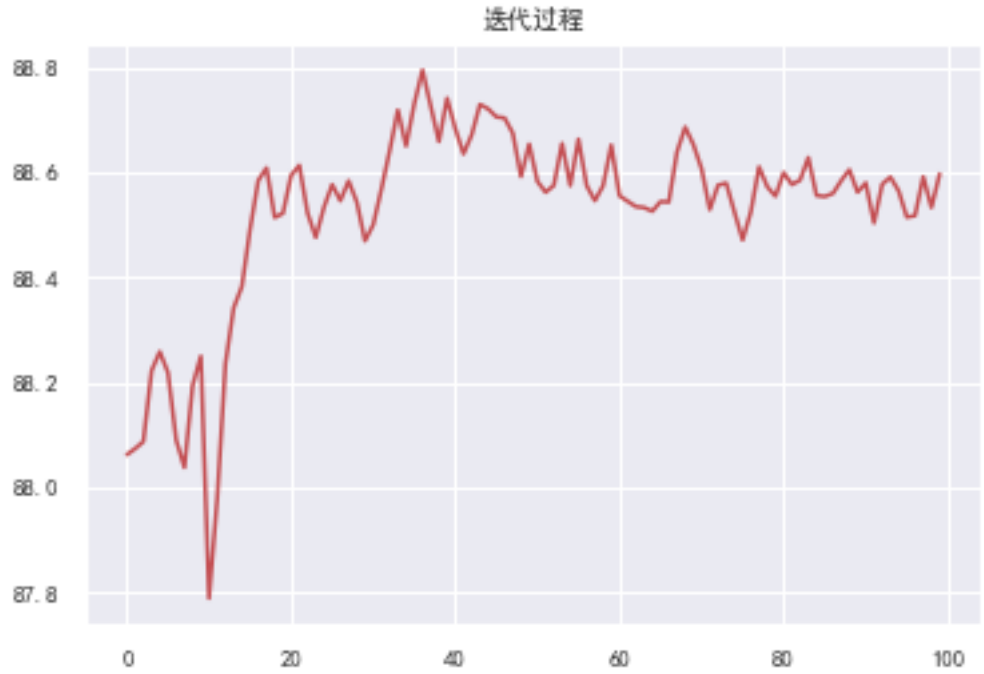


图 7-1 产品辛烷值变化过程

图 7-2 硫含量变化过程

根据硫含量和辛烷值迭代过程的变化曲线，可以发现这两个曲线的轨迹基本吻合，辛烷值持续上升，这也说明了辛烷值损失降幅不断得到了优化，硫含量也在不断上升，但始终没有超过 5μg/g，最终产品辛烷值达到了 88.6，而硫含量达到了 3.64μg/g，因此可以认为该脱硫保辛烷的汽油精制装置在降低辛烷值损失上仍有余力，且在控制硫含量上还有发挥的空间。如果要对硫含量进行进一步的优化，则需要对其它外在因素进行处理。

8 模型的评价与推广

8.1 模型的评价

8.1.1 模型的优点

（1）本文在研究对操作变量进行降维的过程中，首先对其进行了皮尔逊相关系数筛选，提出了与辛烷值相关程度不高的变量，这为后面进行基于随机森林重要度的特征筛选提供了便利，大大降低了时间复杂度

（2）本文在求解优化模型中，整体流程还算全面，既包括了问题本身所要求的，筛选出了符合规定的样本，也对优化后的操作变量进行了描述性分析，通过绘制其分布图来更直观地为未来解决化工问题提供思路与建议。

8.1.2 模型的缺点

（1）本文发现样本数据出现大量异常或超出上下限的情况，但按照附件三直接剔除样本，会导致样本数过少，这对于后面的建模是不友好的，因此本文只能保留样本，反而剔除变量，后面也许会有更好的方法来保留变量。

（2）本文在使用粒子群算法来求解优化模型中，部分参数仍有改进的空间。

8.2 模型的推广

在后续研究中，为保证同时兼顾数据的深度与广度，需要提高样本数量，对数据集进行多次训练，不断优化模型，提升模型预测精度。对于本文筛选出的变量，并未分析 其在模型中的具体表现，因此相关石化从业人员可考虑使用本文的模型进行进一步的 内在联系模式的探究。

本文主要是研究辛烷值在相关主要变量的作用下的预测与优化，但是建立的模型也 可以预测与优化汽油精制过程中其他一些产品的产值，即本文的模型具有-一定的推广价值。

参考文献

[1].Smith P F, Ganesh S, Liu P. A comparison of random forest regression and multiple linear regression for prediction in neuroscience[J]. Journal of neuroscience methods, 2013, 220(1): 85-91.

[2].Rodriguez-Galiano V, Sanchez-Castillo M, Chica-Olmo M, et al. Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines[J]. Ore Geology Reviews, 2015, 71: 804-818.

[3].苏高利,邓芳萍.关于支持向量回归机的模型选择[J].科技通报,2006(02):154-158.

[4].姚登举,杨静,詹晓娟基于随机森林的特征选择算法[J.吉林大学学报(工学版), 2014, 44 (1) :137-141.YAO Dengju, YANG Jing, ZHAN Xiaojuan.Feature Selection Algorithm Based on Random Forest[J].Journal of Jilin University (Engineering and Technology Edition) , 2014, 44 (1) :137-141.

[5].范土俊，张爱武,胡少兴，等基于随机森林的机载激光全波形云数据分类方法[J].中国激光, 2013, 40 (9) :0914001.FAN Shijun, ZHANG Aiwu, HU Shaoxing, et al.A Methodof Classification for Airborne Full Waveform LiDAR Data Based on Random ForestlJ.Chinese Journal of Lasers, 2013, 40 (9) :0914001.

[6]. Keely L C,Tan C M.Understanding Preferences for Income Redistribution[J].Journal of Public Economics,2008,92(516).

[7].Verikas A,Gelzinis A,Bacauskiene M.Mining Data with Random Forests:A Survey and Results of New Tests[J].Pattern Recognition,2011,44(2).

[8]金勇进.缺失数据的插补调整[J].数理统计与管理,2001(06):47-53.DOI:10.13860/j.cnki.sltj.2001.06.012.

[9]杨维,李歧强.粒子群优化算法综述[J].中国工程科学,2004(05):87-94.

[10]Yangyang Li,Xiaoyu Bai,Licheng Jiao,Yu Xue. Partitioned-cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation[J]. Applied Soft Computing,2017,56.

附录

问题 1　python 程序	数据预处理

```
#1、缺失值剔除：对某一位点，所有样本都为缺失值则直接剔除该位点
#2、缺失值填补：对某一位点，样本缺失值较多，超过 80%，则进行填充
#由于本文已通过 0 的方式来表现，故不需要主动删除，0 即代表删除。
def judge1_1(df):
    c = []
    for i in range(354):
        a = 0
        b = 0
        for j in range(40):
            if(df.iat[j,i] == 0):
                a += 1
            else:
                b = b + df.iat[j,i]
        if(a > 20):
            for j in range(40):
                df.iat[j,i] = 0
            print("第"+str(i)+"列 0 比较多，故剔除")
        else:
            for j in range(40):
                if(df.iat[j,i] == 0):
                    df.iat[j,i] = b/(40-a)
                    c.append(i)
    return c
judge1_1(df2_5_1)        #肉眼观测不需要进行缺失值处理
judge1_1(df2_5_2)

#3、最大最小越限处理：根据附件 4 的最大最小值进行判断
df4_1 = pd.DataFrame(df4[3])
df4_1 = df4_1.drop(df4_1.index[0])
df4_1.columns = ["取值范围"]
df4_1.index = map(str, np.arange(354))

def judge2_1(df1, df2):
    a = 0
    b = []
    for i in range(0, 354):
        ul = re.sub('\(|\)',",df1.iat[i,0])
        ul = re.sub('\（|\）',",ul)
        if("--" in ul):
            upper = "-" + ul.rsplit("-",1)[1]    #上限
            lower = ul.rsplit("-",2)[0]          #下限
```

```
            else:
                upper = ul.rsplit("-",1)[1]          #上限
                lower = ul.rsplit("-",1)[0]          #下限
            upper = float(upper)
            lower = float(lower)
            for j in range(0, 40):
                if(float(df2.iat[j,i]) <= lower or float(df2.iat[j,i]) >= upper):
                    df2.iat[j,i] = 0
                    b.append(i)
        return b
    judge2_1(df4_1, df2_5_1)
    judge2_1(df4_1, df2_5_2)

    #4、异常值处理：按照 3 西格玛原则
    def judge2_2(df3):
        a = np.std(df3, axis = 0, ddof =1)
        b = np.mean(df3)
        c = 0
        for i in range(0,354):
            for j in range(0,40):
                if(np.abs(b[i]-df3.iat[j, i]) > 3*a[i]):
                    df3.iat[j, i] = 0
                    c = c+1
        return c
    #judge2_2(df2_5_1)
    judge2_2(df2_5_2)
```

问题 2　python 程序	降维

```
    import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import MinMaxScaler
    from sklearn.preprocessing import StandardScaler
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.ensemble import RandomForestRegressor
    from sklearn.feature_selection import SelectFromModel
    from sklearn.pipeline import Pipeline

    df1_1 = df1_1.drop(df1_1.index[0:1])
    df1_1.index = map(str, np.arange(df1_1.index.size))
    column1 = ['硫含量_原料', '辛烷值_原料', '饱和烃', '烯烃', '芳烃', '溴值',
'密度', '硫含量_产品', '辛烷值_产品', '焦炭_待生', '硫含量_待生', '焦炭_再
生', '硫含量_再生']
```

```
    df1_1.columns = map(str,column1)

    df1_1 = df1_1.apply(lambda x:x.astype(float))
    corr1 = df1_1.corr("spearman")

    plt.rcParams['font.sans-serif'] = ['SimHei']
    plt.rcParams['axes.unicode_minus'] = False
    sns.set(font='SimHei', font_scale=0.8)
    plt.subplots(figsize=(9,9),dpi=1080,facecolor='w')
    p1 = sns.heatmap(corr1 ,annot=True, vmax=1, square=True, cmap="Blues", fmt='.2g')

    feat_labels = n1
    rf = RandomForestRegressor(n_estimators=500)
    rf.fit(X, y)
    importance = rf.feature_importances_

    #np.argsort()返回待排序集合从下到大的索引值，[::-1]实现倒序，即最终 imp_result 内保存的是从大到小的索引值
    imp_result = np.argsort(importance)[::-1][:25]

    #按重要性从高到低输出属性列名和其重要性
    for i in range(len(imp_result)):
        print("%2d. %-*s %f" % (i + 1, 30, feat_labels[imp_result[i]], importance[imp_result[i]]))

    #对属性列，按属性重要性从高到低进行排序
    feat_labels = [feat_labels[i] for i in imp_result]
    #绘制特征重要性图像
    plt.subplots(dpi=1080)
    plt.title('Feature Importance')
    plt.bar(range(len(imp_result)), importance[imp_result], color='lightblue', align='center')
    plt.xticks(range(len(imp_result)), feat_labels, rotation=90)
    plt.rc_context({'xtick.color':'black', 'ytick.color':'black'})
    plt.xlim([-1, len(imp_result)])
    plt.tight_layout()
    plt.show()
```

问题 3　python 程序	预测模型

```
    from sklearn.linear_model import LinearRegression as LR
from sklearn.metrics import r2_score
import numpy as np
import matplotlib.pyplot as plt
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
import sklearn
    from sklearn.svm import SVR
    import statsmodels.api as sm
    from sklearn.ensemble import RandomForestRegressor
    #多元线性回归模型
model1 = LR().fit(x_train, y_train)
    #利用随机森林进行训练
forest = RandomForestRegressor(
    n_estimators=3000,
    random_state=1,
    n_jobs=-1)
forest.fit(x_train, y_train)
    #支持向量回归
linear_svr = SVR(kernel='linear')
model1 = linear_svr.fit(x_train, y_train)
linear_svr_y_predict = linear_svr.predict(x_test)
    # 评价指标
def perfomance_reg(model,x,y,name=None):
    y_predict = model.predict(x)
    check = pd.DataFrame(y)
    check['y_predict'] = y_predict
    check['abs_err'] = abs(check['y_predict'] - check[y.name])
    check['ape'] = check['abs_err'] / check[y.name]
    ape = check['ape'][check['ape']!=np.inf].mean()
    if name:
        print(name,':')
    print(f'均方误差: {sklearn.metrics.mean_squared_error(y,y_predict)}')
    print(f'绝对平均误差: {sklearn.metrics.mean_absolute_error(y,y_predict)}')
    print(f'R 平方: {r2_score(y,y_predict)}')
    print(f'平均绝对误差百分比: {ape}')
    print('- - - - - ')
perfomance_reg(model1, x, y)
    #可视化
plt.figure()
plt.plot(np.arange(len(y_pred)), y_test,'go-',label='true value')
plt.plot(np.arange(len(y_pred)),y_pred,'ro-',label='predict value')
plt.title("multipleRegression")
plt.legend()            # 将样例显示出来
plt.show()
```

问题 4　python 程序	优化模型

```
import numpy as np
import pandas as pd
import scipy
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib as mpl
from random import choices
from turtle import speed
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn import preprocessing

#构建预测模型 1
x = df3_1.iloc[:,1:]
y = df3_1.iloc[:,0]
x = preprocessing.scale(x)#归一化
linear_svr = SVR(kernel='linear')
linear_test = linear_svr.fit(x,y)

#构建预测模型 2
y1 = df3_2.iloc[:,6]
x1 = df3_2.drop(df3_2.columns[6], axis = 1)
x1 = preprocessing.scale(x1)#归一化
linear_svr1 = SVR(kernel='linear')
linear_test1 = linear_svr1.fit(x1,y1)

#计算适应度
mpl.rcParams['font.sans-serif'] = ['SimHei']
mpl.rcParams['axes.unicode_minus'] = False

def fitness_func(X):
    X = np.column_stack((X_1, X))
    X = preprocessing.scale(X)
    return linear_test.predict(X)

def fitness_func1(X):
```

```
    X = np.column_stack((X_1_2, X))
    X = preprocessing.scale(X)
    return linear_test1.predict(X)

#更新速度
def velocity_update(V, X, pbest, gbest, c1, c2, w, max_val):
    size = X.shape[0]
    r1 = np.random.random((size, 1))
    r2 = np.random.random((size, 1))
    V = w*V+c1*r1*(pbest-X)+c2*r2*(gbest-X)
    # 防止越界处理
    for i in range(17):
        max = max_val[i]
        min = -max
        for j in range(266):
            if(V[j,i] > 0):
                V[j,i] = max
            elif(V[j,i] < 0):
                V[j,i] = min
    return V

#更新位置
def position_update(X, V):
    M = X+V
    for i in range(17):
        lower = df3_positon.values[i,0]
        upper = df3_positon.values[i,1]
        for j in range(266):
            if(M[j,i] > upper):
                M[j,i] = upper
            elif(M[j,i] < lower):
                M[j,i] = lower
    return M

c1 = 2
c2 = 2
r1 = None
r2 = None
dim = 17
size = 266
iter_num = 50                          #最大迭代次数
max_val = df3_speed.values             #速度范围
fitness_val_list = []
f133 = []
```

```
s133 = []
# 初始化种群各个粒子的位置
X = X_2
# 初始化各个粒子的速度
V = df3_1.iloc[:,8:]
for i in range(dim):
    speed1 = df3_speed.iat[i,0]
    speed2 = -speed1
    for j in range(size):
        V.iat[j,i] = choices([speed2,speed1], [0.5,0.5])[0]
V = V.values
# print(X)
p_fitness = np.array(y)
g_fitness = p_fitness.min()
s_fitness = fitness_func1(X)

fitness_val_list.append(g_fitness)
f133.append(p_fitness[120])
s133.append(s_fitness[120])
# 初始化的个体最优位置和种群最优位置
pbest = X
gbest = X_2[129,]
# 迭代计算
for i in range(1, iter_num):
    w = 0.5+(i/iter_num)*0.4
    V = velocity_update(V, X, pbest, gbest, c1, c2, w, max_val)
    X = position_update(X, V)
    p_fitness2 = fitness_func(X)
    s_fitness2 = fitness_func1(X)
    f133.append(p_fitness2[120])
    s133.append(s_fitness2[120])
    g_fitness2 = p_fitness2.min()

    # 更新每个粒子的历史最优位置
    for j in range(size):
        if p_fitness[j] > p_fitness2[j]:
            pbest[j] = X[j]
            p_fitness[j] = p_fitness2[j]
        # 更新群体的最优位置
        if g_fitness > g_fitness2:
            gbest = X[p_fitness2.argmin()]
            g_fitness = g_fitness2
        # 记录最优迭代记录
        fitness_val_list.append(g_fitness)
```

```
        i += 1

# 输出迭代结果
print("最优值是：%.5f" % fitness_val_list[-1])
print("最优解是：x=%.5f,y=%.5f" % (gbest[0], gbest[1]))

# 绘图
plt.plot(fitness_val_list, color='r')
plt.title('迭代过程')
plt.show()

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
sns.set(font='SimHei', font_scale=0.8)
plt.figure(figsize=(20,8))
for n,i in enumerate(range(9,17)):
    plt.subplot(2,4,n+1)
#      plt.title(i)
    sns.distplot(pbest_p[i])
    plt.xlabel(i, fontsize = 14)
    plt.ylabel('')
    plt.tick_params(labelsize = 14)
```

问题 5　python 程序	可视化

```
# 绘图
plt.rcParams['font.sans-serif'] = ['SimHei']  # 黑体
plt.rcParams['axes.unicode_minus'] = False    # 解决无法显示符号的问题
sns.set(font='SimHei', font_scale=0.8)        # 解决 Seaborn 中文显示问题
plt.plot(s133, color='r')
plt.title('迭代过程')
plt.show()
# 绘图
plt.rcParams['font.sans-serif'] = ['SimHei']  # 黑体
plt.rcParams['axes.unicode_minus'] = False    # 解决无法显示符号的问题
sns.set(font='SimHei', font_scale=0.8)        # 解决 Seaborn 中文显示问题
plt.plot(f133_1, color='r')
plt.title('迭代过程')
plt.show()
```