|  |  |  |
| --- | --- | --- |
| 所属类别 | **2022年“华数杯”全国大学生数学建模竞赛** | 参赛编号 |
| 本科组 | CM2200889 |

插层熔喷非织造材料的性能控制研究

摘要

本文针对插层熔喷非织造材料的性能控制研究，综合考虑工艺参数与结构变量、结构变量和产品性能之间的关系模型，并以此模型为基础对产品性能的效果和在工艺参数为多少时过滤效率能达到最优的问题进行了研究。

针对问题一，本文先进行数据清洗（异常值处理、缺失值处理、合并data1和data2的数据)。考虑到结构变量、产品性能在插层后不再有改变。设置两个角度分析其在插层后变化规律：直接考虑分析其下各指标在插层前后变化规律和利用因子分析法降维处理各评定指标，根据贡献率结果可直接分析其在插层后的变化规律。对于分析插层率对以上变化的影响采用灰色关联分析。结果发现：插层率对孔隙率、压缩回弹性、过滤效率、透气性、过滤阻力是促进作用，但增加了厚度，根据文献试验数据表明:随着熔喷非织造材料厚度的增加,过滤效率显著提高,材料的平均流量孔径和泡点孔径减小,而泡点压力、平均流量压力和过滤阻力增大[1]。因此厚度变化既有促进作用又有阻碍作用。

针对问题二，该题是一个回归预测类的问题，先建立LinearRegression、LightGBM、随机森林三个回归模型。基于不同结构变量指标，计算预测结果误差值，分别选择其适合模型。分别对原始数据data3进行了训练测试，并选择误差率最小的模型进行对结构变量进行预测。结果填入题目的表格中。

针对问题三，要分析结构变量与产品性能之间的关系，使用因子分析法进行降维，通过载荷图分析出各个指标之间的差异。探究结构变量之间、产品性能之间的关系使用Spearman相关系数进行分析，探究当工艺参数为多少时，产品的过滤效率将会达到最高，继用问题二中预测回归模型进行预测，将过滤效率作为一个因变量其他指标作为自变量进行一个回归分析，构建判别函数。结果发现：结构变量与产品之间有一个公因子；结构变量之间指标存在相关性，且都为正相关；产品性能之间指标之间存在相关性，既有正相关又有负相关；选择最优的即随机森林回归模型得到过滤效率最高的为82.88%（接收距离在20-22cm之间、热风速度在 1155-1200r/min之间）

针对问题四，对data3进行数据预处理，计算出data3的过滤效率正/负向指标，筛选有效数据并分出训练集和测试集，根据条件生成预测集，建立继用问题二中LinearRegression、LightGBM、随机森林三个回归模型，比较和评估选择其中最优的模型，对预测集进行预测，并将评价指标为最大值的所有预测结果筛选出来，最后取工艺参数的平均值为结果。

关键词：灰色关联分析、Spearman系数、因子分析、随机森林、LightGBM、线性回归

一、问题重述

1.1问题背景

近来，全球工业化的发展不断加快，这就使得全球的空气环境日益恶化，严重的危害到人们的健康。此外，疫情的存在对于人们日常生活的影响也是非常大的。由此，口罩对于当下的我们是十分重要的，而熔喷非织造材料是口罩生产的重要原材料，并且熔喷非织造材料因具备纤维细度小,纤网结构致密,孔隙率高,加工性能良好等优点而成为当前常用的非织造过滤材料之一[2]。科学家团队们通过对插层熔喷非织造材料的工艺参数、结构变量和产品性能进行一系列的研究，让熔喷非织造材料的性能达到更高。为检验熔喷非织造材料的效果，本文先对实验过程进行仿真模拟。

1.2问题提出

鉴于以上背景，本文需要解决的问题如下：

问题一：研究插层后结构变量、产品性能的变化规律，并分析插层率对于这些变化是否有影响。

问题二：研究工艺参数与结构变量之间的关系。表1给了8个工艺参数组合，请将预测的结构变量数据填入表1中。

问题三：研究结构变量与产品性能的关系，以及结构变量之间、产品性能之间的关系。

结合第二问，研究当工艺参数为多少时，产品的过滤效率将会达到最高。

问题四：产品生产需要兼顾各方面的条件和要求。如接收距离不大可能大于100cm，热空气速度也不大可能大于2000 r/min。按照应用的要求，厚度尽量不要超过3mm，压缩回弹性率尽量不要低于85%。另外，为了防止熔喷非织造过滤材料因过滤阻力大使得大量颗粒堵塞孔隙而致使过滤效率迅速下降的现象发生，产品是需要同时追求过滤效率高和过滤阻力小的目标的。

二、问题分析

2.1问题一的分析：

首先进行数据预处理，对于研究插层后结构变量、产品性能的变化规律，从四个角度进行分析：1.用处理后的原始数据利用matplotlib作结构变量、产品性能变量各指标插层前后数据可视化分析；2.结构变量、产品性能下各个指标PCA降维后可视化分析；3.用数理统计做插层前后指标均值和方差变化情况进行统计分析；4.计算插层率的情况下，各指标的的提高率进行分析。对于分析插层率对这些变化是否有影响采用灰色关联分析。

2.2问题二的分析：

根据工艺参数预测结构变量，使用原始数据data3做训练集，表格中工艺参数做测试集。考虑到预测结构变量的数值是未知，为尽可能减少误差，建立LinearRegression、LightGBM、随机森林三个回归模型。因此可以考虑学习效果较好的集成学习模型，在此基础上进行比较和评估，选择其中最优的模型进行预测。

2.3问题三的分析：

对于结构变量和产品性能之间的关系，通过因子分析法进行降维，载荷图分析出各个指标之间的差异。探究结构变量之间、产品性能之间的关系使用Spearman相关系数进行分析，探究当工艺参数为多少时，产品的过滤效率将会达到最高，继用问题二中预测回归模型进行预测，将过滤效率作为一个因变量其他指标作为自变量进行一个回归分析，构建判别函数，对比差异并取最优的模型，在取最高过滤效率的参数。

2.4问题四的分析：

根据原始数据data3筛选出接收距离不大可能大于100cm，热空气速度也不大可能大于2000 r/min。按照应用的要求，厚度尽量不要超过3mm，压缩回弹性率尽量不要低于85%的数据分出训练集和测试集，根据接收距离，热空气速度的范围生成预测集，建立LinearRegression、LightGBM、随机森林三个回归模型。比较和评估，选择其中最优的模型，对预测集进行预测，并将评价指标为最大值的所有预测结果筛选出来，最后取筛选结果工艺参数的平均值为结果。

三、模型假设

1：假设除了本文提到的工艺参数，没有其它因素影响到结构变量；

2：假设各个指标之间是相互独立的，如厚度不会对孔隙率造成影响；

3：假设影响工艺参数，能够使得过滤效率尽量的高的同时力求过滤阻力尽量的小

四、符合说明与名词解释

|  |  |
| --- | --- |
| 符号（名词） | 说明 |
|  | 插层率影响下指标的提高率 |
|  | 灰色关联度值 |
|  | 预测值 |
|  | 母序列和特征序列之间的灰色关联系数值 |
| 初值化 | 数据统一除以最开始的值，除以初值能将值都整理到1附近。 |
| 均值化 | 数据除以均值，除掉以后就能归一化到1的量级附近 |

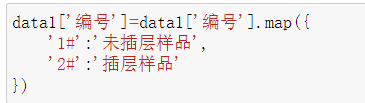
五、模型的建立与求解

5.1问题一的模型建立与求解

5.1.1数据预处理

缺失值、异常值的处理是数据处理不可缺少的一环。通过观察原始数据发现：1#是熔喷样品即未插层，2#为插层样品。为了方便后面解决问题对数据进行处理。

1. 原始数据中1#、2#异常数据进行处理。



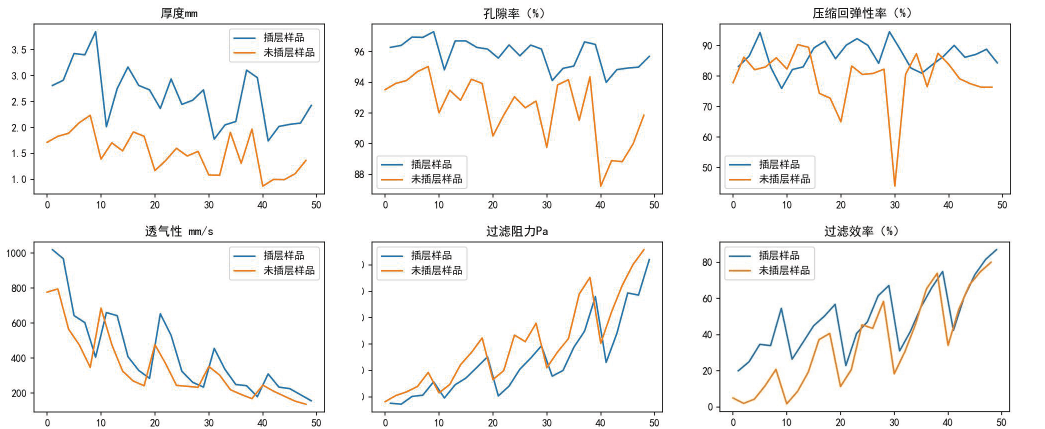
1. 组号中的空值采用向前填充方法补齐。



1. 合并原始数据中datat1和data2数据

5.1.2插层后结构变量、产品性能的变化规律

1. 插层前后结构变量、产品性能下各个指标的变化规律

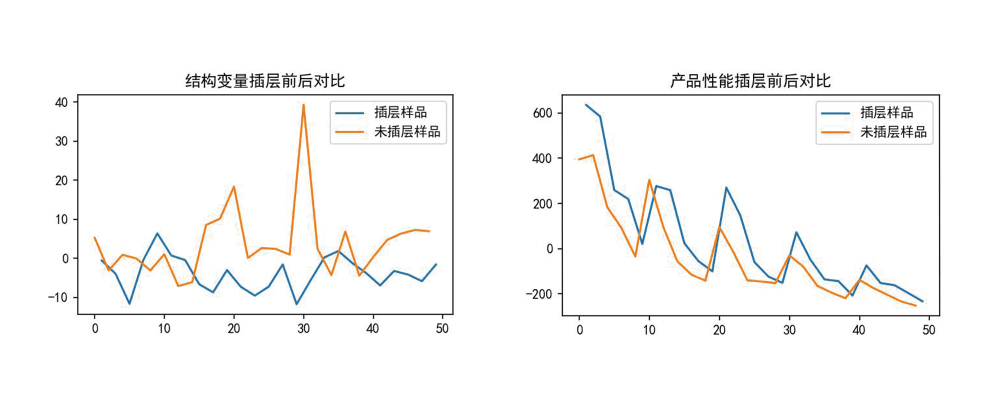


**图 5-1 结构变量与产品性能各指标变化规律**

由表可以明显看出在插层后厚度和孔隙率都有所增加，对压缩回弹性率有起到平稳的作用，能够增加透气性，减小过滤阻力，在一定程度上能够增加过滤效率。

综上插层率对孔隙率、压缩回弹性、过滤阻力、过滤效率、透气性的变化情况是有促进作用的。但增加了厚度，对厚度的变化情况有一定的减小作用。

1. 结构变量、产品性能下各个指标PCA降维后的变化规律



**图 5-2 PCA后插层前后对比**

1. 数理统计做插层前后指标均值和方差变化情况进行统计分析

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 插层（前/后） | 厚度mm | 孔隙率（%） | 压缩回弹性率（%） | 过滤阻力Pa | 过滤效率（%） | 透气性mm/s |
| 均值前 | 1.5188 | 92.3492 | 79.4416 | 29.7844 | 34.99204 | 347.604 |
| 方差前 | 0.382618 | 2.090879 | 9.32606 | 16.462431 | 25.130497 | 189.575125 |
| 均值后 | 2.6074 | 95.8612 | 86.6228 | 24.15732 | 49.38668 | 422.1344 |
| 方差后 | 0.54895 | 0.917997 | 4.433299 | 14.783515 | 18.874436 | 236.816734 |

**表 5-1 数理统计分析**

1. 计算不同插层率情况下，各指标的的提高率。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **差层率（%）** |
| 1 | 0.4496 | 0.0305 | -0.0017 | -0.1689 | 14.234 | -0.0374 | 16.53 |
| 2 | 0.5902 | 0.0264 | 0.0049 | -0.3123 | 11.916 | 0.2175 | 24.74 |
| 3 | 0.6229 | 0.0235 | -0.0016 | -0.2374 | 1.8809 | 0.2712 | 19.37 |
| 4 | 0.6385 | 0.0295 | 0.0689 | -0.07343 | 3.0199 | 0.3121 | 36.44 |
| 5 | 0.7204 | 0.0239 | -0.117 | -0.1836 | 1.6244 | 0.1688 | 31.19 |
| 6 | 0.8122 | 0.0301 | 0.1487 | -0.1463 | 7.0465 | 0.1389 | 31.45 |

**表 5-2 插层率对各指标的影响**

计算公式如下：

 (1-1)

由表2可得：厚度、孔隙率、压缩回弹性、过滤阻力、过滤效率、透气性等于i(i=1,2...6)。由表可得随着插层率的增加，厚度在增加；压缩回弹性和过滤效率有不同程度的提高；空隙率有提高和降低，但变化都不十分明显。过滤阻力在降低；透气性在超过阈值约16.53%后，有不同程度的提高。

5.1.4建立灰色关联分析插层率对于这些变化是否有影响

1. 针对数据进行无量纲化处理（均值化、初值化）。由于原始数据中各因素列中的数据因量纲不同，不便于比较或在比较时难以得到正确的结论。因此在进行灰色关联度分析时，要进行数据的无量纲化处理。

初值化处理：

 (2-1)

均值化处理：

 (2-1)

初值化均值化后结果部分展示：



**表 5-3 初值化均值化后结果**

从表3可知，针对6个评价项（厚度mm、压缩回弹性率（%）、孔隙率（%）、过滤阻力Pa、过滤效率（%）、透气性 mm/s）以及50项数据进行灰色关联度分析,并且以插层率（%）作为“参考值"(母序列)，研究6个评价项(厚度mm、压缩回弹性率（%）、孔隙率（%）、过滤阻力Pa、过滤效率（%）、透气性 mm/s与插层率（%）的关联关系（关联度）。

并基于关联度提供分析参考,分辨系数取0.5，分辨系数ρ∈(0，∞)，ρ越小，分辨力越大，一般ρ的取值区间为 ( 0，1)，当 ρ≤0.5463时，分辨力最好，结合关联系数计算公式计算出的关联系数值,可得出插层率的变化厚度mm、压缩回弹性率（%）、孔隙率（%）、过滤阻力Pa、过滤效率（%）、透气性 mm/s都有不同程度的关联。

1. 求解母序列（对比序列）和特征序列之间的灰色关联系数值

 (2-3)

1. 求解灰色关联度值

 (2-4)

1. 对灰色关联度值进行排序，对排序结果进行分析

|  |  |  |  |
| --- | --- | --- | --- |
| **关联度结果** | | | |
| 评价项 | 关联度 | 排名 |
| 厚度mm | 0.724 | 1 |
| 过滤效率% | 0.718 | 2 |
| 透气性mm/s | 0.697 | 3 |
| 压缩回弹性率% | 0.671 | 4 |
| 孔隙率% | 0.668 | 5 |
| 过滤阻力Pa | 0.662 | 6 |

**表 5-4 灰色关联度值排序结果**

结合上述关联系数结果进行加权处理，最终得出关联度值，使用关联度值针对6个评价对象进行评价排序；关联度值介于0-1之间，该值越大代表其与“参考值”(母序列)之间的相关性越强,也即意味着其评价越高。

从上表可以看出：针对本次6个评价项,厚度mm评价最高(关联度为：0.724)，其次是过滤效率（%）(关联度为：0.718)。

5.2问题二的模型建立与求解

5.2.1 建立LightGBM算法模型

通过对相关文献的阅读分析，基于以往的学者在基础回归问题分析方法上的研究理

论，选取了比较常用和相对具有权威性的技术类指标对模型进行训练，通过LightGBM

算法构建模型，对数据进行预测，具有训练速度快，消耗内存少的特点。其主要思想是通过决策树的迭代训练以得到最优模型[3] 。

1. 通过训练集数据建立LightGBM回归模型，其参数主要包括以下几种:

(1) num\_leaves:代表每一棵树具有的叶子数量;

(2) Learning\_rate:算法的学习率，是每一次权重改变的步长因素，初始训练时可以采用较大的数值，加快训练速度提高效率，参数优化时可以采用较小的数值，进行精细化的调参，有助于获得更好的效果;

(3） max\_depth:树的最大学习深度。这个参数用来控制对每个树模型允许能达到的最大深度，可以对模型的复杂度进行限制，用来规避过拟合问题，提高泛化能力;

(4) min\_data:一片叶子中允许具有的数据最小数量，这个参数可以避免通过决策树分类后的某一个类别数据数量大小分布不均，一定程度上也可以避免过拟合问题;

(5) feature\_fraction:选择的特征占原始特征种类数的比例，取值在0到1之间。这个参数的设置可以提升模型训练的速度，也可以一定程度上优化过拟合的问题;

(6） bagging\_fraction:所选择的数据量占原始总数据量的比例，这个参数的取值同样在0到1之间，决定了采用多少比例的数据进行训练，对模型训练的速度也有一定影响。

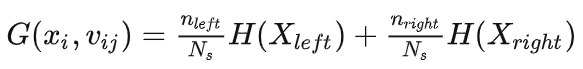
2.其中本文利用Python中sklearn库建立模型并选择合适参数。

3.通过建立的LightGBM来计算特征重要性，将建立的LightGBM回归模型应用到训练、测试数据，得到模型评估结果。

4.由于LightGBM具有随机性，每次运算的结果不一样，若保存本次训练模型，后续可以直接上传数据到本次训练模型进行预测。

5.2.2 随机森林模型建立

随机森林属于Bagging类算法，而Bagging 又属于[集成学习](https://link.zhihu.com/?target=https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E5%AD%A6%E4%B9%A0/3440721" \t "https://zhuanlan.zhihu.com/p/_blank)一种方法。在训练阶段，随机森林使用bootstrap采样从输入训练数据集中采集多个不同的子训练数据集来依次训练多个不同决策树[4]；在预测阶段，随机森林将内部多个决策树的预测结果取平均得到最终的结果。实现的RFR是将多个二叉决策树（即CART，这也是*sklearn,spark*内部实现的模型）打包组合而成的，训练RFR便是训练多个二叉决策树。在训练二叉决策树模型的时候需要考虑怎样选择切分变量(特征）、切分点以及怎样衡量一个切分变量、切分点的好坏。针对于切分变量和切分点的选择，实现采用穷举法，即遍历每个特征和每个特征的所有取值，最后从中找出最好的切分变量和切分点；针对于切分变量和切分点的好坏，一般以切分后节点的不纯度来衡量，即各个子节点不纯度的加权和。其计算公式如下：

 (3-1)

其中，为某一个切分变量，为切分变量的一个切分值，nleft,nright,Ns分别为切分后左子节点的训练样本个数、右子节点的训练样本个数以及当前节点所有训练样本个数，,分为左右子节点的训练样本集合，为衡量节点不纯度的函数。

决策树中某一节点的训练过程在数学等价于下面优化问题：

 (3-2)

即寻找G最小的切分变量和切分点。

在本文实现的回归随机森林中，选用MSE，即针对某一切分点

 (3-3)

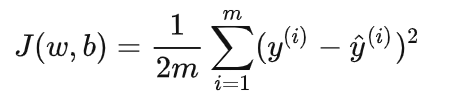
5.2.3 线性回归模型建立

一般来说，当我们拿到一个实际问题时首先会根据问题的背景结合常识选择一个合适的模型。同时，现在常识告诉我们房价的增长更优先符合线性回归这类模型，因此可以考虑建立一个如下的线性回归模型：

(4-1)

其中w叫权重参数（weight），b叫偏置（bias）或者截距（intercept）。当求解得到未知参数,之后也就意味着我们得到了预测模型，即给定工艺系数，就能够预测出其对应的结构变量的指标。

当我们建立好一个模型后，通过给定的数据，也叫训练集（traing data）来对模型h(x)进行求解，通过计算每个样本的真实结构变量（厚度或孔隙率或压缩回弹性）与预测结构变量（厚度或孔隙率或压缩回弹性）之间的均方误差来进行刻画：

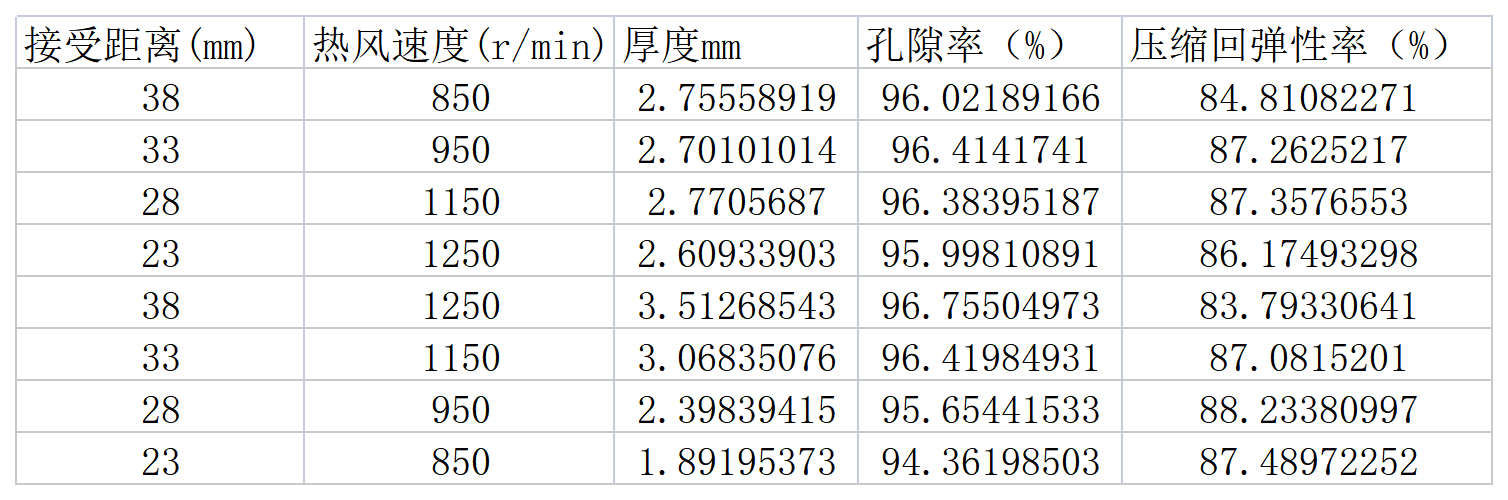
 (4-2)

 (4-3)

其中，m表示总的样本数，x1(i)表示第i个接收距离，同理x2（i）表示第i个热风速度 ，y(i)表示第i个结构变量（厚度或孔隙率或压缩回弹性）的真实值，(i)表示第i个结构变量（厚度或孔隙率或压缩回弹性）预测值。最后综合三个模型分别对结构变量预测结果的误差率

选择误差率最小的模型进行预测，得出以下预测结果：

**表 1 问题二的结果**



5.3问题三的模型建立与求解

5.3.1因子分析

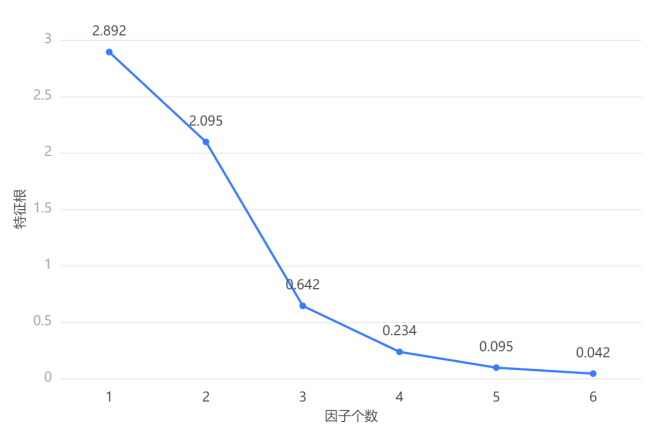
1.通过载荷图分析出各个指标之间的差异，由检验结果可知 KMO值等于0.626，且 Bartlett球形度检验的Р值=0.000<0.05，所以在95%的置信水平下拒绝相关系数矩阵是一个单位矩阵的原假设，说明各指标变量之间存在相关性，表明各个评定指标数据较适合进行因子分析，将原来的指标转化为公共因子的效果较好。



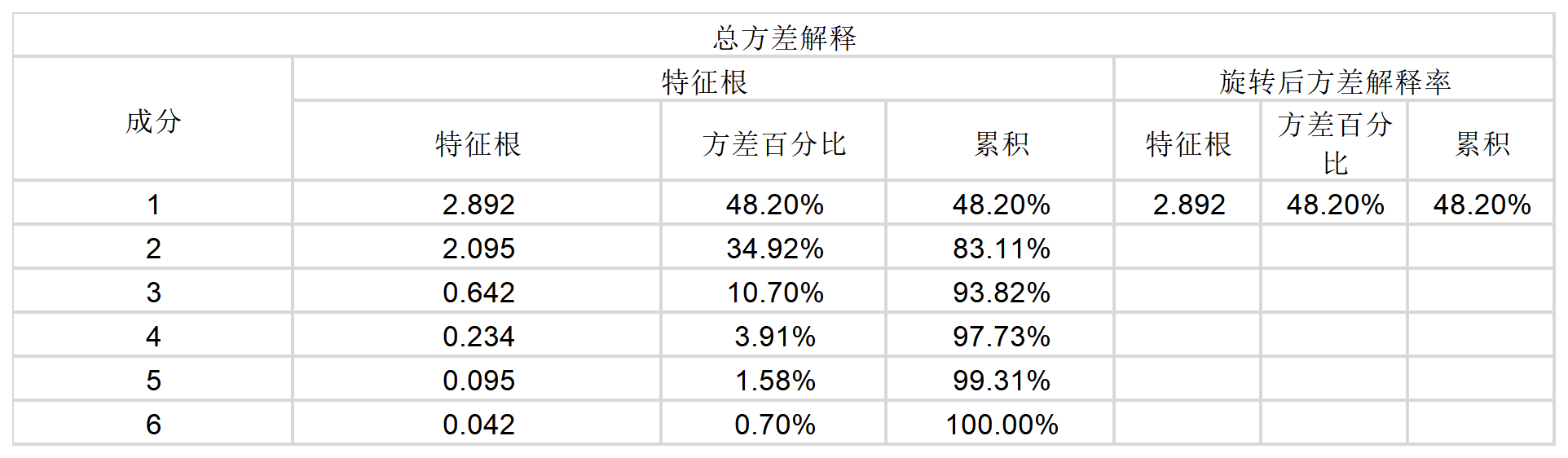
**表 5-6 KMO和Bartlett的检验**

1. 主成分法、碎石图分析提取初等公因子

用主成分法计算初等因子载荷矩阵，确定初等因子。计算相关系数矩阵的特征值，求取相应的特征向量，则初等因子的载荷矩阵为则初等因子的载荷矩阵为，得到6个初等因子。根据碎石检验观察特征值的变化，确定选取的初等公因子数：

****

**图 5-3 碎石图**

****

**表5-7 方差可解释表**

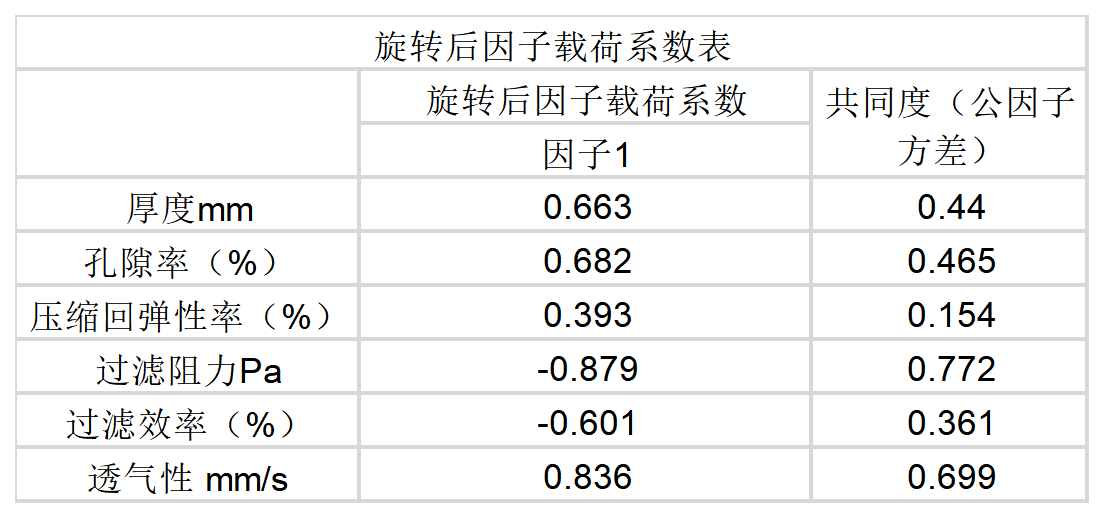
参照碎石图，可得方差解释表，当成分数为6时特征值较前一特征值出现较大下降，并且该特征值数 值较小，此后的特征值变化曲线趋于平缓，变量解释的贡献率达到93.818%，最终选取的公因子6个为。

1. 最大方差法旋转因子载荷矩阵，进行降维分析

得到初等因子模型后，由于其中的公因子不一定可以反映出问题的实质特征，需要

通过因子旋转使每个公因子上的载荷分配更清晰，从而减少解释公因子实际意义的主观

性。利用最大方差法旋转因子，得到旋转后的载荷矩阵：

****

**表 5-8 旋转后因子载荷矩阵**

评价指数为：

F1=0.229×厚度mm+0.236×孔隙率（%）+0.136×压缩回弹性率（%）-0.304×过滤阻力Pa-0.208×过滤效率（%）+0.289×透气性 mm/s

分析到每个主成分中隐变量的重要性

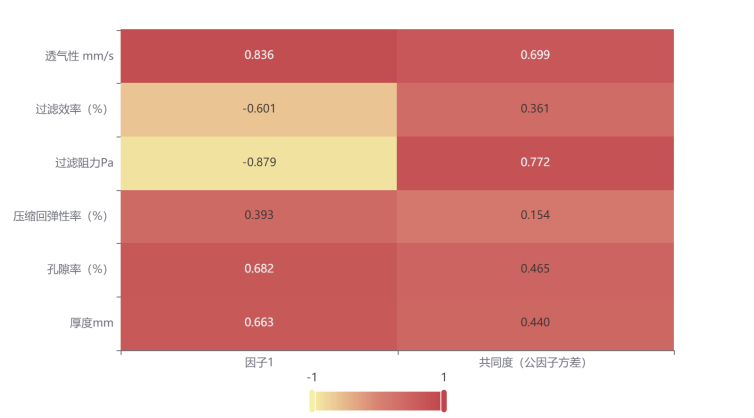
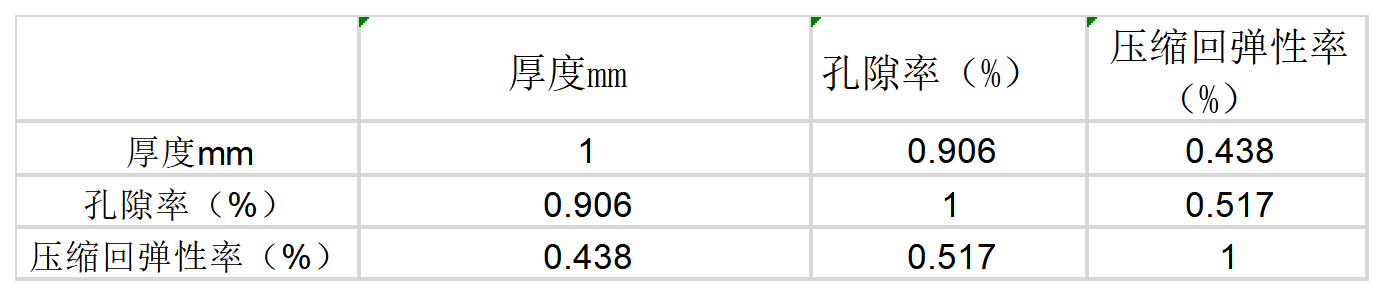
****

图 5-4 **因子载荷矩阵热力图**

5.3.2结构变量的关系，相关系数分析法Spearman

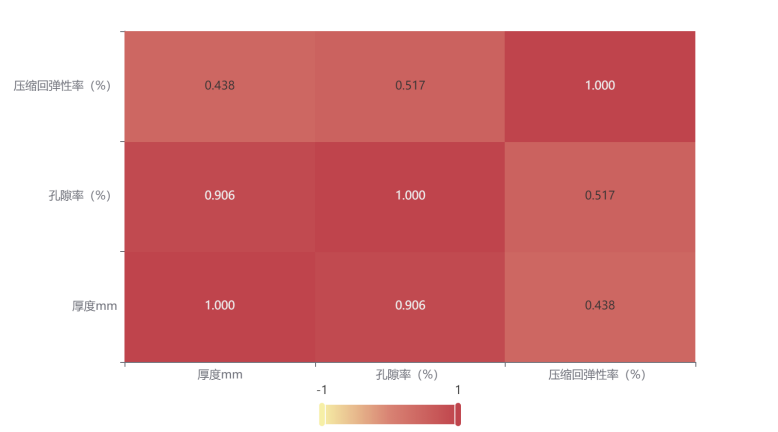
1.首先对数据进行正态分布检验，结果表明数据基本符合正态分布，所以选用Pearson相关系数分析。

2.先对结果变量之间是否存在统计上的显著关系进行检验；



**表5-9结构变量之间指标相关性**

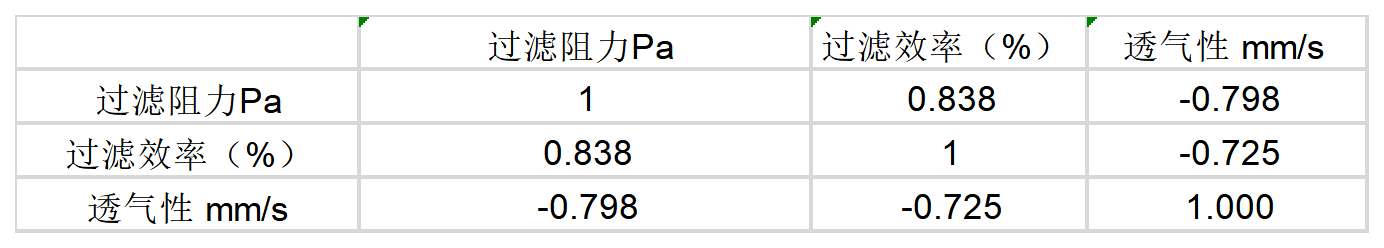
根据判断条件p（p<0.05，p<0.01）说明结构变量之间存在相关性，且都是正相关



**图 5-5 结构变量相关系数热力图**

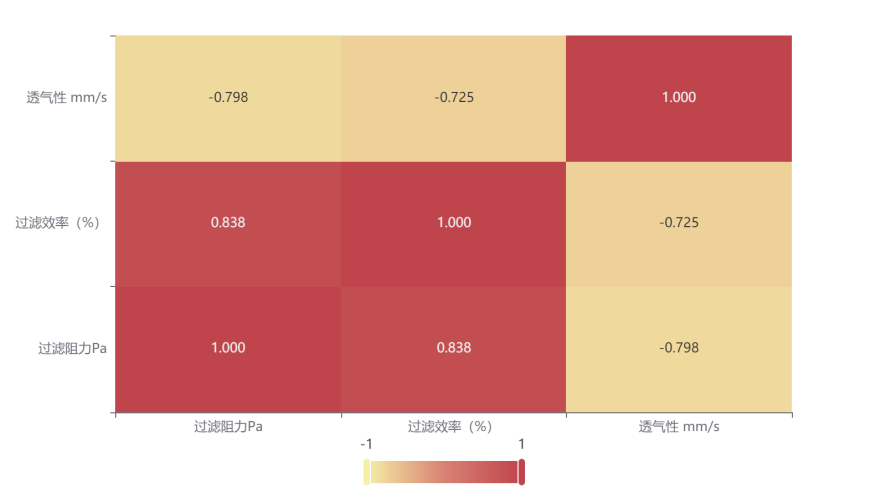
由相关系数Pearson图表明：压缩回弹性率与孔隙率的相关度最高，孔隙率与厚度的相关度最高。

5.3.3产品性能的关系，相关系数分析法Pearson



**表 1-10 结构变量之间指标相关性**

同理判断p值可得出产品性能之间存在相关性。由相关系数Pearson图表明：过滤阻力与透气性的相关度最高，且为负相关；过滤效率与过滤阻力的相关度最高，且为正相关。



**图 5-2 产品性能指标相关热力图**

5.3.4继用问题二的三种回归模型对原始数据data3进行训练和测试，最后分析三种模型的差异，选择最优的即随机森林回归模型得到过滤效率最高的为82.88%（接收距离在20-22cm之间、热风速度在 1155-1200r/min之间），如图预测结果：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 接受距离cm | 热分速度r/min | 过滤效率% |
| 71 | 20 | 1155 | 82.88 |
| 72 | 20 | 1160 | 82.88 |
| 73 | 20 | 1165 | 82.88 |
| 74 | 20 | 1170 | 82.88 |
| 75 | 20 | 1175 | 82.88 |
| 76 | 20 | 1180 | 82.88 |
| 77 | 20 | 1185 | 82.88 |
| 78 | 20 | 1190 | 82.88 |
| 79 | 20 | 1195 | 82.88 |
| 80 | 20 | 1200 | 82.88 |
| 152 | 21 | 1155 | 82.88 |
| 153 | 21 | 1160 | 82.88 |
| 154 | 21 | 1165 | 82.88 |
| 155 | 21 | 1170 | 82.88 |
| 156 | 21 | 1175 | 82.88 |
| 157 | 21 | 1180 | 82.88 |
| 158 | 21 | 1185 | 82.88 |
| 159 | 21 | 1190 | 82.88 |
| 160 | 21 | 1195 | 82.88 |
| 161 | 21 | 1200 | 82.88 |
| 233 | 22 | 1155 | 82.88 |
| 234 | 22 | 1160 | 82.88 |
| 235 | 22 | 1165 | 82.88 |
| 236 | 22 | 1170 | 82.88 |
| 237 | 22 | 1175 | 82.88 |
| 238 | 22 | 1180 | 82.88 |
| 239 | 22 | 1185 | 82.88 |
| 240 | 22 | 1190 | 82.88 |
| 241 | 22 | 1195 | 82.88 |
| 242 | 22 | 1200 | 82.88 |

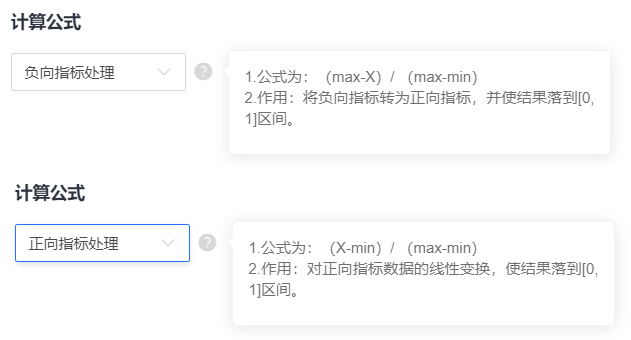
**表 5-11 问题三预测结果**

5.4问题四的模型建立与求解

5.4.1 训练集和测试集的划分

①数据预处理

使用负向指标处理公式和正向指标处理公式计算出data3的过滤效率正/负向指标。

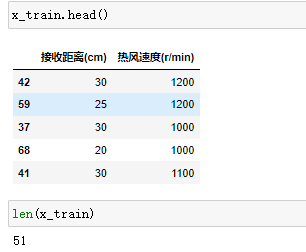
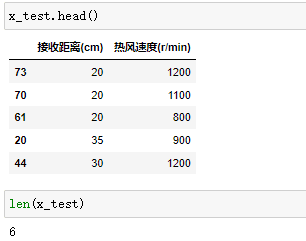
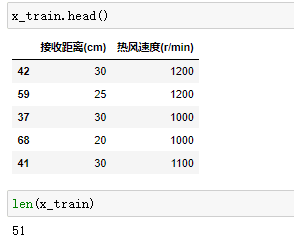


**图 5-7**

添加字段evaluate为评价指标，evaluate=0.5\*过滤阻力+0.5\*过滤效率。

②数据集的划分

将data3处理后的数据按照1:9的比例随机划分为训练集和测试集,训练集数据为51条，测试集数据为6条

5.4.2 模型的选择

①机器学习框架的选择

我们采用的机器学习框架是sklearn,，它是一个开源的基于python语言的机器学习工具包。它通过NumPy, SciPy和Matplotlib等python数值计算的库实现高效的算法应用，并且涵盖了几乎所有主流机器学习算法。继用问题二中的线性回归、随机森林、LightGBM回归模型。

在工程应用中，用python手写代码来从头实现一个算法的可能性非常低，这样不仅耗时耗力，还不一定能够写出构架清晰，稳定性强的模型。更多情况下，是分析采集到的数据，根据数据特征选择适合的算法，在工具包中调用算法，调整算法的参数，获取需要的信息，从而实现算法效率和效果之间的平衡。而sklearn，正是这样一个可以帮助我们高效实现算法应用的工具包。

②模型评价指标

MAPE 指平均绝对百分比误差，它是一种相对度量，它实际上将 MAD 尺度确定为百分比单位而不是变量的单位。平均绝对百分比误差是相对误差度量值，它使用绝对值来避免正误差和负误差相互抵消，您可以使用相对误差来比较各种时间序列模型预测的准确性。数学公式：



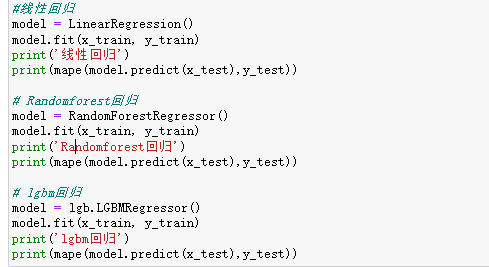
Python代码：



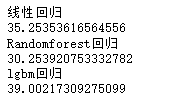
③各模型评价结果

用已划分的训练集对各个模型进行训练，并用测试集进行预测，最后用MAPE对模型进行评价发现Randomforest回归模型最优。

Python代码：



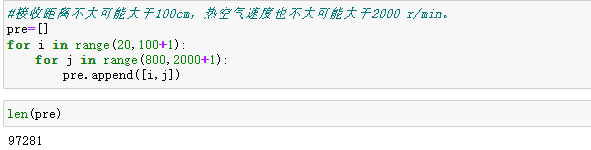
结果：

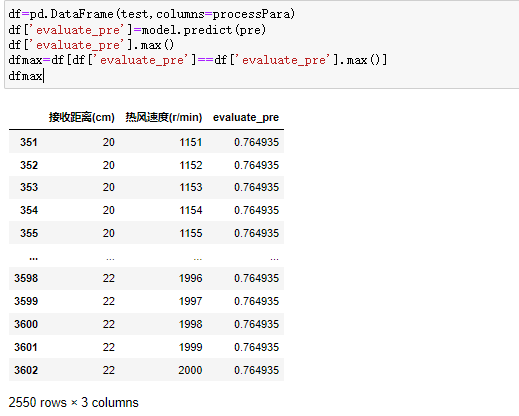


④工艺参数确定

题目中表示接收距离不大可能大于100cm，热空气速度也不大可能大于2000 r/min，所以我们可以认为接收距离的范围就是[20,100],热空气速度范围为[800,2000]，所以可以通过该范围生成预测集，然后对预测集进行预测，并且将评价指标为最大值的所有预测结果筛选出来，最后取筛选结果工艺参数的平均值为结果，我们认为该工艺参数为该结果时，能够使得过滤效率尽量的高的同时力求过滤阻力尽量的小。

Python代码：







结果：

当接收距离为21cm，热风速度为1575.50r/min时，我们认为能够使得过滤效率尽量的高的同时力求过滤阻力尽量的小。

六、模型的评价、改进与推广

6.1模型的优势

（1）我们建立的预测回归模型：LightGBM可以不需要做one\_hot转换，有效避免了原始数据不是**连续变量对预测结果产生的影响；**随机森林可以提供决策树的不同解释，并且具有更好的性能；线性回归建模速度快，不需要很复杂的计算，可以根据系数给出每个变量的理解和解释。

（2）灰色关联分析正好规避了对样本量少、原始数据分布不规律的问题，有效分析出可靠结果。

（3）使用Spearman相关系数探究各项指标的对应关系，可以避免样本数据非正态分布时不能使用Pearson相关系数的问题。

6.2模型的推广

本文所给出的基于LinearRegression、LightGBM、随机森林这三个回归方程模型可以应用于研究吸音降噪、保温隔热、医疗防护等领域。

七、参考文献

[1]刘燕慧,陈龙敏,陈喆,刘建立.熔喷非织造材料厚度对过滤性能的影响[J].产业用纺织品,2012,30(11)

[2]杨静. 熔喷滤料水驻极技术及产品性能研究[D]. 东华大学.

[3]Light GBM:A Highly Efficient Gradient Boosting Decision Tree.KEGL,MENGQ,FINLEY T,et al.Advances in Neural Information Processing Systems 30.2017

[4]刘翔,谢涛,王训洪,易泽邦,何秋芝.基于改进随机森林算法的岩石爆破块度预测[J].矿业研究与开发,2022,42(07):25-29.DOI:10.13827/j.cnki.kyyk.2022.07.026.

1. 附录

# 导入的包

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import r2\_score

import xgboost as xgb

import lightgbm as lgb

from sklearn.ensemble import RandomForestRegressor

from sklearn.linear\_model import LinearRegression

import numpy as np

一、第一题

1.数据预处理（异常值处理、缺失值处理、空值处理）jupyter notebook

data1=pd.read\_excel('C题数据.xlsx',sheet\_name='datat1')

data1

#1#是熔喷样品即未插层，2#为插层样品

data1.drop( index=50,inplace=True)

data1['编号']=data1['编号'].map({

'1#':'未插层样品',

'2#':'插层样品'

})

data1['组号']=data1['组号'].fillna(method='ffill')

data1

data2=pd.read\_excel('C题数据.xlsx',sheet\_name='data2')

data2

del data2['Unnamed: 1']

data2.dropna(how='any',inplace=True)

data2

mapping1={}

mapping2={}

for i in range(5):

for j in range(5):

mapping1[data2.values[i,j+1]]=data2.values[i,0]

for i in range(5):

for j in range(5):

mapping2[data2.values[i,j+1]]=data2.columns[j+1]

print(mapping1)

print(mapping2)

data1[r'接收距离']=data1['组号'].map(mapping1)

data1[r'热风速度(r/min)']=data1['组号'].map(mapping2)

data1.head()

#工艺参数

processPara=['接收距离', '热风速度(r/min)']

#结构变量

StructuralVar=['厚度mm', '孔隙率（%）', '压缩回弹性率（%）']

#产品性能

ProductPerfor=['过滤阻力Pa', '过滤效率（%）', '透气性 mm/s']

import matplotlib.pyplot as plt

# 支持中文

plt.rcParams['font.sans-serif'] = ['SimHei'] # 用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False # 用来正常显示负号

#取消科学计数法

import pandas as pd

import numpy as np

pd.set\_option("display.float\_format", lambda x: "%.2f" % x) #为了直观的显示数字，不采用科学计数法

np.set\_printoptions(suppress=True)

import matplotlib.pyplot as plt

%matplotlib inline

for i in StructuralVar:

plt.plot(data1[data1['编号']=='插层样品'][i],label='插层样品')

plt.plot(data1[data1['编号']=='未插层样品'][i],label='未插层样品')

plt.title(i)

plt.legend()

plt.savefig('./jpg/%s.jpg'%(i[:4]))

plt.show()

for i in ProductPerfor:

plt.plot(data1[data1['编号']=='插层样品'][i],label='插层样品')

plt.plot(data1[data1['编号']=='未插层样品'][i],label='未插层样品')

plt.title(i)

plt.legend()

plt.savefig('./jpg/%s.jpg'%(i[:4]))

plt.show()

for i in StructuralVar:

plt.plot(data1[data1['编号']=='插层样品'][i],label=i)

plt.title('插层后结构变量')

plt.legend()

plt.savefig('./jpg/插层后结构变量.jpg')

plt.show()

for i in ProductPerfor:

plt.plot(data1[data1['编号']=='插层样品'][i],label=i)

plt.title('插层后产品性能')

plt.legend()

plt.savefig('./jpg/插层后产品性能.jpg')

plt.show()

import numpy as np

from sklearn.decomposition import PCA

X = data1[StructuralVar]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

StructuralVar\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

X = data1[ProductPerfor]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

ProductPerfor\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

data1['结构变量']=StructuralVar\_newX

data1['产品性能']=ProductPerfor\_newX

plt.plot(data1[data1['编号']=='插层样品']['结构变量'],label='插层样品')

plt.plot(data1[data1['编号']=='未插层样品']['结构变量'],label='未插层样品')

plt.title('结构变量插层前后对比')

plt.legend()

plt.savefig('./jpg/结构变量插层前后对比.jpg')

plt.show()

plt.plot(data1[data1['编号']=='插层样品']['产品性能'],label='插层样品')

plt.plot(data1[data1['编号']=='未插层样品']['产品性能'],label='未插层样品')

plt.title('产品性能插层前后对比')

plt.legend()

plt.savefig('./jpg/产品性能插层前后对比.jpg')

plt.show()

data1.to\_excel('data1.xlsx',index=None)

from spsspro.algorithm import quantify\_analysis

data1['插层率（%）\_填补']=data1['插层率（%）'].fillna(0)

data1.columns

#生成案例数据

f = data1[StructuralVar+ProductPerfor]

m = data1['插层率（%）\_填补']

i = data1['组号']

#灰色关联分析，输入参数详细可以光标放置函数括号内按shift+tab查看，输出结果参考spsspro模板分析报告

print(quantify\_analysis.grey\_relational\_analysis(f, m, i))

第二题

#第二题改用data3的数据

data3=pd.read\_excel('C题数据.xlsx',sheet\_name='data3')

data3

#工艺参数

processPara=['接收距离(cm)', '热风速度(r/min)']

#结构变量

StructuralVar=['厚度mm', '孔隙率（%）', '压缩回弹性率（%）']

#产品性能

ProductPerfor=['过滤阻力Pa', '过滤效率（%）', '透气性 mm/s']

X = data3[StructuralVar]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

StructuralVar\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

X = data3[processPara]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

processPara\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

plt.plot(StructuralVar\_newX,label='结构变量')

plt.plot(ProductPerfor\_newX,label='工艺参数')

plt.title('结构变量&工艺参数对比')

plt.legend()

plt.savefig('./jpg/结构变量&工艺参数对比.jpg')

plt.show()

for i in StructuralVar:

plt.plot(data3[i ],label=i)

for i in ProductPerfor:

plt.plot(data3[i],label=i)

plt.title('结构变量&工艺参数对比2')

plt.legend()

plt.savefig('./jpg/结构变量&工艺参数对比2.jpg')

plt.show()

for i in StructuralVar:

plt.plot(data3[i ],label=i)

plt.title('工艺参数')

plt.legend()

plt.savefig('./jpg/工艺参数.jpg')

plt.show()

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import r2\_score

import xgboost as xgb

import lightgbm as lgb

from sklearn.ensemble import RandomForestRegressor

from sklearn.linear\_model import LinearRegression

import warnings

warnings.filterwarnings('ignore')

import numpy as np

def mape(actual, pred):

actual, pred = np.array(actual), np.array(pred)

return np.mean(np.abs((actual - pred) / actual)) \* 100

X=data3[processPara]

y=data3['厚度mm']

# 拆分训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=12)

#线性回归

model = LinearRegression()

model.fit(x\_train, y\_train)

print('线性回归')

print(mape(model.predict(x\_test),y\_test))

# Randomforest回归

model = RandomForestRegressor()

model.fit(x\_train, y\_train)

print('Randomforest回归')

print(mape(model.predict(x\_test),y\_test))

# lgbm回归

model = lgb.LGBMRegressor()

model.fit(x\_train, y\_train)

print('lgbm回归')

print(mape(model.predict(x\_test),y\_test))

model = RandomForestRegressor()

model.fit(X, y)

test=np.array([[38,850],

[33,950],

[28,1150],

[23,1250],

[38,1250],

[33,1150],

[28,950],

[23,850]])

model.predict(test)

y=data3['孔隙率（%）']

# 拆分训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=12)

#线性回归

model = LinearRegression()

model.fit(x\_train, y\_train)

print('线性回归')

print(mape(model.predict(x\_test),y\_test))

# Randomforest回归

model = RandomForestRegressor()

model.fit(x\_train, y\_train)

print('Randomforest回归')

print(mape(model.predict(x\_test),y\_test))

# lgbm回归

model = lgb.LGBMRegressor()

model.fit(x\_train, y\_train)

print('lgbm回归')

print(mape(model.predict(x\_test),y\_test))

model = RandomForestRegressor()

model.fit(X, y)

test=np.array([[38,850],

[33,950],

[28,1150],

[23,1250],

[38,1250],

[33,1150],

[28,950],

[23,850]])

model.predict(test)

y=data3['压缩回弹性率（%）']

# 拆分训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=12)

#线性回归

model = LinearRegression()

model.fit(x\_train, y\_train)

print('线性回归')

print(mape(model.predict(x\_test),y\_test))

# Randomforest回归

model = RandomForestRegressor()

model.fit(x\_train, y\_train)

print('Randomforest回归')

print(mape(model.predict(x\_test),y\_test))

# lgbm回归

model = lgb.LGBMRegressor()

model.fit(x\_train, y\_train)

print('lgbm回归')

print(mape(model.predict(x\_test),y\_test))

model = RandomForestRegressor()

model.fit(X, y)

test=np.array([[38,850],

[33,950],

[28,1150],

[23,1250],

[38,1250],

[33,1150],

[28,950],

[23,850]])

model.predict(test)

第三题

X = data3[StructuralVar]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

StructuralVar\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

X = data3[ProductPerfor]

pca = PCA(n\_components=1) #降到1维

pca.fit(X) #训练

ProductPerfor\_newX=pca.fit\_transform(X) #降维后的数据

# PCA(copy=True, n\_components=2, whiten=False)

print(pca.explained\_variance\_ratio\_) #输出贡献率

plt.plot(StructuralVar\_newX,label='结构变量')

plt.plot(ProductPerfor\_newX,label='产品性能')

plt.title('结构变量&产品性能对比')

plt.legend()

plt.savefig('./jpg/结构变量&产品性能对比.jpg')

plt.show()

for i in StructuralVar:

plt.plot(data3[i ],label=i)

for i in ProductPerfor:

plt.plot(data3[i],label=i)

plt.title('结构变量&产品性能对比2')

plt.legend()

plt.savefig('./jpg/结构变量&产品性能对比2.jpg')

plt.show()

for i in StructuralVar:

plt.plot(data3[i ],label=i)

plt.title('结构变量')

plt.legend()

plt.savefig('./jpg/结构变量.jpg')

plt.show()

for i in ProductPerfor:

plt.plot(data3[i],label=i)

plt.title('产品性能')

plt.legend()

plt.savefig('./jpg/产品性能.jpg')

plt.show()

data3.to\_excel('data3.xlsx',index=None)

#工艺参数

processPara=['接收距离(cm)', '热风速度(r/min)']

#结构变量

StructuralVar=['厚度mm', '孔隙率（%）', '压缩回弹性率（%）']

#产品性能

ProductPerfor=['过滤阻力Pa', '过滤效率（%）', '透气性 mm/s']

X=data3[processPara]

y=data3['过滤效率（%）']

# 拆分训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=12)

#线性回归

model = LinearRegression()

model.fit(x\_train, y\_train)

print('线性回归')

print(mape(model.predict(x\_test),y\_test))

# Randomforest回归

model = RandomForestRegressor()

model.fit(x\_train, y\_train)

print('Randomforest回归')

print(mape(model.predict(x\_test),y\_test))

# lgbm回归

model = lgb.LGBMRegressor()

model.fit(x\_train, y\_train)

print('lgbm回归')

print(mape(model.predict(x\_test),y\_test))

X.describe()

print(list(range(20,40+1,1)))

print(list(range(800,1200+1,5)))

test=[]

for i in range(20,40+1,1):

for j in range(800,1200+1,5):

test.append([i,j])

model = RandomForestRegressor()

model.fit(X, y)

model.predict(test)

df=pd.DataFrame(test,columns=processPara)

df['过滤效率（%）']=model.predict(test)

df.head()

df['过滤效率（%）'].max()

temp=df[df['过滤效率（%）']==df['过滤效率（%）'].max()]

temp

temp.describe()

第四题

data3['过滤阻力\_负向']=(data3['过滤阻力Pa'].max()-data3['过滤阻力Pa'])/(data3['过滤阻力Pa'].max()-data3['过滤阻力Pa'].min())

data3['过滤效率\_正向']=(data3['过滤效率（%）']-data3['过滤效率（%）'].min())/(data3['过滤效率（%）'].max()-data3['过滤效率（%）'].min())

data3.head()

data3['evaluate']=0.5\*data3['过滤阻力\_负向']+0.5\*data3['过滤效率\_正向']

#data3['evaluate']=data3['过滤阻力\_负向']\*data3['过滤效率\_正向']

data3.describe()

#接收距离不大可能大于100cm，热空气速度也不大可能大于2000 r/min。按照应用的要求，厚度尽量不要超过3mm，压缩回弹性率尽量不要低于85%

dftemp=data3[(data3['接收距离(cm)']<=100) &(data3['热风速度(r/min)']<=2000)&(data3['厚度mm']<=3)&(data3['热风速度(r/min)']>=85)]

dftemp.shape

X=dftemp[processPara]

y=dftemp['evaluate']

# 拆分训练集和测试集

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=12)

#线性回归

model = LinearRegression()

model.fit(x\_train, y\_train)

print('线性回归')

print(mape(model.predict(x\_test),y\_test))

# Randomforest回归

model = RandomForestRegressor()

model.fit(x\_train, y\_train)

print('Randomforest回归')

print(mape(model.predict(x\_test),y\_test))

# lgbm回归

model = lgb.LGBMRegressor()

model.fit(x\_train, y\_train)

print('lgbm回归')

print(mape(model.predict(x\_test),y\_test))

#接收距离不大可能大于100cm，热空气速度也不大可能大于2000 r/min。

test=[]

for i in range(20,100+1,2):

for j in range(800,2000+1,10):

test.append([i,j])

len(test)

model = RandomForestRegressor()

model.fit(X, y)

model.predict(test)

df=pd.DataFrame(test,columns=processPara)

df['evaluate\_pre']=model.predict(test)

df.head()

df['evaluate\_pre'].max()

temp=df[df['evaluate\_pre']==df['evaluate\_pre'].max()]

temp

temp.describe()