基于评价与回归的古代玻璃制品成分与抗风化能力分析

（古代玻璃制品的成分分析与鉴别）

摘 要

本文采用多种相关分析、综合评价和分类方法解决了古代玻璃制品的成分分析与鉴别问题。并在此基础上，基于灰色关联模型与Topsis综合评价模型进一步论述了古代玻璃抗风化能力与其化学成分之间的关系。

问题一中，根据对数据初步分析以及化学成分分析，发现文物表面风化与玻璃类型有相关性，铅钡玻璃比高钾玻璃更易风化，而风化后部分氧化物成分会发生显著变化，基于此，我们依次构建了显著变化的六个化学成分的含量与样本颜色、纹饰、种类之间的回归模型，以无风化数据作为训练集，去预测风化样本未风化之前的化学成分含量，得到了较好的结果。

问题二中，首先建立玻璃类型与化学成分之间的线性回归模型，通过回归系数来解释玻璃种类的分类规律。而对于亚类的划分，我们综合了分类结果的敏感性和合理性问题，引入了“抗风化能力”这一概念，并构建了基于Topsis的古代玻璃抗风化能力的评价体系，将其与Kmeans++得到的聚类结果进行比对，发现聚类结果与Topsis得分高低有着极高的匹配度，可在玻璃种类大类下划分出抗风化能力强、抗风化能力中、抗风化能力弱三种亚类，增强了结果的合理性。

问题三中，我们首先基于表单2数据构建了玻璃种类与化学成分含量之间的回归模型，通过对回归系数的分析得到对玻璃种类的划分影响程度较大的若干种化学成分。然后基于决策树模型构建了这些成分与玻璃种类的分类模型，然后对决策树模型的最大深度、最小叶子节点数和最小训练样本分枝数等超参数进行敏感性分析，对模型进行调优，最终得到一个稳定的分类结果。

问题四中，我们认为化学成分之间的关系与玻璃本身的抗风化能力有关，基于灰色关联分析方法，同时结合第二问中构建的综合评价方法以分析同种玻璃化学成分之间的关联关系以及不同种玻璃化学成分关联关系的差异性。最终得到了不同化学成分对于同种玻璃抗风化能力强弱的影响程度占比，以及不同种玻璃之间化学成分差异性导致的抗风化能力差异性。

关键词：多元线性回归 K means聚类 决策树 灰色关联

# 问题重述

## 问题背景

玻璃的主要化学成分是二氧化硅（SiO2），由于添加助熔剂的不同，玻璃的化学成分含量存在一定差异。比如，铅钡玻璃以铅矿石作为助熔剂，其氧化铅（PbO）、氧化钡（BaO）的含量较高，而钾玻璃以草木灰作为助熔剂，其含钾量高。

古代玻璃风化后，其化学成分比例发生变化，影响玻璃颜色、形状等特性，从而影响对其类别的正确判断。玻璃表面大面积灰黄色区域为风化层，是明显风化区域，紫色部分是一般风化表面。在部分风化的文物中，其表面也有未风化的区域。

现有一批我国古代玻璃制品的相关数据，其中表单1中含有玻璃的分类信息，表单 2 含有各个样本的化学成分含量信息（空白处表示未检测到该成分），表单3含有一批未分类玻璃文物的化学成分信息。这些数据具有累加和为100%的特性，但因检测手段等原因可能导致其成分比例的累加和非100%的情况。本题中将成分比例累加和介于 85%~105%之间的数据视为有效数据。

## 问题提出

根据题目背景与附件数据需要解决以下问题：

（1）结合表单1数据，分析玻璃文物是否风化与其本身类型、纹饰和颜色之间的关系；对于不同种类的玻璃，分析表面是否风化对于其内部化学成分含量的影响；根据表单2中的检测数据，预测风化玻璃风化前化学成分含量的值。

（2）结合附件数据，分析玻璃的分类主要受到哪些化学成分或者玻璃本身特征的影响；针对不同的玻璃种类，选择合适的化学成分进行亚类划分，给出具体的划分方法以及划分结果，并分析结果的合理性和敏感性。

（3）分析表单3中未知类别玻璃文物的化学成分，进行类型鉴定，并分析结果的敏感性。

（4）对于不同种类的玻璃，分析每种玻璃内部化学成分之间的联系，并且比较不同种类玻璃内部化学成分之间联系的差异性。

# 问题分析

## 问题一的分析

针对问题一,目前已知玻璃类型,纹饰,颜色与是否风化和检测点各氧化物数据，题目要求对玻璃文物各特征和表面是否风化、风化成分进行分析，并且能根据此关系来预测风化前的成分含量。

结合题目相关条件进行斯皮尔曼相关分析，建立风化玻璃与各化学成分所占比例的多元线性回归模型，分析玻璃是否风化与各氧化物比例的显著相关性，得出氧化物因风化导致的含量变化规律。

针对风化前的化学成分含量预测，首先对比同一文物有无风化点的各氧化物含量，得出风化后显著含量变化的氧化物种类， 以未风化数据为训练集，进行线性与非线性回归，输出显著变化氧化物的含量，最后整合预测出文物风化前的化学成分含量。

## 问题二的分析

通过对表单1数据的初步分析，不难发现用于玻璃种类分析的最有效信息为未风化时的颜色以及纹饰信息。而当玻璃风化后，外在信息不易被观察，所以可以通过玻璃内部化学成分含量进行分析，对于此我们决定构建玻璃14个化学成分和3个外在特征与玻璃种类的多元线性回归模型，通过回归系数，可以分析玻璃的分类规律。

而对于亚类的划分，我们首先考虑分类结果的合理性，观察到随着风化程度的不同玻璃表面的颜色呈现一种递进的变化。所以推测，不同的玻璃有着不同的风化程度，这代表着不同的抗风化能力，我们通过观察同一个样本不同风化点的数据发现，部分化学成分含量与风化程度显著相关。所以，我们选取这些风化前后发生显著变化的化学成分，进行Topsis分析，同时，针对不同大类的玻璃我们基于Kmeans++算法进行聚类。最后，将聚类结果与Topsis得分进行对比，观察两者之间是否具有一定的联系。

## 问题三的分析

观察表单3的数据，这里除了化学成分特征外，还包含了表面是否风化的特征。这里我们结合第一题多元线性回归分析的统计规律结果，从中挑选出对于玻璃种类划分影响程度较大的几个变量。然后构建这些变量与玻璃种类之间的决策树模型，并把表单2数据作为训练集，通过对决策树模型的超参数进行敏感性分析确定最优参数，最终将表单3的特征输入决策树模型，得到预测结果。

## 问题四的分析

结合前三题的分析我们知道，对于同一种类的玻璃文物样品，其化学成分含量与其表面是否风化有一定程度的联系。而在第二题中，我们引入了“抗风化能力”这一概念，所以我们决定通过分析化学成分对于玻璃文物样品抗风化能力的贡献程度，来进一步分析同种玻璃文物样品化学成分之间的关联性与不同种玻璃文物样品化学成分关联的差异性。

# 模型假设

1. 假设文物采样点未检测出化学成分的含量为0。
2. 假设玻璃文物各特征与化学成分真实准确，无错漏情况。

# 符号说明

|  |  |  |
| --- | --- | --- |
| **符号** | **说明** | **单位** |
| N | 样本数量 |  |
|  | 斯皮尔曼相关系数 |  |
|  | 多元线性回归系数 |  |
|  | 评价指标 |  |
|  | 归一化得分 |  |
| J | 总畸变程度 |  |
| R-squared | 拟合度 |  |
| a | 两极最小差 |  |
| b | 两极最大差 |  |

# 模型的建立与求解

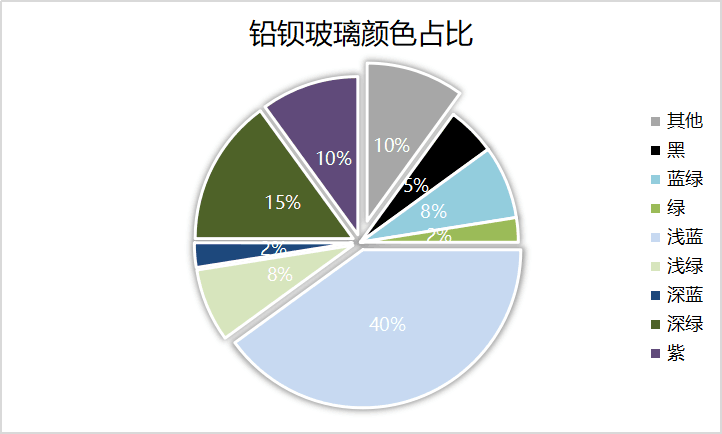
## 问题一模型的建立与求解

### 数据预处理

1. 数据缺失

附件中表单1部分文物颜色缺失，故在分析其他特征与颜色相关性时将19，40，48，58归类为其他颜色，经分析后发现这4种均属于铅钡玻璃，而铅钡玻璃中浅蓝色占比约为40.00%，第二高的深绿色占比为15.00%，考虑到分析玻璃类型与化学含量的统计关系时样本不宜过少，故在对化学含量进行分析时将其归类为浅蓝色。

表单2与表单3的部分氧化物化学含量为空，根据题意空白处表示未检测到该成分，故将空白部分的含量全部置为0。

图1 铅钡玻璃颜色占比

1. 无效数据删除

题目要求成分比例累加和介于 85%~105%之间的数据视为有效数据，根据计算采样点15和17不符合要求，故在后续建模中删去。

表1 无效数据

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **文物采样点** | **二氧化硅(SiO2)** | **氧化钠(Na2O)** | **氧化钾(K2O)** | **氧化镁(MgO)** | **氧化铝(Al2O3)** | **氧化铁(Fe2O3)** | **氧化铜(CuO)** | **氧化铅(PbO)** | **五氧化二磷(P2O5)** | **合计** |
| 15 | 61.87 | 3.21 | 7.44 | 1.02 | 3.15 | 1.04 | 1.29 | 0.19 | 0.26 | 79.5 |
| 17 | 60.71 | 2.12 | 5.71 | 0.85 |  | 1.04 | 1.09 | 0.19 | 0.18 | 71.9 |

### 表面分化与玻璃特征关系的确定

基于附件表单1中文物表面分化和玻璃类型，纹饰，颜色的数据，首先对不同类型的玻璃分化程度进行对比，发现铅钡类型比高钾类型的玻璃更容易风化，推测铅矿石作为助熔剂的玻璃更易风化，且类型与是否风化有相关性。

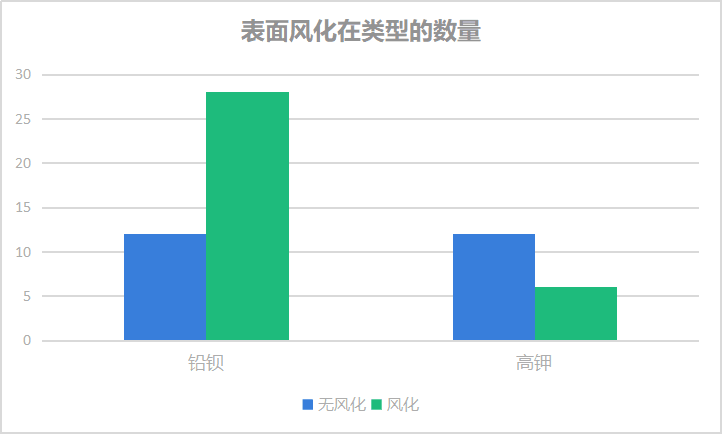


图2 表面风化在类型的数量

基于风化前后玻璃颜色的对比发现重合部分颜色变化较小，有黑色出现推测是因风化导致氧化铜含量上升。为了进一步探究风化与颜色，纹饰的关系，对其进行spearman相关性分析。

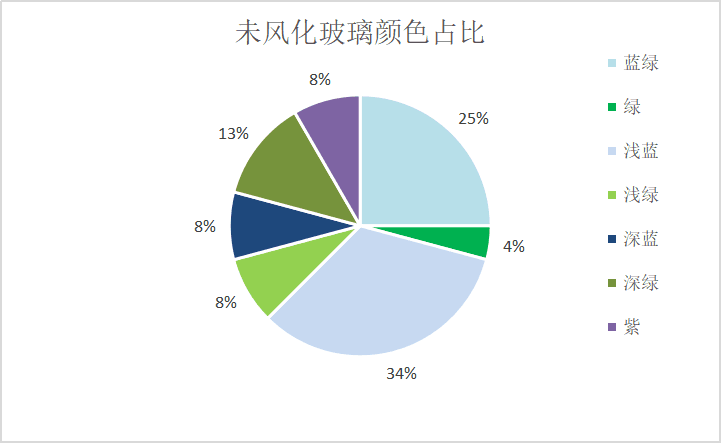
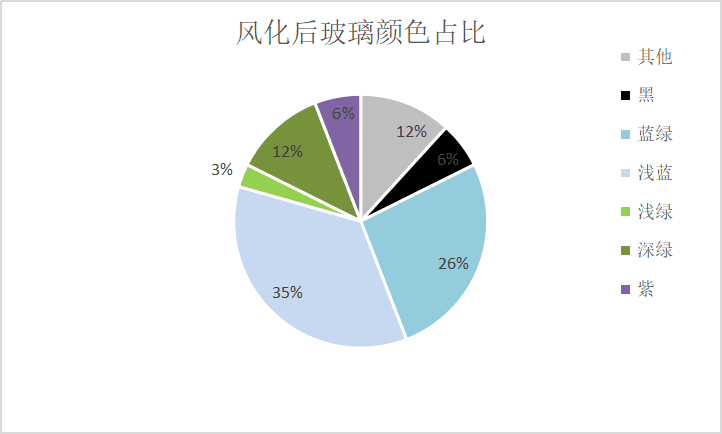
 

图3风化前后玻璃颜色占比

通过对表单1的分析， 对颜色、纹饰、类别和表面分化两两变量求出spearman相关系数，并通过假设检验计算检验值，由此得到变量之间的显著性，相关系数r计算公式如下：

(Ⅰ)

其中，, 和分别是两个变量按大小排序的秩，是样本的容量。之后采取假设检验：

原假设 ，两变量之间不存在线性关联

备择假设 ，两变量之间存在线性关联。

(Ⅱ)

\*\* 当p在 0.01 级别，相关性显著 \* 在 0.05 级别，相关性显著。

由表2可知，文物表面风化与玻璃类型有相关性，而颜色与纹饰和表面风化则无相关性，同时颜色与类型、纹饰与颜色同样也有较强的相关性。

本文根据这一结果的解释为：部分金属具有颜色，相同类型的玻璃化学成分相似，最后形成的颜色也相似；同一年代所偏好的纹饰与助熔剂大体相同，若用草木灰、硝石助熔得到高钾玻璃，铅矿石助熔则得到铅钡玻璃，高钾玻璃与铅钡玻璃有不同的颜色，纹饰也随着助熔剂与颜色相关。

表2 斯皮尔曼相关系数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 相关系数 | 纹饰 | 类型 | 颜色 | 表面风化 |
| 纹饰 | 1 | 0.107 | .449\*\* | 0.025 |
| p1 | . | 0.428 | 0 | 0.852 |
| 类型 | 0.107 | 1 | .341\*\* | .338\* |
| p2 | 0.428 | . | 0.009 | 0.01 |
| 颜色 | .449\*\* | .341\*\* | 1 | -0.205 |
| p3 | 0 | 0.009 | . | 0.125 |
| 表面风化 | 0.025 | .338\* | -0.205 | 1 |
| p4 | 0.852 | 0.01 | 0.125 | . |

### 基于多元线性回归的风化化学成分模型

根据spearman相关性分析的结果，建立风化玻璃与各化学成分所占比例的多元线性回归模型，分析玻璃是否风化与各氧化物比例的显著相关性，得出化学成分的统计规律，模型如下。

Step1.建立多元线性回归模型，涉及m个自变量的多元线性回归模型可表示为

(Ⅲ)

我们通过n组实际数据而引入矩阵记号：

(Ⅳ)

其中成为模型设计矩阵，与是随机向量，是回归系数构成的常数向量，未知待定，且

（为阶单位阵）

Step2.回归系数的最小二乘估计

选取的一个估计值，计为，使随机误差的平方和达到最小，即

（Ⅴ）

由最小二乘法的要求，由多元函数取得极值的必要条件可求解回归参数的标准方程如下：

（Ⅵ）

虽然当X不满秩时，解不唯一，但对任意一组解都能使残差平方和最小，当X秩时，正规方程组的解为，即回归系数的估计值。

Step3.逐步回归分析

由于并非每一个因子都对y影响程度大，可以通过逐步回归的方法来对因子进行筛选,通过对回归系数显著性检验，取t值对应最大概率值,判断是否≤0.05，若满足则可接受，再次建立多元线性回归方程，反之拒绝输出方程。

将（Ⅲ）式中表示为方程的回归系数，表示二氧化硅，氧化钠等化学成分所占的比例。考虑到文物表面化学成分重复性较高，将显著性低于0.1的自变量视作与因变量关系密切，将标准化系数绝对值较大的视作对自变量影响较大的因变量，结合图2与表3可得如下结论：

氧化铅的成分含量对表面风化有较大的正相关影响，氧化钾对表面风化有较大的负相关影响，其中氧化铁，氧化铜，氧化硫的含量与样品表面是否风化相关性较高，二氧化硅无论是否风化，都是文物样品表面主要成分。结合玻璃类型，则认为高钾玻璃不易风化，氧化钾，氧化铁含量较高，铅钡玻璃易风化，氧化铅，氧化铜含量较高。

表3 回归系数

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 模型 | 未标准化系数 | | 标准化系数 | t | 显著性 |
| B | 标准错误 | Beta |
| (常量) | 0.297 | 2.655 | 0 | 0.112 | 0.911 |
| 二氧化硅 | 0.008 | 0.027 | 0.396 | 0.296 | 0.768 |
| 氧化钠 | 0.052 | 0.043 | 0.177 | 1.205 | 0.234 |
| 氧化钾 | 0.061 | 0.037 | -0.488 | 1.663 | 0.102 |
| 氧化钙 | 0.028 | 0.046 | 0.131 | 0.596 | 0.554 |
| 氧化镁 | 0.006 | 0.107 | -0.008 | 0.059 | 0.953 |
| 氧化铝 | 0.034 | 0.030 | 0.215 | 1.136 | 0.261 |
| 氧化铁 | 0.114 | 0.057 | -0.277 | 1.997 | 0.051 |
| 氧化铜 | 0.069 | 0.035 | 0.317 | 1.940 | 0.058 |
| 氧化铅 | 0.018 | 0.027 | 0.734 | 0.668 | 0.507 |
| 氧化钡 | 0.015 | 0.036 | -0.258 | 0.417 | 0.679 |
| 五氧化二磷 | 0.040 | 0.039 | 0.288 | 1.012 | 0.316 |
| 氧化锶 | 0.413 | 0.261 | -0.227 | 1.584 | 0.119 |
| 氧化锡 | 0.071 | 0.159 | 0.049 | 0.448 | 0.656 |
| 二氧化硫 | 0.048 | 0.029 | 0.268 | 1.688 | 0.097 |

根据上述结论和风化前后各氧化物比例对比，可知玻璃风化前后化学含量主要变化的有二氧化硅，氧化钾，氧化铁，氧化铜，氧化铅，氧化锶，二氧化硫，故建立未风化玻璃类型、颜色、纹饰为自变量，样本表面氧化物比例为因变量的逐步多元线性回归方程。

### 模型的求解

首先，利用MATLAB软件将所有未分化样本进行输入建立，得到了文物表面各特征对氧化物比例的线性回归方程。

将（Ⅲ）式中自变量表示类别\_铅钡  类别\_高钾  纹饰\_A  纹饰\_C  颜色\_浅绿  颜色\_浅蓝  颜色\_深绿  颜色\_深蓝  颜色\_紫  颜色\_蓝绿，为各氧化物含量进行多元线性回归，模型的显著性检验为

表4 二氧化硅(SiO2)回归结果

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | 二氧化硅(SiO2) | R-squared : | 0.731 |
| Model: | GLS | Adj. R-squared : | 0.424 |
| Method: | Least Squares | F-statistic: | 2.38 |
| Date: | Sun, 18 Sep 2022 | Prob (F-statistic): | 0.135 |

得到的预测结果为

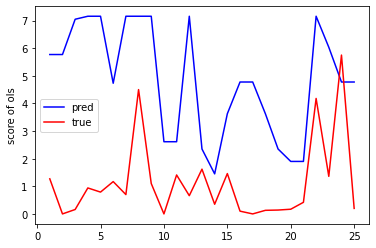
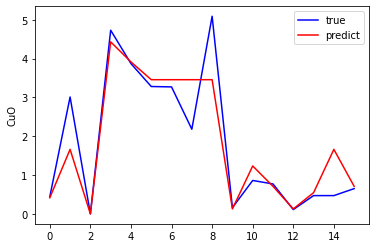
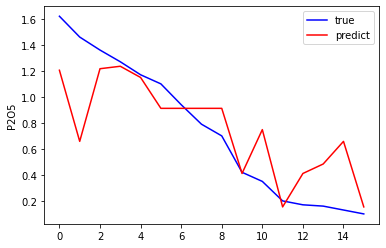
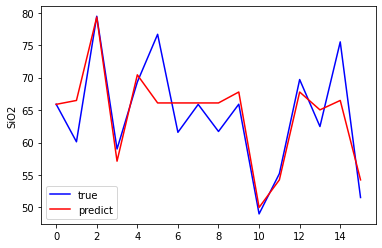


图4 线性回归模型化学含量预测比对图

由图可知，用线性回归预测的值并不理想，对回归方程进行改进；本次改进使用

多层感知机（MLP，Multilayer Perceptron），除了输入输出层，中间可以有多个隐层，本次开始只是使用最简单的MLP，只含一个隐层，即三层的结构，对主要化学成分分化前的含量进行预测，有图可得二氧化硅，氧化铜，氧化钙，氧化钡，三氧化二铝吧预测效果均较为准确。



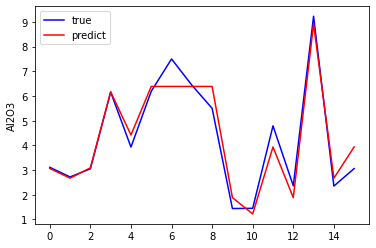
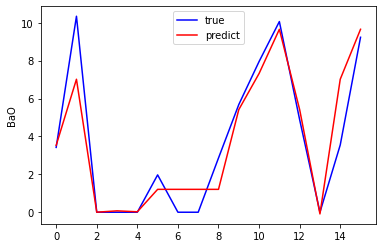
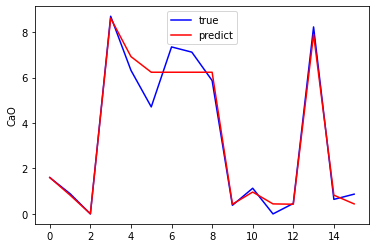


图5 基于MLP模型主要化学成分的预测拟合图

## 问题二模型的建立与求解

### 基于多元线性回归的玻璃分类模型

基于对数据的分析，本文认为附录2提供的数据表明化学成分对玻璃分类有关，并建立多元回归模型：

在（Ⅲ）式中表示二氧化硅，氧化钠等化学成分所占的比例， 表示第i个样本为铅钡玻璃，为高钾玻璃，利用 matlab 统计工具箱建立多元线性回归方程，对结果统计量进行分析得到R-squared为0.927， 表明预测值和实际值接近92.7％，回归方程显著。

由t分布的p值可知，除氧化锶以外，其他13种化学成分均与类别有显著相关性，根据回归系数绝对值大小的升序排列，可知氧化钾对玻璃类型的分类的决定作用最大；同时其标准误差偏小，结果更具解释性。同时在对回归系数和标准误差的综合考量下，氧化钡也对玻璃分类起重要作用。

表5 回归系数

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **coef** | **std err** | **t** | **P>|t|** | **[0.025** | **0.975]** |
| 二氧化硅**(SiO2)** | 0.0086 | 0.001 | 12.207 | 0.000 | 0.007 | 0.010 |
| 氧化钠**(Na20)** | -0.0386 | 0.014 | -2.775 | 0.008 | -0.067 | -0.010 |
| 氧化钾**(K20)** | 0.0425 | 0.011 | 3.975 | 0.000 | 0.021 | 0.064 |
| 氧化钙**(CaO)** | 0.0115 | 0.019 | 0.618 | 0.540 | -0.026 | 0.049 |
| 氧化镁**(MgO)** | -0.0103 | 0.047 | -0.220 | 0.827 | -0.105 | 0.084 |
| 氧化铝**(AI2O3)** | -0.0324 | 0.009 | -3.507 | 0.001 | -0.051 | -0.014 |
| 氧化铁**(Fe2O3)** | 0.0276 | 0.026 | 1.066 | 0.293 | -0.025 | 0.080 |
| 氧化铜**(CuO)** | 0.0354 | 0.015 | 2.423 | 0.020 | 0.006 | 0.065 |
| 氧化铅**(PbO)** | -0.0029 | 0.002 | -1.635 | 0.110 | -0.007 | 0.001 |
| 氧化锲**(BaO)** | -0.0174 | 0.005 | -3.232 | 0.002 | -0.028 | -0.007 |
| 五氧化二磷**(P2O5)** | -0.0008 | 0.008 | -0.098 | 0.923 | -0.017 | 0.015 |
| 氧化锂**(SrO)** | 0.0812 | 0.110 | 0.739 | 0.464 | -0.141 | 0.303 |
| 氧化揚**(SnO2)** | 0.0268 | 0.067 | 0.399 | 0.692 | -0.109 | 0.162 |
| 二氧化硫**(S02)** | 0.0300 | 0.012 | 2.574 | 0.014 | 0.006 | 0.054 |

### 基于keans++聚类玻璃亚分类模型

先将玻璃分成高钾和铅钡两大类，基于对数据的分析，对于同一样本，风化点与未风化的化学成分具有一定的变化趋势，我们观察多组这样的数据发现，部分化学成分的变化趋势相同。本文推测随着风化程度的不同，样本的部分化学成分有着往一个趋势增加或者减少的特性。

将高钾玻璃的亚分类聚成两类，对比原数据的分化类型与聚类结果的一致性；将铅钡玻璃的亚分类聚成三类，对比原数据的分化类型与聚类结果的一致性。

对每类进行kmeans++聚类，具体算法流程如下图所示：

Step1.输入聚类数量k和迭代次数n;

Step2.初始化聚类中心,随机选取一个样本点作为第一个聚类中心，计算每个样本与当前已有聚类中心的最短距离，用轮盘法选出下一个聚类中心，直到聚类中心数量达到k。

Step3.计算其余的各个数据对象到这K个初始聚类中心的距离，把数据对象划归到距离它最近的那个中心所处在的簇类中。

Step4.调整新类并且重新计算出新类的中心;

Step5:Step3和Step4直到达到最大迭代次数n或者收敛时输出聚类结果。

聚类结果如附录所示；

根据化学成分与风化程度的相关性，将高钾玻璃划分为两类，以风化程度的高低作为划分依据；铅钡玻璃划分为三类，以风化程度强、中、弱为划分依据。

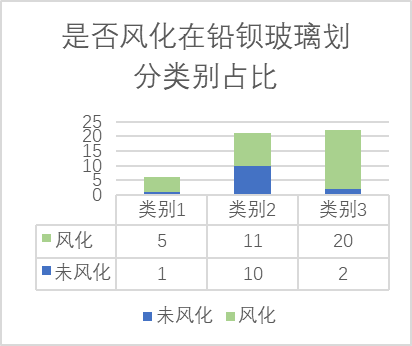
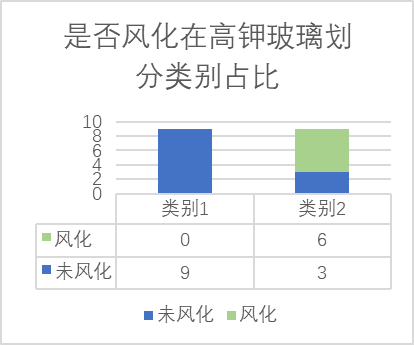


图6 是否分化在玻璃划分类别占比

由是否风化在高钾、铅钡玻璃中划分类别占比可得划分依据为风化程度，通过横向纵向交叉对比，按照所分类别，从类别1到类别2、类别3，风化样品占比逐步增加，所以本文所亚分类划分的依据为风化程度。

合理性分析：

通过猜想化学成分含量一定程度上反映玻璃的抗风化能力，本文使用Topsis模型对玻璃抗风化能力进行评估，由于化学成分类别过多，采用前文中使用的基于多元线性回归的玻璃分类模型中对回归系数和标误的综合分析，提取出影响玻璃分类最大的前五个指标。

这五个指标依次为二氧化硅,氧化钾,氧化钙,氧化铝,五氧化二磷，且他们分别为极大型、极小型、极小型、极小型、极小型指标。最终得分高低为玻璃抗风化能力的强弱，对比划分的亚类结果发现一致性，增强了模型的合理性。

### 指标计算

对不同化指标进行归一化处理，处理公式如下：

效益性指标： 成本性指标：

Step 1：将数据导入matlab，得到二氧化硅,氧化钾,氧化钙,氧化铝五氧化二磷共五个指标的权重；

Step2：对五个指标数据进行指标属性同向化处理并构造归一化初始矩阵

(Ⅶ)

Step 3：计算得分并进行归一化；

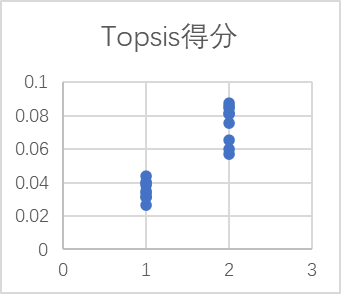
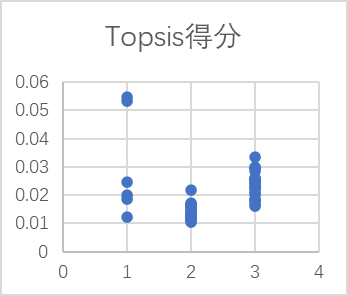
定义最大值：

定义最小值：

定义第 个评价对象与最大值的距离

定义第 个评价对象与最小值的距离   
那么, 我们可以计算得出第 个评价对象末归一化的得分: ，

通过对用Topsis得分对玻璃划分结果进行分析，在高钾玻璃中采用Topsis模型进行评价，得分越高，属于类别2的样本数量越集中；而在铅钡玻璃中，尽管未能发现得分的高低与类别有显著区分关系，但纵向观察，类别一致的样本Topsis得分呈现集中性。

高钾玻璃 铅钡玻璃

图7 topsis得分与玻璃亚类结果比对图

同时用系统聚类模型进行进一步验证，高钾玻璃的系谱图中，以重新标度的距离组合为15划分发现与前文所选用的初始K值相同，说明使用的keans聚类数的选取符合合理划分值,谱系图见附件1。

敏感性分析：

对于敏感性的分析使用总畸变程度。各个类的畸变程度等于该类重心与其内部成员位置距离的平方和；设一共将n个样本划分到K个类中（K≤n-1，即至少有一类中有两个元素）用表示第k个类（k=1，2，…，K），且该类重心的位置记为,那么第k个类的畸变程度为

定义所有类的总畸变程度

以横坐标为玻璃亚类类的类别数K，纵坐标为畸变总程度J。由图可直观感受到高钾玻璃在聚类类别数大于2时曲线趋于平缓，敏感度降低；铅钡玻璃在聚类类别数大于4时曲线趋于平缓，敏感度降低.

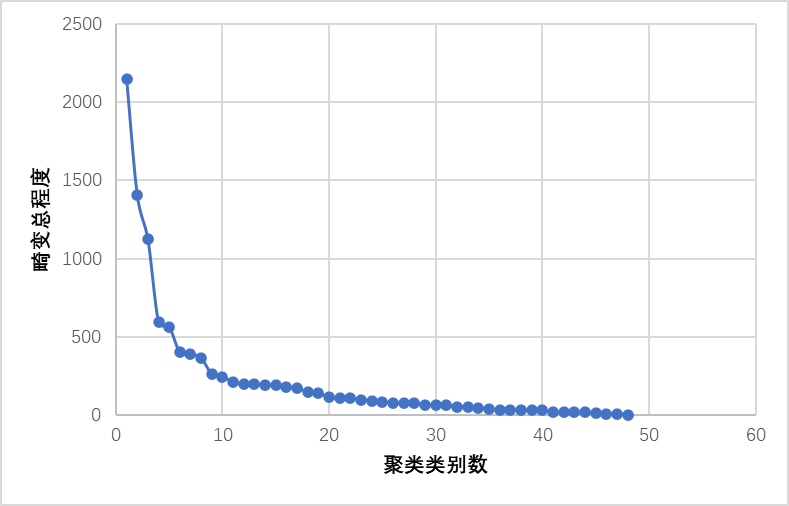


图8 铅钡敏感度

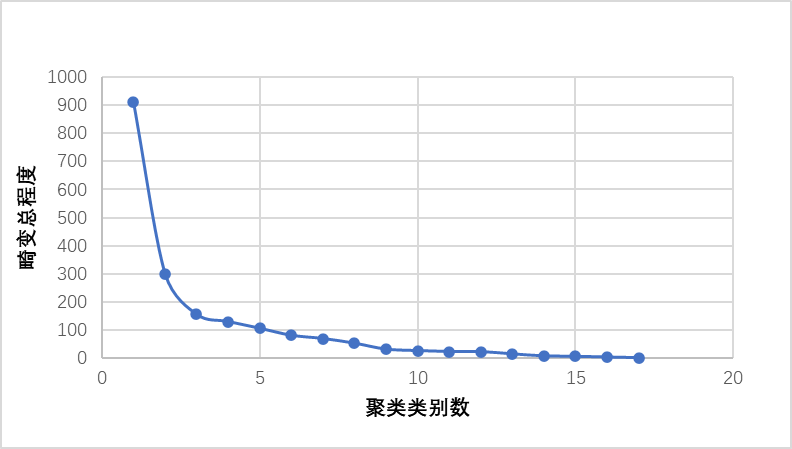


图9 高钾敏感度

## 问题三模型的建立与求解

为了对未知类别玻璃文物的分类有更好的合理性和敏感度，基于问题一所求各氧化物含量比例与玻璃风化的显著性，得到各属性对分类的重要程度，使用ID3决策树的算法建立以显著性高的化学成分为因变量的玻璃文物分类模型，对各化学成分进行风化贡献评级分析分类的合理性，通过选择最敏感或最不敏感的属性作为当前拓展属性进行分支，来判断分类模型是否稳健。

### 基于ID3决策树算法分类模型

在数据预处理中，基于问题一，将氧化铅变量去除。信息熵是由香农提出，表示对随机变量X不确定性的度量。熵越高，信息越多;熵越低，信息越少。信息熵如公式( 1)所示:

其中s表示随机变量,c表示随机变量的取值数量，pi表示每个取值时的概率。

信息增益:假设S是一组实例,A是一个特征，s v是S的子集，其中A = v, Values (A)是A 的所有可能值的集合，则信息增益为

当使用决策树中的节点将训练实例划分为更小的子集时，熵会发生变化。某个特征的信息增益是熵变化程度的量度。该算法具体的步骤如下：

Step1.特征选择

这一步主要根据特征选择算法来选取合适的特征来对当前样本集进行划分。在特征选择中本文选用使用信息增益为准则筛选出跟分类标签最相关性的特征。

Step2.决策树生成

选好特征后，就从当前树的根节点出发，遍历所有特征,找到那个使分信息增益最大的特征，将其设置为根节点，并且将这feature删除掉；根据该特征的不同取值建立子节点;对每个子节点使用相同的方式生成新的子节点，直到信息增益很小（满足一定的阈值）或者当前样本集不需要再划分为止；采用信息增益来选取最优分裂特征。

Step3.决策树剪枝

决策树剪枝的主要防止过拟合，通过提前去掉部分分支来降低过拟合的风险。剪枝的阈值设置是一个关键。

Step4.调整超参数分析敏感性

影响决策树敏感性的有超参数max\_depth，min\_samples\_leaf， min\_samples\_split。如果max\_depth设定的太低，那么决策树不能够很好的学习数据，如果设置的太大则极易造成模型的过拟合，min\_samples\_leaf超参数设置得太小会引起过拟合，设置得太大就会阻止模型学习数据，min\_samples\_split决定模型是否可以分支，还可以有效预防过拟合，本文对这三个超参数进行测试来分析模型的敏感性。

|  |  |
| --- | --- |
| 模型的求解  1. 模型的分类结果   class = 0 表示铅钡玻璃  class = 1 表示高钾玻璃  决策树  图10 基于决策树模型的对表单三化学成分预测  表单三化学成分预测具体结果位于附录中。   1. 敏感性检验结果   image-20220918005441176  图11 超参数max\_depth  首先，我们对决策树的最大深度进行敏感性分析。超参数max\_depth限制树的最大深度，超过设定深度的树枝将被全部剪掉用于限制树的最大深度。考虑到题目数据量较少，树的最大深度不宜过高，所以我们对该参数进行敏感性分析。如图所示，当max\_depth取5时，模型在验证集和训练集上的效果达到最佳，因此最优参数为max\_depth=5。  image-20220918005757811  图12 超参数min\_samples\_leaf  其次，对决策树超参数min\_samples\_leaf进行敏感性分析。min\_samples\_leaf表示一个节点在分枝后的每个子节点都必须包含至少min\_samples\_leaf个训练样本，否则分枝就不会发生，或者分枝会朝着满足每个子节点都包含min\_samples\_leaf个样本的方向去发生。min\_samples\_leaf一般和max\_depth参数配合使用，可以让模型变得更加平滑，同时，min\_samples\_leaf可以保证每个叶子的最小尺寸，可以在回归问题中避免低方差，过拟合的叶子节点出现。如图所示，随着min\_samples\_leaf的增加，train和test的准确率都有所下降，综合考虑模型的泛化能力以及准确度，我们认为min\_samples\_leaf的最优值为2.  image-20220918005824667  图13 超参数min\_samples\_split  最后，对超参数min\_samples\_split进行敏感性分析 min\_samples\_split限制决策树一个节点必须要包含至少min\_samples\_split个训练样本，这个节点才允许被分枝，否则分枝就不会发生，min\_samples\_split可以有效防止过拟合。但是由于玻璃样本的数据比较少，并且结合敏感性曲线分析，我们认为min\_samples\_split最优参数为4.  问题四模型的建立和求解  在第二题中，我们引入了“抗风化能力”这一概念，并且用Topsis得分去表征它，所以在这一题中，认为化学成分之间的关系与玻璃本身的抗风化能力有关，本文基于灰色关联模型构建了针对化学成分之间的关系与不同类别的玻璃文物抗风化能力的分析模型。  化学成分之间的灰色关联模型  Step1.将第二题所求得的铅钡与高钾玻璃Topsis评分当做母序列，其他化学成分当做子序列，以此来分析其他化学成分对于玻璃抗风化能力的影响程度，以及不同化学成分之间的联系。  Step2.对正向化后的矩阵进行预处理，得到矩阵Znxm=(Zij)nxm，具体矩阵数值见附件。  Step3.计算子序列中各个指标与母序列的关联系数。  母序列： （Ⅷ）  子序列：    两极最小差 （Ⅸ）  两极最大差 （Ⅹ）  (ρ为分辨系数，取0.05 ) （XI）  和的灰色关联度 （XII）  模型的求解  根据python计算，设定化学成分对应特征顺序为：二氧化硅，氧化钠，氧化钾，氧化钙，氧化镁，氧化铝，氧化铁，氧化铜，氧化铅，五氧化二磷，氧化锶，氧化锡，二氧化硫。  高钾的灰色关联度系数为：  [0.97835483, 0.84242264, 0.88733777, 0.884876  , 0.89560761,   0.91239705, 0.88283529, 0.9178258 , 0.86610521, 0.8480089 ,0.90449882, 0.85303659, 0.86511522, 0.84456953]  铅钡的灰色关联度系数为：  [0.94105583, 0.88245123, 0.91247829, 0.95657924, 0.92914516,  0.94592981, 0.91610475, 0.92718159, 0.97269554, 0.9536275,0.93650384, 0.9554802 , 0.88464863, 0.89391412]  图14 化学成分关联关系的差异性  高钾与铅钡玻璃其抗风化能力关联度由高到低如图所示：  图15 玻璃抗风化能力关联度  对于高钾玻璃而言，对抗风化能力贡献程度较大的化学成分主要为二氧化硅、氧化铜、氧化铝、五氧化二磷等成分。  而对于铅钡玻璃而言，对抗风化能力贡献程度较大的化学成分主要为氧化铅、氧化锶、氧化钡、氧化铝等成分。    图16 表面风化在类型的数量  如图所示，铅钡玻璃的风化样本明显多于高钾玻璃。而在高钾玻璃抗风化能力成分中占主要地位的二氧化硅，其含量在铅钡玻璃中相对较少。所以我们认为对于不同种类玻璃，高钾玻璃含有更多的“抗风化能力”成分，而相应成分在铅钡玻璃中比重则较小，这就是不同种玻璃化学成分关联关系的差异性，而这种差异性进一步影响了两种玻璃的抗风化能力。 |  |

# 模型的评价、改进与推广

本文主要构建了基于多元线性回归算法的分析模型、基于Kmeans与Topsis的亚类划分模型、基于决策树的分类模型以及基于灰色关联算法的分析模型。

## 模型的优点

（1）从定性与定量两个角度出发，构建化学成分和玻璃外在特征与玻璃类别之间的多元回归模型，通过回归系数可以直观的解释玻璃种类的分类规律。

（2）我们对每一个模型都进行了敏感性分析，从结果可以看出，这些模型都具有鲁棒性好、结果可靠等优点

（3）模型具有一定的通用性。比如我们用于解释亚类划分的Topsis模型也适用于玻璃抗风化能力分析中，并且与灰色关联模型结合使得结果具有较高的合理性与严谨性。

## 模型的缺点

（1）由于我们考虑的问题较为复杂，需要的数据量庞大，因此模型并不适用与数据量少的场景。

（2）对于Topsis模型权重的处理具有一定的主观性，虽然考虑到原始数据中0值较多所以对权重进行了特殊的处理，但是必然影响到模型的准确性。

## 模型的改进

（1）在铅钡玻璃亚类划分的结果里，Topsis的结果并不能完全把亚类划分开，因此我们认为，对于Topsis模型，可以选取更合适的化学成分作为评分特征，使得亚类的划分具有更高的可解释性。

# 参考文献

1. sklearn实战-----1.sklearn入门与决策树.CSDN.2020-06-16

【2】多元线性回归分析.CSDN. 2022-03-28

【3】陈飞.采用信息散布指数的改进决策树算法[J].数学的实践与认识,2020,50(14):76-82.

附录

|  |
| --- |
| 附录1高钾和铅钡玻璃的平均组间聚类 |
| 介绍：支撑材料的文件列表 |
|  |

|  |
| --- |
| 附录2 |
| 介绍：导入Python必要的包 |
| import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  from sklearn import linear\_model  import sklearn  import statsmodels.api as sm  from scipy import stats  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import accuracy\_score  from sklearn.preprocessing import MinMaxScaler  from sklearn import tree  import graphviz  from sklearn import preprocessing  #%matplotlib  from sklearn.neural\_network import MLPRegressor |

|  |
| --- |
| 附录3 |
| 介绍：基于神经网络进行玻璃风化前化学成分预测 |
| #model = sm.GLS(y\_train.iloc[:,0],x\_train)  #result = model.fit()  #result.summary()  df\_train = pd.read\_excel("train5.xlsx")  df\_train = df\_train.dropna()  x\_train = pd.get\_dummies(df\_train.iloc[:,-4:])  x\_train = x\_train.iloc[:,1:]  x\_train = x\_train.drop(labels=["颜色\_绿"],axis=1)  y\_train = df\_train.iloc[:,1:-5]  for num in range(14):  model\_mlp = MLPRegressor(  hidden\_layer\_sizes=(18,2),activation='relu', solver='adam', alpha=0.0001, batch\_size='auto',learning\_rate='constant',learning\_rate\_init=0.001,power\_t=0.5,max\_iter=50000, shuffle=True,random\_state=1, tol=0.0001, verbose=False, warm\_start=False, nesterovs\_momentum=True,early\_stopping=False,beta\_1=0.9,beta\_2=0.999, epsilon=1e-08,momentum=0.9)  model\_mlp.fit(x\_train,y\_train.iloc[:,num])  pred = model\_mlp.predict(x\_train)  #x\_train = x\_train.reshape(-1,1)  mlp\_score=model\_mlp.score(x\_train.values,y\_train.iloc[:,num].values)  mlp\_score  draw, = plt.plot(y\_train.iloc[:,num], 'b')  draw.set\_label("true")  draw, = plt.plot(pred, 'r')  draw.set\_label("predict")  #plt.plot(x1, result, 'ro')  draw.set\_label("predict")  plt.ylabel('P2O5')  plt.legend()  plt.show()  print(mlp\_score) |

|  |
| --- |
| 附录4 |
| 介绍：基于Python的Topsis评价模型实现，用于衡量玻璃的抗风化能力 |
| #Topsis函数  #Topsis函数定义  #0 极大型  #1 极小型  #2 中间型 额外标志：Xbest #PS:个人感觉，中间型的转换，本质上是先转换成极小型再转换成极大型  #3 区间型 额外标志：[a,b]区间 #正向化的方法是不唯一的，这里参考网课上的  #正向化  def ForwardDirection(m,y):#m表示输入矩阵，y表示指标类型  num=len(y)  for i in range(num):  #i代表着第i个指标，对应着m的第i列（从0计数）  labeltype=y[i]  if(labeltype[0]==0):  print("第",i+1,"个指标为极大型指标，无需正向化")  elif(labeltype[0]==1):  print("第",i+1,"个指标为极小型指标，进行正向化")  maxnum=np.amax(m,axis=0)#返回每一列元素的最大值  maxnum=maxnum[i]  m[:,i]= maxnum-m[:,i]  elif(labeltype[0]==2):  print("第",i+1,"个指标为中间型指标，进行正向化")  xbest=labeltype[1]  M=np.max(np.abs(m[:,i]-xbest))  m[:,i]=(1-np.abs(m[:,i]-xbest)/M)  elif(labeltype[0]==3):  print("第",i+1,"个指标为区间型指标，进行正向化")  a=labeltype[1]  b=labeltype[2]  # print("a,b:",a,b)  minx=np.min(m[:,i])  maxx=np.max(m[:,i])  #print(m[:,i])  # print("min,max",minx,maxx)  M=max(a-minx,maxx-b)  col=len(m[:,i])  for j in range(col):  if(m[j,i]<a):  m[j,i]=1-(a-m[j,i])/M  elif(m[j,i]<=b and m[j,i]>=a):  m[j,i]=1  elif(m[j,i]>b):  m[j,i]=1-(m[j,i]-b)/M  return m  def Standardize(m):#m为输入矩阵，且已经正向化    #第一步，先把m中元素乘方  temp\_all=np.power(m,2)  #第二步，按列求和得到一个行向量  temp\_all=np.sum(temp\_all,axis=0)#在第一个轴展开方向上求和  #第三步，将行向量元素开方  temp\_all=np.power(temp\_all,1/2)  #第四步，将m中每一列元素除以第三部中行向量对应列元素  for i in range(len(temp\_all)):  m[:,i]=m[:,i]/temp\_all[i]    return m  #计算得分  def getGrade(m,w):#m为输入矩阵，且已经标准化    #第一步，得到每一列的最大值Zmax与最小值Zmin  Zmax=np.max(m,axis=0)  Zmin=np.min(m,axis=0)    #第二步，通过ZmaxZmin得到每个样本（即本例中的小王小明等）的每个指标到各个指标最大值的距离与到最小值的距离  #1.将m中的每一列元素减去Zmax/Zmin中对应列的值  #2.然后将结果乘方  #3.按列方向展开求和  #4.再开方，从而得到每个评价对象（即样本）的Dmax与Dmin（注意：这里的算法我还没有算上权重）  Dmax = np.subtract(m,Zmax)#Zmax是行向量，这里有广播  # print(Dmax)  Dmin = np.subtract(m,Zmin)#同上  Dmax=np.power(Dmax,2)  Dmin=np.power(Dmin,2)    Dmax=Dmax\*w  Dmin=Dmin\*w    Dmax=np.sum(Dmax,axis=1)  Dmin=np.sum(Dmin,axis=1)    Dmax=np.power(Dmax,1/2)  Dmin=np.power(Dmin,1/2)  #print("Dmax,Dmin",Dmax,"\n",Dmin)  #第三步，根据公式：分数=到最小值距离/(到最大值距离+到最小值距离)，算出每个评价对象的综合得分S  S=Dmin/(Dmax+Dmin)  #print(S)  #第四步，对得分进行归一化  S\_sum=np.sum(S,axis=0)  S=S/S\_sum  return S |

|  |
| --- |
| 附录5 |
| 介绍：对之前定义的Topsis模型进行调用，得到相关数据 |
| #读入铅钡数据  df\_topsis\_qb = pd.read\_excel("铅钡Topsis.xlsx")  data = df\_topsis\_qb.loc[:,['二氧化硅(SiO2)','氧化钙(CaO)','氧化铜(CuO)','氧化铅(PbO)','五氧化二磷(P2O5)','二氧化硫(SO2)']]  #考虑到原始数据具有较多0值，故对权重进行处理  w=[[1/6,1/6,1/6,1/6,1/6,1/6]]\*data.shape[0]  w=np.array(w)  data=np.array(data,dtype=np.float32)  label=[[1],[0],[1],[0],[0],[0]]  data=ForwardDirection(data,label)  data=Standardize(data)  S=getGrade(data,w)  S\_df = pd.DataFrame(S,columns=['Topsis得分'])  S\_df.to\_excel(excel\_writer=r"2022C数据处理Topsis1.xlsx")  #读入高钾数据  df\_topsis\_gj = pd.read\_excel("高钾Topsis.xlsx")  data = df\_topsis\_gj.loc[:,['二氧化硅(SiO2)','氧化钾(K2O)','氧化钙(CaO)','氧化铝(Al2O3)','五氧化二磷(P2O5)']]  #考虑到原始数据具有较多0值，故对权重进行处理  w=[[1/5,1/5,1/5,1/5,1/5]]\*data.shape[0]  w=np.array(w)  data=np.array(data,dtype=np.float32)  label=[[0],[1],[1],[1],[1]]  data=ForwardDirection(data,label)  data=Standardize(data)  S=getGrade(data,w)  S\_df = pd.DataFrame(S,columns=['Topsis得分'])  S\_df.to\_excel(excel\_writer=r"2022C数据处理Topsis0.xlsx") |

|  |
| --- |
| 附录6 |
| 介绍：基于决策树的玻璃种类分类 |
| #用决策树进行种类预测  df\_tree = pd.read\_excel("决策树预测.xlsx")  df\_tree = df\_tree.iloc[:,1:]  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=0.3)  #生成模型  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=5)  clf = clf.fit(x\_train,y\_train)  result = clf.score(x\_test,y\_test)  feature\_name = ['二氧化硅(SiO2)','氧化钠(Na2O)','氧化钾(K2O)','氧化钙(CaO)','氧化镁(MgO)','氧化铝(Al2O3)',  '氧化铁(Fe2O3)','氧化铜(CuO)','氧化铅(PbO)','氧化钡(BaO)','五氧化二磷(P2O5)','氧化锶(SrO)',  '氧化锡(SnO2)','二氧化硫(SO2)','表面风化']  tree.plot\_tree(clf,class\_names=['高钾','铅钡']) |

|  |
| --- |
| 附录7 |
| 介绍：对决策树四个超参数进行敏感性分析 |
| #针对树深度的灵敏度分析  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=0.3)  score\_train = []  score\_test = []  for depth in range(1,10):  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=depth,  splitter="random")  clf = clf.fit(x\_train,y\_train)  score\_train.append(clf.score(x\_train,y\_train))  score\_test.append(clf.score(x\_test,y\_test))  score\_train,score\_test  x = [1,2,3,4,5,6,7,8,9]  draw, = plt.plot(x,score\_train,'b')  draw.set\_label("train")  draw, = plt.plot(x,score\_test,'r')  draw.set\_label("test")  plt.xlabel('max\_depth')  plt.legend()  score\_train=[]  score\_test = []  #针对数据划分的灵敏度分析  for i in np.linspace(0.1,0.9,9):  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=i)  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=5,  splitter="random")  clf = clf.fit(x\_train,y\_train)  score\_train.append(clf.score(x\_train,y\_train))  score\_test.append(clf.score(x\_test,y\_test))  x = [1,2,3,4,5,6,7,8,9]  plt.plot(x,score\_train,'b')  plt.plot(x,score\_test,'r')  # ,min\_samples\_leaf=10,min\_samples\_split=10  #针对min\_samples\_leaf参数进行敏感性分析  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=0.3)  score\_train=[]  score\_test = []  for i in range(20):  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=4,  splitter="random",  min\_samples\_leaf=i+1)  clf = clf.fit(x\_train,y\_train)  score\_train.append(clf.score(x\_train,y\_train))  score\_test.append(clf.score(x\_test,y\_test))  x = np.arange(20)  draw, = plt.plot(x,score\_train,'b')  draw.set\_label("train")  draw, = plt.plot(x,score\_test,'r')  draw.set\_label("test")  plt.xlabel('min\_samples\_leaf')  plt.legend()  #针对min\_samples\_spilt参数进行敏感性分析  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=0.3)  score\_train=[]  score\_test = []  for i in range(20):  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=4,  splitter="random",  min\_samples\_split=i+2)  clf = clf.fit(x\_train,y\_train)  score\_train.append(clf.score(x\_train,y\_train))  score\_test.append(clf.score(x\_test,y\_test))  x = np.arange(20)  draw, = plt.plot(x,score\_train,'b')  draw.set\_label("train")  draw, = plt.plot(x,score\_test,'r')  draw.set\_label("test")  plt.xlabel('min\_samples\_spilt')  plt.legend() |

|  |
| --- |
| 附录8 |
| 介绍：确定决策树最终的超参数，并进行未知种类的预测 |
| #确定参数maxdp = 5  df\_tree = pd.read\_excel("决策树预测.xlsx")  df\_tree = df\_tree.iloc[:,1:]  x\_train,x\_test,y\_train,y\_test = train\_test\_split(df\_tree.iloc[:,:-1],df\_tree.iloc[:,-1],test\_size=0.3)  #生成模型  clf = tree.DecisionTreeClassifier(criterion="entropy",  random\_state=30,  max\_depth=5)  clf = clf.fit(x\_train,y\_train)  result = clf.score(x\_test,y\_test)  feature\_name = ['二氧化硅(SiO2)','氧化钠(Na2O)','氧化钾(K2O)','氧化钙(CaO)','氧化镁(MgO)','氧化铝(Al2O3)',  '氧化铁(Fe2O3)','氧化铜(CuO)','氧化铅(PbO)','氧化钡(BaO)','五氧化二磷(P2O5)','氧化锶(SrO)',  '氧化锡(SnO2)','二氧化硫(SO2)','表面风化']  tree.plot\_tree(clf,class\_names=['高钾','铅钡'])  #clf.score(x\_train,y\_train),clf.score(x\_test,y\_test) |

|  |
| --- |
| 附录9 |
| 介绍：基于Python实现的灰色关联分析模型 |
| #灰色关联手撕  #第一步，数据预处理：每个元素除以均值  #第二步，计算关联系数：  #1）求每一列特征的两极最小差与两极最大差a与b  #1.5)求得母与子对应列的绝对差fabs  #2）定义分辨系数p （超参数，一般为0.5）  #3）求得子列每个元素的Yi = (a+pb)/(fabs+pb)  #4）对每一列的Yi求和/n，最终得到每一个特征的关联程度  def gray(df\_values,p):  temp1 = np.sum(df\_values,axis=0)#得到每一列的和  temp1 = temp1/df\_values.shape[0]#得到每一列的均值  df\_values = df\_values/temp1 #对数据预处理  #print(df\_values)  df\_mother = df\_values[:,-1]#母列  df\_son = df\_values[:,:-1]#子列  for i in range(df\_son.shape[1]):  df\_son[:,i] = np.fabs(df\_son[:,i] - df\_mother)  #print(df\_son)  a=np.min(df\_son)#求每一列特征的两极最小差与两极最大差a与b  b=np.max(df\_son)  #print(a,b)  df\_son = (a+p\*b)/(df\_son+p\*b)  #print(df\_son)  df\_son = np.sum(df\_son,axis=0)/df\_values.shape[0]    return df\_son |

|  |
| --- |
| 附录10 |
| 介绍：调用灰色关联模型函数，生成灰色关联度数据 |
| df\_4 = pd.read\_excel("高钾\_4.xlsx")  w = gray(df\_4.iloc[:,1:].values,0.5)#高钾  w = pd.DataFrame(w,columns=["灰色关联度"])  w.to\_excel(excel\_writer=r"灰色关联度\_高钾.xlsx")  df\_4\_qb = pd.read\_excel("铅钡\_4.xlsx")  w = gray(df\_4\_qb.iloc[:,1:].values,0.5)#高钾  w = pd.DataFrame(w,columns=["灰色关联度"])  w.to\_excel(excel\_writer=r"灰色关联度\_铅钡.xlsx") |

|  |
| --- |
| 附录11 |
| 介绍：前期实现的线性回归模型预测样品成分，以氧化钙为例 |
| #氧化钙  df\_train = pd.read\_excel("train6.xlsx")  df\_train = df\_train.dropna()  x\_train = pd.get\_dummies(df\_train.iloc[:,-4:])  x\_train = x\_train.iloc[:,1:]  x\_train = x\_train.drop(labels=["颜色\_绿"],axis=1)  y\_train = df\_train.iloc[:,1:-5]  x\_train = sm.add\_constant(x\_train)  model = sm.GLS(y\_train.iloc[:,5],x\_train)  result = model.fit()  result.summary()  #画图  w = np.array(result.params)  pred = x\_train.values\*w  pred = pred.sum(axis=1)  x = np.arange(1,26)  plt.plot(x,pred,'b')  plt.plot(x,y\_train.iloc[:,10].values,'r')  plt.show()  #预测  y\_pred = x\_pred.values\*w  y\_pred = y\_pred.sum(axis=1)  bias = np.random.normal(loc=0.0, scale=1.0, size=[y\_pred.shape[0]])  y\_pred = y\_pred + bias  y\_pred  #导出数据  Outer = pd.DataFrame(y\_pred,columns=["预测氧化钙"])  Outer.to\_excel(excel\_writer=r"2022数据处理4.xlsx") |