基于小样本学习算法的古代玻璃制品的成分分析与鉴别

摘要

古代玻璃制品的成分分析与鉴别对我国考古研究和古代玻璃制品文化的研究具有重要意义。本文对古代玻璃制品特征进行分析，研究了各化学成分的变化规律和相关性，研究不同类型玻璃分类规律，并采用决策树、随机森林、层次聚类和 Logistic 回归等模型对问题进行建模解决。

针对问题一，首先对数据进行预处理，补全缺失值且将无效值删除，进而利用信息增益对表面风化与纹饰、类型和颜色的关系进行分析，得出类型为影响表面风化的最主要因素。其次，分析化学元素，得出高钾玻璃风化后二氧化硅含量大幅增加，其余化学成分含量均有所下降，而铅钡玻璃受到风化其二氧化硅含量大幅减少，但氧化铅含量大幅增加，其余化学成分大多有增加趋势。对于风化点，计算其平均差值，加入惩罚因子对其变化程度进行控制，最终得到表 5 和表 6 数据。

针对问题二，首先将两表进行整合并对非数值型数据进行编码，通过 LightGBM 算法查找相关特征重要性，得到 PbO，BaO，SiO2，K2O，重要性较强，在通过这些特征求得模型的系数和偏置。通过层次聚类选择出五个亚分类，再将其放入决策树模型，通过计算相关的基尼，通过基尼选择化学元素，最终得到二氧化硅(SiO2)作为节点得到基尼最大，故选二氧化硅(SiO2)划分。

针对问题三，对化学成分分析以鉴别类型，由于数据量较少，使用是随机森林模型每次都可以选择一个子集并重复采样，可以增加用于训练的数据，最终预测 A1,A6,A7 为高钾，其余为铅钡，并通过添加随机噪声的方法证明了模型具有较高的鲁棒性。

针对问题四，对不同类别玻璃文物样品其化学成分关联关系进行分析，分别得出每个特征之间的变量相关性，画出热力图，通过对比高钾和铅钡两图差异，得出高钾玻璃中氧化钙和氧化钾之间的关联程度最大，而铅钡中氧化铝和二氧化硅相关性最高，高钾和铅钡这两种类型的玻璃中，相应化学成分关联度差值在-0.012-0.062 之间波动。

关键词： 决策树　随机森林　层次聚类　特征重要性　玻璃风化化学成分

一、问题重述

1.1 问题背景

玻璃是人类最早发明的人工材料之一，其发展历史源远流长[1]。对我国来说，玻璃最早是由丝绸之路从西亚等地区传入，是早期贸易往来交流的见证。我国古代玻璃是学习并吸收西方先进技术后就地取材制作而成，体现了古代发达的工艺技术。然而古代玻璃不易保存，其极易受环境的影响而风化。因此通过检测手段并分析玻璃中化学成分与风化关系显得尤为重要。

1.2 问题重述

现实情况下，古代玻璃容易受环境影响而风化。在风化过程中，玻璃内部元素会与环境元素进行交换，导致成分比例发生变化，从而影响其类别判定。

本文通过结合检测出文物信息的相关情况，建立数学模型研究以下问题：

（1）根据附件表单 1 中数据，将玻璃的表面风化和其类型、纹饰以及颜色之间的关系进行分析即研究其相关性；结合玻璃类型和表单 2 中的化学成分含量，分析样品表面有无风化化学成分含量的统计规律，以及根据风化点检测数据，预测风化前的化学成分含量。

（2）通过附件数据分析高钾和铅钡玻璃的分类规律；将每个类别的玻璃选择合适化学成分对其进行亚类划分即对主要类型再次进行细分，给出具体的划分方法和结果，并对分类结果的合理性和敏感性进行分析。

（3）依据附件表单 3 中未知类别玻璃的化学成分进行分析，鉴别所属类型，并对分类结果的敏感性进行分析。

（4）针对不同类别的玻璃分析其化学成分之间的关联关系，并比较不同类别之间的化学成分关联关系的差异性。

二、问题分析

2.1 问题一的分析

针对问题一，首先需要对玻璃文物的表面风化与三个特征的关系进行分析，此问题可以从相关性进行处理，运用信息增益分析三个特征与表面风化的相关程度来分析其关系。进而结合类型，将文物样品高钾和铅钡分类进行处理，求其各化学成分差值，以发现其统计规律。最后根据统计规律，对风化点的数据，与化学成分变化值进行运

算，预测得风化点风化前的检测数据。

2.2 问题二的分析

针对问题二，需要综合考虑表单 1 和表单 2 数据，对分类规律进行寻找。因此对于分类规律，首先需找出影响类别划分的主要特征，根据特征的差异以寻找规律。对于特征重要性的寻找，可以运用 LightGBM 算法中的特征重要性函数，运算不同特征的重要性，因此可以对重要性高的特征，建立 Logistic 回归方程，以此来表示不同类型玻璃的分类规律。对于这两类下属的亚类进行划分，对于多层次分类问题可以运用层次分析法对数据进行处理，将高钾和铅钡两种玻璃分别进行层次分析，以一定阈值确定具体划分结果，最后通过划分结果的正态性对合理性进行分析。

2.3 问题三的分析

针对问题三，对未知类别玻璃文物化学成分分析后鉴别类型，可将该题理解为预测题求解，运用已知数据预测未知数据，对于数据量较小但特征数量较多的预测问题，可以采用随机森林模型对类型进行预测，预测结果后可以通过随机噪声干扰对其敏感性进行分析。

2.4 问题四的分析

针对问题四，分析各化学成分间的关联关系，由于化学成分在风化过程中会发生变化，且不同化学成分会产生交叉影响，因此针对此变化数据可以采用灰色关联分析法对每两两成分间的关联度进行运算，可以结合关联度分析其关联关系。对于不同类别化学成分差异性的比较，可以对高钾铅钡两类型文物中两两化学成分对应关联度作差，进而可以获得其变化值，再以此来比较差异性。

三、模型的假设

1. 假设附件表单 2 中空白数据无法检测到的成分其比例无限趋近于 0
2. 假设无本文外的其他因素影响玻璃文物化学成分变化

四、符号说明

符号	含义
S	表 1 数据集

X_1	玻璃的类型
X_2	玻璃的颜色
X_3	玻璃的纹饰
$\hat{i}_t^2$	树节点分裂之后平方损失的减少值
γ	惩罚因子
B_i	文物表面风化程度
M	决策树数量
L	叶子节点数量
P_k	相同特征不同样本占比

五、模型的建立与求解

5.1 问题一模型的建立与求解

5.1.1 数据的预处理

对于附件中的表单 1 数据进行分析，发现在对不同玻璃文物进行信息统计时部分文物颜色数据存在缺失值，故运用 SVM 对缺失值进行预测，预测结果均为浅蓝色。

对于表单 2 中的数据，针对数据成分性的特点，在成分比例介于 85%~105%的数据为有效数据，故对各采样点化学成分含量进行求和，经过计算采样点 15 与采样点 17 其化学成分含量总和分别为 79.47%和 71.89%，成分比例不介于该范围内，因此在后续模型建立中将对此数据进行清除。

5.1.2 玻璃文物的表面风化与其玻璃类型、纹饰和颜色的关系分析

在该问题中需对玻璃类型、纹饰和颜色与表面风化的关系进行分析，因各数据为离散数据，所以对离散数据的相关性进行计算，采用决策树的中的信息增益进行相关性分析。

信息熵指特定信息出现的概率，样本类别越多，分布越均匀，熵也就越大，因此分别以玻璃类型、颜色、纹饰为特征对表 1 数据进行划分，表 1 数据集 S，玻璃类型、颜色、纹饰分别为X_1、X_2、X_3，可针对经过特征划分后的数据集中该特征不同分类$p(k)$进行信息熵的计算，信息熵：

$$Ent(SX_i) = -\sum_{k=1}^{n} p(k)\log_2[p(k)] \tag{1}$$

根据以上数据可求得结果如表 1 所示：

表 1 不同特征信息熵数值

	纹饰	类型	颜色	表面风化
信息熵	1.361	0.8936	2.3227	0.9784

进而根据已求得的类型、颜色、纹饰的信息熵可以加入表面风化程度的特征 B_i 进行信息增益的计算，信息增益：

$$Gain(SX_i,B) = Ent(SX_i) - \sum_{i=1}^{V}\frac{|B_i|}{|S|}Ent(SX\_B_i) \tag{2}$$

由于本问题中影响风化程度的特征较多，且两特征概率分布不同，所以信息增益对于两两特征为非对称的，因此对于两两特征求其相关程度需对信息增益进行处理：

$$corr(X_iB) = \frac{Gain(SX_i,B)}{\sqrt{Ent(SB)*Ent(SX_i)}} \tag{3}$$

故可求得其相关程度结果如表 2 所示：

表 2 不同特征之间的信息增益

	纹饰	类型	颜色	表面风化
纹饰	1.0000	0.1809	0.3113	0.0768
类型	0.1809	1.0000	0.2304	0.0917
颜色	0.3113	0.2304	1.0000	0.0743
表面风化	0.0768	0.0917	0.0743	1.0000

对表 2 进行数据可视化处理可得图 1：

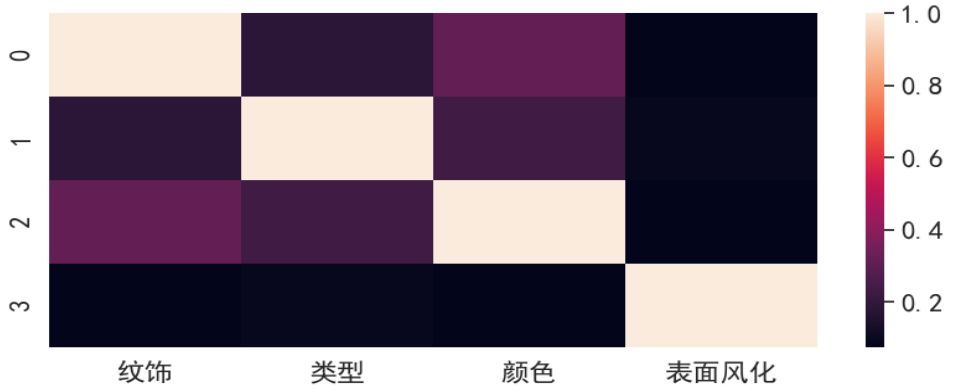


图 1 不同特征之间信息增益热力图

对图与表进行观察，可知表面风化与文物类型相关程度最高，因此文物类型的不同更容易导致表面风化程度的不同。且玻璃文物类型与颜色和纹饰也具有较高相关性，加之玻璃文物受到各方面影响因素较多，诸如颜色跟随表面风化程度会变化，纹饰等不同凹槽深度位置也会对表面风化程度产生一定影响，因此其他因素与风化程度也具有一定的相关性，但文物类型为影响表面风化程度的首要因素。

5.1.3 文物样品表面有无风化化学成分含量的统计规律

运用已预处理完成的数据对不同类型的玻璃表面风化化学成分进行分析，由于部分化学成分未被检测到，故其占比为 0。进而对每一个采样点的风化程度进行标记，其中严重风化点暂不对其放入总数据集中进行分析，将其留存以验证统计规律。

由于不同颜色花纹以及埋藏地和测量时的差异，因此同一类型的玻璃文物受到风化时化学成分比例有所差异，但基本可以断定各个化学成分比例在同一类型玻璃文物受到风化时的变化趋势是相同的，因此针对此问题的玻璃文物化学成分比例，通过计算均值对同一类型文物风化前后统计规律进行分析。

首先对高钾玻璃风化前后各个化学成分平均比例进行计算，数据如表 3 所示：

表 3 高钾玻璃风化前后各化学成分平均比例

	未风化	风化	变化比例
二氧化硅(SiO_2)	67.98	93.96	25.98
氧化钠(Na_2O)	0.70	0.00	-0.7
氧化钾(K_2O)	9.33	0.54	-8.79
氧化钙(CaO)	5.33	0.89	-4.44
氧化镁(MgO)	1.08	0.19	-0.89
氧化铝(Al_2O_3)	6.62	1.93	-4.69
氧化铁(Fe_2O_3)	1.93	0.265	-1.665
氧化铜(CuO)	2.45	1.56	-0.89
氧化铅(PbO)	0.41	0.00	-0.41
氧化钡(BaO)	0.60	0.00	-0.6
五氧化二磷(P_2O_5)	1.40	0.28	-1.12
氧化锶(SrO)	0.04	0.00	-0.04
氧化锡(SnO_2)	0.20	0.00	-0.2

二氧化硫(SO_2)	0.10	0.00	-0.1

对该表数据进行可视化处理可得：

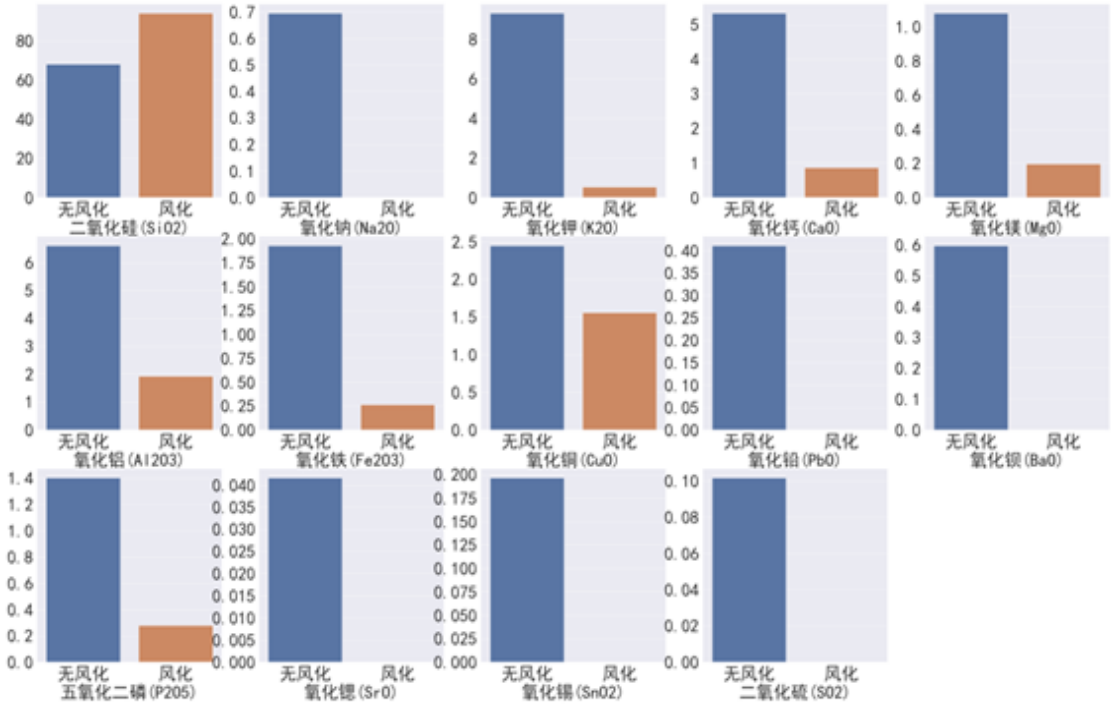


图 2 高钾玻璃风化前后各化学成分比例对比图

由此数据及图表进行观察分析可得，高钾玻璃在受到风化后二氧化硅的含量大大增加，其余化学成分比例均下降，其中氧化钠、氧化铅、氧化钡、氧化锶、氧化锡和二氧化硫的成分比例降至 0，可知该类型玻璃文物表面有无风化化学成分含量的统计规律：其二氧化硅的含量大幅度增加，剩余化学成分含量比例都下降。

其次将铅钡玻璃进行相同处理，对其风化前后各个化学成分平均比例进行计算，结果如表 4 所示：

表 4 铅钡玻璃风化前后各化学成分平均比例

	未风化	风化	变化比例
二氧化硅(SiO2)	54.66	24.91	-29.75
氧化钠(Na2O)	1.68	0.22	-1.46
氧化钾(K2O)	0.22	0.13	-0.09
氧化钙(CaO)	1.32	2.70	1.38
氧化镁(MgO)	0.64	0.65	0.01
氧化铝(Al2O3)	4.46	2.97	-1.49
氧化铁(Fe2O3)	0.74	0.58	-0.16
氧化铜(CuO)	1.43	2.28	0.85
氧化铅(PbO)	22.08	43.31	21.23

氧化钡(BaO)	9.00	11.81	2.81
五氧化二磷(P2O5)	1.05	5.28	4.23
氧化锶(SrO)	0.27	0.42	0.15
氧化锡(SnO2)	0.05	0.07	0.02
二氧化硫(SO2)	0.16	1.37	1.21

对该表数据进行可视化处理可得图 3：

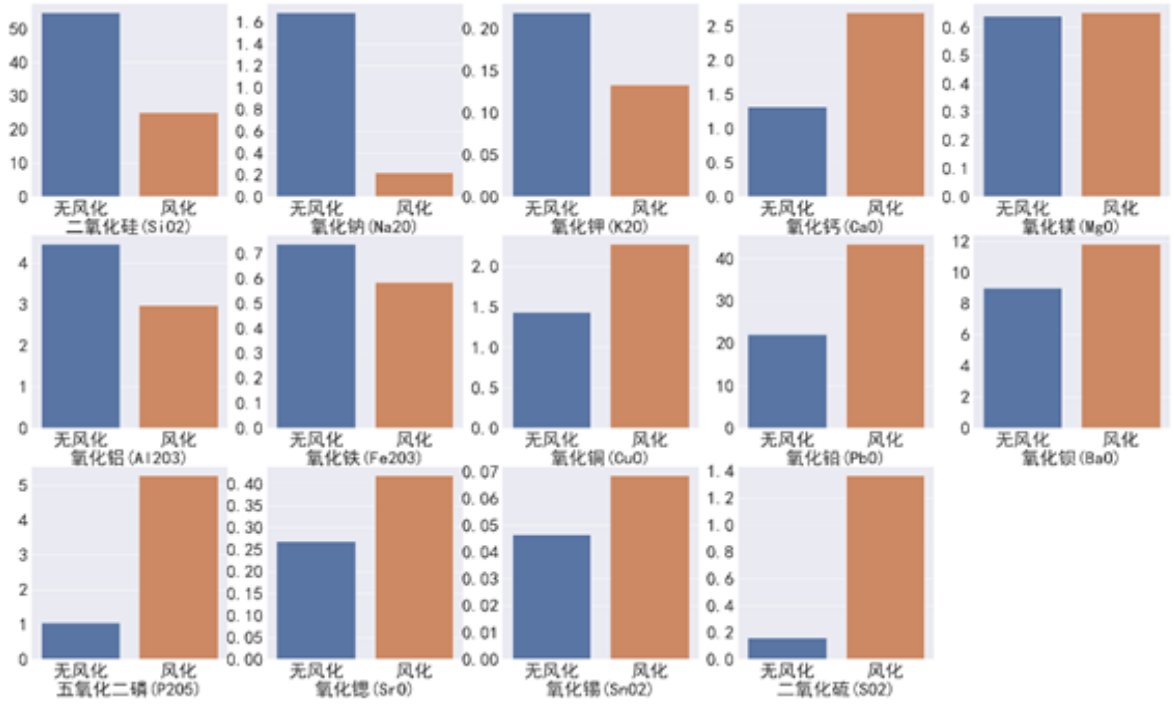


图 3 铅钡玻璃风化前后各化学成分比例对比图

由上述图表数据对比可知，该类型玻璃在风化后，其二氧化硅含量大大减少，氧化钠、氧化钾、氧化铝、氧化铁和氧化铜含量比例均减少；但氧化铅成分比例增加幅度大，其余化学成分含量都增加。经此可得铅钡玻璃文物样品有无风化化学成分含量的统计规律：风化后，二氧化硅含量大幅度减少，氧化铅含量明显增加，其余化学成分变化不大。

5.1.4 玻璃文物风化前的化学成分含量预测

对于高钾玻璃和铅钡玻璃两类数据，根据上文所得基本统计规律，对玻璃文物风化前的化学成分含量进行预测。我们定义预测公式为：

$$Y_{前} = Y_{后} + \gamma \Delta X \tag{4}$$

其中△x 为风化前后每个化学成分平均差值，计算流程如图 4 所示：

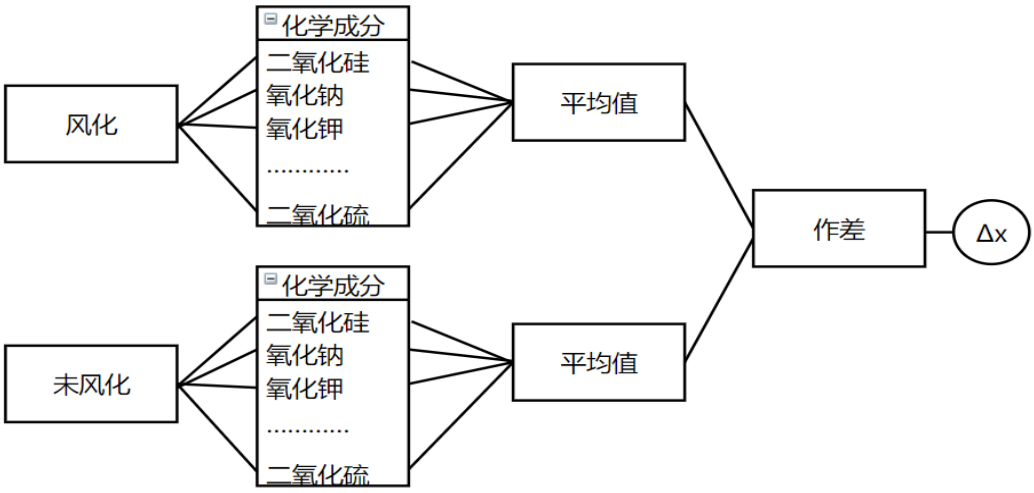


图 4 化学成分差值计算流程图

在风化前后各化学成分平均变化含量的基础上，加之以惩罚因子对其变化幅度进行控制，提高了模型的鲁棒性，并定义 γ 计算公式为:

$$\gamma = \frac{x_i - \overline{x_{后}}}{x_{后} \cdot std(x_{后})} \cdot \frac{x_i - x_{后} \cdot \min}{(x_{后} \cdot \max x_{后} - x_{后} \cdot \min x_{后}) \times 10} \tag{5}$$

最终预测高钾和铅钡玻璃的风化前数据如下表 5、6 所示：

表 5 高钾玻璃风化前化学成分预测结果

采样点	7	9	10	12	22	27
二氧化硅(SiO2)	0.667789	0.735409	0.875139	0.707122	0.672505	0.677844
氧化钠(Na2O)	0	0	0	0	0	0
氧化钾(K2O)	0.089705	0.101727	0.175696	0.205081	0.129039	0.089705
氧化钙(CaO)	0.080798	0.037252	0.047776	0.04202	0.223514	0.063363
氧化镁(MgO)	0.009048	0.009048	0.009048	0.009048	0.159139	0.111273
氧化铝(Al2O3)	0.07004	0.04914	0.056122	0.050793	0.244572	0.113469
氧化铁(Fe2O3)	0.018734	0.086646	0.015712	0.044617	0.142215	0.002727
氧化铜(CuO)	0.221035	0.024266	0.009367	0.029767	0.014819	0.024511
氧化铅(PbO)	0	0	0	0	0	0
氧化钡(BaO)	0	0	0	0	0	0
五氧化二磷(P2O5)	0.174352	0.033866	0.011447	-0.0023	0.001945	0.038551
氧化锶(SrO)	0	0	0	0	0	0
氧化锡(SnO2)	0	0	0	0	0	0
二氧化硫(SO2)	0	0	0	0	0	0

表 6 铅钡玻璃风化前化学成分预测结果（部分）

采样点	2	8	11	19	26	34
二氧化硅(SiO2)	0.728034	0.343884	0.708309	0.621706	0.481255	0.734729
氧化钠(Na2O)	0.01502	0.01502	0.01502	0.01502	0.01502	0.01502
氧化钾(K2O)	0.389285	0.0009	0.0104	0.0009	0.0009	0.015607
氧化钙(CaO)	-0.00049	0.030224	0.04988	0.0192	-0.01708	-0.0204
氧化镁(MgO)	0.03908	-0.00027	0.009477	0.003493	-0.00027	-0.00027
氧化铝(Al2O3)	0.112084	0.02248	0.041432	0.055924	0.020239	0.026699
氧化铁(Fe2O3)	0.126837	0.001505	0.001505	0.060488	0.001505	0.003601
氧化铜(CuO)	-0.00769	0.032682	0.092001	0.041523	0.38841	0.002962
氧化铅(PbO)	0.263336	0.078324	0.015163	0.204496	0.042922	0.267081
氧化钡(BaO)	-0.02874	0.448769	0.137949	0.01521	0.476533	0.068334
五氧化二磷(P2O5)	-0.01876	0.05986	0.121625	0.094099	-0.02399	-0.04314
氧化锶(SrO)	-0.01418	0.022268	-0.00303	-0.0142	0.005704	-0.014
氧化锡(SnO2)	-0.00022	-0.00022	-0.00022	-0.00022	-0.00022	-0.00022
二氧化硫(SO2)	-0.01218	0.455134	-0.01218	-0.01218	0.00915	-0.01218

5.2 问题二模型的建立与求解

在该问题中分析高钾玻璃和铅钡玻璃的分类规律，需要对表单 1 和表单 2 中的数据进行整合，进而对可能影响玻璃类型分类的所有数据运用模型进行分析，进而确定与类型相关性最高的特征，进而确定分类规律。

5.2.1 数据收集与预处理

将表单 1 和 2 进行整合，并对表单 1 中非数值数据进行编码，便于与表单 2 数据共同利用模型进行运算。

5.2.2 玻璃文物分类模型的建立与求解

由于本题中数据量较小，为了能够让模型充分的拟合数据，对处理好的数据运用五折交叉验证结合 LightGBM 算法来使模型更好地拟合数据，其中五折交叉验证时将玻璃文物采样点的所有数据随机分为五层，每次运算将其中一层留作验证，剩余四层进

行计算，循环五次，运算流程如图 5：

图 5 五折交叉验证流程

在每一层训练集进行运算时，均采用 LightGBM 算法对特征重要性进行运算，运用其中的 feature_importances_函数对玻璃文物样品的各个特征进行特征重要性分析，首先计算特征 j 在单棵树的特征重要性:

$$\hat{J}_j^2(T) = \sum_{t=1}^{L-1} \hat{i}_t^2 (v_t = j) \tag{6}$$

其次计算全局特征重要性，特征 j 的全局重要度通过特征 j 在单颗树中的重要度的平均值来衡量：

$$\hat{J}_j^2 = \frac{1}{M} \sum_{m=1}^{M} \hat{J}_j^2(T_m) \tag{7}$$

其中，M 是树的数量，L 为树的叶子节点个数，L-1 即为树的非叶子节点数量（构建的树都是具有左右孩子的二叉树），v_t 是和节点 t 相关联的特征，$\hat{i}_t^2$ 是节点分裂之后平方损失的减少值。

表 7 不同化学成分与类型的特征重要性（部分）

化学成分	特征重要性

氧化铅(PbO)	2
氧化钡(BaO)	1.6667
二氧化硅(SiO_2)	0.6667
氧化钾(K_2O)	0.3333
颜色	0.3333

由上表可知，氧化铅、二氧化硅、氧化钡和氧化锶的特征重要性相对较高，故采用该四种化学成分作为自变量，类型作为因变量，除去四组数据，高钾玻璃与铅钡玻璃各两组作为校验数据，其他数据运用 Logistic 回归对玻璃的分类进行预测，其 Logistic 回归计算公式如下：

$$\sigma(z)=\frac{1}{1+e^{-\omega^T x}} \tag{8}$$

通过训练模型可得模型相关系数如下表

ω_1	ω_2	ω_3	ω_4	b
-2.31	2.23	-0.78	0.61	-1.7

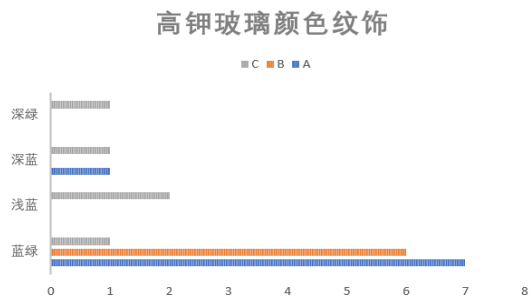
随机选取四条样本进行测试，可得实际结果与预测结果完全吻合，故 Logistic 回归方程可满足高钾玻璃、铅钡玻璃的分类规律，即下表所示：

表 8 不同文物采样点与实际采样点

文物采样点	预测类型	实际类型
9	高钾玻璃	高钾玻璃
22	高钾玻璃	高钾玻璃
44 未风化点	铅钡玻璃	铅钡玻璃
50	铅钡玻璃	铅钡玻璃

5.2.3 玻璃文物类型的亚类划分模型建立与求解

前文已对不同玻璃类型的分类规律进行了求解，故继续对高钾玻璃和铅钡玻璃分别进行亚类的划分，首先对高钾玻璃和铅钡玻璃的颜色和纹饰数据进行可视化处理，根据图 6 其中高钾和铅钡玻璃中各有较为明显的细划分类，可以假设其基本分类结果与颜色和纹饰有关。



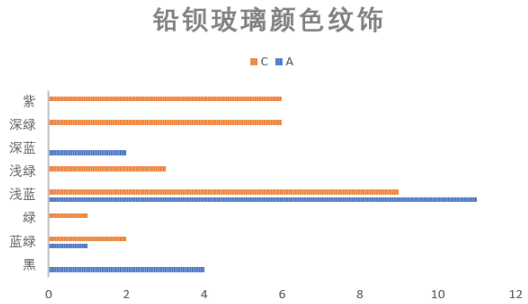


图 6 高钾玻璃与铅钡玻璃颜色纹饰分类

选择层次分析法对高钾玻璃和铅钡玻璃的下属分类继续进行划分，首先对于高钾玻璃，将高钾玻璃所有化学成分作为原数据，层次分析后可以得到可视化树状图：

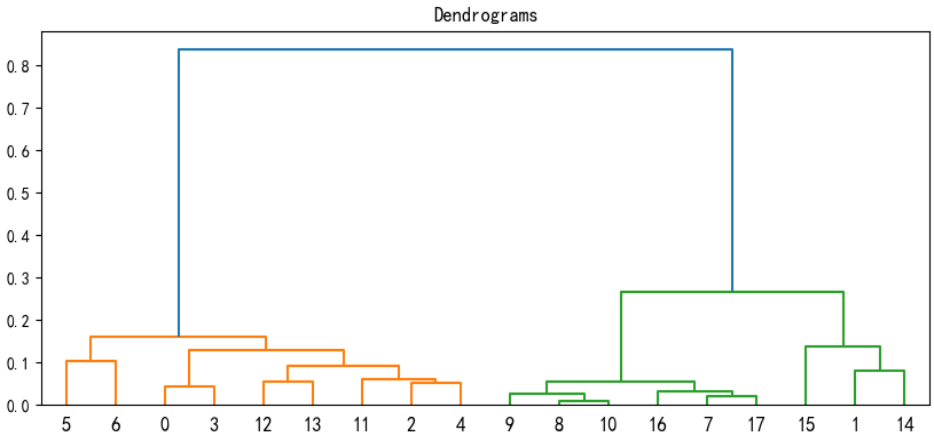


图 7 高钾玻璃层次分析树状图

对树状图的纵坐标选定一个阈值，以该阈值为分界线进行亚分类的划分，此处处于控制亚分类数量的目的，设置其阈值为 0.3，据此阈值可得整体数据被分为两类，其中 1、3、4、5、6、13、14、16、18、21 为一类，另一类为 7、9、10、12、22、27，将此部分采样点运用决策树模型进行运算，运算其划分方法。运算后可得图 8：

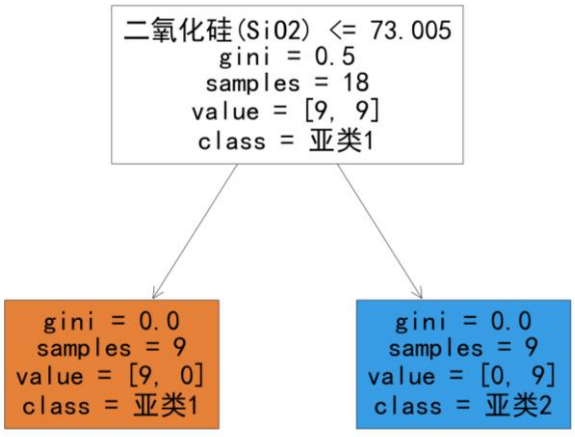


图 8 高钾玻璃决策树可视化

因此可知划分方法为依据玻璃文物中二氧化硅的含量，故划分结果如表 9 所示：

表 9 高钾玻璃文物划分结果

类型	文物编号	备注

类型 1	1、3、4、5、6、13、14、16、18、21	全为 A、C 纹饰，颜色绝大部分为蓝绿色
类型 2	7、9、10、12、22、27	全为 B 纹饰，蓝色全为蓝绿色

继续选择层次分析法对铅钡玻璃下属分类进行划分，选取铅钡玻璃所有化学成分对其层次分析后可以得到可视化树状图：

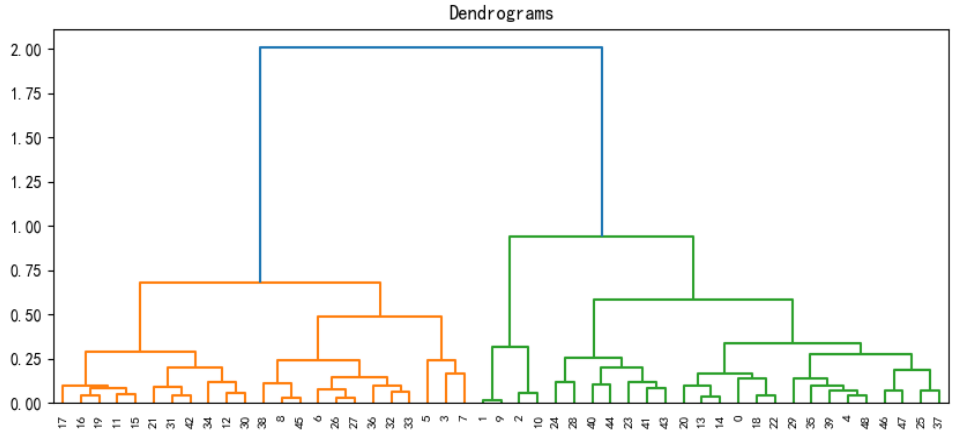


图 9 铅钡玻璃层次分析树状图

对树状图的纵坐标选定一个阈值，以该阈值为分界线进行亚分类的划分，此处处于控制亚分类数量的目的，设置其阈值为 0.75，据此阈值对整体数据进行划分，将划分后的结果再次运用决策树模型进行运算划分方法。运算结果如图 10 所示：

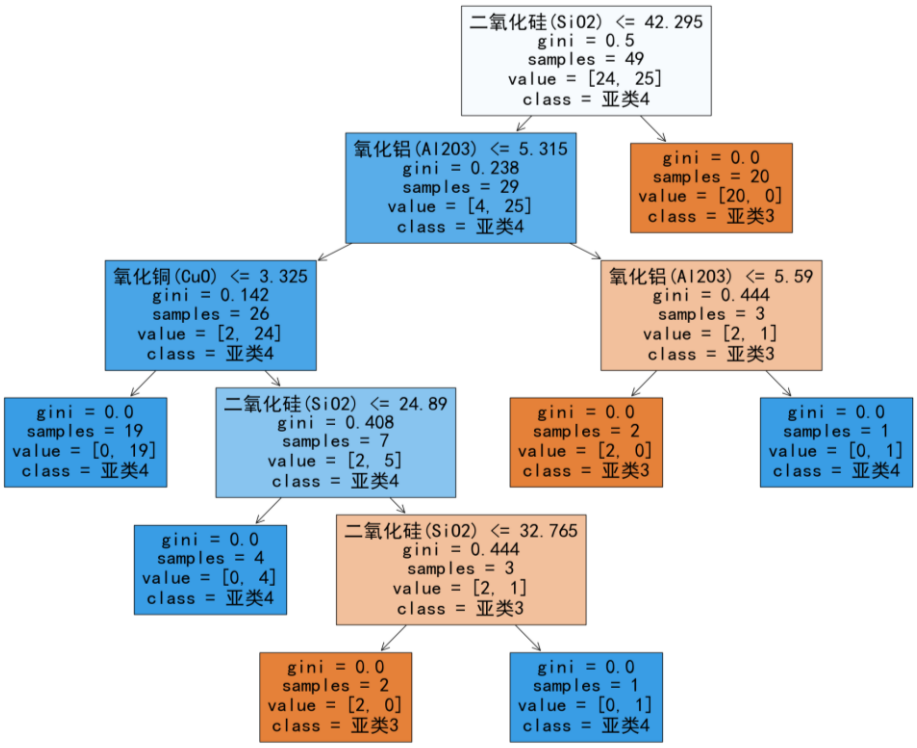


图 10 铅钡玻璃决策树可视化

因此可知划分方法为依据玻璃文物中二氧化硅、氧化铜的含量，故划分结果可见表 10：

表 10 铅钡玻璃文物划分结果

类型	文物编号	备注
类型 1	30、34、36、38、39、40、41、43、50、51、52、54、56、57、58	SiO2≤42.295%，CuO≤3.325% 大多数纹饰为 C，无 B 纹饰，颜色基本均为蓝绿色系
类型 2	8、11、19、24、26、43	SiO2≤42.295%，CuO≥3.325% 纹饰六个 C，仅有一个 A，颜色大多数为紫色
类型 3	1、3、7、9、10、12、18、22、23、25、27、28、31、32、33、35、37、42、45、46、47、55	SiO2≥42.297，纹饰 ABC 分布均匀，颜色基本全为蓝色系

5.2.4 分类结果合理性及敏感性分析

通过对数据加入随机噪声，然后再去通过模型去重新去推测划分依据，我们通过对每个特征进行测试，取出高钾的化学元素列，变 5%的幅度，通过多轮的实验结果，可得出模型依然能够正确的做出判断。对于铅钡类型的数据，加入同样比例的随机噪声，模型依然能够正确的作出判断，故模型鲁棒性较强，不易受异常数据影响。

在本数据中，针对铅钡玻璃，如表 11 所示，其所有数据 SiO2 的平均值为 49.68%，中位数为 52.36%，其中位数与均值较为接近，且 SiO2 划分比例为 42.295%，较为接近平均值与中位数的数值，因此此划分数值基本处于数据中心，对数据划分效果明显，故该分类结果也相当具有合理性。

表 11 铅钡亚分类依据化学成分数据比较

	平均值	中位数	划分依据
SiO2	49.68%	52.36%	42.295%
CuO	1.9%	0.9%	3.325%

5.3 问题三模型的建立与求解

5.3.1 未知玻璃文物化学成分分析模型的建立与求解

在该问题中，需要根据表单 2 中的已分类玻璃文物的化学成分比例对表单 3 中未知类别的玻璃文物化学成分进行分析，并对类型进行鉴别，此问题可作为预测问题进行解决，可以理解为通过已分类玻璃文物化学成分和类型，结合未分类玻璃文物化学成分对其类型进行预测。

针对预测问题，由于总数据量较小，对数据进行预测容易出现过拟合现象，随机森林其随机性较高，不易出现过拟合，因此可采用随机森林模型对数据进行运算处理。

以已分类玻璃文物的化学成分比例作为测试集，在测试集中，选取各个采样点数据作为样本，并对其中的特征选取最佳特征作为分裂点，逐层进行分裂得出对玻璃文物化学成分起较大影响的特征，以此为依据对玻璃文物类型进行初步判定，随机森林中多决策树同时对类型进行判定，以投票方式可得出最终所属类型的鉴定，具体算法流程图见图 11。

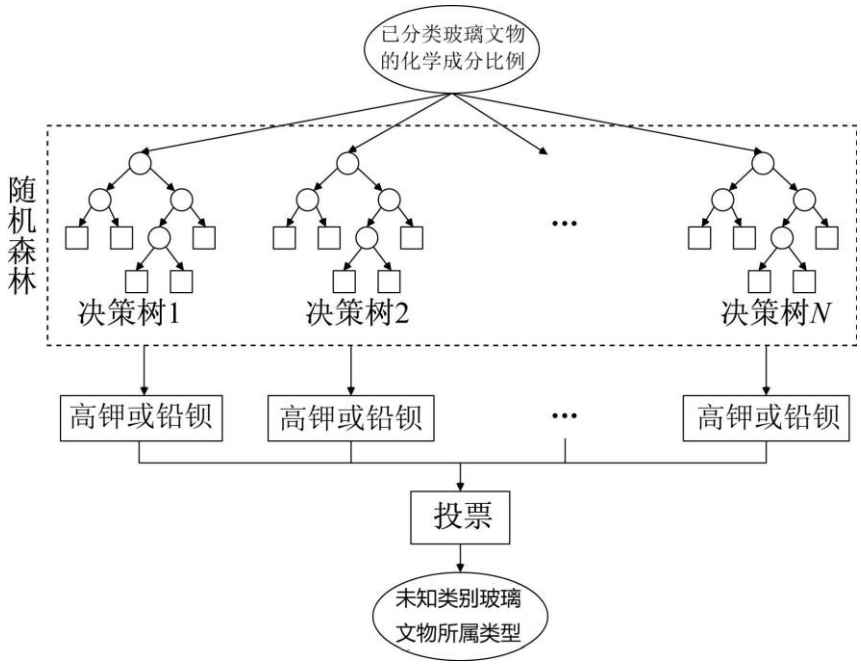


图 11 随机森林模型流程图

随机森林模型在训练过程中，默认采用不纯度减少的方法对特征重要性排序，，进而优化特征空间。在决策树构建时采用基尼指数选择最优特征，基尼指数表示在样本集合中随机样本被分错的概率，基尼指数越小，集合的不纯度越低，即样本被分错的可能性更小；反之，样本被分错的可能性越大。样本集合 D 的基尼指数定义为：

$$G(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}{p_k}^2 \tag{9}$$

将表单 3 中数据在此理论基础上进行测试，得出未知玻璃文物所属类型如表 11 所示：

表 11 未知玻璃文物的编号与其类型

文物编号	类型
A1	高钾

A2	铅钡
A3	铅钡
A4	铅钡
A5	铅钡
A6	高钾
A7	高钾
A8	铅钡

5.3.2 分类结果的敏感性分析

对敏感性进行分析，首先要对数据中 14 个化学成分比例数值进行变化，并对数值变化后的整体数值再次进行类型的预测。首先选取一组数据，对数值的变化范围分别选取 3%、5%、10%，变换单个样本的数据，其余数据均不改变，分别用随机森林模型进行预测，模型依然能够正确的预测，预测结果如表 12 所示。

表 12 局部数据变化后最终预测结果

	未变化	变化 3%	变化 5%	变化 10%
predict	[1,0,0,0,0,1,1,0]	[1,0,0,0,0,1,1,0]	[1,0,0,0,0,1,1,0]	[1,0,0,0,0,1,1,0]

其中 1 代表高钾，0 代表铅钡，当对局部数据进行变化后不影响总体预测结果，当变化比例达到 10%仍具有高稳定性，因此可推断该模型稳定性强且合理性高。

5.4 问题四模型的建立与求解

5.4.1 不同类别玻璃文物化学成分的关联关系模型的建立与求解

在该问题中，依据表单 2 中数据针对高钾和铅钡玻璃文物两种类型，分别分析其化学成分之间的关联关系。针对此问题，先将不同化学成分的含量趋势做分析，由于化学成分比例在风化时会产生变化，不同化学成分可能会产生交叉影响，且样本数据量较少，故采用系统分析中常用的灰色关联度分析法。

对此问题利用初值法，首先确定原始分析序列，令 14 个化学成分相继作为母序列，然后对其他变量序列进行无量纲化处理，处理完成之后对变量序列首项 x_1 与其余各项 x_k 依次求差：

$$\Delta_i(k) = |x_1(k) - x_i(k)|;\ i = 1,2,3... \tag{10}$$

进而对两极差进行计算：

$$M = \max_i \max_k \Delta_i(k) \tag{11}$$

$$m = \min_{i} \min_{k} \Delta_i(k) \tag{12}$$

最后可计算关联系数：

$$\Upsilon_{1i}(k) = \frac{m + \xi M}{\Delta_i(k) + \xi M} \tag{13}$$

故关联度为：

$$\Upsilon_{1i} = \frac{1}{n} \sum_{k=1}^{n} \Upsilon_{1i}(k) \tag{14}$$

经上述运算后，可得高钾玻璃和铅钡玻璃两种类别分别与各化学成分之间的关联度，对该数据绘制热力图进行可视化结果如下：

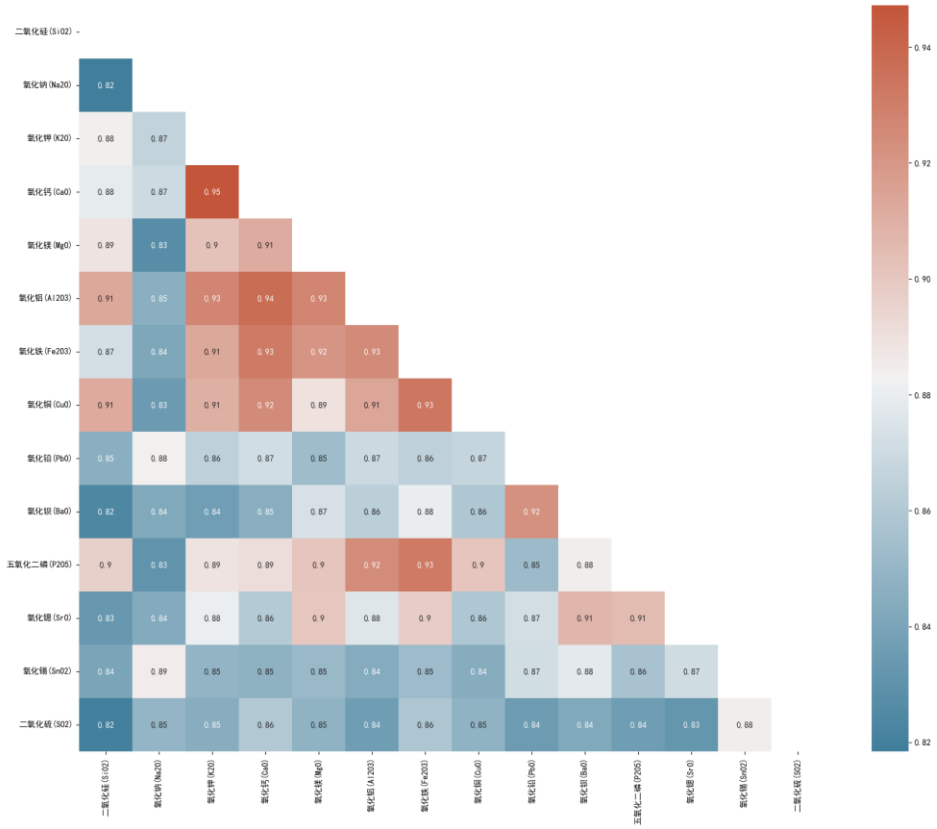


图 12 高钾玻璃文物中不同化学成分之间的关联度

由上图分析可得，高钾玻璃文物中各化学成分的关联度在 0.82~0.95 之间，表明各个化学成分之间有较强的相关性，其中氧化钙和氧化钾之间的关联程度最大，其次是氧化铝和氧化钙。

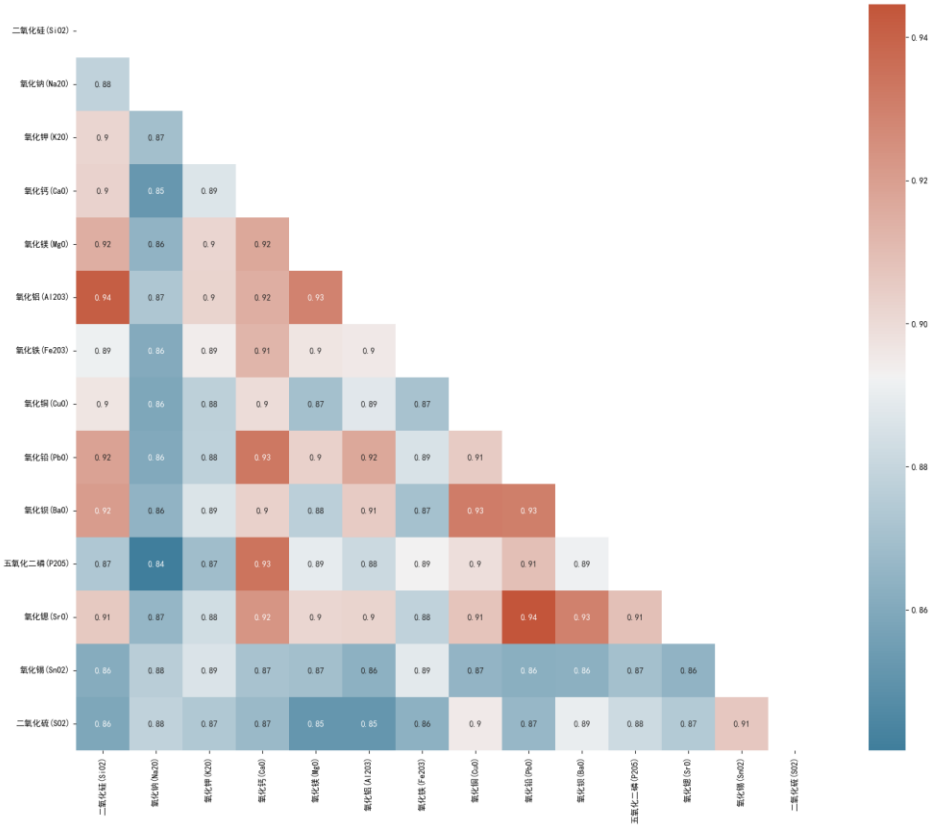


图 13 铅钡玻璃文物中不同化学成分之间的关联度

由上图分析可得，铅钡玻璃文物中各化学成分的关联度在 0.84~0.94 之间，表明其化学成分之间具有较强的相关性，其中氧化铝和二氧化硅、氧化锶和氧化铅之间的相关程度最大，其关联度均为 0.94。

5.4.2 比较不同类别化学成分关联关系的差异性

针对此问题，在上述得到高钾玻璃和铅钡玻璃这两种类别中各化学成分之间的关联度数据的基础上，再将高钾玻璃文物中各化学成分的关联度与铅钡玻璃中相应关联度相减，得到相应的关联度差值，以此绘制热力图，如下：

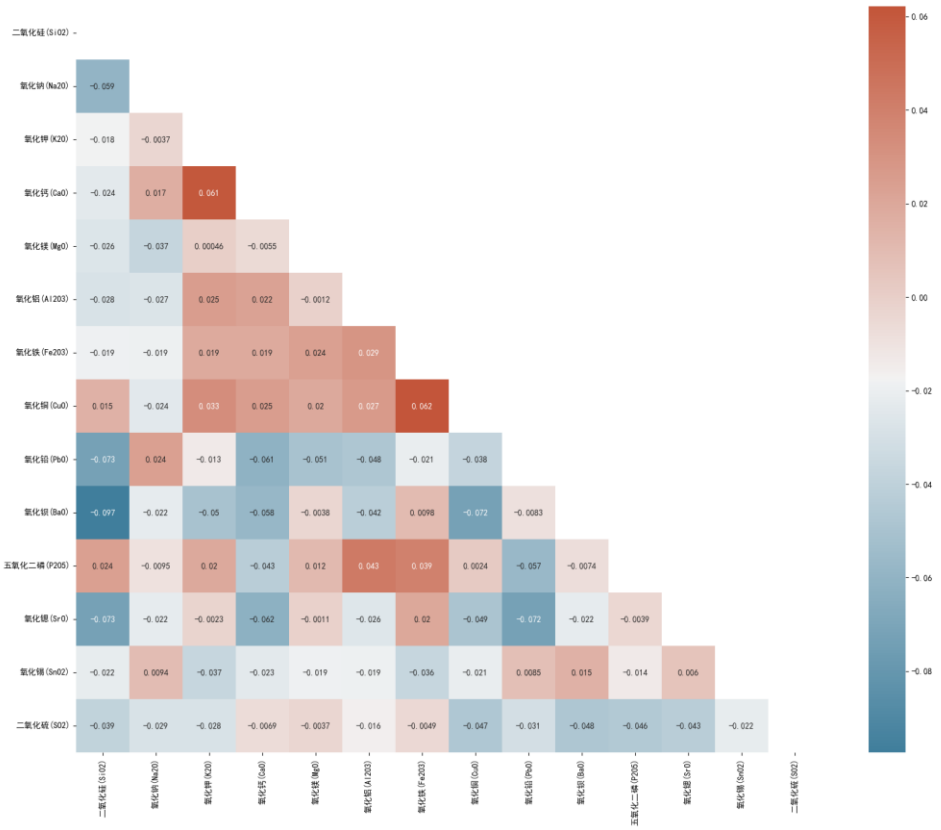


图 14 不同类性之间化学成分的关联度差值

分析上图可得，在高钾和铅钡这两种类型的玻璃中，相应化学成分关联度差值在-0.012~0.062 之间波动，由此可得出不同类型玻璃中化学成分之间关联关系的差异值较小，其中氧化铝和氧化镁之间关联关系的差异最小，氧化铜和氧化铁之间的关系差异最大。

六、模型的评价与推广

6.1 模型的评价

6.1.1 模型的优点

问题一中在分析玻璃文物表面与类型、纹饰和颜色的关系中，使用信息熵以及信息增益方法，能够使信息的不确定性降低，提高分析的精准度；其次在分析相应的统计规律中采用归一化方法进行运算，使定量更加准确，同时提高了模型的精度。

问题二中建立的玻璃文物分类模型中，采用五折交叉验证和 LightGBM 算法选出特

征重要性，能够有效地达到特征选择降维的目的，降低模型的空间复杂度[4]，充分挖掘数据信息；选出特征值进行 Logistic 回归，该回归简单易行便于操作，可以有效地解决分类问题，提高了模型的精确度；其次在对模型进行亚类划分时，运用层次聚类分析法，该算法程序容易实现，存储空间小且运算时间短，伸缩性好，便于分析判断。

问题三中对未知玻璃文物根据其化学成分鉴定所属类别建立的模型，运用随机森林算法，该算法引入了随机性，不容易过拟合，且不需要将数据集规范化，训练速度快能够得到特征的重要性排序。

问题四在建立模型中采用灰色关联度分析法，该方法对于样本数据的多少没有要求且不需要具体的分类规律，操作简单可靠，计算量小适合本文小样本数据分析，其结果会更接近于实际。

6.1.2 模型的缺点及改进方向

采用层次分析法存在一定的主观性和盲目性；随机森林算法在建立模型中，取值划分比较多的特征容易对决策造成影响，从而影响模型拟合的效果。

使用层次分析法时，在进行分类之前需要收集多种信息资料，理清决策的目标以及所要采取的方案；需要比较各元素之间的相关程度，相差太大的不能在同一层次比较。

6.2 模型的推广

本文中所涉及的模型，能够合理地分析文物化学成分含量的统计规律并预测风化前的化学成分含量，还可以准确地研究高钾和铅钡玻璃的分类规律并对其进行亚类划分，同时分析不同类别玻璃的化学成分间的关联关系和其差异性以及将未知玻璃文物划分其所属类别，有利于对古代玻璃制品文物风化程度的研究且对出土文物的研究具有深远意义。

参考文献

[1] 陈姝聿.我国古代玻璃的起源和发展[J].文物鉴定与鉴赏,2019(04):44-45.

[2] 刘树鑫,卓裕,李津,田二胜,刘洋,冯毅.基于随机森林算法的配电线路短路故障分类[J].电器与能效管理技术,2020(08):52-57+67.

[3] 郭弘,陈勇明.灰色关联分析模型数据预处理算子的若干性质[J].成都信息工程大学学报,2020,35(02):235-238.

[4] 产胜宁.基于 LightGBM-Gibbs Sampling 的特征选择算法研究[J].现代计算机,2022,28(06):56-59.

附录

问题 1 第一问 01 模型预测缺失值

```
import warnings
warnings.filterwarnings('ignore')

# 导入包
import gc
from sklearn.cluster import KMeans  # 算法
from sklearn.datasets import load_iris   # 数据集
from sklearn.model_selection import train_test_split  # 数据集划分
from sklearn.metrics import accuracy_score   #评估
from sklearn.preprocessing import StandardScaler  # 标准化
from sklearn.model_selection import GridSearchCV  # 交叉验证网格搜索(没用到）
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression,LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import KFold, StratifiedKFold
from sklearn.metrics import roc_auc_score
from sklearn.preprocessing import normalize
import scipy.cluster.hierarchy as shc
import xgboost as xgb
import lightgbm as lgb
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from lightgbm import early_stopping
from lightgbm import log_evaluation
from sklearn.metrics import roc_auc_score, f1_score, cohen_kappa_score, precision_score, recall_score,
confusion_matrix
plt.rcParams['font.sans-serif'] = ['SimHei']  # 中文字体设置-黑体
plt.rcParams['axes.unicode_minus'] = False  # 解决保存图像是负号'-'显示为方块的问题
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/归一化合并表一二筛选之后.csv",encoding='gb2312')
file1 = pd.read_excel('E:/File/数学建模/2022 数学建模/data/附件.xlsx',sheet_name = 0)
# file2 = pd.read_excel('E://File//数学建模//新建文件夹//C 题/附件.xlsx',sheet_name = 1)
# file3 = pd.read_excel('E://File//数学建模//新建文件夹//C 题/附件.xlsx',sheet_name = 2)
file1 = file1.fillna('NAN')

def chane_col(x):
    dic = {'蓝绿':0, '浅蓝':1, '紫':2, '深绿':3, '深蓝':4, 'NAN':5, '浅绿':6, '黑':7, '绿':8}
```

```
    return dic[x]
file1['颜色'] = file1['颜色'].apply(lambda x:chane_col(x))

feat = [i for i in file1.columns.tolist() if i not in ['颜色','文物编号']]
for i in feat:
    lb = LabelEncoder()
    file1[i] = lb.fit_transform(file1[i])

file1 = file1[['纹饰', '类型','颜色','表面风化']]
train = file1[file1['颜色']!=5]
test = file1[file1['颜色']==5]

# 训练模型
estimator = svm.SVC()
estimator.fit(train[feat],train['颜色'])
# 预测模型
print(estimator.predict(test[feat]))
```

问题 1 第一问 02 计算信息增益

```
file1 = pd.read_excel('E:/File/数学建模/2022 数学建模/data/附件.xlsx',sheet_name = 0)
file2 = pd.read_excel('E:/File/数学建模/2022 数学建模/data/附件.xlsx',sheet_name = 1)

file1 = file1.fillna('NAN')

def chane_col(x):
    dic = {'绿':0, '黑':1, '紫':2, '深绿':3, '深蓝':4, 'NAN':5, '浅绿':6, '浅蓝':7, '蓝绿':8}
    return dic[x]

def chane_wenshi(x):
    dic = {'A':0, 'B':1, 'C':2}
    return dic[x]

def chane_leixing(x):
    dic = {'高钾':0, '铅钡':1}
    return dic[x]

def chane_fenghua(x):
    dic = {'无风化':0, '风化':1}
    return dic[x]

file1['颜色'] = file1['颜色'].apply(lambda x:chane_col(x))
feat = [i for i in file1.columns.tolist() if i not in ['颜色']]
for i in feat:
    lb = LabelEncoder()
```

```
    file1[i] = lb.fit_transform(file1[i])

file1 = file1[['纹饰', '类型','颜色','表面风化']]

# 信息熵
def info_entropy(attr):
    prob = pd.value_counts(attr) / len(attr)   # 对于一个特征不同类所占的比例类
    return sum( np.log2( prob )* prob * (-1) )  # 经验熵

for i in ['纹饰', '类型','颜色','表面风化']:
    print(str(i)+'特征的信息熵:'+str(round(info_entropy(file1[i]),4)))

# 信息增益 （返回值越大，attr1 与 attr2 相关性越强）
def info_gain(dataset, attr1, attr2):
    ent1= dataset.groupby(attr1).apply(lambda x: info_entropy(x[attr2]))
    prob = pd.value_counts(dataset[attr1]) / len(dataset[attr1])
    ent2= sum( ent1 * prob )                # 经验条件熵
    return info_entropy(dataset[attr2]) - ent2    # 信息增益
for i in ['纹饰', '类型','颜色','表面风化']:
#     print(i,info_gain(file1,'表面风化', i))
    print(str(i)+'特征的信息增益:'+str(round(info_gain(file1,'表面风化', i),4)))

# 3.
def infor(data):
    a = pd.value_counts(data) / len(data)
    return sum(np.log2(a) * a * (-1))
# 定义计算信息增益的函数：计算 g(D|A)
def g(data, str1, str2):
    e1 = data.groupby(str1).apply(lambda x: infor(x[str2]))
    p1 = pd.value_counts(data[str1]) / len(data[str1])
    # 计算 Infor(D|A)
    e2 = sum(e1 * p1)
    return infor(data[str2]) - e2
# print("学历信息增益：{}".format(g(file1, "表面风化", "颜色")))
for j in ['表面风化','纹饰', '类型','颜色','表面风化']:
    for i in ['表面风化','纹饰', '类型','颜色','表面风化']:
#     print(i,info_gain(file1,'表面风化', i))
        print(str(j)+'  和  '+str(i)+'  特  征  的  信  息  增  益  :'+str(round(g(file1,j,
i),4)/np.sqrt(round(info_entropy(file1[i]),4)*round(info_entropy(file1[j]),4))))
```

问题 1 第三问 04 预测风化前的化学成分含量相关代码

```
feats = [i for i in data.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点
','Unnamed: 0']]
```

```
data_bei_hou_qian = pd.DataFrame()
data_bei_wei = data_bei[data_bei['表面风化']==0]
data_bei_hou = data_bei[data_bei['表面风化']==1]
dic = {}
for i in data_bei[feats]:
  # 风化前的平均
#   print(i,data_bei.groupby('表面风化')[i].mean().tolist())
#   qian  = data_bei.groupby('表面风化')[i].mean().tolist()[0]

#   # 风化后的平均
#   hou  = data_bei.groupby('表面风化')[i].mean().tolist()[1]
#   # 高钾风化前和风化后的差值
  chazhi  =  data_bei.groupby('表面风化')[i].mean().tolist()[0]-data_bei.groupby('表面风化')[i].mean().tolist()[1]
  # 计算标准化
  scal = (data_bei_hou[i]-data_bei_hou[i].mean())/data_bei_hou[i].std()
#   scal = (data_bei_hou[i]-data_bei_hou[i].mean())/data_bei_hou[i].std()
  # 归一化
  one = (data_bei_hou[i]-data_bei_hou[i].min())/(data_bei_hou[i].max()-data_bei_hou[i].min())
#   one = data_bei_hou[i]/data_bei_hou[i].mean()
#   res = scal*chazhi
#   print(chazhi)
#   print(i,(data_bei[i]-data_bei[i].mean()).mean())
  print(data_bei)
  data_bei_hou_qian[i] = data_bei_hou[i]+scal*one/10+chazhi
#   print(scal*one/10)
# data_bei_hou_qian = data_bei_hou_qian.fillna(0)
# #  data_bei_hou_qian.to_csv('E:/File/数学建模/新建文件夹/问题 1-3 铅钡风化前化学成分含量.csv',index=False)
# data_jia_hou_qian.to_csv("E:/File/数学建模/2022 数学建模/data/问题一预测的铅钡.csv")
print(data_bei_hou_qian)
```

问题 1 第二问 03 柱状图

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/合并表一二筛选之后.csv",encoding='gb2312')

data_jia = data[data['类型']=='高钾']
data_bei = data[data['类型']=='铅钡']

feats = [i for i in data.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]

feats = [i for i in data.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]
```

```
plt.figure(figsize=(50,30))
j = 0
for i in data_jia[feats]:
    j+=1
    plt.subplot(3,5,j)
    x = data_jia.groupby('表面风化')[i].mean().tolist()
    y = ['无风化','风化']
    sns.barplot(x=y,y=x)
#     plt.bar(y,x)
#     plt.title(i)
    # 设置刻度字体大小
    plt.xticks(fontsize=45)
    plt.yticks(fontsize=45)
#     # 设置坐标标签字体大小
        plt.xlabel(i,fontsize=45)
```

问题 2 第一问 01 利用 LightGBM 查看特征重要性

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/合并表一二筛选之后.csv")
data = data.drop(columns=['文物采样点','Unnamed: 0'])
feat = ['颜色','纹饰','类型','表面风化']
for i in feat:
    lb = LabelEncoder()
    data[i] = lb.fit_transform(data[i])

data = data.fillna(0)

train = data
feature_names = [i for i in data.columns.tolist() if i not in ['文物编号','类型']]
ycol = ['类型']

model = lgb.LGBMClassifier(objective='binary',
                           boosting_type='gbdt',
                           tree_learner='serial',
                           num_leaves=2 ** 9,
                           max_depth=18,
                           learning_rate=0.1,
                           n_estimators=10000,
                           subsample=0.75,
                           feature_fraction=0.55,
                           reg_alpha=0.2,
                           reg_lambda=0.2,
                           random_state=i,    # 1983,852
                           is_unbalance=True,
```

```
                    # scale_pos_weight=130,
                    metric='auc')

df_importance_list = []

kfold = StratifiedKFold(n_splits=3, shuffle=True, random_state=1983)
for fold_id, (trn_idx, val_idx) in enumerate(kfold.split(train[feature_names], train[ycol])):
    X_train = train.iloc[trn_idx][feature_names]
    Y_train = train.iloc[trn_idx][ycol]

    X_val = train.iloc[val_idx][feature_names]
    Y_val = train.iloc[val_idx][ycol]

    #       print('\nFold_{} Training ================================\n'.format(fold_id + 1))

    lgb_model = model.fit(X_train,
                          Y_train,
                          eval_names=['train', 'valid'],
                          eval_set=[(X_train, Y_train), (X_val, Y_val)],
                          verbose=500,
                          eval_metric='auc',
                          early_stopping_rounds=50)

    pred_val = lgb_model.predict_proba(
        X_val, num_iteration=lgb_model.best_iteration_)

    df_importance = pd.DataFrame({
        'column': feature_names,
        'importance': lgb_model.feature_importances_,
    })
    df_importance_list.append(df_importance)

    del lgb_model, pred_val, X_train, Y_train, X_val, Y_val
    gc.collect()

df_importance = pd.concat(df_importance_list)
df_importance = df_importance.groupby(['column'])['importance'].agg(
    'mean').sort_values(ascending=False).reset_index()
print(df_importance)
```

问题 2 第一问 02 线性回归

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/归一化合并表一二筛选之后.csv",encoding='gb2312')
```

```
feat = ['颜色','纹饰','类型','表面风化']
for i in feat:
    lb = LabelEncoder()
    data[i] = lb.fit_transform(data[i])

train = data[['氧化铅(PbO)','二氧化硅(SiO2)','氧化钡(BaO)','氧化钾(K2O)']]
y = data['类型']

estimator = LogisticRegression()
estimator.fit(train,y)

print(estimator.coef_)    # w1-w4    氧化铅(PbO) 二氧化硅(SiO2)     氧化钡(BaO)  氧化钾

print(estimator.intercept_)

test = data.iloc[[11,22,48,56]][['氧化铅(PbO)','二氧化硅(SiO2)','氧化钡(BaO)','氧化锶(SrO)']]
lab = data.iloc[[11,22,48,56]][['类型']]

estimator.predict(test)
```

问题 2 第二问 03 使用层次集类进行分类

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/归一化合并表一二筛选之后.csv",encoding='gb2312')

data1 = data[data['类型']=='高钾']
data2 = data[data['类型']=='铅钡']

feat = ['颜色','纹饰','类型','表面风化']
for i in feat:
    lb = LabelEncoder()
    data1[i] = lb.fit_transform(data1[i])

feats = [i for i in data1.columns.tolist() if i not in ['纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]
# data.index = [i for i in range(0,49)]
data1.index = [i for i in range(0,18)]

feats = [i for i in data1.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]
# data1 = data1[feats]
# data1[feats]
```

```
data_scaled = pd.DataFrame(data1[feats], columns=data1[feats].columns)


plt.figure(figsize=(10, 4))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/高钾的两个亚类划分化学元素')
plt.show()


# 铅钡层次分析图

feat = ['颜色','纹饰','类型','表面风化']
for i in feat:
    lb = LabelEncoder()
    data2[i] = lb.fit_transform(data2[i])

feats = [i for i in data2.columns.tolist() if i not in ['纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]
data2.index = [i for i in range(0,49)]
# data2

feats = [i for i in data2.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]
# data2 = data2[feats]
# data2[feats]

ta_scaled = pd.DataFrame(data2[feats], columns=data2[feats].columns)


plt.figure(figsize=(10, 4))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(ta_scaled, method='ward'))
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/铅钡的两个亚类划分化学元素')
plt.show()
```

问题 2 第二问 04 使用决策树进行划分

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/亚分类.CSV",encoding='gb2312')
data = data.drop(columns=['文物采样点','Unnamed: 0'])
data_jia=data[data['类型']=='高钾']
data_bei=data[data['类型']=='铅钡']

feat = ['颜色','纹饰','表面风化']
```

```
for i in feat:
    lb = LabelEncoder()
    data_jia[i] = lb.fit_transform(data_jia[i])
data_jia = data_jia.fillna(0)

def chan(x):
    if x==0:
        return '亚类 1'
    else:
        return '亚类 2'

data_jia['label'] = data_jia['label'].apply(lambda x:chan(x))

feats = [i for i in data_jia.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0','label']]
# data = data[data['类型']==1]
X = data_jia[feats]
y = data_jia['label']

# Fit the classifier with default hyper-parameters
clf = DecisionTreeClassifier(random_state=1234)
model = clf.fit(X, y)
# data[feats]

text_representation = tree.export_text(clf)
print(text_representation)

fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(
    clf,
    feature_names=feats,
    class_names=['亚类 1','亚类 2'],
    filled=True
)
plt.show()
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/决策树实现高钾的两个亚类划分化学元素')

feat = ['颜色','纹饰','表面风化']
for i in feat:
    lb = LabelEncoder()
    data_bei[i] = lb.fit_transform(data_bei[i])
data_bei = data_bei.fillna(0)
```

```
def chan(x):
    if x==3:
        return '亚类 1'
    else:
        return '亚类 2'

data_bei['label'] = data_bei['label'].apply(lambda x:chan(x))



feats = [i for i in data_bei.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0','label']]
# data = data[data['类型']==1]
X = data_bei[feats]
y = data_bei['label']

# Fit the classifier with default hyper-parameters
clf = DecisionTreeClassifier(random_state=1234)
model = clf.fit(X, y)
# data[feats]



fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(
    clf,
    feature_names=feats,
    class_names=['亚类 3','亚类 4'],
    filled=True
)
plt.show()
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/决策树实现铅钡的两个亚类划分化学元素')
```

问题 3 第一问 01 使用使用随机森林预测表三

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/归一化合并表一二筛选之后.csv",encoding='gb2312')
test = pd.read_excel('E://File//数学建模/2022 数学建模/data/附件.xlsx',sheet_name = 2)

test = test.fillna(0)
lb = LabelEncoder()
test['表面风化'] = lb.fit_transform(test['表面风化'])

feat = ['颜色','纹饰','类型','表面风化']
for i in feat:
    lb = LabelEncoder()
```

```
    data[i] = lb.fit_transform(data[i])

feats = [i for i in test.columns.tolist() if i not in ['文物编号']]

X = data[feats]
test = test[feats]
y = data['类型']

clf = RandomForestClassifier()
clf.fit(X, y)#训练

print(clf.predict(test))
```

问题 4 第一问 01 使用灰度预测预测相关性

```
data = pd.read_csv("E:/File/数学建模/2022 数学建模/data/归一化合并表一二筛选之后.csv",encoding='gb2312')
feat = ['颜色','纹饰','表面风化']
for i in feat:
    lb = LabelEncoder()
    data[i] = lb.fit_transform(data[i])
data_jia = data[data['类型']=='高钾']
data_bei = data[data['类型']=='铅钡']
feats = [i for i in data.columns.tolist() if i not in ['文物编号','纹饰','类型','颜色','表面风化','文物采样点','Unnamed: 0']]

result =pd.DataFrame(columns=[i for i in range(0,14)])

for tem in range(0,14):
    x=data_jia[feats].iloc[:,:].T
    x_mean=x.mean(axis=1)
    for i in range(x.index.size):
        x.iloc[i,:] = x.iloc[i,:]/x_mean[i]

    ck=x.iloc[tem,:]    # 二氧化挂
    a = [j for j in range(0,14)]
    cp=x.iloc[a,:]    # 剩下所有

    t=pd.DataFrame()
    for j in range(cp.index.size):
        temp=pd.Series(cp.iloc[j,:]-ck)
        t=t.append(temp,ignore_index=True)

    mmax=t.abs().max().max()
```

```
    mmin=t.abs().min().min()
    rho=0.4

    ksi=((mmin+rho*mmax)/(abs(t)+rho*mmax))

    ksi.columns.size
    r=ksi.sum(axis=1)/ksi.columns.size
    result[tem] = r
#     print(tem)
result.columns = feats
# result['类型'] = feats
result_jia = result

result.index = data_jia[feats].corr().index
mask = np.zeros_like(result, dtype=np.bool)   #定义一个大小一致全为零的矩阵 用布尔类型覆盖原来的类型
mask[np.triu_indices_from(result)]= True     #返回矩阵的上三角，并将其设置为 true
plt.figure(figsize=(20,16))
# cmap = sns.choose_diverging_palette()
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(result,square=True, cmap=cmap,mask=mask,annot=True)
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/铅钡的灰度预测相关性')
plt.show()


result =pd.DataFrame(columns=[i for i in range(0,14)])

for tem in range(0,14):
    x=data_bei[feats].iloc[:,:].T
    x_mean=x.mean(axis=1)
    for i in range(x.index.size):
        x.iloc[i,:] = x.iloc[i,:]/x_mean[i]

    ck=x.iloc[tem,:]    # 二氧化挂
    a = [j for j in range(0,14)]
    cp=x.iloc[a,:]    # 剩下所有

    t=pd.DataFrame()
    for j in range(cp.index.size):
        temp=pd.Series(cp.iloc[j,:]-ck)
        t=t.append(temp,ignore_index=True)

    mmax=t.abs().max().max()
    mmin=t.abs().min().min()
```

```
    rho=0.4

    ksi=((mmin+rho*mmax)/(abs(t)+rho*mmax))

    ksi.columns.size
    r=ksi.sum(axis=1)/ksi.columns.size
    result[tem] = r
#     print(tem)
result.columns = feats
# result['类型'] = feats
result_bei = result

result.index = data_jia[feats].corr().index
mask = np.zeros_like(result, dtype=np.bool)  #定义一个大小一致全为零的矩阵 用布尔类型覆盖原来的类型
mask[np.triu_indices_from(result)]= True     #返回矩阵的上三角，并将其设置为 true
plt.figure(figsize=(20,16))
# cmap = sns.choose_diverging_palette()
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(result,square=True, cmap=cmap,mask=mask,annot=True)
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/铅钡的灰度预测相关性')
plt.show()

result = result_jia-result_bei

result.index = data_jia[feats].corr().index
mask = np.zeros_like(result, dtype=np.bool)  #定义一个大小一致全为零的矩阵 用布尔类型覆盖原来的类型
mask[np.triu_indices_from(result)]= True     #返回矩阵的上三角，并将其设置为 true
plt.figure(figsize=(20,16))
# cmap = sns.choose_diverging_palette()
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(result,square=True, cmap=cmap,mask=mask,annot=True)
plt.savefig('E:/File/数学建模/2022 数学建模/图片文件/高钾减去铅钡的相关性')
plt.show()
```