古代玻璃制品的成分分析与鉴别

摘要

玻璃既是人类最早发明的人造材料之一，也曾是最昂贵的材料之一。玻璃极易受埋藏环境的影响而风化，在风化过程中，玻璃化学成分比例及特征发生变化，从而影响对其类别的正确判断。因此，对玻璃制品的成分进行分析与鉴别在考古工作中显得尤为重要。

针对问题一：我们首先通过**Logistic回归**对附件表单1中缺失的文物颜色数据进行填补，预测出4个缺失颜色均为浅蓝。然后根据玻璃的纹饰、类型和颜色属性通过**one-hot编码**重构数据，建立其与风化关系的**岭回归模型**。结果显示，纹饰B的回归系数最高，为**0.251**，该纹饰文物最易发生风化；铅钡与高钾玻璃的回归系数分别为**0.191**和**-0.191**，说明铅钡比高钾玻璃更易风化。其次，考虑到化学成分维度较多，我们通过**主成分分析**分别对高钾、铅钡玻璃的14种化学成分进行降维，利用**皮尔逊相关系数**作出降维后化学成分与玻璃风化间相关系数矩阵。结果显示，高钾玻璃风化后二氧化硅含量显著增加，铅钡玻璃风化后二氧化硅含量降低。最后，我们根据原数据分布特点，用**蒙特卡罗模拟**出呈**正态分布**的风化前后数据，将其用于训练**BP神经网络**，然后通过训练好的模型，预测风化玻璃文物风化前的化学成分含量。

针对问题二：我们将玻璃数据分为定量数据和定性数据，分别建立它们与玻璃分类之间关系的**决策树模型**。结果显示，氧化铅含量低，或氧化钾含量高且氧化锶含量低的玻璃为高钾玻璃；氧化铅含量高，或氧化钾含量低且二氧化硅含量低于**76%**的玻璃为铅钡玻璃。然后，我们分别对高钾和铅钡玻璃所有采样点依据14种化学成分进行**系统聚类分析**，并根据**肘部法则**确定出两种玻璃最优聚类数量均为**4**，再通过**Kmeans**进行亚分类分析，依据硅、铝、钾等五种元素将高钾玻璃分为高钾玻璃风化、待风化、中性、活泼金属类；依据硅、铝、磷等5种元素将铅钡玻璃分为铅钡玻璃风化、未风化、金属、非金属类。最后，我们改变聚类数量并计算各聚类中心的距离，结果显示我们的模型对聚类数量的改变具有较高的灵敏性。

针对问题三：我们首先对未知类别的8种文物进行数据侧写，分析其化学成分含量。然后以**XGBoost**、**随机森林**、**Adaboost为基分类器**，建立**Majority Voting集成学习算法**，对A1到A8玻璃文物类别进行分类。结果显示A1、A6、A7为高钾玻璃，A2、A3、A4、A5、A8为铅钡玻璃。随后我们对集成分类器进行灵敏度分析，发现树种类采样比例较低时，A5小概率被分为高钾玻璃，其余情况下分类精度较高，说明集成分类器分类效果好，灵敏度较高。

针对问题四：我们通过**斯皮尔曼相关系数**分别对高钾、铅钡玻璃化学成分之间的关系进行分析，并通过python将相关系数矩阵可视化为**热力图**。通过分析热力图不同区域颜色深浅以及颜色种类，结果显示，对于高钾玻璃，酸性氧化物（二氧化硅）和碱性氧化物（氧化钾等）含量增减呈反比关系；对于铅钡玻璃，氧化钙和氧化铅之间呈负相关关系，氧化镁和氧化铝之间呈正相关关系。最后，通过对比两种玻璃热力图的色系分布和颜色深浅，结果显示高钾玻璃化学成分之间的相关性相对铅钡玻璃更强。

关键词：岭回归　神经网络　决策树　聚类分析　Majority Voting集成学习

一　问题重述

1.1　问题背景

玻璃历史源远流长，既是人类最早发明的人造材料之一，亦曾是最昂贵的材料之一。玻璃和大气作用发生侵蚀称为风化[1]，古代玻璃极易受埋藏环境的影响而风化，例如玻璃埋在土中（如古墓内的葬品）所受的侵蚀。在风化过程中，玻璃内部元素与外部环境元素进行反应交换，导致其成分比例发生变化，从而影响对其类别的正确判断，所以，对玻璃化学成分以及类别的分析具有重要意义。

1.2　问题的提出

现在考古工作者收集了一批我国古代玻璃制品的相关数据，并且依据这些文物样品的化学成分和其它检测手段已经将其分为高钾玻璃和铅钡玻璃两种类型。

基于上述背景我们需要建立数学模型解决以下问题:

问题一需要解决三个问题，首先，我们要分析玻璃文物表面风化与其类型、纹饰和颜色的关系，其次是要结合玻璃的类型，分析玻璃表面是否风化化学成分含量的统计规律，最后根据风化点的检验数据，去预测其风化前的化学成分含量。

问题二我们需要根据附件数据去分析高钾玻璃、铅钡玻璃的分类规律，并对两种玻璃的化学成分再次进行亚类划分，给出具体的划分方法以及划分结果，同时分析最终分类结果的合理性和敏感性。

问题三我们需要利用前面两问建立的模型，对附件表单3中未知类别的玻璃文物化学成分进行分析，从而鉴定其所属的的类型，并对最终的分类结果做敏感性分析。

问题四我们需要针对不同类别的玻璃文物样品，分析其化学成分之间的关联关系，同时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

本问我们需要解决三个问题，首先是分析玻璃文物表面风化与其类型、纹饰和颜色的关系。发现原数据中存在颜色数据缺失，我们考虑通过多元Logistics回归对附件表单1中编号为19、40、48、58的文物颜色缺失信息进行补全，然后简要统计不同玻璃类型、不同纹饰以及不同颜色的表面风化情况，再建立基于独热编码的岭回归模型，根据岭回归系数的大小分析玻璃风化与其类型、纹饰和颜色的关系。其次，我们分别考虑高钾玻璃和铅钡玻璃两种类型，对玻璃风化程度与其他化学成分含量的统计规律进行分析。由于化学成分维度较多，我们对两种类型玻璃的12种氧化物含量分别使用主成分分析进行降维，根据累积贡献率确定主成分个数，随后利用皮尔逊相关系数得出风化程度与降维后化学成分之间的相关性矩阵，推测出风化统计规律。最后，我们先对照表单1与2，确认各检测点风化情况。分别考虑高钾玻

璃和铅钡玻璃，将风化后数据作为输入，风化前数据作为输出，拟合神经网络模型。然而对于高钾玻璃，由于数据量较少，模型训练数据不足，我们考虑通过蒙特卡洛模拟数据；此时发现不同文物同一化学成分含量分布大致呈正态分布，因此我们根据原风化前后数据的方差和均值模拟多组服从正态分布的风化前后数据，多次训练神经网络模型。最后通过训练好的神经网络模型，以待预测风化文物风化后化学成分含量作为输入，预测其风化前的化学成分含量。

2.2 问题二分析

问题二已知高钾、铅钡玻璃各化学成分含量以及它们的纹饰、颜色、是否风化属性，首先题目要求探讨两种玻璃的分类规律。我们考虑到化学成分含量为定量数据（连续型数据），属性为定性数据（离散型数据）；因此分别对定量数据与玻璃分类情况，定性数据与玻璃分类情况之间建立决策树模型。在定量数据中，通过决策树模型判断玻璃分类指标的定量依据以及优先级；在定性数据中，通过玻璃多个性质先后对玻璃进行分类。第二小问需要针对高钾玻璃和铅钡玻璃的化学成分进行亚类划分，并给出具体的划分方法以及划分结果。我们先分别对高钾和铅钡玻璃18、48个采样点依据14种化学成分进行系统聚类分析，并根据肘部法则得到最优的聚类数量，再将聚类数量带入Kmeans进行聚类分析，得到最终的亚分类结果。随后，我们再改变聚类数量，根据得到的聚类效果图以及各聚类中心之间的距离分析模型对聚类数量的敏感性。

2.3 问题三分析

本问要求依据不同样本化学成分及风化程度的不同，对附件表单3中8种未知类别玻璃文物进行分类，并对分类结果进行敏感性分析。本问与问题二的区别在于问题二中我们需要根据已有的分类结果提取出分类特征，而此处我们需要利用不同样本的不同特征完成对未知类别文物的分类。相较于问题二中单一的决策树分类模型，我们首先使用基分类器均可以是决策树的XGBoost、随机森林、Adaboost三种集成学习模型完成对集成分类器的搭建，随后由于分类结果为一个二值变量，因此我们利用绝对多数投票法完成集成分类器对分类结果的确定。随后，我们结合玻璃文物具体化学成分对分类结果进行解释说明。最后，我们对集成学习模型进行敏感性分析。

2.4 问题四分析

问题四针对不同类别的玻璃文物进行分析，我们将文物分为高钾玻璃、铅钡玻璃两类，分别探讨它们化学成分之间的关系。考虑到化学成分之间不一定为线性关系，我们采用斯皮尔曼相关系数对化学成分之间的相关关系进行衡量。可以通过相关系数正负判断：当两个化学成分相关系数为正时，其中一个含量增加时，另一个含量增加；相关系数为负时，一个增加则另一个减少。并且可以通过相关系数大小来判断化学成分之间关联性强弱。与此同时，可以考虑采用热力图（双色系）的方式可视化相关系数矩阵，通过颜色的对比判断正负相关，通过颜色的深浅判断相关性强弱。最后，针对差异性，我们通过对比高钾玻璃和铅钡玻璃两个热力图的颜色深浅和颜色分布，分析它们化学成分关联关系之间的差异性。

三　模型基本假设与合理性说明

假设1：玻璃文物表面风化不会改变玻璃的纹饰。玻璃风化只是玻璃内部元素与环境元素进行反应交换，导致其成分比例发生变化，而玻璃纹饰是玻璃固有属性，风化不会对其造成较大改变。

假设2：玻璃文物正常局部检测点的情况可以反映出文物整体化学成分特征。由于缺少文物整体化学成分的组成情况，我们假设除了严重风化点和未风化点外，其余检测点能代表文物整体平均化学成分组成。

假设3：玻璃文物化学成分的改变只是由风化引起的。玻璃与环境之间的元素交换是由多种化学物理反应共同作用的，为了剔除其它因素的影响，简化模型，这里我们只认为玻璃文物化学成分的改变是由风化引起的。

假设4：同一玻璃文物不同部位的检测点是相互独立的样本。玻璃不同部位与环境的作用程度不同，同一个玻璃上可能存在风化和未风化的部位，所以不同部位的检测点是相互独立的。

四　符号说明

符号	含义
ω_i	权重系数
R	拟合优度
ρ	相关系数
K	聚类类别数量
Gin	基尼指数
P_k	样本点属于第K类的概率
chem (x_i)	第i种化学成分含量

五　问题一建模与求解

5.1　数据预处理

数据预处理是在主要的处理前对数据进行的一些处理，有利于更好的解答和分析问题。本部分将对附件表单一中的玻璃文物的基本信息进行初步的观察，获得玻璃文物的表面风化与其玻璃类型、纹饰和颜色的大致关联分析。

5.1.1　遗漏数据的补全

在附件表单1中，我们可以发现其中编号为19、40、48、58的文物颜色信息缺失，对数据的统计有较大影响，所以我们在SPSS中，以纹饰、类型、表面风化为自变量，颜色为预测值，通过多元 logistics 回归对四个缺失值补全，最后，我们得到4个缺失颜色均为浅蓝。具体玻璃文物信息如下表:

表 1: 玻璃文物信息表

文物编号	纹饰	类型	颜色	表面风化
19	A	铅钡	浅蓝	风化
40	C	铅钡	浅蓝	风化
48	A	铅钡	浅蓝	风化
58	C	铅钡	浅蓝	风化

5.1.2 不同玻璃类型的表面风化情况

古代玻璃制品的主要区别是在玻璃炼制过程中所添加的助熔剂不同，本文中玻璃分为铅钡玻璃和高钾玻璃，其中铅钡玻璃是在烧制过程中加入铅矿石，其氧化铅和氧化钡的含量较高；高钾玻璃则是以含钾量高草木灰作为助熔剂烧制。下图是不同玻璃类型的表面风化情况：

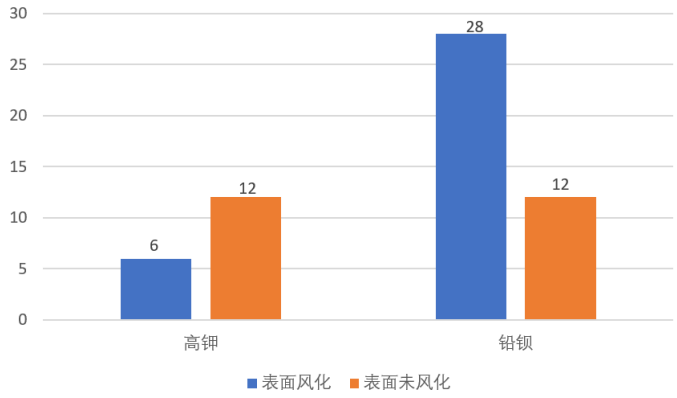


图 1: 不同玻璃类型的表面风化情况

从图中我们可以看出，在采集的58个玻璃文物中，相较于高钾玻璃，铅钡玻璃的表面更容易风化，这是由于当铅钡玻璃表面吸附水时，水中的氢离子与玻璃中的氧化铅和氧化钡等碱性金属进行反应，形成的碱性金属氢氧化物进一步侵蚀玻璃[1]。

5.1.3 不同纹饰的表面风化情况

玻璃文物能明显的看出纹饰，本文纹饰主要分为三种，分别命名为A、B、C，下图是不同纹饰的表面风化情况：

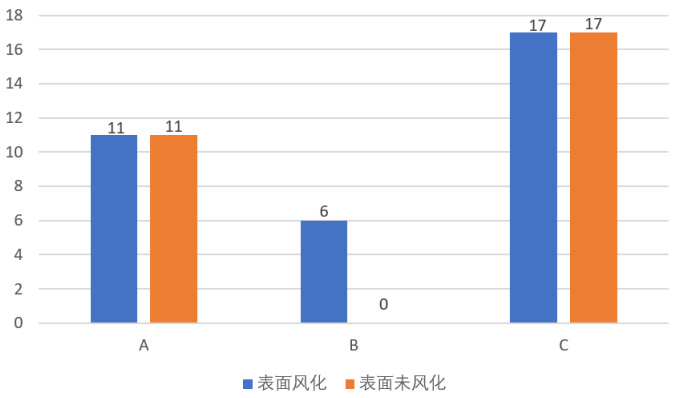


图 2: 不同纹饰的表面风化情况

从图中我们可以看出，在采集的58个玻璃文物中，A、C纹饰中表面风化和未风化的数量相同，而B纹饰所有玻璃表面都已风化，所以可以初步判断，A、C纹饰对于玻璃表面风化并没有显著影响，而B纹饰的玻璃表面更容易风化。

5.1.4 不同颜色的表面风化情况

玻璃文物易受到埋藏环境的影响而风化，在风化过程中，玻璃表面的颜色会随着风化程度而发生改变，例如，表面大面积灰黄色区域为风化层，是明显风化区域，而紫色部分是一般风化表面。不同颜色的表面风化情况如下图:

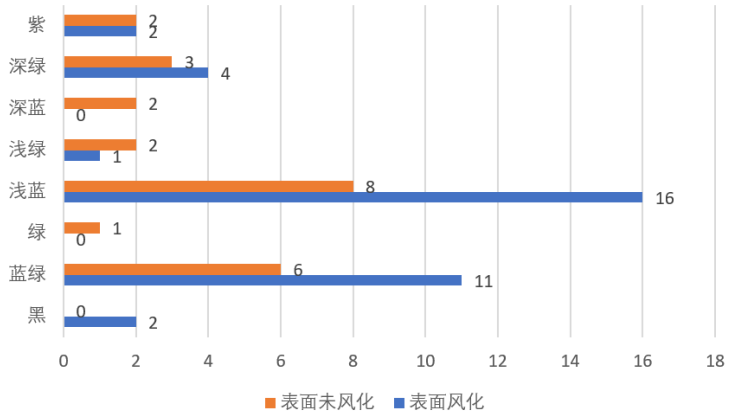


图 3: 不同颜色的表面风化情况

从图中可以发现，在采集的58个玻璃文物中，玻璃的紫色部分表面风化与未风化数量一致，所以紫色部分是一般风化表面；而深绿、浅蓝、蓝绿以及黑色部分风化表面的数量大于未风化，所以这些颜色部分是明显风化区域；最后，对于深蓝、浅绿以及绿色部分，表面未风化数量大于风化数量，所以这些颜色部分可初步断定为不明显风化区域。

5.2 基于独热编码的岭回归模型建立

本文建立基于独热编码的岭回归模型，对这些玻璃文物的表面风化与其玻璃类型、纹饰和颜色的关系进行分析。

5.2.1 基于独热编码的数据重构

独热编码，又称$one-hot$向量，对于其一个属性而言，在一任意维度的向量中，仅容许一个单元为 1，其他单元必须为 0。玻璃文物除风化情况外包含三个属性：纹饰，类型以及颜色。对于每个文物的数据，我们对其三个属性分别做$one-hot$向量并进行向量拼接，每组数据重构为其对应的独热编码。

纹饰	类型	颜色
C	高钾	蓝绿

重构

A	B	C	高钾	铅钡	黑	蓝绿	绿	浅蓝	浅绿	深蓝	深绿	紫
0	0	1	1	0	0	1	0	0	0	0	0	0

图 4: 数据独热编码转换示意图

经过此处理，独热编码数据可以更突出其中某个特定颜色、某个纹饰或者某个类型对表面是否风化的影响。例如，当文物纹饰为B类时，文物均为风化，则文物纹饰类型对文物是否风化影响大；通过独热编码可以突出维度B对是否风化的影响。

以下为表头的数据处理结果，所有数据见附录A。

表 2: 部分数据重构结果

编号	A	B	C	铅钡	高钾	浅绿	浅蓝	深绿	深蓝	紫	绿	蓝绿	黑	风化
01	0	0	1	0	1	0	0	0	0	0	0	1	0	0
02	1	0	0	1	0	0	1	0	0	0	0	0	0	1
03	1	0	0	0	1	0	0	0	0	0	0	1	0	0
04	1	0	0	0	1	0	0	0	0	0	0	1	0	0
05	1	0	0	0	1	0	0	0	0	0	0	1	0	0

5.2.2 岭回归分析文物风化情况与各属性之间关系

1. 岭回归原理

岭回归，是标准线性回归的基础上增加$L2$正则化项，通过增加惩罚项，用于在估计中加入偏差，从而得到更好的估计。同时解决过拟合问题，增强对存在数据点数据的拟合。岭回归的损失函数在线性回归的基础上增加正则化项$\lambda\|\theta\|^2$，假设数据特征为n，样本个数为m个，则得到损失函数如下：

$$J(\theta)=\frac{1}{2}\sum_{i=1}^{m}\left(\left(h_\theta\left(x_i\right)-y_i\right)^2+\lambda\sum_{i=1}^{n}\theta_i^2\right) \tag{1}$$

其中：

$$h_e(x)=\theta_0x_0+\theta_1x_1+\ldots+\theta_nx_n \tag{2}$$

损失函数$J(\theta)$对各θ_i求偏导，经过求解可以得到回归系数的计算方式（矩阵形式）如下：

$$\theta=\left(X^TX+\lambda I\right)^{-1}X^Ty \tag{3}$$

其中λ为岭系数，I为单位矩阵，X为样本特征数据，y为风化数据，θ为回归系数矩阵。通过该式子得到回归系数。

2. 岭回归求解与结果分析

我们将玻璃属性数据（重构后数据）作为样本特征数据X，玻璃是否风化作为结果数据，经过岭回归得到如下结果：

表 3: 玻璃风化情况岭回归模型系数

A	B	C	铅钡	高钾	浅绿	浅蓝
-0.162	0.251	-0.021	0.191	-0.191	0.016	-0.006
深绿	深蓝	紫	绿	蓝绿	黑	
-0.043	0.028	-0.058	-0.136	0.062	0.084	

随后计算均方误差mse，得到误差大小为0.28414，误差较小，表明岭回归准确率高，回归效果较好。与此同时，由于风化时$y=1$，当回归系数为正时，说明当该属性存在时更容易出现风化的情况；反之更容易出现未风化的情况。因此，从回归系数中可以看出：

(a) B的回归系数最高，即当纹饰为B时，玻璃文物最高概率为风化；对应原数据观察发现，纹饰B的所有玻璃文物均为风化。

(b) 其次，铅钡的回归系数较大，说明玻璃类型为铅钡时，玻璃文物为风化的概率高。对应原数据发现，当文物为铅钡类型时，80%的文物是风化的。同时，我们发现高钾的回归系数负值较大，意味着当玻璃类型为高钾时，风化的概率低。因此对于类型而言，我们认为铅钡比高钾更容易风化,与铅钾玻璃的实际机理情况相符合[1]。

(c) 颜色为黑，蓝绿时玻璃相对高概率为风化，对应原数据发现黑色时文物均风化。

(d) 对于属性回归系数相对较小的文物而言，可以根据综合各个属性的回归结果更加准确地判断文物实际的风化情况。

5.3 风化化学成分的统计模型建立

5.3.1 数据预处理

观察附件表单2可知，每个样本采样点均对应着14种类型的氧化物含量，且存在较多的空白缺失值。由于空白处表示未检测到该成分，因此所有的空缺值在接下来统一用0补齐。

由于检测手段等原因，文物采样点 15和17中各成分累加和分别只有79.47%与71.89%，不在85%到105%的有效区间内，与理想情况有较大误差，故将这两个采样数据删除。

玻璃整体风化情况被鉴定为风化，但存在采样点为未风化点时，将未风化的采样点视作未风化数据来处理。

5.3.2 主成分分析

风化作用可以通过水、空气等节理通道与玻璃发生直接接触，发生水化、氧化反应，产生众多类型的氧化物[2]。因此，该玻璃的风化程度可以由玻璃表面特殊类型氧化物含量的高低直观地反应出来。而附件表单2中文物采样点所对应的氧化物种类高达14种，成分较多，不利于体现风化化学成分，因此我们采用主成分分析来对氧化物种类进行降维处理。

以文物采样点与氧化物含量构建样本矩阵，建立主成分分析的流程图如5，同时，为了研究玻璃类型对风化化学成分含量规律的影响，我们将高钾、铅钡两种类型的样本分离，分别进行主成分降维，互相作为对照组，以探究不同类型的玻璃风化化学成分的不同。

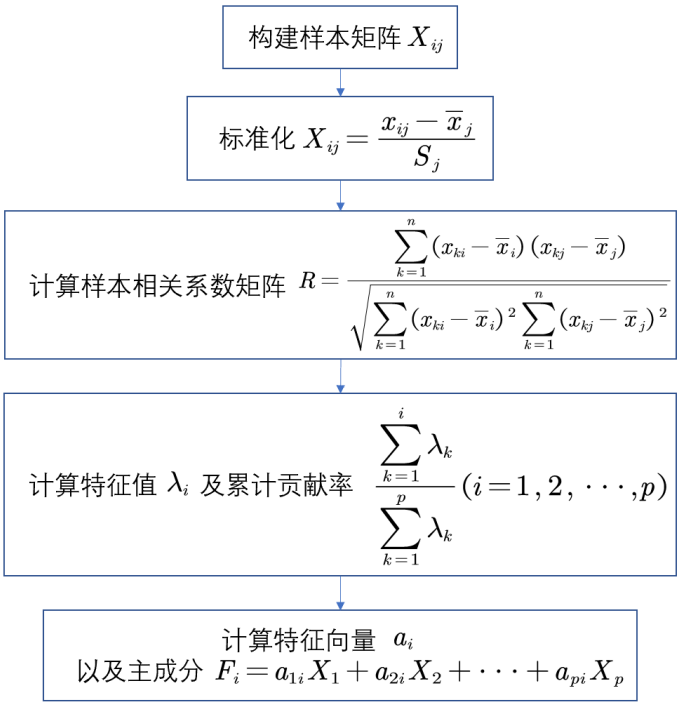


图 5: 主成分分析的流程图

对于高钾类型玻璃，通过$Matlab$求解，我们得到了主成分贡献率随主成分个数的分布。其中在主成分个数为5时，主成分累计贡献率达到0.9。此时我们认为氧化物的信息已经能被较好地表达出来。

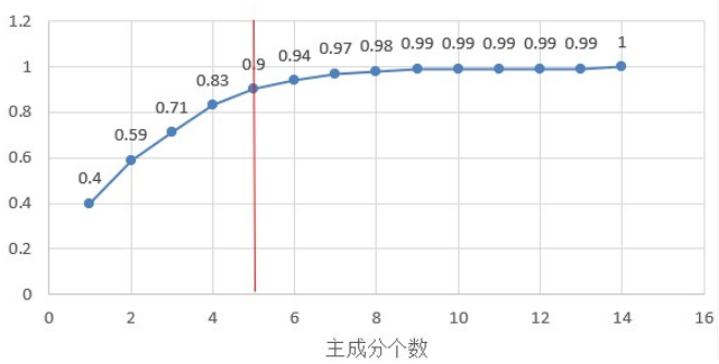


图 6: 主成分累计贡献率

同时计算得到了5个主成分的构成展示如下:

$$
\begin{aligned}
F_1 &= -0.40x_1+0.10x_2+0.30x_3+0.27x_4+0.31x_5+0.37x_6+0.34x_7+0.21x_8\\
&\quad +0.18x_9+0.23x_{10}+0.27x_{11}+0.29x_{12}-0.03x_{13}+0.12x_{14}\\
F_2 &= -0.17x_1+0.49x_2+0.27x_3+0.43x_4-0.27x_5+0.01x_6-0.13x_7+0.05x_8\\
&\quad +0.13x_9-0.23x_{10}-0.36x_{11}-0.33x_{12}-0.19x_{13}+0.11x_{14}\\
F_3 &= -0.08x_1-0.13x_2+0.18x_3+0.07x_4+0.22x_5-0.02x_6+0.02x_7-0.11x_8\\
&\quad -0.52x_9-0.47x_{10}+0.04x_{11}-0.01x_{12}+0.23x_{13}+0.55x_{14}\\
F_4 &= -0.06x_1+0.31x_2+0.24x_3-0.08x_4+0.09x_5+0.12x_6-0.21x_7-0.51x_8\\
&\quad +0.14x_9-0.13x_{10}+0.03x_{11}+0.19x_{12}+0.54x_{13}-0.34x_{14}\\
F_5 &= -0.03x_1-0.27x_2+0.16x_3+0.05x_4+0.19x_5-0.28x_6-0.28x_7+0.21x_8\\
&\quad +0.31x_9+0.34x_{10}-0.42x_{11}+0.02x_{12}+0.40x_{13}+0.29x_{14}
\end{aligned}
$$

分析主成分构成可知，在第一对主成分F_1中，x_6、x_7具有较高的正向载荷，说明F_1主要代表着氧化铝和氧化铁的混合物（$Al_2O_3 \cdot Fe_2O_3$）。第二对主成分F_2中，x_2、x_4具有较高的正向载荷，则意味着F_2主要由氧化钠以及氧化钙的混合物构成（$\mathrm{Na_2O \cdot CaO}$）。F_3中x_5、x_{13}和x_{14}的正向载荷显著，则F_3可以被认为主要由氧化镁，氧化锡以及二氧化硫的混合物组成（$\mathrm{MgO{\cdot}SnOO_2{\cdot}SO_2}$）。$F_4$中由于正向载荷较高的氧化物成分已被前三对主成分解释过，因此此处不作另外考虑。F_5中x_9、x_{10}的正向载荷显著，则F_5可以被认为主要由氧化铅，氧化钡的混合物组成（$PbO{\cdot}BaO$）。F_1、F_2、F_3、F_5各自贡献率比例分别为40%，19%和12%和7%。

故通过主成分分析，对于高钾类型玻璃，我们从原始14类氧化物中挑选出9种氧化物组成了4类混合物，与剩下没有被主成分选中的5种氧化物组合，最终氧化物的类型被降维成9类。

同理，对于铅钡类型玻璃，我们也得到了主成分贡献率随主成分个数的分布图。其中在主成分个数为7个时，累计贡献率达到0.84（铅钡类型主成分累计贡献率表见附录）。同样的，我们可以得到主成分构成关系式。由于选取的主成分个数较多，数据量较大，因此此处仅展示4个主成分的有效部分，详细主成分表达式见附录。

$$
\begin{aligned}
F_1 &= 0.30x_8 + 0.29x_9 + 0.33x_{10} + \cdots \\
F_2 &= 0.21x_1 + 0.27x_2 + \cdots \\
F_3 &= 0.35x_3 + 0.31x_6 + 0.35x_{13} + 0.40x_{14} + \cdots \\
F_4 &= 0.38x_5 + 0.54x_{12} + \cdots
\end{aligned}
$$

分析其成分构成可知，第一对主成分x_8、x_9、x_{10}正向载荷较高，故其应为铜、铅、钡的混合物（$\mathrm{CaO \cdot PbO \cdot BaO}$）；第二对主成分主要为硅、钠混合物（$\mathrm{SiO_2 \cdot Na_2O}$）；第三对主成分主要为钾、铝、锡、硫组成的混合物（$\mathrm{K_2O \cdot Al_2O_3 \cdot SnO_2 \cdot SO_2}$），第四对主成分由镁、锶混合而成（$MgO \cdot SrO$）。

通过主成分分析，铅钡玻璃的氧化物种类也由原来的14类中选取出来11类，组成了新的4类混合物。与剩下没有被主成分选中的3种氧化物结合，最终氧化物的类型被降维成7类。

5.3.3 相关性分析

对高钾类型玻璃，我们有经过主成分降维得到的9种类型的化合物。为了探究风化与化合物成分的规律，我们利用皮尔逊相关系数计算出风化与各类化合物之间的相关系数矩阵(完整相关系数矩阵见附录)。其中风化与各类化合物间的相关系数见下表:

表 4: 高钾风化与各类化合物间的相关系数表

化合物	F_1混合物	F_2混合物	F_3混合物	F_5混合物	
相关系数	-0.81	-0.16	-0.13	-0.16	
化合物	二氧化硅	氧化钾	氧化铜	五氧化二磷	氧化锶
相关系数	0.87	-0.8	-0.28	-0.42	-0.46

可以看出，除了二氧化硅的相关系数为正数以外，其余混合物的相关系数均为负值。结合具体系数分析，对于高钾类型玻璃，风化与二氧化硅具有较强的正相关关系，而对F_1混合物（$Al_2O_3 \cdot Fe_2O_3$）以及氧化钾均具有较强的负相关关系。故我们推测高钾玻璃风化时，其二氧化硅的成分会有大幅度的增加，而氧化钾以及铁铝混合物均会有较大幅度降低。总体而

言，其余混合物含量均呈现不同程度的降低。

对铅钡类型玻璃，我们有降维得到的7种类型的化合物。同样的使用皮尔逊相关系数计算出风化与各类化合物间相关系数见下表（完整相关系数矩阵见附录）：

表 5: 铅钡与各类化合物间的相关系数表

化合物	F_1混合物	F_2混合物	F_3混合物	F_4混合物
相关系数	0.63	-0.43	-0.09	0.02
化合物	氧化钙	氧化铁	五氧化二磷	
相关系数	0.42	-0.08	0.558	

分析相关系数表可知，铅钡类型玻璃风化时，F_1混合物（$CaO \cdot PbO \cdot BaO$）含量明显增多，氧化钙以及五氧化二磷含量略微增加。但F_2混合物（$SiO_2 \cdot Na_2O$）含量会有较大程度的降低。

通过高钾玻璃与铅钡玻璃的对照分析，我们可以总结出，高钾玻璃风化时，二氧化硅含量明显增加，但反之铅钡玻璃风化时，二氧化硅含量则明显降低。与此同时，高钾玻璃风化时，铁、钾、铝氧化物的含量会略微降低，而铅钡玻璃风化时，铜、钡、锡、钙、磷化物的含量均会有不同程度上的增加。

5.4 基于神经网络的风化前化学成分含量预测

由于高钾玻璃和铅钡玻璃在烧制过程中，采用的助溶剂不同，主要化学成分含量差距也较大；因此，我们将高钾玻璃与铅钡玻璃分开预测。根据所有风化后玻璃的化学成分含量以及所有风化前玻璃化学成分含量拟合神经网络，并通过训练好的神经网络预测风化玻璃风化前的化学成分含量。

5.4.1 神经网络原理

Step 1 正向传播 正向传播示意图如下：

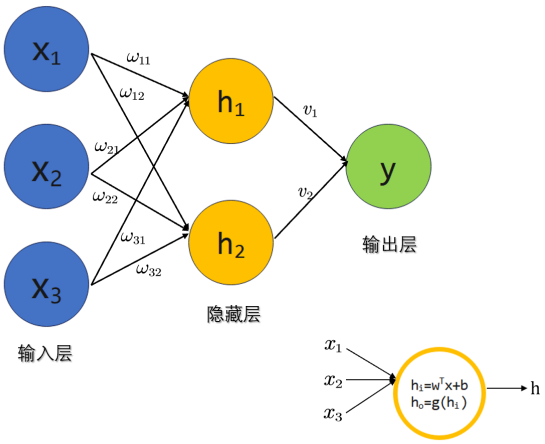


图 7: 正向传播示意图

输入层为风化后的化学成分含量，输出层为风化前的化学成分含量。其中任意两个节点之间的连线为连接权重参数，随机初始化该权重，并在后续不断修正。

Step 2 反向传播 通过反向传播优化权重矩阵参数通过网络输出的值y_0与真实值y之间的误差反向计算出权重改变程度。对系数ω不断进行修正。

Step 3 重复上述两步骤至误差结果小于允许值或者迭代次数到达最大值$epoch$时。每经过一次正向传播和反向传播，会不断更新网络中连接权重，训练完毕后，得到最优权重矩阵ω。

5.4.2 高钾玻璃

- 玻璃文物基本信息

  根据对数据的基本观察我们发现，高钾玻璃中风化的玻璃对应纹饰类型均为B，而风化前均为A,C；而且风化的玻璃普遍为蓝绿色，风化前则存在多种颜色情况，风化前后颜色必然存在变化。因此我们认为，根据原有数据中的纹饰和颜色，都无法直接通过一一对应关系找到相似玻璃，判断某块风化玻璃风化前的数据情况。

  因此，我们考虑通过所有已知的风化后的数据作为输入数据，所有已知风化前的数据作为输出数据，拟合神经网络。再通过训练好的神经网络模型进行预测。

- 数据预处理

  由于不风化的数据为12组，而风化的数据为6组，数据组数较少而且组数无法匹配。因此我们首先观察各个维度（各个化学成分含量）数据分布情况，通过观察可以发现，各个维度数据大致呈正态分布形态。例如，下图为所有铅钡-风化点数据氧化铝含量的分布情况，可以看出，含量比例大致集中分布在均值$\mu = 2.5$附近，因此我们认为同种化合物含量数据大致呈正态分布。

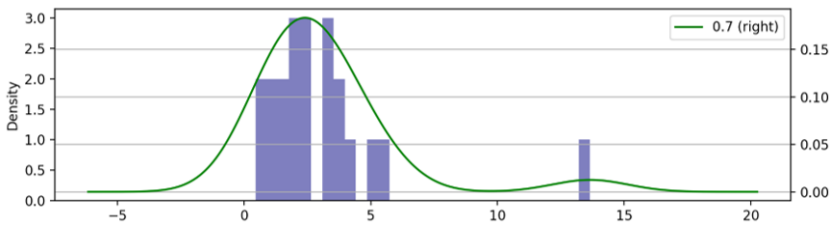


图 8: 铅钡-风化Al_2O_3含量分布图

因此，我们采用蒙特卡洛模拟100组符合正态分布的数据，我们假设各个化学成分含量均服从一维正态分布，即

$$\text{chem}\,(x_i) \sim N\left(\mu_i, \sigma_i^2\right) \tag{4}$$

概率密度公式表示如下式:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5}$$

其中 chem (x_i)为第i种化学成分含量的比例，而μ_i为第i种化学成分含量已知数据的平均值，σ_i为第i种化学成分含量已知数据的方差。例如第1种为SiO_2。

根据已知数据的均值和方差模拟，模拟的风化文物数据中K_2O分布结果如下:

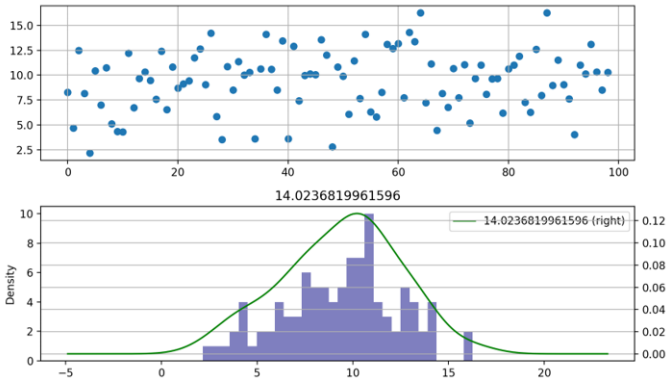


图 9: 模拟数据中高钾-风化K_2O含量分布图

根据已知数据，蒙特卡洛模拟的风化数据中K_2O分布结果如下（由于数据量较大，全部风化数据以及未风化数据见附录A）:

表 6: 三组模拟结果

SiO2	Na2O	K2O	CaO	MgO	Al2O3	Fe2O3	CuO	PbO	BaO	P2O5	SrO	SnO2	SO2
95.28	0	0.57	0.36	0.03	1.71	0.29	3.39	0	0	0.41	0	0	0
93.66	0	0.18	1.34	0	0.95	0.21	0	0	0	0.68	0	0	0
93.92	0	1.31	1.08	0.58	2.18	0.21	1.31	0	0	0.42	0	0	0

5.4.3 铅钡玻璃风化前化学成分预测

铅钡玻璃风化前化学成分预测主要的依据分别有两个，玻璃文物的基本信息以及化学成分比例信息。以下是分别对两个信息进行分析:

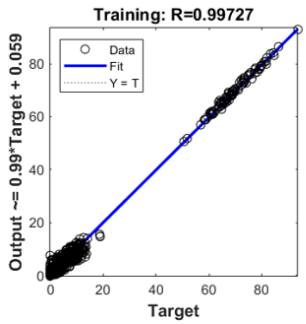
- 铅钡玻璃文物的基本信息
  由附件表2中的数据我们可以发现，铅钡玻璃的纹饰只有两种类型，分别是A和C，并且两种纹饰均有风化和未风化的数据。所以在后续的解题过程中，我们将不同纹饰中风化和未风化数据交叉组合训练，从而减少预测误差。而对于颜色，由于不同的风化程度，对应的颜色部分可能会有所不同，而从附件中我们可以发现，不同的颜色均有风化和未风化的数据，所以为了防止颜色对预测结果造成较大的干扰，我们并不将颜色作为我们预测的依据。

- 铅钡玻璃文物的化学成分比例
  在附件表2中，我们可以得到铅钡玻璃风化和未风化采样点的化学成分比例，其中我们可以观察到二氧化硅在未风化的铅钡玻璃中含量高，风化之后含量减少。同时，铅钡玻璃风化中包含3个严重风化的采样点，分别是编号为08、26、54，由于严重风化的数据较少，所以在对神经网络的训练过程中会引入较大误差，并且，每一个严重风化的编号都有对应的正常风化采样点，即使是剔除这三个严重风化的数据也并不会对训练结果造成较大影响。

5.4.4 含量预测结果

我们将模拟所得100组呈正态分布的输入数据和100组输出数据，取80%为训练集，20%为

验证集，训练得到具有最优权重的神经网络，其拟合优度如下图，可以看出，拟合效果较好，模型合理性高：



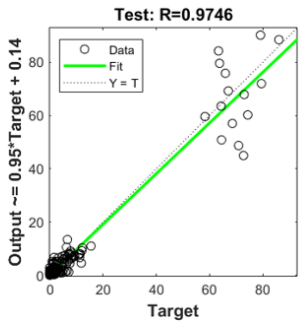
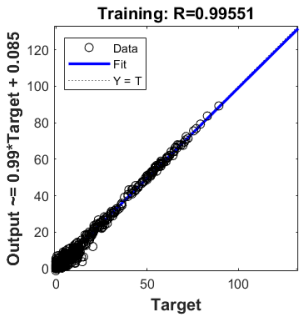


图 10: 高钾玻璃神经网络拟合优度R



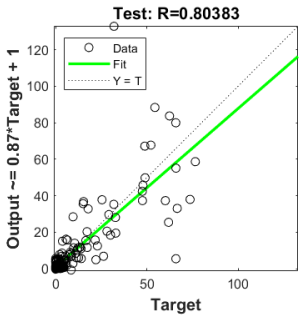


图 11: 铅钡玻璃神经网络拟合优度R

通过训练好的神经网络模型，我们预测风化玻璃风化前的化学成分含量。以实际待预测的风化文物风化后的化学成分含量作为输入数据进行预测。其中，高钾玻璃文物中风化的有6个，预测得到6个文物风化前的化学成分含量如下：

表 7: *6*个文物风化前的化学成分含量

采样点	SiO2	Na2O	K2O	CaO	MgO	Al2O3	Fe2O3	CuO	PbO	BaO	P2O5	SrO
7	75.46	0.42	1.58	0.37	0.99	6.39	2.49	1.12	0.48	0.67	1.48	0.07
9	62	1.09	10.5	5.58	1.06	6.67	1.42	2.48	0.46	0.78	1.61	0.07
10	69.96	0.41	9.77	6.54	1.07	7.76	1.86	3.22	0.52	0.61	1.17	0.07
12	59.82	0.49	10.18	5.04	1.1	6.75	1.77	2.96	0.55	0.63	1.29	0.07
22	61.96	0.56	7.4	3.73	1.22	6.98	3.69	1.81	0.41	0.34	0.96	0.08
27	67.94	0.29	9.92	5.97	1.17	7.97	2.98	2.44	0.3	0.31	1.37	0.07

铅钡玻璃文物中风化的有26个，其中5个文物风化前的化学成分含量如下（所有预测结果见附件A）：

表 8: *5*个文物风化前的化学成分含量

采样点	SiO2	Na2O	K2O	CaO	MgO	Al2O3	Fe2O3	CuO	PbO	BaO	P2O5	SrO
26	37.7	1.68	0.27	0.54	0.62	2.11	0.84	3.37	30.38	17.07	0.31	0.44
2	63.35	0	0.26	2.14	0.85	6.74	0.28	0	36.57	5.78	1.22	0.24
8	32.24	2.1	0.28	0.53	0.62	2.34	0.9	3.66	28.8	17.98	0.22	0.45
8	65.61	1.45	0.31	1.84	0.74	2.84	1.32	1.28	17.51	4.47	1.7	0.34
11	50.07	1.24	0.26	0.56	0.59	3	0.74	1.78	29.25	11.8	0.67	0.35

六　问题二建模与求解

6.1　基于决策树的玻璃类型分类模型

由于玻璃已知数据可分为定性数据和定量数据。例如，颜色和纹饰等属性值为定性数据，而化学成分含量（氧化钡等）为定量数据。我们分别通过定性数据和定量数据两类数据对玻璃类型进行分类，探究高钾玻璃、铅钡玻璃的分类规律。分类采用决策树模型，决策树可以更加清晰地推断出各个成分含量影响的优先级以及各个属性对分类的影响。

6.1.1 决策树原理

生成决策树的过程经历如下步骤:

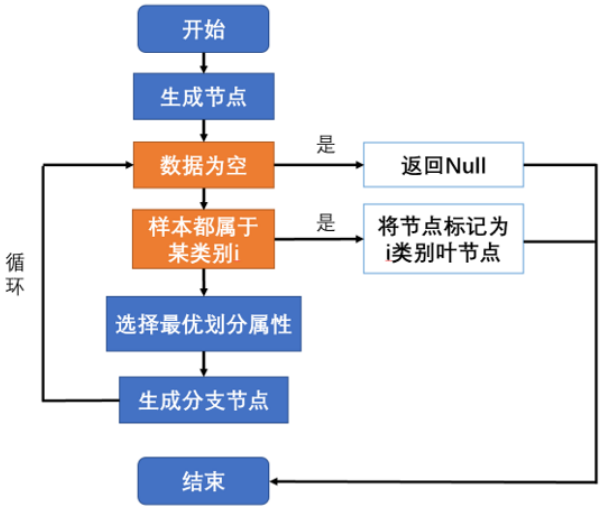


图 12: 决策树流程示意图

对于其中选择最优划分属性的方式，通常采用信息增益比或者基尼系数的方式衡量。本文采用基尼指数作为算法选择划分的依据（CART算法），其中基尼系数计算公式如下:

$$\text{Gin}\,\dot{x}(p) = \sum_{k=1}^{K} p_k \left(1 - p_k\right) = 1 - \sum_{k=1}^{K} p_k^2 \tag{6}$$

其中K为类别个数，P_k表示样本点属于第K类的概率。

6.1.2 定量数据

定量数据，为玻璃文物的化学成分含量数据，包含14个维度的连续型数据，我们将各文物化学成分含量作为X，通过决策树进行分类。第一次决策树分类发现，氧化铅$\mathrm{Pb_2O_3}$指标即可完全划分文物为高钾、铅钡的属性。当Pb_2O_3小于5.46%时，文物为高钾，反之为铅钡。这与实际数据情况高度以及实际文物情况吻合。

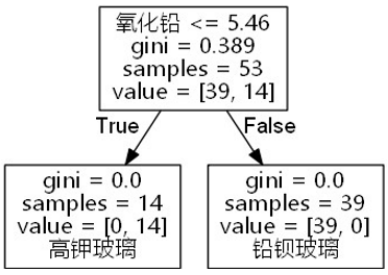


图 13: 决策树定量分类结果

然而，单独依据一个化合物作为分类标准可能存在误差，例如当氧化铅含量位于5.46%附近时，无法仅通过氧化铅判断。因此，我们在第一次决策树分类的结果基础上进行第二次决策树分类。即通过剪去氧化铅这一维度数据的方式，剩余数据作为新的X进行决策树分类，得到结果如下图:

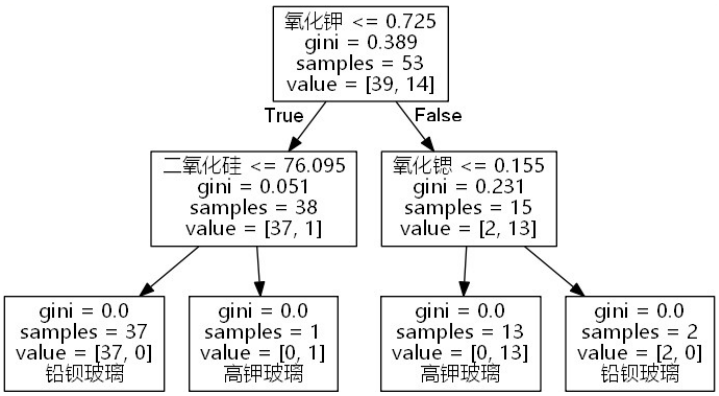


图 14: 决策树定量分类结果

从结果可以看出，第二个分类依据的首要标准为氧化钾。当氧化钾含量低于0.725%时，再判断二氧化硅含量情况，此时如果二氧化硅含量低于76.095%，那么该文物为高钾玻璃，反之则为铅钡玻璃。而当氧化钾含量高于0.725%时，需再判断氧化锶含量情况，此时如果氧化锶含量低于0.155%，则该文物类型为高钾玻璃，反之则为铅钡玻璃。

同理，通过剪去上述维度数据的方式，剩余数据作为新的X再进行决策树分类。由于前两次的维度已经可以作为主要依据完全判断出高钾/铅钡文物的分类情况。因此我们取氧化铅、氧化钾、二氧化硅、氧化锶先后作为定量数据的主要分类指标，后续决策树分类结果见附录A。

6.1.3 定性数据

定性数据，包括颜色、纹饰以及是否风化，均为离散型数据，我们同图4将定性数据重构为独热编码的形式，以0-1变量的形式衡量决策树分类情况。决策树输入数据X重构后的颜色、纹饰以及风化属性数据，输出数据 y 为高钾-铅钡玻璃类型（0-高钾玻璃，1-铅钡玻璃）。预测得到如下结果（数据结果见附件A）：

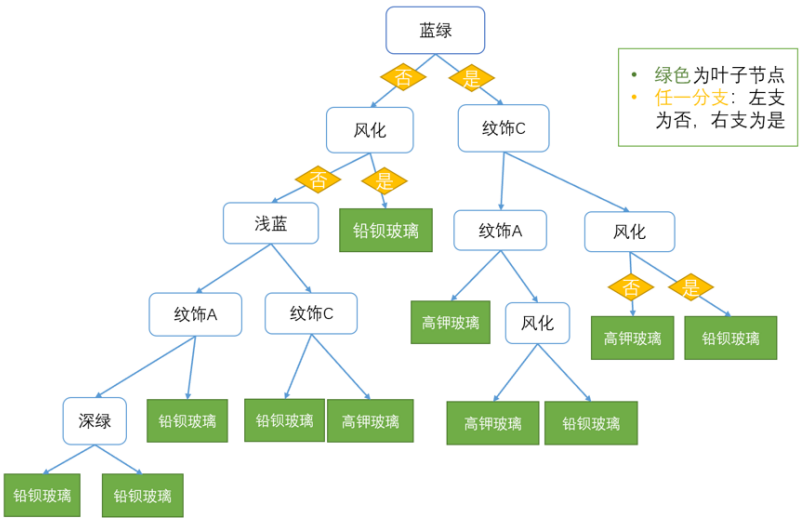


图 15: 决策树定性分类结果

从决策树定性分类结果可以看出，玻璃文物是否为蓝绿色可以作为最优先分类依据。当颜色不为蓝绿色时，再考虑是否风化，若风化则为铅钡玻璃，若未风化则依据后续分类指标

继续分类。当文物颜色为蓝绿色时，再考虑纹饰，若纹饰为C且纹饰不为A，则该文物为高钾玻璃，其它情况同样可依据后续决策树结果继续分类。

6.2 基于系统聚类的Kmeans优化聚类亚类划分模型

6.2.1 系统聚类

系统聚类的合并算法通过计算两类数据点间的距离，对最为接近的两类数据点进行组合，并反复迭代这一过程，直到将所有数据点合成一类，并生成聚类谱系图。

本文中，我们利用$SPSS$进行系统聚类，并得到相应的畸变系数J，通过肘部法则，估计出最优的聚类数量K。其中，肘部法则是通过图形的大致估计出最优的聚类数量，原理是将各个类的畸变系数定义为该类重心与其内部成员位置距离的平方和，作出以总的畸变系数为纵坐标，聚类的类别数为横坐标的折线图，当畸变程度变化显著降低时，所对应的类别数K即为最优的聚类数量。

6.2.2 Kmeans聚类亚类划分模型

Kmeans算法的优点在于算法简单、快速，在处理大数据集时，该算法是相对高效率的。其中，算法流程图如下图所示:

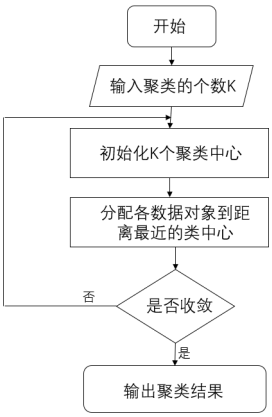


图 16: *Kmeans*算法流程图

但Kmeans的不足之处则在于需要人为事先给出要生成的聚类数目K，但是不同的聚类数目对最后的聚类效果影响很大，所以我们通过前文系统聚类得到的最优聚类数量K作为生成簇，从而减少主观性对分类结果的影响。

所以综上，我们综合附件表单1和表单2分别将两种玻璃类型高钾玻璃和铅钡玻璃依据14种化学成分进行亚类划分，并分析具体的划分方法以及划分结果。

6.3 模型求解

6.3.1 高钾玻璃的亚类划分及合理性分析

我们对高钾玻璃的18个数据进行重新编号，再利用系统聚类对14个化学成分为依据对18个采样点进行系统聚类分析，可以得到聚类谱系图以及聚类系数折线图如下所示:

依据肘部法则，当K=3或者4时，畸变程度变化趋于平缓，这里我们令K=4，即将高钾

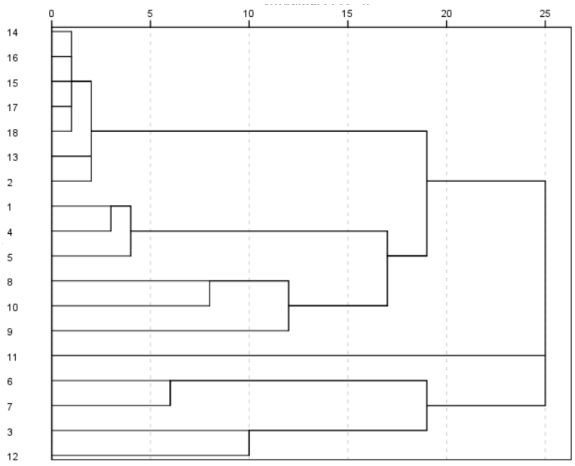


图 17: 高钾玻璃聚类谱系图

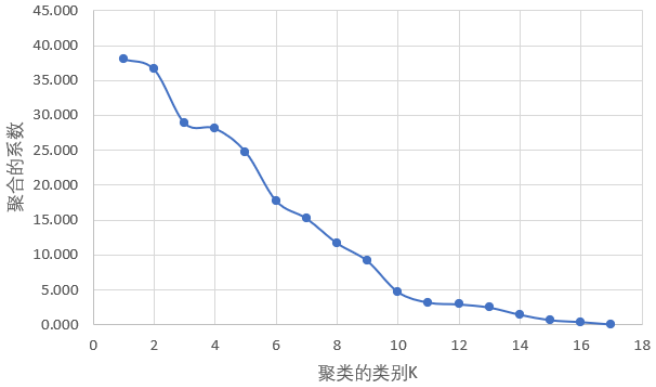


图 18: 高钾玻璃聚类系数折线图

玻璃分为4种亚类划分，由于数据过多，具体分类结果参见附录A，随后对这四种亚类的14种化学成分比例取平均，可以得到下表:

表 9: 高钾玻璃分类表数据展示

类别	二氧化硅	氧化钠	氧化钾	氧化钙	氧化镁
第一类	92.98	0.00	1.21	1.03	0.17
第二类	79.46	0.00	9.42	0.00	1.53
第三类	66.46	0.00	6.86	4.00	1.51
第四类	63.91	1.39	11.66	7.67	0.90

从上表处理的数据可以发现，第一类的二氧化硅含量最高，并且通过数据比对，第一类中的采样点恰好均为高钾玻璃风化采样点，与我们之前所认为风化的高钾玻璃表面二氧化硅含量高一致，所以第一类为高钾玻璃风化类。而第二类中，二氧化硅的含量也较高，而其它的化学成风含量均较低，与我们之前认为风化之后化学成分变化趋势相近，所以有理由认为第二类为高钾玻璃待风化类。而对于第三和第四类，它们的二氧化硅含量均较低，并通过数据比对也可以发现，第三第四类中的采样点均为高钾玻璃未风化点，而第三第四类的主要区别在于第三类的氧化铝、氧化铁等中性金属氧化含量较高；而第四类氧化钾、氧化钙等活泼金属氧化物含量较高，所以可以将第三第四类划分为中性高钾玻璃以及活泼金属高钾玻璃。

6.3.2 铅钡玻璃的亚类划分及合理性分析

我们对铅钡玻璃的48个数据进行重新编号，再利用系统聚类对14个化学成分为依据对48个采样点进行系统聚类分析，可以得到聚类系数折线图如下所示:（由于聚类谱系图谱系较多，故参见附录）

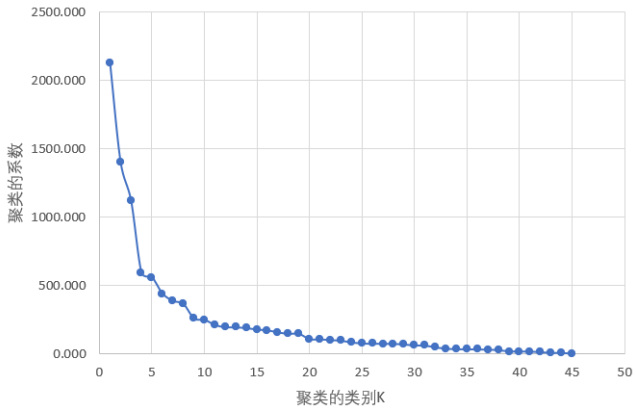


图 19: 铅钡玻璃的聚类折线图

依据肘部法则，当$K = 4$或者5时，畸变程度变化趋于平缓，这里我们令$K = 4$，即将铅钡玻璃分为4种亚类划分，由于数据过多，具体分类结果参见附录A，随后对这四种亚类的14种化学成分比例取平均，可以得到下表:

表 10: 玻璃文物信息表

类别	二氧化硅	氧化钠	氧化钾	氧化钙	氧化镁
第一类	28.56	0	0.18	1.38	0.14
第二类	4.17	0	0.2	3.1	0
第三类	24.67	0.2	0.18	2.9	0.67
第四类	58.17	1.7	0.18	1.31	0.78

从上表处理的数据可以发现，第二类和第四类的二氧化硅含量分别为最多和最少，通过数据比对，我们发现，第二类恰好全为铅钡未风化的采样点，而第四类全为铅钡风化的采样点，与我们之前所认为的铅钡玻璃在经过风化后二氧化硅含量下降的特点一致，所以第二类和第四类分别为铅钡玻璃未风化类和铅钡玻璃风化类。而第一类和第三类的二氧化硅含量相近，但是第三类的二氧化硅含量更少一些，即风化进程更明显，为了减小判别误差，进一步比较其它化学成分含量可以发现，第一类中氧化铝和氧化铜等金属氧化物含量更高，而第三类中五氧化二磷、二氧化硫等非金属氧化物含量更高，鉴于风化过程中金属氧化物含量减少，非金属氧化物含量增加，同时，第三类比第一类风化进程更明显，所以分类符合风化特点规律，即第一类为铅钡金属氧化物玻璃，第三类为铅钡非金属氧化物玻璃。

6.3.3 敏感性分析

根据肘部法则，我们可以得到最优的聚类数量，而无论是铅钡玻璃和高钾玻璃都存在两个畸变程度趋于平缓的点，所以我们通过改变聚类数量，利用Kmeans算法分析不同的聚类效果，作出聚类效果在不同的聚类数量下的敏感性分析。

- 高钾玻璃的亚类划分
  对于高钾玻璃，不同种类中二氧化硅和氧化铝的区别最大，所以我们依据二氧化硅和氧化铝作为最终的聚类效果G图展示，并求出不同聚类数量K=3、4时聚类中心的之间的距离。

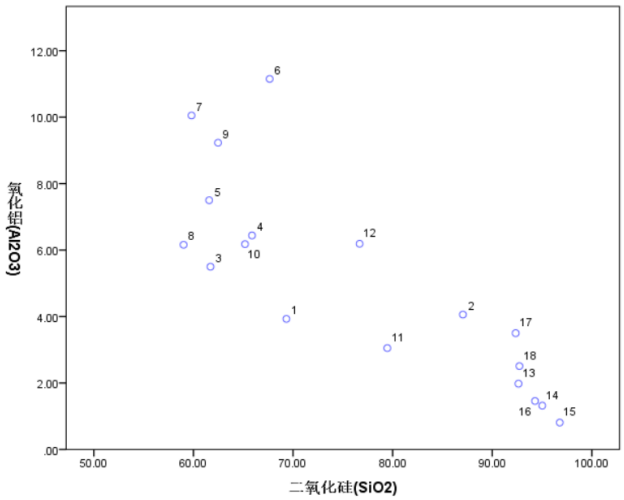


图 20: *K=3*时的聚类效果*G*图

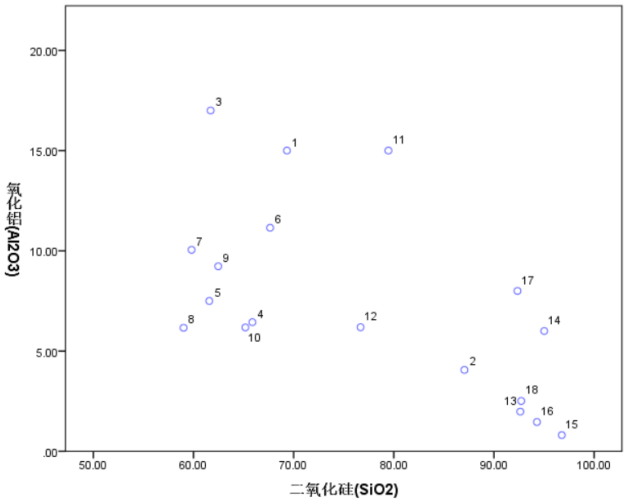


图 21: *K=4*时的聚类效果*G*图

表 11: *K=3*时聚类中心的之间的距离

聚类	1	2	3
1		21.07	13.02
2	21.07		33.52
3	13.02	33.52	

表 12: *K=4*时聚类中心的之间的距离

聚类	1	2	3	4
1		15.74	18.25	11.12
2	15.74		33.52	26.50
3	18.25	33.52		8.25
4	11.12	26.50	8.25	

根据最终的聚类效果图以及聚类中心的距离可得，当聚类数量为3时，聚类中心间的距离较小，而最终的聚类效果较为集中，说明此时聚类结果受到孤立值的影响较小；而当聚类效果为4时，聚类中心间的距离较大，但是聚类效果较为分散，不同的孤立值对最终聚类结果影响较大。所以，我们的模型对聚类数量的划分比较敏感。

- 铅钡玻璃的亚类划分

  对于铅钡玻璃，不同种类中二氧化硅和氧化钡的区别最大，所以我们依据二氧化硅和氧化铝作为最终的聚类效果G图展示，并求出不同聚类数量K=4、5时聚类中心的之间的距离。

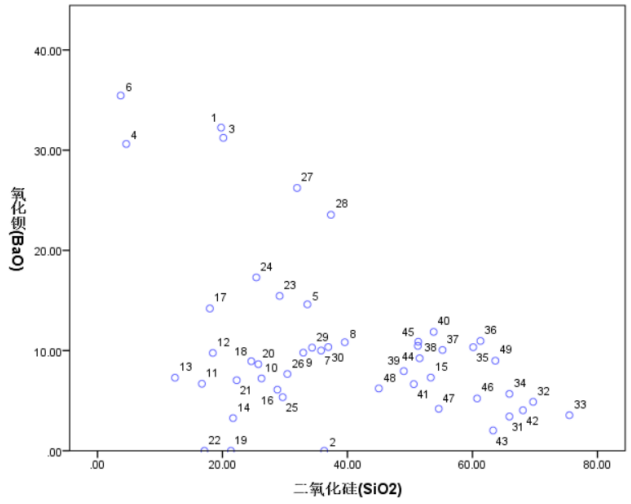


图 22: *K=4*时的聚类效果*G*图

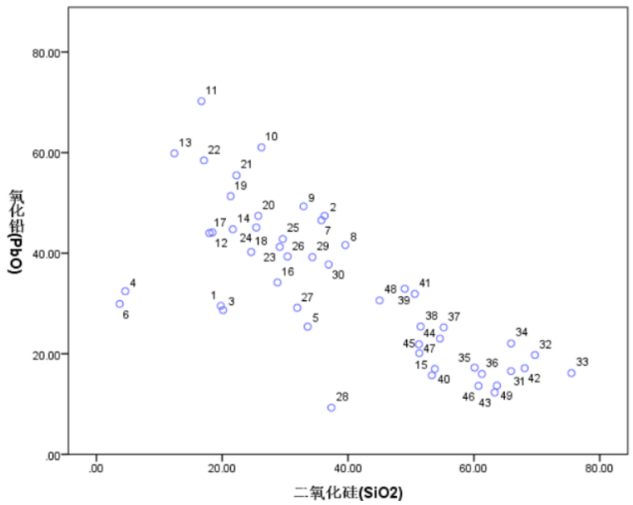


图 23: *K=5*时的聚类效果*G*图

表 13: K=4时聚类中心的之间的距离

聚类	1	2	3	4
1		21.80	34.93	31.93
2	21.80		54.76	42.25
3	34.93	54.76		53.63
4	31.93	42.25	53.63	

表 14: K=5时聚类中心的之间的距离

聚类	1	2	3	4	5
1		58.31	57.28	41.31	24.19
2	58.31		42.25	18.84	37.47
3	57.28	42.25		32.98	35.77
4	41.31	18.84	32.98		19.18
5	24.19	37.47	35.77	19.18	

根据最终的聚类效果图以及聚类中心的距离可得，当聚类数量为4时，聚类中心间的距离较小，而最终的聚类效果较为分散，说明此时聚类结果受到孤立值的影响较小，聚类效果并不太理想；而当聚类效果为5时，聚类中心间的距离较大，但是聚类效果却较为集中，不同的孤立值对最终聚类结果影响较小，聚类结果较为理想。所以，我们的模型对聚类数量的划分比较敏感，并且当聚类数量为5时，可以得到最好的聚类效果。

七　问题三建模与求解

7.1　数据侧写

对附件表单3中未分类文物的化学成分进行总体分析，我们发现A1、A6、A7均具有极高的二氧化硅含量，初步认为符合高钾类型玻璃的特征。而A2、A3、A4、A8二氧化硅含量较低，氧化铅含量较高，初步认为符合铅钡类型玻璃的特征。但A5的二氧化硅含量处于中等偏高水平，同时具有不低的氧化铅和氧化铝含量，因此无法直观地判断出A5所属类型。

7.2　集成学习

附件表单3中提供了8种文物的14种氧化物成分含量以及是否风化相关数据。故我们可以借助附件表单2中已有的样本点各氧化物含量，以及样本点在附件表单1中对应文物的风化数据来作为训练集，以样本点的玻璃类型为目标值，进行二值分类器训练，最后根据附件表单3中各氧化物成分以及风化数据来对未知文物进行分类预测。

在问题二模型的建立中，我们使用了决策树这种基分类器来对玻璃类型进行训练分类，但由于基类模型容易产生过拟合现象，不能很好地利用诸如氧化物含量这类连续性特征，且其较高的方差也使得数据分布的轻微改变很容易造成树结构的不同。而集成学习[4]则是通过构建并结合多个基分类器来进行训练，其得到的模型训练精度通常比其基分类器有着显著提升。为了更精确地对鉴别未知玻璃类别，本文采用$XGBoost$[5]、随机森林、$AdaBoost$三种均可以以决策树为基分类器的集成学习模型来对未知玻璃进行分类。

7.3　基于集成学习的玻璃类型划分

$XGBoost$着眼于串联多个决策树进行共同决策，而对于串联的方式，XGBoost通常采用迭代误差法，即除了第一棵初始的基决策树以外，每次新添加进来的决策树的作用均是为了拟合上一棵树与分类目标值的误差，直到所有决策树的总体误差缩小到一个理想情况以内。

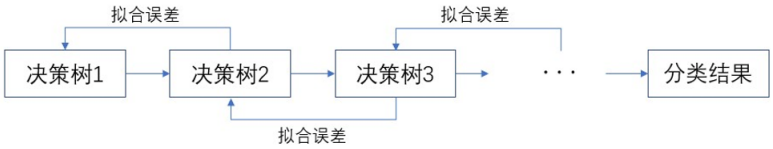


图 24: *XGBoost*原理图

随机森林由多个相互之间没有任何关联的决策树组合而成。对于待分类的目标值，各决策树均并行地对该目标完成分类任务，得到分类结果，而随机森林则是根据多数投票原则，将决策树分类结果中数量最多的目标值作为森林分类结果。

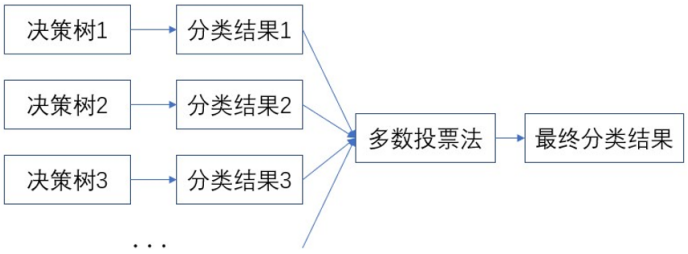


图 25: 随机森林原理图

对于初始时具有相同权重的样本，在多次迭代训练中，*AdaBoost*着重于提高前一轮基分类器错误分类样本的权重，降低正确分类样本的权重，从而在下一次迭代中增加分类成功的概率，使得分类结果被一系列基分类器以分治的形式解决，最终集成分类器为各基分类器及对应权重线性组合而成。

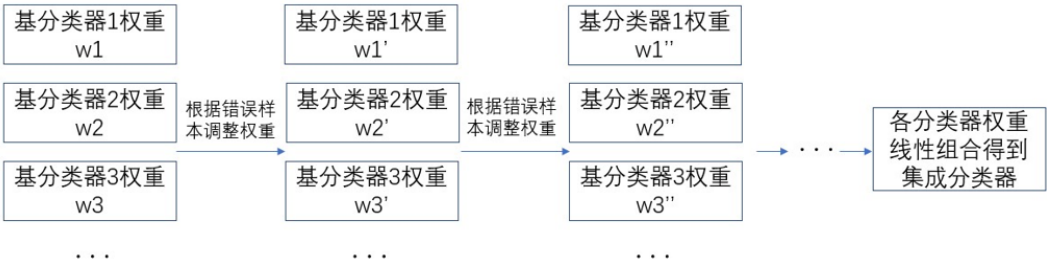


图 26: *AdaBoost*原理图

得到三种集成分类结果后，由于对玻璃种类的分类属于二值分类，我们对三种不同的集成方式采用绝对Majority Voting（多数投票法）[6]来确定最终玻璃的分类结果，即某分类结果的投票超过半数，则最终分类结果为该标记，否则拒绝该分类。我们将附件表单2中的数据作为训练数据，将附件表单3中的数据作为待分类数据进行集成分类，采用绝对多数投票法确定最终分类结果。

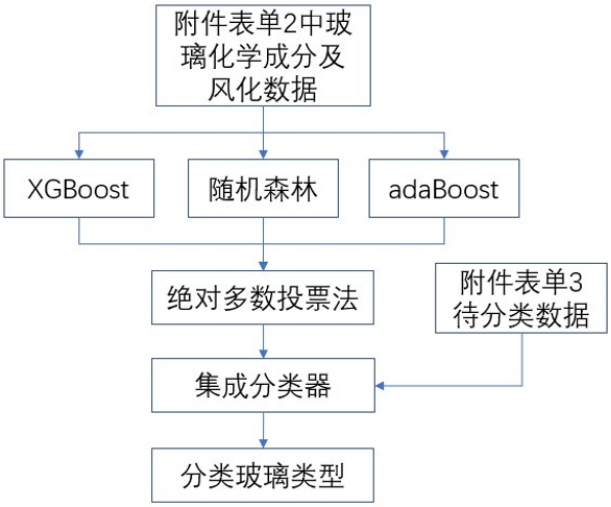


图 27: 多数投票法流程图

通过python编程求解，我们得到了附件表单3中从A1至A8的分类结果如下表所示:

表 15: *A1至A8的分类结果*

编号	A1	A2	A3	A4	A5	A6	A7	A8
分类	高钾	铅钡	铅钡	铅钡	铅钡	高钾	高钾	铅钡

结合具体化学成分及风化程度分析，对于风化玻璃，高钾类型通常具有极高的二氧化硅含量，相反铅钡类型二氧化硅含量极低。而A6、A7两种风化玻璃均具有高达90%的二氧化硅含量且未检测到铅、钡元素，故被分类为高钾类型。同样是风化玻璃，A2的二氧化硅含量只有37.75%，属于较低水平，且其铅元素含量高达34.3%，故A2属于铅钡类型玻璃。而A5的情况较为特殊，其硅含量为64.29%，属于略高的水平，但其不属于高钾风化玻璃90%以上极端高的情况，也不属于铅钡风化玻璃10%到30%的平均水平，不过样本点48具有与A5相似的特征，其硅含量为53.33%，与A5的64.29%接近，且二者氧化铝的含量均为12%到14%之间，故A5被分为了铅钡类型，与样本48同一类。

对于未风化玻璃，A3、A4、A8均具有较高的氧化铅、氧化钡含量，氧化钾含量较低，符合铅钡类型玻璃的特点。A1玻璃中虽然没有检测到钾、铅、钡中的任何元素，但其二氧化硅含量达到78.45%，钙元素含量达到6.08%，均属于较高水平，符合高钾玻璃的特征。

7.4 集成学习敏感性分析

分析结果可知，一般而言，随着树深度越高，产生过拟合现象的概率就越大，但集成分类器的精确度从树深度较低时就一直维持在极高水平，这也印证了集成学习相较于普通分类器而言能够较好地避免过拟合的问题。与此同时，迭代步长与分类结果的关联度也同样较低。而基分类器的树种类以及树特征的采样比例不宜过低或过高，否则均会对训练精度产生影响。

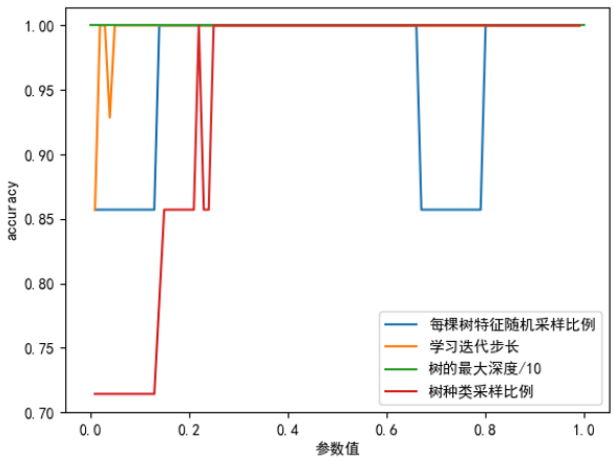


图 28: 集成分类器的精确度

而在树种类采样比例为0到0.3区间内时，我们发现文物A5有极小概率会被分类为高钾类型，甚至有时A1到A8所有文物均会被识别为铅钡类型。第一种情况推测可能是因为A5偏高的二氧化硅含量导致分类器淡化了其对铅钡风化玻璃二氧化硅特征的识别，从而导致其被误判成为高钾类型。而第二种所有文物同时被识别为铅钡类型的情况则可能是因为树采样比例过低导致基分类器拟合效果不佳，从而影响了整体判断精度。

八　问题四建模与求解

8.1　化学成分之间的关联关系

问题四第一部分需要针对不同类别的玻璃文物样品分析化学成分之间的关系。我们将玻璃文物样品分为两种类型：高钾玻璃和铅钡玻璃。分别探究高钾文物中各化学成分之间的关联关系以及铅钡文物各化学成分之间的关联关系。

对于化合物之间的关联关系，我们考虑采用斯皮尔曼相关系数进行探究。由于不同化学成分之间不一定为线性关系，而斯皮尔曼相关系数相比于皮尔逊相关系数更加适合非线性关系，因此它更适合于探究不同化学成分之间的关联关系。计算公式如下：

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} = 1 - \frac{6\sum_n d_i^2}{n(n^2-1)} \tag{7}$$

其中x_i和y_i是i数据的位次，$\bar{x}$和$\bar{y}$是平均位次，d_i是第i个数据对的位次差，n为总的观测样本数。同时，为便于比较不同化学成分之间的相关关系，我们采用热力图的方式可视化相关系数矩阵（数据表格结果见附录A），通过颜色深浅可以判断相关性强弱，通过色系可以判断数据呈正相关或是负相关。

8.1.1　高钾玻璃文物中化学成分的关联分析

高钾玻璃文物的斯皮尔曼相关系数结果如图29。玻璃极易受埋藏环境的影响而风化，玻璃化学成分之间关系与所处化学环境息息相关。从可视化结果可以看出：

- SiO_2与除SnO_2外其余化学成分均的负相关关系（第一行或列数据），这代表着当SiO_2成分含量比例增加时，其余化学成分比例均呈现减少的趋势，反之亦然。该关联关系主要源自于化合物属性：SiO_2为酸性氧化物，而MgO、Na_2O等多为碱性氧化物。当玻璃处于酸性环境中，往往碱性氧化物会与氢离子反应；当玻璃处于碱性环境中，酸性氧化物会与氢氧根离子反应。所以酸性氧化物与碱性氧化物常适应于不同的化学环境，因此二者呈现负相关关系。

- $Mg-Al$以及$Al-Fe$呈较大的相关关系，相关系数分别为0.73和0.75。根据化学知识，我们知道这三个金属的活泼性情况接近[3]，因此它们更容易存在于相同的化学环境下，从而呈现正相关关系。

- $K-Ca$之间相关系数为0.69，非常接近于高相关关系，钾玻璃常同时以草木灰作为助熔剂、石灰石作为稳定剂烧制而成，二者分解后即为K_2O、CaO和共同生成的CO_2，因此$K-Ca$之间相关系数较高。

- SiO_2和其余各化合物之间均为弱相关系数，我们认为SiO_2在文物中含量较少，且难吸附于金属，更多与非金属化合物共同存在，因此与化合物之间成弱相关关系。

- 我们发现热力图的中心部分均为绿色色系，即碱性氧化物之间均呈现中等程度的正相关关系，根据前述分析，我们认为它们均适应于相同的化学环境，所以呈现正相关关系。

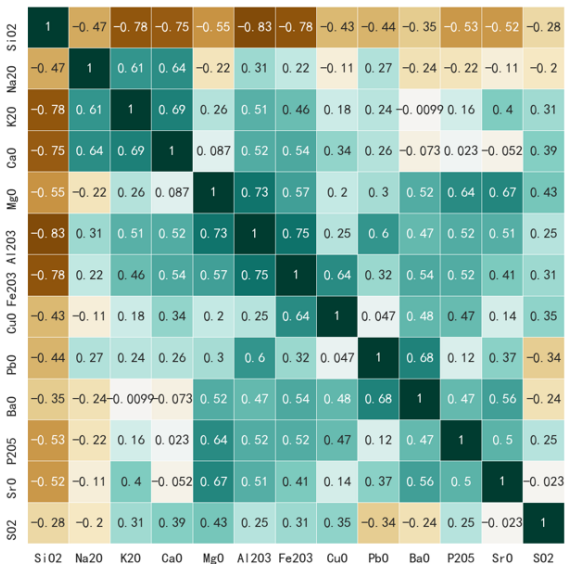


图 29: 相关性热力图-高钾

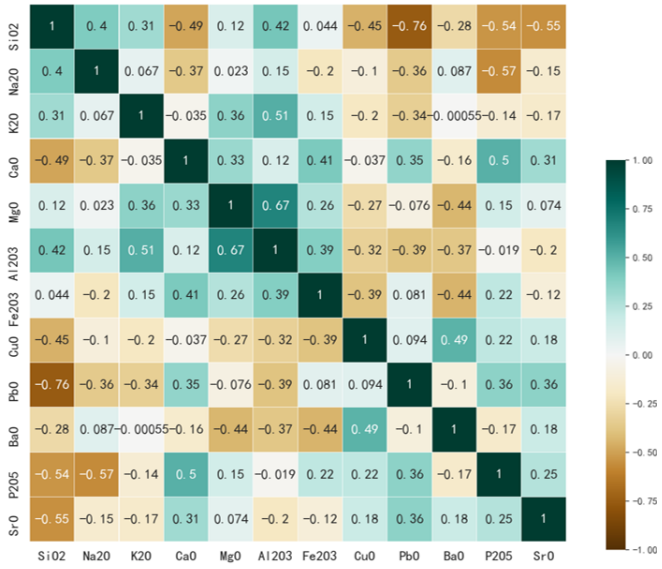


图 30: 相关性热力图-铅钡

8.1.2 铅钡玻璃文物中化学成分的关联分析

铅钡玻璃文物的斯皮尔曼相关系数结果如图30。从可视化结果可以看出：

- $K-Ca$之间呈现中负相关关系，系数为-0.34。由于氧化钙可以提高玻璃的化学稳定性，所以从宋代开始用氧化钙代替钾铅玻璃中的氧化铅[3]，二者呈负相关。

- $Mg-Al$之间相关系数为0.67，由于二者金属活泼性质相近，适应相同环境，呈正相关关系。
- 铅钡玻璃文物中化学成分之间大致均呈现低/中相关关系($\rho < 0.7$),化学成分之间关联关系较弱。

8.2 不同类别化学成分关联关系差异性

通过可视化图的对比我们可以发现如下差异:

- **色系分布:**高钾化学成分相关系数热力图于中间部分主要为绿色，而图的周围黄色分布较多；即金属氧化物之间相关关系主要为正相关，而SiO_2于金属氧化物之间主要为负相关关系。对比之与铅钡玻璃，铅钡玻璃颜色分布相对较为均匀，可见铅钡玻璃化学成分之间的关联性规律相对高钾玻璃较弱。
- **颜色深浅**：高钾玻璃热力图颜色相对较深，铅钡玻璃颜色相对较浅。可以看出高钾化学成分之间相关关系较为紧密，而铅钡化学成分之间相关关系相对较弱，与前述分析印证。我们通过如下公式，对所有相关系数的绝对值取平均值发现，高钾玻璃$\text{Corr}_1 = 0.438$,，铅钡玻璃$\text{Corr}_2 = 0.236$。高钾玻璃整体的相关性相对铅钡玻璃整体的相关性强。

$$\text{Corr}_i = \frac{1}{n} \sum_{k-\Theta_{(1)}} |\rho_k| \quad , i = 1, 2 \tag{8}$$

其中n为相关系数矩阵i元素个数，k为矩阵i的第k 个元素，$\Theta(i)$为矩阵所有元素ρ_k为每个斯皮尔曼相关系数。

九 误差分析

- 岭回归的均方误差较低，只有0.28414，拟合优度较高。高钾、铅钡玻璃主成分的累计贡献率分别为0.9、0.84，能够较好地反应原有成分的信息。高钾玻璃的神经网络拟合优度维持在0.9以上的较高水平，但铅钡玻璃拟合优度只有0.8，仍有提高空间。
- 决策树的分类精度较高，对重要化学成分具有较强的特征提取效果。随着聚类类数变化，亚分类效果也会随之改变。
- 除了训练参数过于极端的情况下，分类结果与预期会有出入以外，其余情况下集成学习的分类效果较决策树更为理想。
- 对所有的化学成分之间的相关性作出了量化，直观精确。

十 模型评价

10.1 模型优点

- **灵敏性高。**可以通过调整亚分类类数，以及集成训练阈值，从而调整分类器分类结果以及训练精度。且不同的分类结果均有据可循，便于解释。

- **精确度高。**使用了多种分类器进行联合分类，相较于普通分类方法，在训练精度上有着较大提升。神经网络的拟合优度较高，预测结果较为合理。
- **特征展现直观。**利用相关性矩阵展现出各个指标间的关联与差异，特征关系一目了然，呈现清晰。决策树的树状分类逻辑图层次清晰，便于发现分类特征的重要程度。
- **求解方式灵活。**利用主成分分析对采样点化学成分降维，使得大量氧化物被转化为少量混合物，便于后续分析和处理，灵活多变。。

10.2 模型缺点

- 对数据依赖性强。由于原始数据并非完全服从正态分布，与我们预测文物未风化前的化学成分时作出的假设有一定偏差，故神经网络训练时会受到影响，容易产生过拟合现象。
- 集成分类器原理复杂，类似于黑箱，难以对其内在分类逻辑进行分析。

参考文献

[1] 王承遇, 陶瑛. 硅酸盐玻璃的风化[J]. 硅酸盐学报, 2003, 31(1):8.

[2] 叶孔凯. 风化作用及节理对饰面用花岗岩矿床的影响[J]. 中国非金属矿工业导刊, 2021(000-005).

[3] 顾亮亮, 周静. 关于中国古代玻璃艺术研究的几个问题[J]. 艺术与设计: 理论版, 2021(4):3.

[4] 金龙. 基于集成半监督学习的标签噪声研究[D]. 西安电子科技大学, 2013.

[5] 王圣洁.基于XGBoost算法的财务造假识别研究[J].山东理工大学学报(自然科学版),2022,36(06):74-79.DOI:10.13367/j.cnki.sdgc.2022.06.012.

[6] 李江宽,林萌. 基于多数投票法的集成学习方法在核电厂故障诊断中的应用[C]//.中国核科学技术进展报告（第七卷）

附录

附录 A

1. 独热编码重构结果

文物编号	A	B	C	铅钡	高钾	浅绿	浅蓝	深绿	深蓝	紫	绿	蓝绿	黑	风化
01	0	0	1	0	1	0	0	0	0	0	0	1	0	0
02	1	0	0	1	0	0	1	0	0	0	0	0	0	1
03	1	0	0	0	1	0	0	0	0	0	0	1	0	0
04	1	0	0	0	1	0	0	0	0	0	0	1	0	0
05	1	0	0	0	1	0	0	0	0	0	0	1	0	0
06	1	0	0	0	1	0	0	0	0	0	0	1	0	0
07	0	1	0	0	1	0	0	0	0	0	0	1	0	1
08	0	0	1	1	0	0	0	0	0	1	0	0	0	1
09	0	1	0	0	1	0	0	0	0	0	0	1	0	1
10	0	1	0	0	1	0	0	0	0	0	0	1	0	1
11	0	0	1	1	0	0	1	0	0	0	0	0	0	1
12	0	1	0	0	1	0	0	0	0	0	0	1	0	1
13	0	0	1	0	1	0	1	0	0	0	0	0	0	0
14	0	0	1	0	1	0	0	1	0	0	0	0	0	0
15	0	0	1	0	1	0	1	0	0	0	0	0	0	0
16	0	0	1	0	1	0	1	0	0	0	0	0	0	0
17	0	0	1	0	1	0	1	0	0	0	0	0	0	0
18	1	0	0	0	1	0	0	0	1	0	0	0	0	0
19	1	0	0	0	0	0	1	0	0	0	0	0	0	1
20	1	0	0	1	0	0	1	0	0	0	0	0	0	0
21	1	0	0	0	1	0	0	0	0	0	0	1	0	0
22	0	1	0	0	1	0	0	0	0	0	0	1	0	1
23	1	0	0	1	0	0	0	0	0	0	0	1	0	1
24	0	0	1	1	0	0	0	0	0	1	0	0	0	0
25	0	0	1	1	0	0	1	0	0	0	0	0	0	1
26	0	0	1	1	0	0	0	0	0	1	0	0	0	1
27	0	1	0	0	1	0	0	0	0	0	0	1	0	1
28	1	0	0	1	0	0	1	0	0	0	0	0	0	1
29	1	0	0	1	0	0	1	0	0	0	0	0	0	1
30	1	0	0	1	0	0	0	0	1	0	0	0	0	0
31	0	0	1	1	0	0	0	0	0	1	0	0	0	0
32	0	0	1	1	0	1	0	0	0	0	0	0	0	0
33	0	0	1	1	0	0	0	1	0	0	0	0	0	0
34	0	0	1	1	0	0	0	1	0	0	0	0	0	1
35	0	0	1	1	0	1	0	0	0	0	0	0	0	0
36	0	0	1	1	0	0	0	1	0	0	0	0	0	1
37	0	0	1	1	0	0	0	1	0	0	0	0	0	0
38	0	0	1	1	0	0	0	1	0	0	0	0	0	1
39	0	0	1	1	0	0	0	1	0	0	0	0	0	1

40	0	0	1	1	0	0	1	0	0	0	0	0	0	1
41	0	0	1	1	0	1	0	0	0	0	0	0	0	1
42	1	0	0	1	0	0	1	0	0	0	0	0	0	1
43	0	0	1	1	0	0	1	0	0	0	0	0	0	1
44	1	0	0	1	0	0	1	0	0	0	0	0	0	1
45	1	0	0	1	0	0	1	0	0	0	0	0	0	0
46	1	0	0	1	0	0	1	0	0	0	0	0	0	0
47	1	0	0	1	0	0	1	0	0	0	0	0	0	0
48	1	0	0	1	0	0	1	0	0	0	0	0	0	1
49	1	0	0	1	0	0	0	0	0	0	0	0	1	1
50	1	0	0	1	0	0	0	0	0	0	0	0	1	1
51	0	0	1	1	0	0	1	0	0	0	0	0	0	1
52	0	0	1	1	0	0	1	0	0	0	0	0	0	1
53	1	0	0	1	0	0	1	0	0	0	0	0	0	1
54	0	0	1	1	0	0	1	0	0	0	0	0	0	1
55	0	0	1	1	0	0	0	0	0	0	1	0	0	0
56	0	0	1	1	0	0	0	0	0	0	0	1	0	1
57	0	0	1	1	0	0	0	0	0	0	0	1	0	1
58	0	0	1	1	0	0	1	0	0	0	0	0	0	1

2. 主成分分析/相关系数结果

1）主成分结果

高钾玻璃主成分：

主成分	F1	F2	F3	F4	F5
SiO_2	-0.39752	-0.17612	-0.08453	-0.06658	-0.03421
Na_2O	0.09759	0.492189	-0.13591	0.310958	-0.27247
K_2O	0.301581	0.275517	0.182669	0.245781	0.167453
CaO	0.275585	0.434578	0.074837	-0.08469	0.050872
MgO	0.310513	-0.271	0.223846	0.096174	0.190576
Al_2O_3	0.374468	0.009113	-0.02436	0.127703	-0.284
Fe_2O_3	0.345623	-0.13372	0.02729	-0.21951	-0.28609
CuO	0.213794	0.052121	-0.11442	-0.51661	0.210587
PbO	0.182719	0.13888	-0.52885	0.141635	0.31572
BaO	0.230936	-0.22832	-0.4744	-0.13225	0.344941
P_2O_5	0.27273	-0.36424	0.047212	0.035688	-0.42565
SrO	0.294615	-0.33659	-0.01926	0.195161	0.027659
SnO_2	-0.03785	-0.19506	0.237227	0.545776	0.404122
SO_2	0.120069	0.111441	0.554312	-0.34042	0.294736

铅钡玻璃主成分：

主成分	F1	F2	F3	F4	F5	F6	F7
SiO_2	-0.45679	0.21777	-0.02641	-0.07713	0.144606	-0.03404	-0.01651
Na_2O	-0.18863	0.275585	-0.09075	0.527698	0.063887	0.220933	-0.06738

K_2O	-0.1557	-0.13202	0.35424	-0.12712	-0.42228	0.683815	0.203508
CaO	0.11631	-0.46675	0.140033	-0.01715	0.148617	-0.03554	-0.09611
MgO	-0.20326	-0.37152	0.057821	0.385505	0.221033	0.233142	0.061575
Al_2O_3	-0.34548	-0.16173	0.314383	0.262065	0.059395	-0.14699	-0.04681
Fe_2O_3	-0.15146	-0.32051	0.132601	-0.39673	0.125977	0.006195	0.284544
CuO	0.302254	0.20691	0.275848	0.097564	0.266803	-0.09764	0.613063
PbO	0.295113	-0.26864	-0.40757	-0.01946	-0.40148	0.012621	0.0343
BaO	0.335953	0.263292	0.43491	0.005739	-0.02502	-0.01474	0.093377
P_2O_5	0.240114	-0.35415	0.011616	0.04477	0.486685	0.036595	-0.1369
SrO	0.27305	-0.14623	-0.06167	0.542885	-0.22812	0.017986	0.1751
SnO_2	-0.17244	-0.18644	0.353711	0.131258	-0.42936	-0.60534	-0.03037
SO_2	0.28481	0.092547	0.407194	-0.02356	-0.00972	0.164766	-0.64925

2）主成分之间以及与是否风化相关系数

高钾玻璃：

	F1	F2	F3	F5	SIO_2	K_2O	CUO	P_2O_5	SRO	风化
F1	1	0	0	0	-0.944	0.716	0.508	0.648	0.7	-0.815
F2	0	1	0	0	-0.285	0.447	0.084	-0.59	-0.546	-0.165
F3	0	0	1	0	-0.112	0.242	-0.151	0.062	-0.025	-0.137
F5	0	0	0	1	-0.034	0.164	0.206	-0.417	0.027	-0.17
SIO_2	-0.944	-0.285	-0.112	-0.034	1	-0.877	-0.465	-0.448	-0.525	0.871
K_2O	0.716	0.447	0.242	0.164	-0.877	1	0.251	0.171	0.421	-0.803
CUO	0.508	0.084	-0.151	0.206	-0.465	0.251	1	0.213	0.187	-0.29
P_2O_5	0.648	-0.59	0.062	-0.417	-0.448	0.171	0.213	1	0.741	-0.425
SRO	0.7	-0.546	-0.025	0.027	-0.525	0.421	0.187	0.741	1	-0.461
风化	-0.815	-0.165	-0.137	-0.17	0.871	-0.803	-0.29	-0.425	-0.461	1

铅钡玻璃：

	F1	F2	F3	F4	CAO	FE_2O_3	P_2O_5	风化
F1	1	0	0	0	0.219934	-0.2864	0.454036	0.62979
F2	0	1	0	0	-0.8015	-0.55038	-0.60814	-0.43128
F3	0	0	1	0	0.18067	0.171081	0.014987	-0.08582
F4	0	0	0	1	-0.01786	-0.4132	0.046628	0.015007
CAO	0.219934	-0.8015	0.18067	-0.01786	1	0.387489	0.535134	0.424136
FE_2O_3	-0.2864	-0.55038	0.171081	-0.4132	0.387489	1	0.152629	-0.08076
P_2O_5	0.454036	-0.60814	0.014987	0.046628	0.535134	0.152629	1	0.545373
风化	0.62979	-0.43128	-0.08582	0.015007	0.424136	-0.08076	0.545373	1

3.神经网络预测结果

高钾预测结果在正文已经全部展示。

铅钡预测结果（正文展示前 5 行）：

文物编号	SiO_2	Na_2O	K_2O	CaO	MgO	Al_2O_3	Fe_2O_3
26	37.697577	1.6770039	0.2744114	0.539465	0.6213157	2.1120376	0.8418381
02	63.351696	0	0.2642332	2.1439092	0.8486499	6.7385836	0.2834556

08	32.238652	2.0967229	0.2769362	0.5253131	0.6218133	2.339537	0.902795
08 严重风化	65.612493	1.4480426	0.310964	1.8407107	0.7381283	2.8361298	1.322569
11	50.066132	1.2426605	0.2575273	0.5573607	0.5908624	3.0009163	0.7399466
26 严重风化	68.473424	0	0.3605131	2.1102261	0.7037044	0.5334138	1.4653255
34	23.129677	1.6329955	0.1977199	1.2606904	0.7268479	5.6936183	0.1113919
36	66.303461	1.9035557	0.2212261	1.2644475	0.6047672	2.9434767	0.1311415
38	57.829229	1.9683674	0.2272756	1.2451333	0.6042573	3.0129836	0.2347632
39	45.741279	1.0866734	0.2177653	1.4346703	0.7577745	4.053925	0.3708718
40	65.95883	1.8528645	0.2107717	1.506325	0.7503485	4.4716343	0.7761635
41	69.98694	1.0038116	0.2219137	1.3743989	0.6982392	5.7370602	0.9413181
43 部位 1	38.920437	1.5677158	0.2209325	1.605558	0.7537703	7.1163129	0.653165
43 部位 2	50.435242	2.0442591	0.2331389	1.3670642	0.7019591	6.5307132	0.9501457
48	61.826587	2.550632	0.2882982	1.5907772	0.7219532	4.3142205	0.8926286
49	53.257256	1.3663393	0.2274123	1.4136153	0.7274249	6.5326113	0.7968053
50	45.836872	1.1815273	0.2217965	1.4356895	0.7111123	5.3277571	0.4443905
51 部位 1	38.971901	1.5628653	0.2235539	1.2822642	0.6805689	6.8989811	0.744161
51 部位 2	32.568497	2.7483515	0.2299061	1.2534653	0.6869951	7.2185204	0.8364092
52	77.836336	0	0.2201888	1.4750811	0.692625	2.4692722	0.2214528
54	71.646419	0	0.2521004	1.780732	0.771612	4.4422553	0.6077797
54 严重风化	88.227811	0	0.2529843	1.7631198	0.7975191	2.9014632	0.6324552
56	49.020409	0.9510044	0.2258369	1.1700315	0.613533	3.043843	0.1556244
57	44.530838	1.7961765	0.2255731	1.1662084	0.5831255	3.775856	0.2203928
19	53.521519	1.7013526	0.2268447	1.148224	0.6931393	5.076806	0.7577215
58	56.498561	1.5814943	0.2385249	1.1446122	0.6738952	5.4862334	0.9198777

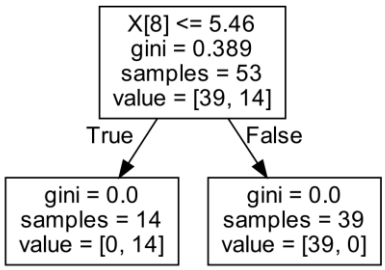
接上表（横向拼接）

文物编号	CuO	PbO	BaO	P_2O_5	SrO	SnO_2	SO_2
26	3.36679	30.376278	17.072375	0.314812	0.4400284	0.1309404	0.4864542
02	0	36.574372	5.7814914	1.2211148	0.2358788	0.1339744	0.6059458
08	3.6616684	28.797177	17.975852	0.2206157	0.4485564	0.1334013	0.4986658
08 严重风化	1.2819009	17.511304	4.4717675	1.6952587	0.3398038	0.1478227	0.2688461
11	1.7841583	29.24626	11.800623	0.6699323	0.3518723	0.1116905	0.5453755
26 严重风化	0	27.499294	3.1547591	1.958629	0.3384134	0.1700946	0.3094766
34	2.2680363	35.4711	13.245431	0.5468697	0.3019347	0.0929006	0.5928972
36	1.2248745	20.330136	10.173915	0.6785986	0.3090474	0.1066197	0.6134039
38	1.5788992	19.823073	9.1515756	0.8767579	0.3085242	0.1046092	0.5497256
39	1.9327286	24.873747	5.5854569	1.5637773	0.2953775	0.089407	0.4143388
40	1.8964128	15.444593	6.4472822	1.4073316	0.2691507	0.0823025	0.33932
41	0.2831367	19.905232	6.3489427	0.8533148	0.2433541	0.1088202	0.3175137
43 部位 1	1.3260734	24.121322	2.2883331	1.5617592	0.2380315	0.0906798	0.3267258
43 部位 2	1.0655136	16.795091	3.1717117	1.3631894	0.2432853	0.0968502	0.3599114
48	1.8604541	16.561452	11.806095	0.2819559	0.4218739	0.1718916	0.4440574
49	0.7146321	22.143165	4.1043439	1.1915572	0.250757	0.1031055	0.3732375

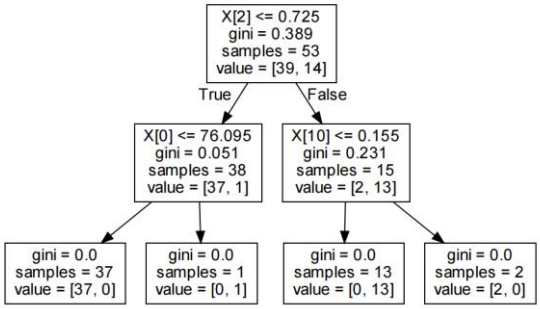
50	1.5720137	27.662234	4.4666527	1.4927604	0.26578	0.0878874	0.3775337
51 部位 1	1.7559411	31.978129	10.544007	0.5546015	0.2774561	0.1045556	0.3450953
51 部位 2	1.647978	19.790205	5.813936	1.1108497	0.2534959	0.0936228	0.469492
52	0.9676249	31.11022	3.2478872	1.7587907	0.2783397	0.0875707	0.2686486
54	0.1516052	36.085491	3.4139843	1.7661151	0.2600694	0.1031418	0.4336847
54 严重风化	0.2788724	40.648642	6.8762613	1.6400053	0.2889836	0.1027971	0.4503978
56	1.7981545	30.734857	9.7153988	0.928613	0.3131894	0.1012495	0.4642256
57	2.0167336	28.119591	11.5989	0.610246	0.3111401	0.1042531	0.5007167
19	1.798953	24.322311	8.7134975	0.9964789	0.2921664	0.0934842	0.4275861
58	1.1460908	21.689102	6.08504	1.1025058	0.2734479	0.1011668	0.3910222

3.决策树数据结果图

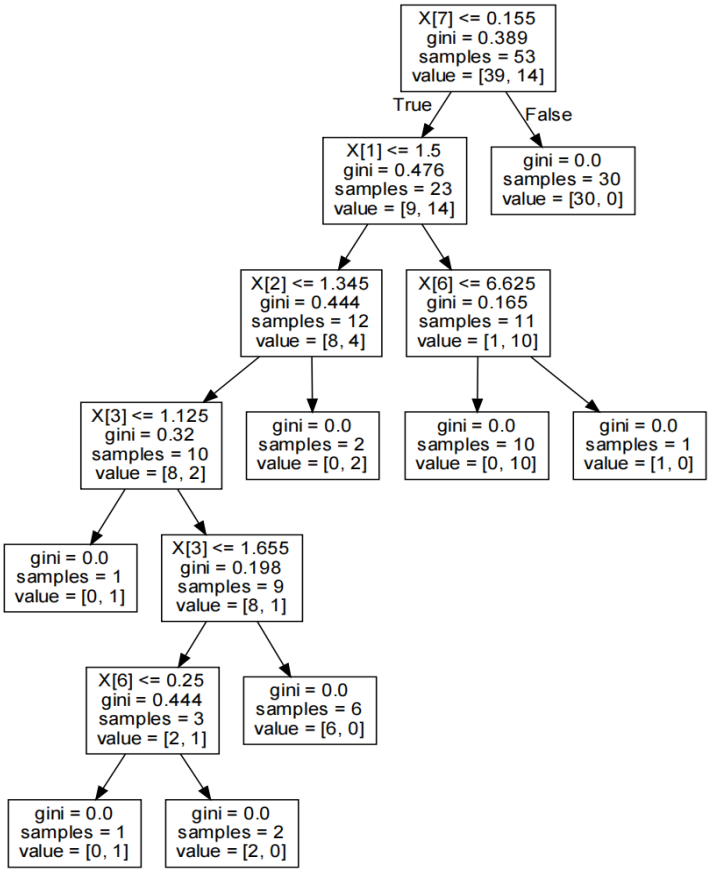
- **定量分析数据结果图**



第一次决策树结果图



第二次决策树结果图



第三次决策树结果图

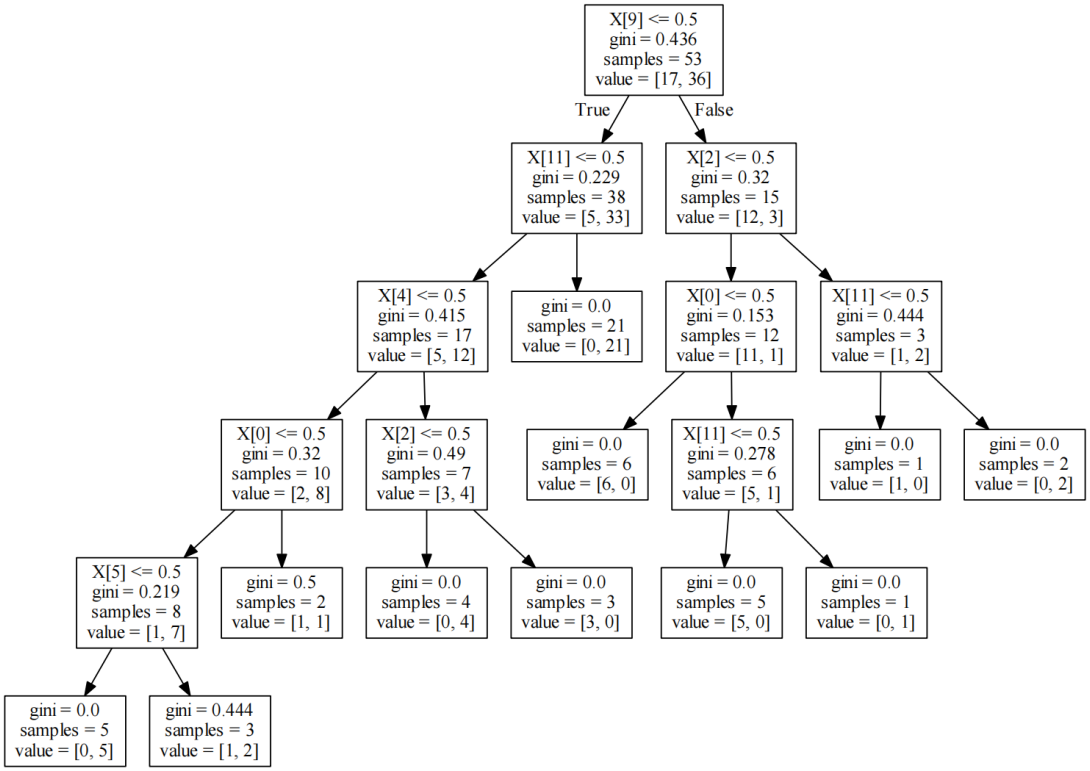
- **定性分析数据结果图**

以**铅钡**作为输出数据（y=0 高钾，y=1 铅钡）

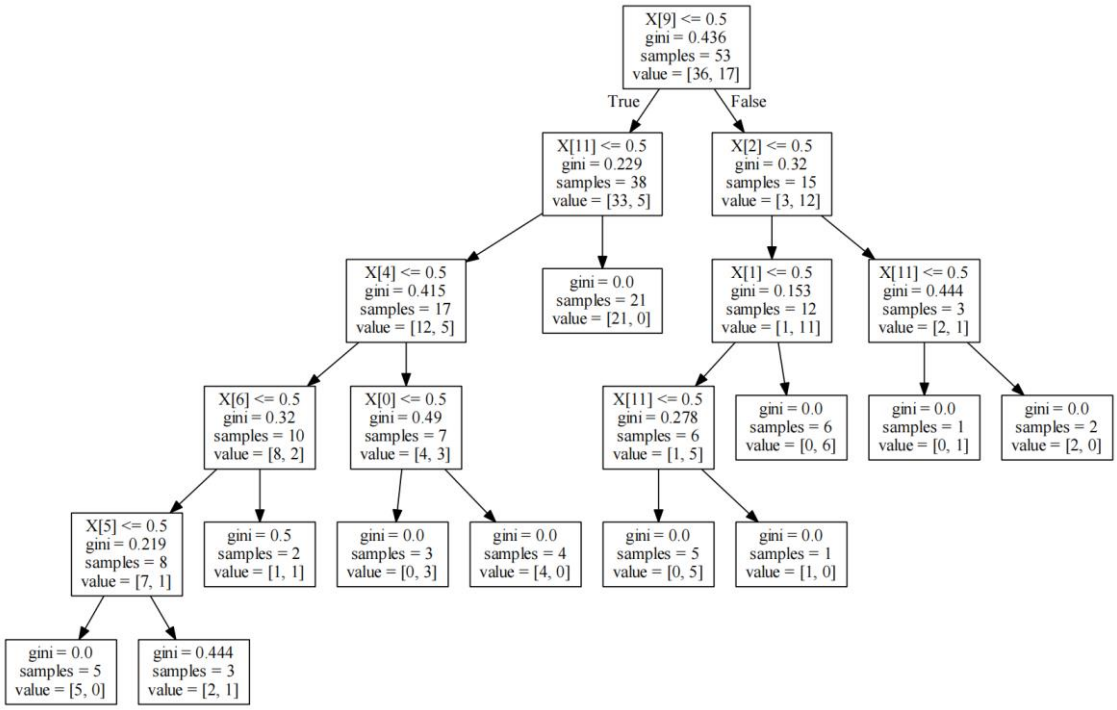
$X[i]$参照标准如下表:

$X[0]$	$X[1]$	$X[2]$	$X[3]$	$X[4]$	$X[5]$	$X[6]$	$X[7]$	$X[8]$	$X[9]$	$X[10]$	$X[11]$
A	B	C	浅绿	浅蓝	深绿	深蓝	紫	绿	蓝绿	黑	风化

由于该数据为定性数据，$X[i] \leq 0.5$意味着$X[i] = 0$，即不包含该属性。例如$X[1]$=0，意味着玻璃不含有属性 B；因此，当$X[1]$=0 为 false 时，该玻璃纹饰为 B。



以**高钾**作为输出数据（y=0 铅钡，y=1 高钾），$X[i]$参照标准同上。决策树预测结果如下，同理当$X[i] \leq 0.5$意味着$X[i] = 0$，即不包含该属性：



3. 聚类划分结果（亚分类）

1）铅钡玻璃划分结果：

原始编号	铅钡玻璃	原始编号	铅钡玻璃
26	**第一类**	02	**第三类**
08		34	
11		36	
24		38	
20		39	
08 严重风化点	**第二类**	40	
26 严重风化点		41	
48	**第四类**	43 部位 1	
31		43 部位 2	
32		49	
33		50	
37		51 部位 1	
46		51 部位 2	
47		52	
55		54	
25 未风化点		54 严重风化点	
28 未风化点		56	
29 未风化点		57	
42 未风化点 1		19	
42 未风化点 2		58	
44 未风化点		30 部位 1	
49 未风化点		30 部位 2	
50 未风化点			
53 未风化点			

2）高钾玻璃划分结果：

原始编号	高钾玻璃
03 部位 1	第一类
07	
09	
10	
12	
22	
27	
18	第二类
03 部位 2	第三类

06 部位 1	
06 部位 2	
21	
01	第四类
04	
05	
13	
14	
16	

4. 化学成分之间相关关系

铅钡玻璃

	SiO2	Na	K2O	CaO	MgO	Al	Fe	CuO	PbO	BaO	P2O5	SrO	SnO2	SO2
SiO2	0	0.4	0.31	0.49	0.12	0.42	0.044	0.45	0.76	0.28	0.54	0.55	0.1	0.31
Na	0.4	0	0.07	0.37	0.02	0.15	0.204	0.1	0.36	0.09	0.57	0.15	0.09	0.21
K2O	0.31	0.07	0	0.03	0.36	0.51	0.149	0.2	0.34	0	0.14	0.17	0.14	0.02
CaO	0.49	0.37	0.03	0	0.33	0.12	0.411	0.04	0.35	0.16	0.5	0.31	0.28	0.08
MgO	0.12	0.02	0.36	0.33	0	0.67	0.258	0.27	0.08	0.44	0.15	0.07	0.26	0.37
Al	0.42	0.15	0.51	0.12	0.67	0	0.387	0.32	0.39	0.37	0.02	0.2	0.33	0.38
Fe	0.04	0.2	0.15	0.41	0.26	0.39	0	0.39	0.08	0.44	0.22	0.12	0.36	0.34
CuO	0.45	0.1	0.2	0.04	0.27	0.32	0.39	0	0.09	0.49	0.22	0.18	0.36	0.45
PbO	0.76	0.36	0.34	0.35	0.08	0.39	0.081	0.09	0	0.1	0.36	0.36	0.09	0.11
BaO	0.28	0.09	0	0.16	0.44	0.37	0.443	0.49	0.1	0	0.17	0.18	0.02	0.47
P2O5	0.54	0.57	0.14	0.5	0.15	0.02	0.225	0.22	0.36	0.17	0	0.25	0.03	0.2
SrO	0.55	0.15	0.17	0.31	0.07	0.2	0.116	0.18	0.36	0.18	0.25	0	0.01	0.23
SnO2	0.1	0.09	0.14	0.28	0.26	0.33	0.363	0.36	0.09	0.02	0.03	0.01	0	0.11
SO2	0.31	0.21	0.02	0.08	0.37	0.38	0.337	0.45	0.11	0.47	0.2	0.23	0.11	0

高钾玻璃

	SiO2	Na2O	K2O	CaO	MgO	Al	Fe	CuO	PbO	BaO	P	SrO	SO2
SiO2	0	0.47	0.78	0.75	0.55	0.83	0.78	0.43	0.44	0.35	0.53	0.52	0.28
Na2O	0.475	0	0.61	0.64	0.22	0.31	0.22	0.113	0.27	0.24	0.22	0.11	0.2
K2O	0.781	0.61	0	0.69	0.26	0.51	0.46	0.179	0.24	0.01	0.16	0.4	0.31
CaO	0.752	0.64	0.69	0	0.09	0.52	0.54	0.344	0.26	0.07	0.02	0.05	0.39
MgO	0.546	0.22	0.26	0.09	0	0.73	0.57	0.204	0.3	0.52	0.64	0.67	0.43
Al	0.831	0.31	0.51	0.52	0.73	0	0.75	0.246	0.6	0.47	0.52	0.51	0.25
Fe	0.783	0.22	0.46	0.54	0.57	0.75	0	0.643	0.32	0.54	0.52	0.41	0.31
CuO	0.43	0.11	0.18	0.34	0.2	0.25	0.64	0	0.05	0.48	0.47	0.14	0.35
PbO	0.443	0.27	0.24	0.26	0.3	0.6	0.32	0.047	0	0.68	0.12	0.37	0.34
BaO	0.346	0.24	0.01	0.07	0.52	0.47	0.54	0.483	0.68	0	0.47	0.56	0.24
P	0.53	0.22	0.16	0.02	0.64	0.52	0.52	0.466	0.12	0.47	0	0.5	0.25
SrO	0.524	0.11	0.4	0.05	0.67	0.51	0.41	0.141	0.37	0.56	0.5	0	0.02

SO2	0.284	0.2	0.31	0.39	0.43	0.25	0.31	0.354	0.34	0.24	0.25	0.02	0

附录 B

[注：所有代码数据源自于原附件数据或数据预处理后数据，python 数据读入均采用相对路径的形式。在支撑材料中，数据与代码同时打包于相同文件夹下。]

1 岭回归代码

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression, SGDRegressor, Ridge
from sklearn.metrics import mean_squared_error
import pandas as pd
import numpy as np

file_path = r'./one-hot.xlsx'  # 重构后的数据,见支撑材料
data = pd.read_excel(file_path, header=0)
# print(data)
x = data.iloc[:, 0:13]
y = data.iloc[:, 13:]
print(x, y)


def zeroone(y_predict):
    result = np.zeros((12, 1))

    #     print(result)
    for x in range(0, 12):
        for y in range(0, 1):
            if y_predict[x] >= 0.5:
                result[x, y] = 1
    return result


# 岭回归
def linear3(x, y):
    # 数据获取和分割
    x_trian, x_test, y_train, y_test = train_test_split(x, y, random_state=22)
    # 数据预处理
    transfer = StandardScaler()
    x_train = transfer.fit_transform(x_trian)
    x_test = transfer.transform(x_test)
    # 预估器
```

```
    estimator = Ridge()
    estimator.fit(x_train, y_train)
    # 得出模型
    print('coef:', estimator.coef_)
    print('bias:', estimator.intercept_)
    # 模型评估
    y_predict = estimator.predict(x_test)
    print('预测值:', y_predict)
    # print(y_predict.shape)
    # print(type(y_predict))
    # result = zeroone(y_predict)
    # print('更改:', y_predict)
    error = mean_squared_error(y_test, y_predict)
    print('均方误差: ', error)


if __name__ == '__main__':
    linear3(x, y)
```

2.主成分分析代码/皮尔逊相关系数

[Matlab 代码]正文主成分分析结果源于 matlab 结果

```
%% 高钾
clear
clc

load x
[n,p] = size(x);
x = zscore(x);
r = corrcoef(x);
[e,D] = eig(r);
ld = diag(D);
ld = ld(end:-1:1);
contr = ld / sum(ld);
sum_contr = cumsum(ld)/ sum(ld);
disp('贡献率计算如下')
disp(sum_contr')
disp('特征向量计算如下: ')
e=rot90(e)';
disp(e) % 主成分组成
m = 5;
u = zeros(n,m);
for i = 1:m
    ai = e(:,i)';
```

```
    Ai = repmat(ai,n,1);
    u(:, i) = sum(Ai .* x, 2);
end
u = [u(:,1),u(:,2),u(:,3),u(:,5)];
load y
y = zscore(y);
u = [u,y];
r = corrcoef(u) % 主成分相关系数


%% 铅钡
clear
clc

load xx
[n,p] = size(xx);
xx = zscore(xx);
r = corrcoef(xx);
[e,D] = eig(r);
ld = diag(D);
ld = ld(end:-1:1);
contr = ld / sum(ld);
sum_contr = cumsum(ld)/ sum(ld);
disp('贡献率计算如下：')
disp(sum_contr')
disp('与特征值对应的特征向量矩阵为：')
e=rot90(e)';
disp(e) % 主成分组成
m = 7;
u = zeros(n,m);
for i = 1:m
    ai = e(:,i)';
    Ai = repmat(ai,n,1);
    u(:, i) = sum(Ai .* xx, 2);
end
u = [u(:,1),u(:,2),u(:,3),u(:,4)];
load yy
yy = zscore(yy);
u = [u,yy];
r = corrcoef(u) % 主成分相关系数
```

[python 代码]

```
import pandas as pd
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from matplotlib.colors import ListedColormap

data=pd.read_excel("./fenghua22.xlsx") # 文件位于支撑材料
# data=pd.read_excel("./fenghua.xlsx")

data=data.fillna(0)
x=data.iloc[:,:14]
y=data.iloc[:,14:]

# // 用 python 实现主成分分析（PCA）
import numpy as np
from numpy.linalg import eig
from sklearn.datasets import load_iris
def pca(X,k):
    X = X - X.mean(axis = 0) #向量 X 去中心化
    X_cov = np.cov(X.T, ddof = 0) #计算向量 X 的协方差矩阵，自由度可以选择 0 或 1
    eigenvalues,eigenvectors = eig(X_cov) #计算协方差矩阵的特征值和特征向量

    # 可视化
    tot = sum(eigenvalues)
    var_exp = [(i / tot) for i in sorted(eigenvalues, reverse=True)]
    cum_var_exp = np.cumsum(var_exp)

    plt.bar(range(1, 15), var_exp, alpha=0.5, align='center', label='individual var'
)
    plt.step(range(1, 15), cum_var_exp, where='mid', label='cumulative var')
    plt.ylabel('variance rtion')
    plt.xlabel('principal components')
    plt.legend(loc='best')
    plt.show()

    klarge_index = eigenvalues.argsort()[-k:][::-1] #选取最大的 K 个特征值及其特征向量
    k_eigenvectors = eigenvectors[klarge_index] #用 X 与特征向量相乘
    return np.dot(X, k_eigenvectors.T)
iris = load_iris()
X = x
k = 2
X_pca = pca(X, k)
```

```
print(X_pca)
```

3.蒙特卡洛模拟代码（以高钾-风化为例，其余类似，见支撑材料）

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

# 模拟高钾风化的数据
for i in range(100):
    arr = [92.63, 95.02, 96.77, 94.29, 92.35, 92.72]
    std1 = np.std(arr)
    mean = np.mean(arr)

    arr2 = [0.59, 0.92, 1.01, 0.74, 0, 0]
    std2 = np.std(arr2)
    mean2 = np.mean(arr2)

    arr3 = [1.07, 0.62, 0.21, 0.72, 1.66, 0.94]
    std3 = np.std(arr3)
    mean3 = np.mean(arr3)

    arr4 = [0.64, 0.54, 0, 0, 0, 0]
    std4 = np.std(arr4)
    mean4 = np.mean(arr4)

    arr5 = [1.98, 1.32, 0.81, 1.46, 3.5, 2.51]
    std5 = np.std(arr5)
    mean5 = np.mean(arr5)

    arr6 = [0.17, 0.32, 0.26, 0.29, 0.35, 0.2]
    std6 = np.std(arr6)
    mean6 = np.mean(arr6)

    arr7 = [3.24, 1.55, 0.84, 1.65, 0.55, 1.54]
    std7 = np.std(arr7)
    mean7 = np.mean(arr7)

    arr8 = [0.61, 0.35, 0, 0.15, 0.21, 0.36]
    std8 = np.std(arr8)
    mean8 = np.mean(arr8)

    # 生成 100 组数据
```

```
    Si = np.random.normal(mean, std1, 100)
    Na = np.random.normal(0, 0, 100)
    K = np.random.normal(mean2, std2, 100)
    Ca = np.random.normal(mean3, std3, 100)
    Mg = np.random.normal(mean4, std4, 100)
    Al = np.random.normal(mean5, std5, 100)
    Fe = np.random.normal(mean6, std6, 100)
    Cu = np.random.normal(mean7, std7, 100)
    Pb = np.random.normal(0, 0, 100)
    Ba = np.random.normal(0, 0, 100)
    P = np.random.normal(mean8, std8, 100)
    Sr = np.random.normal(0, 0, 100)
    Sn = np.random.normal(0, 0, 100)
    S = np.random.normal(0, 0, 100)

    # 处理小于 0 的数据
    Si[Si < 0] = 0
    Na[Na < 0] = 0
    K[K < 0] = 0
    Ca[Ca < 0] = 0
    Mg[Mg < 0] = 0
    Al[Al < 0] = 0
    Fe[Fe < 0] = 0
    Cu[Cu < 0] = 0
    Pb[Pb < 0] = 0
    Ba[Ba < 0] = 0
    P[P < 0] = 0
    Sr[Sr < 0] = 0
    Sn[Sn < 0] = 0
    S[S < 0] = 0

    # 写入 csv 文件
    # use pandas
    # write date by using the form of dict
    df = pd.DataFrame(
        {'二氧化硅\n(SiO\u2082)': Si, '氧化钠\n(Na\u2082O)': Na, '氧化钾\n(K\u2082O)': K, '氧化钙\n(CaO)': Ca, '氧化镁\n(MgO)': Mg,
         '氧化铝\n(Al\u2082O\u2083)': Al, '氧化铁\n(Fe\u2082O\u2083)': Fe, '氧化铜\n(CuO)': Cu, '氧化铅\n(PbO)': Pb,
         '氧化钡\n(BaO)': Ba,
         '五氧化二磷\n(P\u2082O\u2085)': P, '氧化锶\n(SrO)': Sr, '氧化锡\n(SnO\u2082)': Sn, '二氧化硫\n(SO\u2082)': S})
    print("printing {}...".format(i))
```

```
    df.to_csv("C:\\Users\\Kaytlin\\Desktop\\random\\test{}.csv".format(i), index=Fal
se)
```

5. 神经网络[matlab 代码]（神经网络主要通过工具箱完成）

```
clear
clc

load a
load b
set(0,'defaultfigurecolor','w')
% 调用神经网络工具箱
load aa

pre1=[];
for i = 1: 6
    result = sim(net, aa(i,:)');
    result = result';
    pre1=[pre1;result];
end

clear
clc

load c
load d

set(0,'defaultfigurecolor','w')
% 调用神经网络工具箱
load cc

pre2=[];
for i = 1: 26
    result = sim(net, cc(i,:)');
    result = result';
    pre2=[pre2;result];
end
for i = 1:26
    for j=1:14
        if pre2(i,j)<0
            pre2(i,j)=0;
        end
    end
```

```
end

pre2tmp = sum(pre2,2);
pre2 = pre2./pre2tmp*100;
```

5.决策树代码

1）定性数据：离散型数据

```
# 利用 决策树模型 在三分类(多分类)上 进行训练和预测
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn import metrics
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

data=pd.read_excel("./one-hot.xlsx")
plt.rcParams['font.sans-serif'] = 'SimHei'
plt.rcParams['axes.unicode_minus'] = False  ## 设置正常显示符号
x_train, x_test, y_train, y_test = train_test_split(
    data[['A','B','C','浅绿','浅蓝','深绿','深蓝','紫','绿','蓝绿','黑','风化
'']], data[['高钾']],
    test_size = 0.01, random_state = 2020) # 所有数据作为训练集，更好的对数据分类
clf = DecisionTreeClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

train_predict_proba = clf.predict_proba(x_train)
test_predict_proba = clf.predict_proba(x_test)

print('The test predict Probability of each class:\n',test_predict_proba)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,t
rain_predict))
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,te
st_predict))

confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
print('The confusion matrix result:\n',confusion_matrix_result)

```

```
# plt.figure(figsize=(8, 6))
# sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
# plt.xlabel('Predicted labels')
# plt.ylabel('True labels')
# plt.show()

# 导出决策树 pdf 可视化结果
import graphviz
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("decistiontest1")
```

2）定量数据：连续型数据

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn import metrics
import seaborn as sns

data = pd.read_excel('./index_chemistry.xlsx')
data = data.fillna(0)

drop_cols = [15]
x = data.drop(drop_cols, axis=1)
y = data[[15]]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_st
ate = 2020)

clf = DecisionTreeClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

train_predict_proba = clf.predict_proba(x_train)
test_predict_proba = clf.predict_proba(x_test)

print('The test predict Probability of each class:\n',test_predict_proba)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,t
rain_predict))
```

```
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,test_predict))

confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
print('The confusion matrix result:\n',confusion_matrix_result)

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.show()

import graphviz
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("decisiontree_连续 1")

# 第二次 剔除氧化铅
drop_cols = [9, 15]
x = data.drop(drop_cols, axis=1)

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 2020)

clf = DecisionTreeClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

train_predict_proba = clf.predict_proba(x_train)
test_predict_proba = clf.predict_proba(x_test)

print('The test predict Probability of each class:\n',test_predict_proba)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,train_predict))
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,test_predict))

confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
print('The confusion matrix result:\n',confusion_matrix_result)

plt.figure(figsize=(8, 6))
```

```
sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.show()

import graphviz
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("decisiontree_连续 2")

# 第三次 剔除二氧化硅 氧化钾 氧化钡

drop_cols = [9, 15, 1, 3, 10]
x = data.drop(drop_cols, axis=1)

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_st
ate = 2020)

clf = DecisionTreeClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

train_predict_proba = clf.predict_proba(x_train)
test_predict_proba = clf.predict_proba(x_test)

print('The test predict Probability of each class:\n',test_predict_proba)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,t
rain_predict))
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,te
st_predict))

  confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
  print('The confusion matrix result:\n',confusion_matrix_result)

  plt.figure(figsize=(8, 6))
  sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
  plt.xlabel('Predicted labels')
  plt.ylabel('True labels')
  plt.show()

  import graphviz
```

```
    dot_data = tree.export_graphviz(clf, out_file=None)
    graph = graphviz.Source(dot_data)
    graph.render("decisiontree_连续 3")
```

6.集成学习代码（包含 XGBoost,随机森林,LightGBM,AdaBoost）

```
# XGBoost 代码
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from xgboost.sklearn import XGBClassifier
from sklearn import metrics

data = pd.read_excel('./')
data = data.fillna(0)


data_target_part = data[15]
data_features_part = data[[x for x in data.columns if x != 15]]

x_train, x_test, y_train, y_test = train_test_split(data_features_part, data_target_part, test_size = 0.2, random_state = 2020)

clf = XGBClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,train_predict))
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,test_predict))

confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
print('The confusion matrix result:\n',confusion_matrix_result)

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.show()

```

```
x_pre = pd.read_excel('./chemistry.xlsx')
x_pre = x_pre.fillna(0)
y_pre = clf.predict(x_pre)

# ----------------------------------------------------------------------
# 随机森林代码
# RandomForest
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics

data = pd.read_excel('./chemistry.xlsx')
data = data.fillna(0)

data_target_part = data[15]
data_features_part = data[[x for x in data.columns if x != 15]]

x_train, x_test, y_train, y_test = train_test_split(data_features_part, data_target_part, test_size = 0.2, random_state = 2020)

clf = RandomForestClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,train_predict))
print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,test_predict))

confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
print('The confusion matrix result:\n',confusion_matrix_result)

plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.show()

x_pre = pd.read_excel('./input.xlsx')
```

```
x_pre = x_pre.fillna(0)
y_pre = clf.predict(x_pre)

# ----------------------------------------------------------------------
# Adaboost 代码
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn import metrics

data = pd.read_excel('./chemistry.xlsx')
data = data.fillna(0)

data_target_part = data[15]
data_features_part = data[[x for x in data.columns if x != 15]]

x_train, x_test, y_train, y_test = train_test_split(data_features_part, data_target_part, test_size = 0.2, random_state = 2020)


  clf = AdaBoostClassifier()
  clf.fit(x_train, y_train)

  train_predict = clf.predict(x_train)
  test_predict = clf.predict(x_test)

  print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_train,train_predict))
  print('The accuracy of the Logistic Regression is:',metrics.accuracy_score(y_test,test_predict))

  confusion_matrix_result = metrics.confusion_matrix(test_predict,y_test)
  print('The confusion matrix result:\n',confusion_matrix_result)

  plt.figure(figsize=(8, 6))
  sns.heatmap(confusion_matrix_result, annot=True, cmap='Blues')
  plt.xlabel('Predicted labels')
  plt.ylabel('True labels')
  plt.show()

  x_pre = pd.read_excel('./input.xlsx')
```

```
x_pre = x_pre.fillna(0)
y_pre = clf.predict(x_pre)
```

7.集成学习敏感度分析（以 xgboost 代码为例）

```
# 敏感性分析
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from xgboost.sklearn import XGBClassifier
from sklearn import metrics


data = pd.read_excel('./chemistry.xlsx')
data = data.fillna(0)

data_target_part = data[15]
data_features_part = data[[x for x in data.columns if x != 15]]

x_train, x_test, y_train, y_test = train_test_split(data_features_part, data_target_part,  test_size = 0.2, random_state = 2020)

#XGBoost 为例做敏感度分析
clf = XGBClassifier()
clf.fit(x_train, y_train)

train_predict = clf.predict(x_train)
test_predict = clf.predict(x_test)

x_pre = pd.read_excel('./input.xlsx')
x_pre = x_pre.fillna(0)
y_pre = clf.predict(x_pre)


# 每棵树特征随机采样比例 敏感度
xl = np.zeros(99)
yl = np.zeros(99)
cnt = 0
i=0.01
while i<1:
    xl[cnt]=i
```

```
    clf = XGBClassifier(colsample_bytree=i, learning_rate=0.1, max_depth=7, subsample=0.3)
    clf.fit(x_train, y_train)
    train_predict = clf.predict(x_train)
    test_predict = clf.predict(x_test)
    yl[cnt]=metrics.accuracy_score(y_test,test_predict)
    cnt+=1
    i+=0.01


plt.rcParams['font.sans-serif'] = 'SimHei'
plt.rcParams['axes.unicode_minus'] = False
plt.plot(xl,yl,label='每棵树特征随机采样比例')
plt.xlabel('参数值')
plt.ylabel('accuracy')
plt.legend()


# 学习迭代步长 敏感度
xl = np.zeros(99)
yl = np.zeros(99)
cnt = 0
i=0.01
while i<1:
    xl[cnt]=i
    clf = XGBClassifier(colsample_bytree=0.4, learning_rate=i, max_depth=7, subsample=0.3)
    clf.fit(x_train, y_train)
    train_predict = clf.predict(x_train)
    test_predict = clf.predict(x_test)
    yl[cnt]=metrics.accuracy_score(y_test,test_predict)
    cnt+=1
    i+=0.01

plt.plot(xl,yl,label='学习迭代步长')
plt.legend()


# 树的最大深度/10 敏感度
xl = np.zeros(11)
yl = np.zeros(11)
cnt = 1
i=1
xl[0]=0
```

```
yl[0]=1
while i<=10:
    xl[cnt]=i/10.0
    clf = XGBClassifier(colsample_bytree=0.9, learning_rate=0.2, max_depth=i, subsample=0.3)
    clf.fit(x_train, y_train)
    train_predict = clf.predict(x_train)
    test_predict = clf.predict(x_test)
    yl[cnt]=metrics.accuracy_score(y_test,test_predict)
    cnt+=1
    i+=1

plt.plot(xl,yl,label='树的最大深度/10')
plt.legend()


# 树种类采样比例 敏感性
xl = np.zeros(99)
yl = np.zeros(99)
cnt = 0
i=0.01
  while i<1:
      xl[cnt]=i
      clf = XGBClassifier(colsample_bytree=0.4, learning_rate=0.3, max_depth=7, subsample=i)
      clf.fit(x_train, y_train)
      train_predict = clf.predict(x_train)
      test_predict = clf.predict(x_test)
      yl[cnt]=metrics.accuracy_score(y_test,test_predict)
      cnt+=1
      i+=0.01
      y_pre = clf.predict(x_pre)
      print(y_pre)

  plt.plot(xl,yl,label='树种类采样比例')
  plt.legend()
  plt.show()
```

8.斯皮尔曼相关系数/相关系数矩阵热力图代码

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

```

```
# filename = "./高钾 相关系数.xlsx"# 数据见支撑材料
filename = "./铅钡 相关系数.xlsx"

df = pd.read_excel(filename)
data=df.fillna(0)
# print(data)
# index_col 参数默认为 False，即自动分配行 index
# csv_data.head(10)

corrit = data.corr(method='spearman')
# 丢失掉缺失项的行列
corrit.dropna(axis=1, how="all", inplace=True)
corrit.dropna(axis=0, how="all", inplace=True)
# print(corrit)

# 解决中文乱码
sns.set_style('whitegrid',{'font.sans-serif':['simhei','Arial']})
# 解决保存图像是负号'-'显示为方块的问题
plt.rcParams['axes.unicode_minus']=False
# 设置大小
plt.figure(figsize=(8, 8))
plt.rcParams['savefig.dpi'] = 50 #图片像素
plt.rcParams['figure.dpi'] = 50 #分辨率
# 绘制图像
ax = sns.heatmap(corrit, linewidths = 0.01,vmin=-
1, vmax=1.0, square=True, annot=True,cmap='BrBG')#橙色绿色 cmap
# ax.set_title("相关性热力图-高钾")
ax.set_title("相关性热力图-铅钡")
plt.show()

#计算绝对值平均值
corrit[corrit<0]=-corrit[corrit<0]
corrit[corrit==1]=0
means=corrit.mean()
print(means.mean())
```

9.正态分布验证及可视化代码

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

filename="./正太分布试试.xlsx" # 该组数据取自于某一次模拟结果中的某一化学成分
s=pd.read_excel(filename)

```

```
# 已知数据分布情况
# s = pd.DataFrame([92.63,95.02,96.77,94.29,92.35,92.72])
print(s.head())
# 创建随机数据

fig = plt.figure(figsize = (10,6))
ax1 = fig.add_subplot(2,1,1)  # 创建子图 1
ax1.scatter(s.index, s.values)
plt.grid()
# 绘制数据分布图

ax2 = fig.add_subplot(2,1,2)  # 创建子图 2

# plt.legend('value')
s.hist(bins=30,alpha = 0.5,ax = ax2,color='navy')
s.plot(kind = 'kde', secondary_y=True,ax = ax2,color='g')
plt.grid()
# 绘制直方图
# 呈现较明显的正太性
plt.show()
```