基于元素风化规律的玻璃文物分类模型

摘要

随着考古技术的进步，我们对文物的鉴别不再停留于表面，而是深入文物的化学成分，对不同的文物进行种类的细分。我们通过分析文物中各指标以及化学成分与风化率之间的关系，得出了各指标中风化率的基本情况，并基于 SiO_2 变化的类时间序列构建文物风化程度模型。同时，也基于原有的分类方式对各文物进行了亚类细分，并对各细分类之间进行了探究。

针对问题一，首先我们对不同的单维指标进行了风化率研究，并通过卡方检验筛选出了二维指标的组合方式，对玻璃文物的风化率进行了进一步分析。其次，我们进行数据预处理，分析两类不同玻璃中 SiO_2 含量的变化规律，并使用 SiO_2 来作为判定不同类型文物风化程度的依据。此后使用 K-means 均值聚类法，将玻璃风化分成了前中后三个阶段，并以风化前这一阶段的各元素优化后的均值，作为不同类型检测点风化前数据的预测值。

针对问题二，我们使用决策树模型，判别出了对于高钾玻璃和铅钡玻璃分类的决策依据为氧化铅含量这一单一指标。鉴于问题一中 SiO_2 的变化规律，我们使用了类时间序列分析，来对文物的化学成分进行判断。此后使用 Spearman 相关性分析，结合问题一的聚类结果，以最小二乘为最终标准，对两类玻璃进行亚类细分。其中高钾玻璃依据镁和磷的含量细分为透光型和非透光型，铅钡玻璃则依据氧化钠的含量，分成了活泼型和非活泼型两种亚类，均对其进行了合理性和敏感性分析。

问题三、四则是对问题二的延伸，基于问题二得到的标准，我们对附件 3 中的元素进行了亚类分析以及敏感性分析。最后基于斯皮尔曼相关性系数绘制了不同亚类中各化学元素的相关性热力图，结合具体值对不同亚类之间的关联关系以及差异性进行了对比分析。

总而言之，文章结构层层递进，具有较强的逻辑性与可执行性，在玻璃文物分类过程之中具有一定的参考价值。

关键字：玻璃文物分类　K-means 均值聚类　Spearman 相关系数　卡方检验　SiO_2

一、 问题背景

中国的历史文化博大精深，源远流长。西汉时期，张骞出使西域开辟了古代中西方文化交流的通道——丝绸之路，迎来了中西方文明交汇融合的时代，玻璃就是早期贸易往来的宝贵物证。

我国古代人民吸取其技术后也开始制作玻璃，虽然外观与贸易而来的玻璃制品相似，不过取材不尽相同。玻璃的主要化学成分是二氧化硅，来自于原料石英砂，但是由于纯石英砂的熔点较高，炼制时常常需要添加助溶剂降低其熔点。添加的助熔剂不同，烧制出来的玻璃的主要化学成分也有不同。如楚文化的玻璃就是以铅钡玻璃为主，这种玻璃品种通常被认为是我国自己发明出来的品种，其在烧制的过程中会加入铅矿石作为助熔剂，使该玻璃品种氧化铅和氧化钡的含量较高。而流行于我国岭南和东南亚和印度等地的钾玻璃加入的助熔剂为草木灰，其含钾量较高。

目前考古学家会通过大量考古挖掘，丰富和深化对历史的认识。但古代玻璃很容易受埋藏环节的影响而风化，风化会使玻璃制品的内部元素和环境中的外部元素发生大量交换，导致制品的成本比例发生变化，从而影响到考古学家对古代玻璃制品原材料比例的判断。所以对古代玻璃制品成分比例进行分析还原是一件十分重要的事情。

现有一批古代玻璃制品的相关数据，并且已经将其分为铅钡玻璃和高钾玻璃。附件 1 给出了文物的分类信息，附件 2 给出了检测出的主要成分对应比例，用来解决以下问题。

二、 问题重述

- **问题一**

  对玻璃文物的表面风化情况与玻璃类型、纹饰、颜色这三类指标进行相关性分析；以玻璃类型进行分类，分析文物样品表面风化前后化学成分的变化规律，并分别预测两类型玻璃检测点风化前的化学元素含量。
- **问题二**

  探究高钾和铅钡玻璃的分类规律；对这两类玻璃分别进行亚类划分，并分析其结果的合理性和敏感性。
- **问题三**

  依问题二的划分标准，分析附件 3 的数据，并判断敏感性
- **问题四**

  剖析不同类别玻璃文物间化学成分关联关系，并比较其差异性。

三、 模型假设

1. 附件 1 中颜色的空白处，均是受风化影响而无法对其颜色进行分析的玻璃文物
2. 其检测结果中若化学成分之和大于 100% 则为测量时的整体性误差，若各化学成分之和小于 100% 则默认表格中所检测到的每一样元素数据结果均为准确的，其中误差值中的化学成分，不在所检测的化学成分列表之中，与问题的分析无关。
3. 若部分元素在玻璃文物中占比较少或几乎不出现，则可在后续的分析中忽略其对整体的影响。

四、 问题一的分析与求解

4.1 问题一的分析

问题一总共分为了三个小问，是循序渐进的过程，我们的首先需探究的便是玻璃的类型、纹路以及颜色与风化率的关系。在第一小问当中，玻璃的风化这一本是循序渐进的过程，被题目定义为了一个二值化的概念，即不考虑风化的中间过程，只考虑风化了或者未风化这两种情况。因此，我们将引入风化率这一个概念，来探究他们三类单一指标与玻璃的风化率之间的基础关系。更进一步，我们将对他们这三个单一指标之间，以及这些指标与实际风化率之间进行卡方检验，从而判定其间是否存在显著性差异，从而针对这些差异，有选择地进行指标的组合，从而判断多指标结合后与玻璃风化率之间的关系。

而对于第二小问与第三小问，根据题目的定义，我们首先需对数据进行有效性判断，将成分比例累加不介于 85%-105% 之间的数据剔除，再对具体的化学成分进行分析。我们发现，两类组成成分不同的玻璃制品，其风化率均与其二氧化硅的含量强相关，但具体的化学成分含量仍具有一定差距。因此，我们对两类玻璃制品分开考虑，探究其风化情况与二氧化硅的关系。因此，我们引入风化程度这一概念，依据二氧化硅的含量来判定文物的风化程度。

在此，我们考虑到了 K-Means 均值聚类算法。由于风化文物存在未风化部分，且我们所选取的检测部位是随机的，我们并不能直接将检测点的各元素数据直接当作该物品的平均元素值。因此，我们将风化分成三个过程，分别为风化前、风化中以及风化后三个状态，并将风化中的检测数据剔除，转而使用风化前和风化后这两部分的数据，以此来减弱取样随机性对我们的计算带来的误差。

此后，我们便可探究风化文物的化学成分统计规律，并以风化前的数据作为参照，将风化点按照高钾玻璃和铅钡玻璃两列，预测出其风化前各化学物质的含量。

4.2 问题一的求解

4.2.1 指标关系的分析

首先，我们统计了各种类型文物的个数，我们发现总的文物统计样本数是 58 个，高钾文物共有 18 个，而铅钡文物共有 40 个。此外，A 纹路的数量为 22 个，B 纹路的数量为 6 个，而 C 纹路的数量有 30 个。而颜色的个数则各有参差，浅蓝的个数有 20 个，蓝绿的个数有 15 个，而黑色和无颜色均是已被风化的，其样本量分别为 2 和 4。而浅绿和绿色的样本量也较小，分别是 3 和 1。

统计完毕后，我们计算了三类指标中各个指标的分化率，并结合平均风化率，我们发现高钾玻璃的风化率较低，而铅钡玻璃的风化率高于平均水平，且其总量较多，共有 40 个。此外，B 纹路的样品样本量虽然只有 6 个但均被风化了。而对于颜色部分，我们发现部分颜色的样本量较小，因此我们不在总体分析部分对颜色的情况进行详细的阐述。

我们对这三类单维指标的风化率进行刻画，并使用颜色对三类指标进行了区分，结合平均分化率的数据，绘制了如下的统计结果分析图：

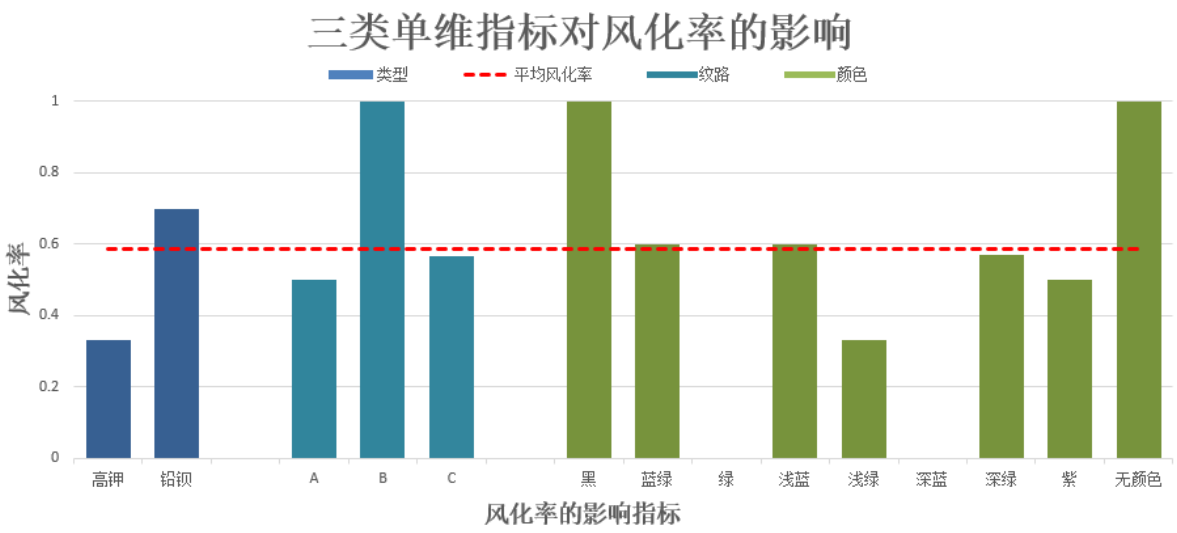
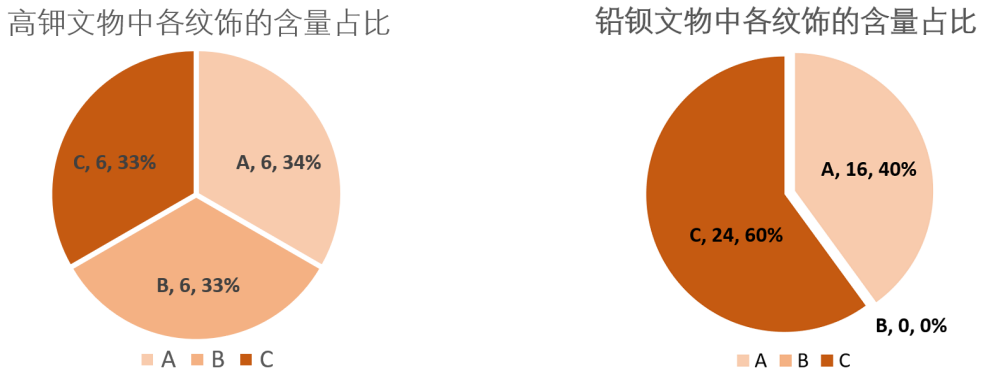


图 1　三类单维指标对文物风化率的影响

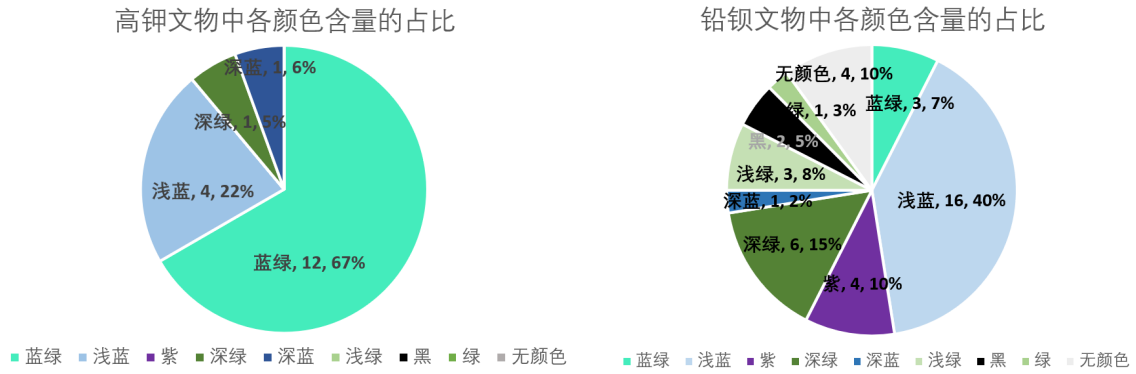
诚然，如果仅仅使用这样的方式进行判定是远远不够的，因为当我们使用单一指标对风化率进行分析时，可能会受该指标中的组成成分所影响，例如铅钡的风化率较高，很有可能是因为铅钡中，某种纹路的文物含量较高，抑或是其中某种颜色的文物含量较高。因此我们需由点及面，对各指标的组成成分进行分析。

为此，我们选取了部分具有典型性的数据，发现了不同类型的文物中其所包含纹路以及颜色大相径庭，如下便是我们根据高钾和铅钡的组成成分占比所绘制的饼状分析图：



(a) 高钾文物 **(b) 铅钡文物**

图 2　对不同类型文物中纹饰的分析



(a) 高钾文物 **(b) 铅钡文物**

图 3　对不同类型文物中纹饰的分析

由此，我们发现不同类型文物中的数据类型规律，即铅钡文物中缺失了类型 B 的纹路，而高钾类型的文物当中缺失了绿、黑、浅绿、紫色以及无颜色的部分。也正是因此，我们需对这三类单维指标之间的关系进行差异性检验分析，同时再对这三类指标与其呈现的风化率进行差异性检验，从而判断出我们需要具体结合哪些指标，对风化率进行二维指标的深入分析。

在诸多的差异性检验方式之中，卡方检验更适于对定类的数据进行分析，因此我们选择卡方检验这一方式来进行差异性分析，其具体检验方法如下所示：

（1）计算出每个 k、f_i 以及 np_i 的值

首先根据我们所进行卡方检验的数据列出卡方表格，并计算其具体的卡方值。其中 k 代表了行数和列数得乘积，即该条件下的方格数，而 f_i 则代表了该方格中实际数据的频数，np_i 则代表的是行合计数乘上列合计数再除以总个数，从而得到的理论频数。

例如，我们在探究类型与纹路之间关系时，我们计算得 k 值为 $2 \times 3 = 6$，其 f_1 的值为高钾中 A 类型的个数 6，其 np_i 的值为 $\frac{22 \times 18}{58} = 6.827$。

（2）计算出 χ^2 的值

依据如下公式，我们便可以的到 χ^2 的值

$$\chi^2 = \sum_{i=1}^{k} \frac{(f_i - np_i)^2}{np_i} \tag{1}$$

（3）依据 χ^2 的值对数据差异性进行检验

表 1　卡方检验结果表

	类型与纹饰	类型与颜色	纹饰与颜色	类型与风化	纹饰与风化	颜色与风化
概率 p	0.82	0.0019	0.0005	0.0201	0.0565	0.5066
卡方 χ	2.2059	22.6929	38.1086	4.134	5.7471	6.2871

通过对卡方检验临界值表进行对照，我们可以得出，在他们三类变量当中，唯有类型与纹饰之间没有相关性，而在他们与风化率分析比较的数据中，我们发现唯有类型与风化率这一判断指标具有弱相关性。

因此在二维指标分析时，我们以类型这一指标为大类，并对其叠加了颜色或纹饰后的二维指标的风化率进行分析，结果如下图所示：

表 2　高钾玻璃细分类型的分化率表

高钾玻璃（平均风化率 33%）							
指标	A	B	C	蓝绿	浅蓝	深蓝	深绿
风化率	0	100%	0	50%	0	0	0

表 3　铅钡玻璃细分类型的分化率表

铅钡玻璃（平均风化率 70%）										
指标	A	C	黑	蓝绿	绿	浅蓝	浅绿	深蓝	深绿	紫	无颜色
风化率	69%	71%	100%	100%	0%	75%	33%	0%	67%	50%	100%

根据以上表格，我们得到了如下的结论：

（1）对高钾玻璃而言，我们得到在 ABC 三类纹路占比均等的基础之上，风化的高钾玻璃全部都是由 B 组成的；颜色方面，所有风化的高钾玻璃均为蓝绿色的，其余的高钾玻璃均为风化。

（2）对于铅钡玻璃，其纹路类型对他风化率的影响较小，而其占比最大的浅蓝色类型以 75% 的风化率，成为了其中风化数量最多的颜色类型。深蓝色和绿色作为唯二的两个小众元素，他们均未被风化，而黑色的玻璃则是全被风化。此外，附件二中的无颜色部分，均是被风化了的，因此我们假定其均为受风化影响而无法分辨其颜色。

4.2.2 化学成分的统计规律及含量预测

第二小问将数据分析的主体，转变为了化学元素的组成成分，根据题目条件，我们首先要对数据进行数据预处理。我们将预处理分为了如下两步：

- 进行初步剔除，将成分比例累加和介于 85%-105% 之间的数据视为有效数据，即排除了编号为 15 与 17 的文物采样数据。
- 对剩余文物的化学成分组成进行归一化处理，使得每个文物的各化学成分之和为 100%。

通过对整体表格进行观察，我们发现玻璃制品的风化程度与二氧化硅的含量呈现除了强相关性。我们首先对文物先将样本分成高钾和铅钡两类，分别按照二氧化硅含量从大到小排列，我们发现如下规律：

（1）高钾类玻璃制品随着二氧化硅含量增大，表面风化情况从无风化到风化进行转变，并且呈现风化的高钾类玻璃制品二氧化硅含量都在 90% 以上且呈现密集聚拢，无风化的高钾类玻璃制品二氧化硅含量向 60% 左右呈现密集聚拢。

图 4　高钾玻璃的 SiO_2 成分含量的统计图

（2）同样的，铅钡类玻璃制品的二氧化硅与高钾类中的二氧化硅含量呈现了相同的趋势，并且在严重风化点处，二氧化硅含量达到了最小值。在这之中，所有表面已风化但采样点为未风化的点，其二氧化硅含量均处于 45% ~ 70% 之间；而二氧化硅含量处于 31% ~ 40% 的区间时，玻璃制品表面既有已风化又有未风化；二氧化硅含量低于 31%

时，玻璃制品表面都属于已风化的状态。

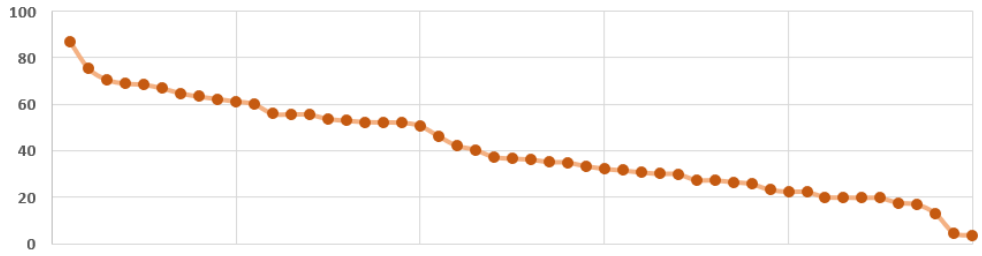


图 5　铅钡玻璃的 SiO_2 成分含量的统计图

结合以上两图以及玻璃是否被风化的具体数据，我们分析得到了如下结果：

（1）高钾类玻璃制品在风化的过程中二氧化硅含量会逐渐增大，无风化的玻璃制品二氧化硅含量处于 60% 左右，风化状态的玻璃制品二氧化硅含量处于 90% 以上，且有一个玻璃制品表面从无风化到风化的临界状态，其二氧化硅含量处于 67% ~ 90% 之间。

（2）铅钡类玻璃制品在风化的过程中二氧化硅含量同样会逐渐减小，无风化的玻璃制品二氧化硅含量均处于 50% 以上，已风化的玻璃制品二氧化硅含量处于 25% 以下，处于无风化状态向风化状态过渡的临界状态的玻璃制品，其二氧化硅含量处于 25%-50% 之间。

由于这两类玻璃制品的二氧化硅含量都有一定的规律，所以可以将这两类玻璃制品分别按照二氧化硅含量多少聚成三类，分别是无风化状态，临界状态和已风化状态。

我们采用的是 K-means 聚类算法进行聚类，原因是：K-means 聚类算法是很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。

该算法认为簇是由距离靠近的对象组成的，所以把紧凑且独立的簇作为最终目标。而通过以上观察，发现处于同一状态下玻璃制品的二氧化硅含量是较为接近，并且不同状态的二氧化硅含量具有明显的差异，所以我们可以通过该算法将玻璃制品分为典型的三类，算法步骤具体如下：

1. 先随机选取 3 个对象作为初始的聚类中心 k_1，k_2 和 k_3。
2. 计算其中一个对象与 3 个聚类中心 k_1，k_2 和 k_3 的距离得 x_1，x_2 和 x_3。
   找出 $\min\{x_i, i = 1, 2, 3\}$，并将该对象分配给 k_i。
3. 重复以下步骤，直到全部对象被分配。
4. 通过欧氏距离重新计算这 3 个聚类的聚类中心，再回到第二步，不断重复。

直到满足以下终止条件：

- 没有对象被重新分配给不同的聚类。
- 没有聚类中心再发生变化。
- 误差平方和局部最小。

最后高钾类玻璃制品按照二氧化硅含量聚成 3 类的情况如下图：

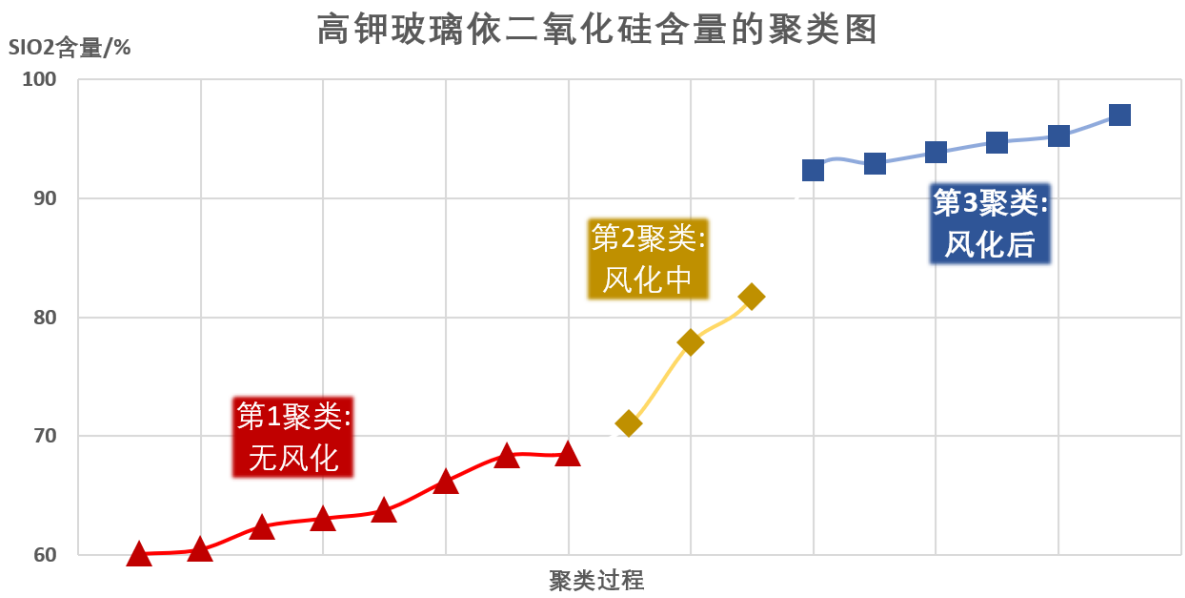


图 6　高钾玻璃依 SiO_2 的 K-means 聚类结果图

同理，铅钡类玻璃制品也聚成 3 类如下图：

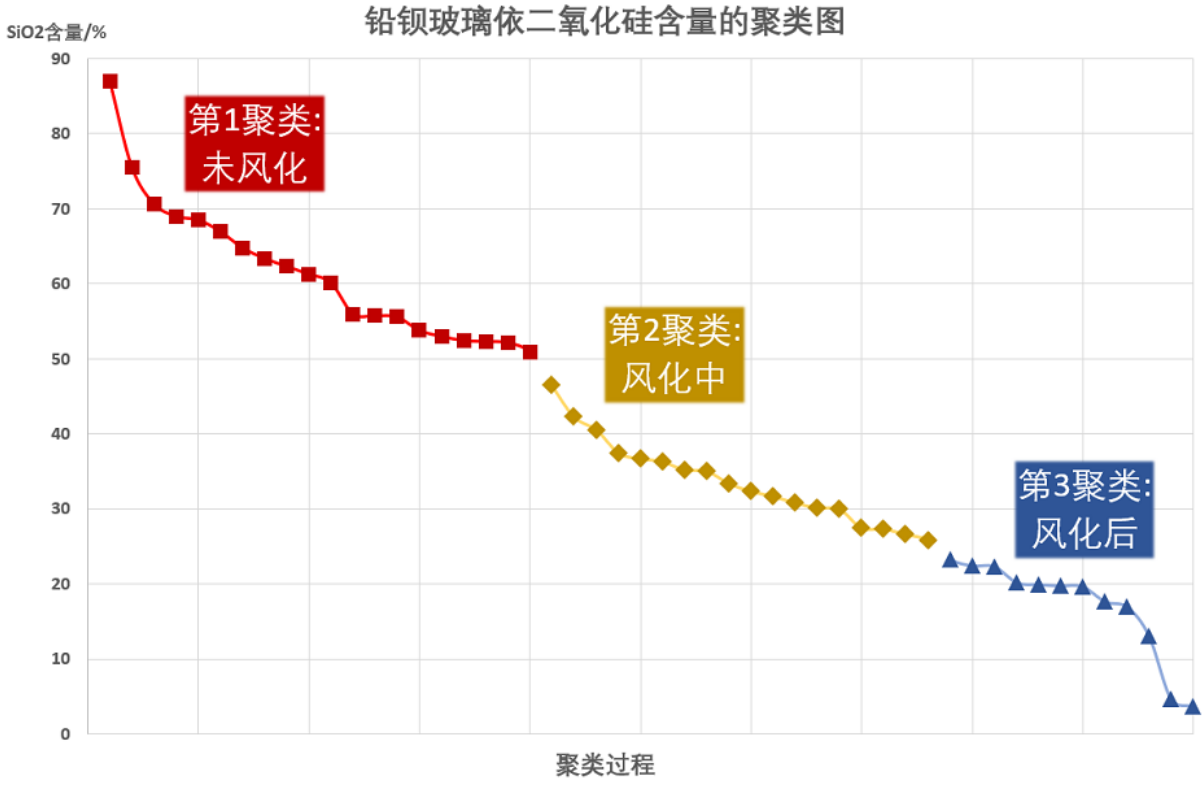


图 7　铅钡玻璃依 SiO_2 的 K-means 聚类结果图

根据以上统计分析规律以及数据的聚类结果，我们将所有的检测点分为两类进行预测，分别为高钾玻璃的检测点以及铅钡玻璃的检测点，我们对以上聚类前的元素中的各化学元素取均值进行计算，从而便可对该风化点的检测数据，进行风化前结果的预测。

对于高钾玻璃部分，由于有个别的化学成分的值为零，我们将其直接剔除，不进行预测; 而对于铅钡玻璃部分，由于含有氧化锡的文物和含有二氧化硫的文物，数量占比过少，故我们同样将其剔除。最终我们对所得结果归一化，并保留两位小数，便得到了我们基于高钾和铅钡两类玻璃的预测结果。

具体的预测结果如下表所示:

表 4　两类玻璃的化学成分含量预测表

高钾玻璃（化学元素/百分比含量）											
SiO_2	Na_2O	K_2O	CaO	MgO	Al_2O_3	Fe_2O_3	CuO	PbO	BaO	P_2O_5	SnO_2
64.99	0.94	11.03	6.5	1.16	7.49	2.36	2.88	0.41	0.59	1.55	0.05

铅钡玻璃（化学元素/百分比含量）											
SiO_2	Na_2O	K_2O	CaO	MgO	Al_2O_3	Fe_2O_3	CuO	PbO	BaO	P_2O_5	SrO
61.57	2.02	0.42	1.16	0.72	5.1	5.9	1.01	19.27	7.02	0.62	0.23

五、 问题二的分析与求解

5.1 问题二的分析

问题二首先是分析高钾玻璃、铅钡玻璃的分类规律。由于事先不知道需要参考几类元素才能完成分类，所以考虑通过机器学习进行分类。并且题目的本质是二分类，所以采用了合适的 CART 决策树分类模型。

亚类划分方面，首先将高钾玻璃与铅钡玻璃分开考虑，由于样本的风化程度不一导致元素含量较为复杂，我们先进行相关性分析找出尽可能不受风化程度影响的其含量的氧化物，再综合评价判断出适合作为亚类划分的氧化物，最后再进行亚类划分。

敏感性分析方面，高钾玻璃我们采取对氧化镁和五氧化二磷的权重比值进行调整分析，探究不同比值对最后分类的影响，铅钡玻璃我们采用对剔除的玻璃制品的二氧化硅含量数值进行调整分析，探究风化的玻璃制品的多少对分类的影响。

5.2 问题二的求解

5.2.1 高钾与铅钡的决策树分类模型

对于高钾玻璃与铅钡玻璃进行分类，我们采用的是 CART 决策树分类模型。通过递归的方法，对分类树用基尼指数最小化准则，进行特征选择，同时决定该特征的最优二值切分点，生成二叉树。

分类问题中，假设有 K 个类，样本点属于第 k 类的概率为 p_k，则概率分布的基尼指数定义为：

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

由于是需要将玻璃制品分成高钾玻璃与铅钡玻璃两类，所以是二分类问题。则此时基尼指数为：$Gini(p) = 2p(1-p)$。

接下来根据数据集，从根节点开始，递归地每个结点进行以下操作，用来构建二叉决策树：

1. 设结点的训练数据集为 D，计算现有特征对该数据集的基尼指数，此时对每一个特征 A，对其可能取的每一个值 a，根据样本点 A=a 的测试为“是”或“否”将 D 分割成 D1 和 D2 两部分，然后计算 $Gini(D, A)$。
2. 在所有可能的特征 A 以及他们所有可能的切分点 a 中，选择基尼指数最小的特征及其对应的切分点作为最优特征与最佳切分点。依最优特征与最优切分点，从现在的结点生成两个子节点，将训练数据集依特征分配到两个子节点中去。
3. 对两个子节点递归调用 1，2，直至满足停止条件。
4. 生成 CART 决策树。

算法停止的条件是结点中的样本个数小于预定的阈值，或样本集的基尼指数小于预定的阈值，或者没有更多的特征，故我们最后可得出如下决策树：

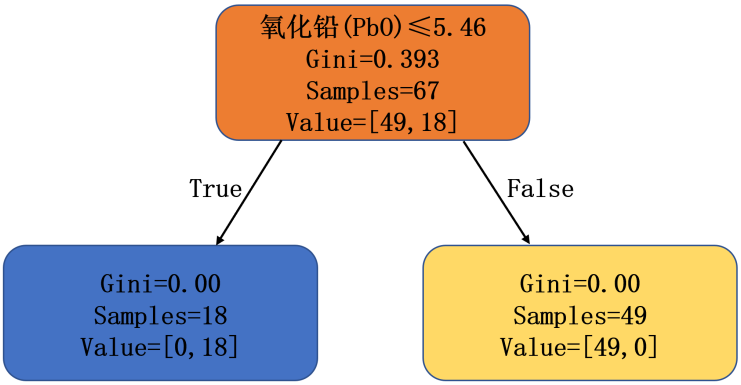


图 8　决策树结果图

因此我们便得到了，将高钾玻璃制品和铅钡玻璃制品区分开来的条件是氧化铅是否

小于 5.46%，此时区分出来的两类数量分别为 18 与 49，与样本数据情况完全吻合，并且氧化铅的特征重要性为 100%。

将模型的评估进行验证，得到的结果如下：

表 5　决策树模型分配评价指标表

	准确率	召回率	精确率	F1
训练集	1	1	1	1

5.2.2 基于相关性分析的亚类划分

接下来是对高钾玻璃制品和铅钡玻璃制品选择合适的化学成分对其进行亚类划分。首先是高钾玻璃，从对玻璃制品分类成高钾玻璃制品和铅钡玻璃制品的过程中可知，存在一些元素的增加与减少与风化过程呈相关性，会导致该元素的含量受到风化原因的干扰而波动较大，而亚类划分是将同类型玻璃制品再根据化学成分进行一次类型细分，所以需要尽可能排除风化原因对元素含量的干扰影响到玻璃制品的划分。下一步我们需要将会干扰的元素尽可能排除，选出不易受风化因素干扰的元素作为亚类划分的依据。

我们首先绘制出了如下的亚类划分算法流程图：

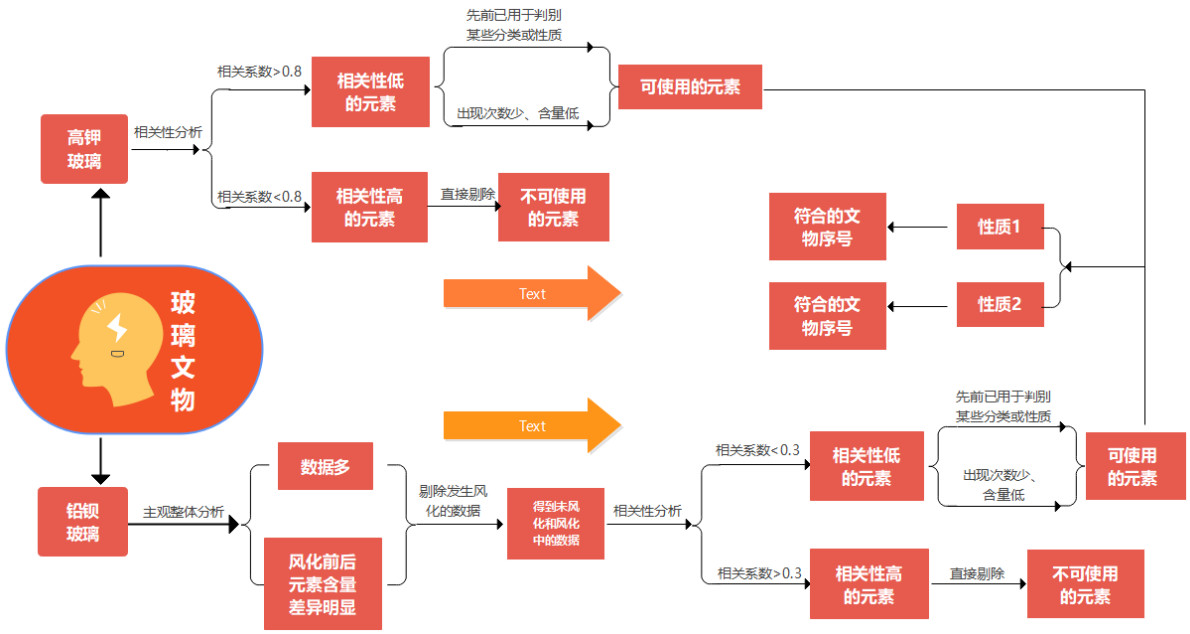


图 9　亚类划分流程图

（1）高钾玻璃的亚类划分

我们首先根据斯皮尔曼相关系数进行相关性分析：

- X 和 Y 是两组数据，我们首先计算 X_i 与 Y_i 之间的等价差 d_i，并将该数据所在的一列按照从小到大排序后，这个数所在的位置，再计算总数据的个数 n。
- 根据如下公式计算出斯皮尔曼相关系数

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n\left(n^2-1\right)} \tag{3}$$

- 通过 r_s 的值，判断其之间的相关性。

在相关性之后，由于风化程度不能被具体表示，但是 SiO_2 的含量是随着风化过程持续体现出强相关性的变化，所以我们将其他所有氧化物对二氧化硅作相关性分析。通过查阅书籍可知相关性绝对值处于 0.3 以下认为没有相关性，0.3 到 0.8 之间可以认为有弱的相关性，0.8 以上认为有强的相关性，综上我们画出下图：

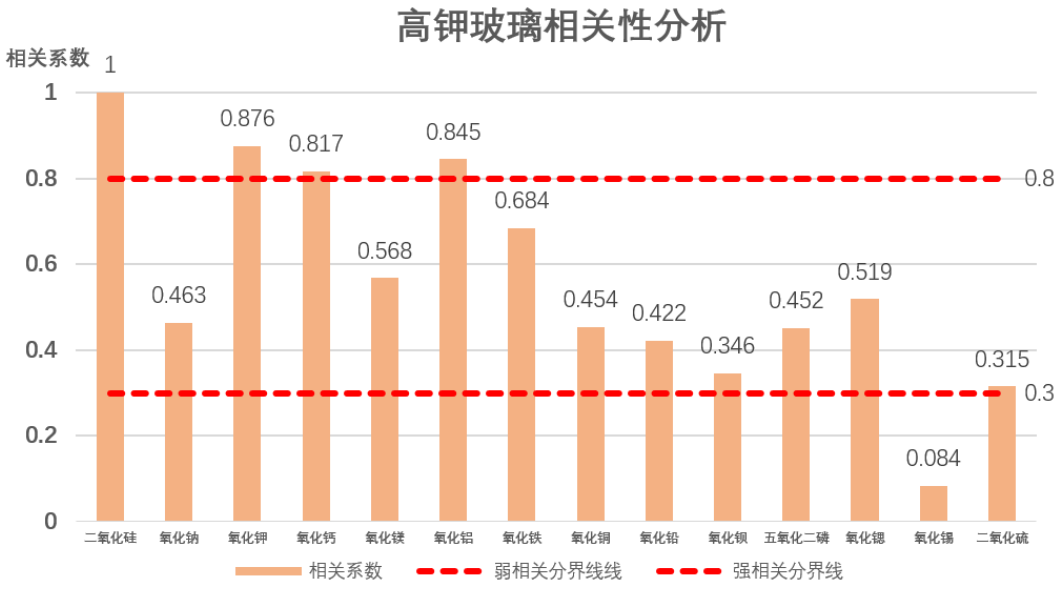


图 10　高钾玻璃相关性分析图

此时氧化锡的相关性最低，为唯一的没有相关性的氧化物。但由于氧化锡的含量十分少，检测的数据可能会存在误差，所以不考虑氧化锡作为划分依据。之后考虑弱相关性的氧化物，同理不考虑含量过低或几乎不怎么存在的氧化物，剔除后剩下：MgO、Fe_2O_3、PbO、BaO 和 P_2O_5。又由于高钾玻璃和铅钡玻璃已经是依据 K_2O、PbO 和 BaO 进行分类的，所以这三类并不适合作为继续亚分类的氧化物，故剔除。查阅资料可知 Fe_2O_3、SrO 在玻璃文物中充当调色剂的作用，由于在表 1 各文物的颜色已经展现出来了，无需依据 Fe_2O_3 含量对高钾玻璃文物进行颜色分类，故也不适合。因此最终经筛选可得到进行亚类划分的氧化物：MgO 和 P_2O_5。

由于 MgO 能提高玻璃的透明度 [1]；P_2O_5 能提高表面结构的“光洁化”或“平面

化”，我们可以认为当 MgO 和 P_2O_5 大于一定含量时可认为该玻璃的透光性较好，视高透光型和不透光型作为亚类划分。

由于考虑的是两类氧化物作为划分的依据，需要先处理 MgO 和 P_2O_5 的权重。我们采用的该方法是首先对所有高钾玻璃制品的 MgO 和 P_2O_5 百分含量进行求和，再对求和后所占比例进行归一化比较为 $0.433:0.577$。为了消除两类氧化物的含量不一致导致指标的影响力不一样的情况，我们将采用修正系数的方法，设 MgO 百分比含量是 X_1，P_2O_5 百分比含量是 X_2，设定透光性值为：

$$Y = 0.557X_1 + 0.443X_2 \tag{4}$$

但因为 MgO 和 P_2O_5 依旧与风化程度有一定的相关性，随着风化程度它们的含量也会减少，所以依据透光性值进行亚分类的同时也要考虑到玻璃制品的风化状态，于是考虑采用二氧化硅含量作为风化状态的依据。以二氧化硅含量作为横坐标，透光性值作为纵坐标进行画图分析，并且使用如下最小二乘法公式进行线性拟合：

$$\begin{cases} \hat{a} = \dfrac{\sum xy - \frac{1}{n}\Sigma x \Sigma y}{\Sigma x^2 - \frac{1}{n}(\Sigma x)^2} \\ \hat{b} = \bar{y} - \hat{a}\bar{x} \\ y = \hat{a}x + \hat{b} \end{cases} \tag{5}$$

最终拟合结果如下所示：

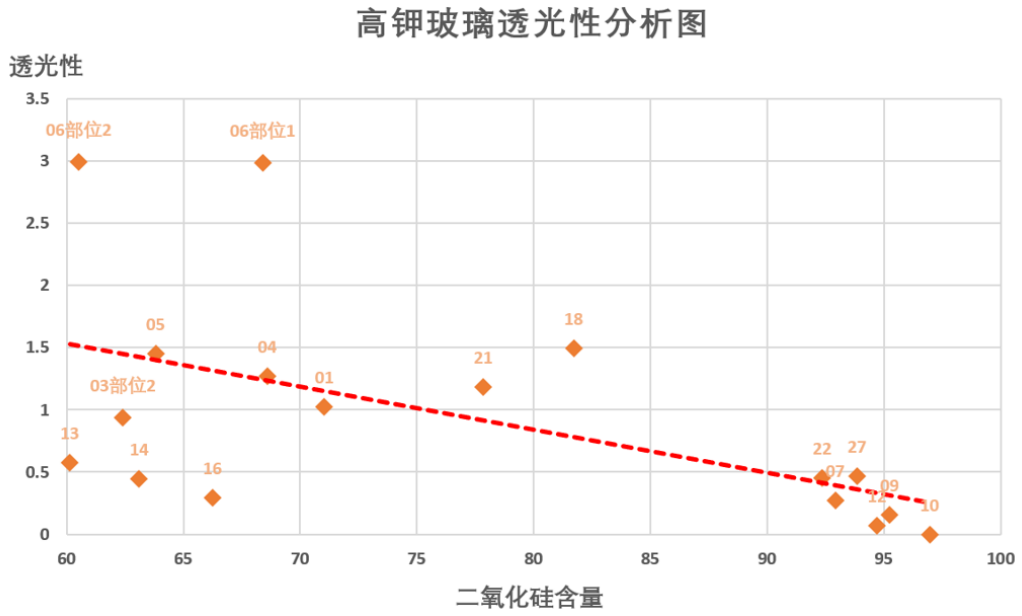


图 11　高钾玻璃透光性分析图

我们将拟合曲线上方点对应的玻璃制品作为高钾透光型，共有 8 个玻璃制品，下方点对应的玻璃制品作为高钾不透光型，共有 9 个玻璃制品，具体的细分点的坐标见附录。

（2）铅钡玻璃的亚类划分

对于铅钡玻璃，若是直接进行相关性分析，会发现大多数氧化物都与二氧化硅的相关性较高，因此可能需要预先处理该数据。基于此前提我们对铅钡玻璃的数据进行了分析，发现发生风化后的数据跨度很大，会对整体的相关性检验存在较大的干扰，因此结合第一问的结论，将玻璃文物分成未风化阶段、分化过程中与风化后，将风化后的玻璃制品数据剔除，留下未风化阶段和风化过程中的数据，再进行不同氧化物与二氧化硅进行相关性分析，得到具有较为良好观测性的相关性分析图如下：

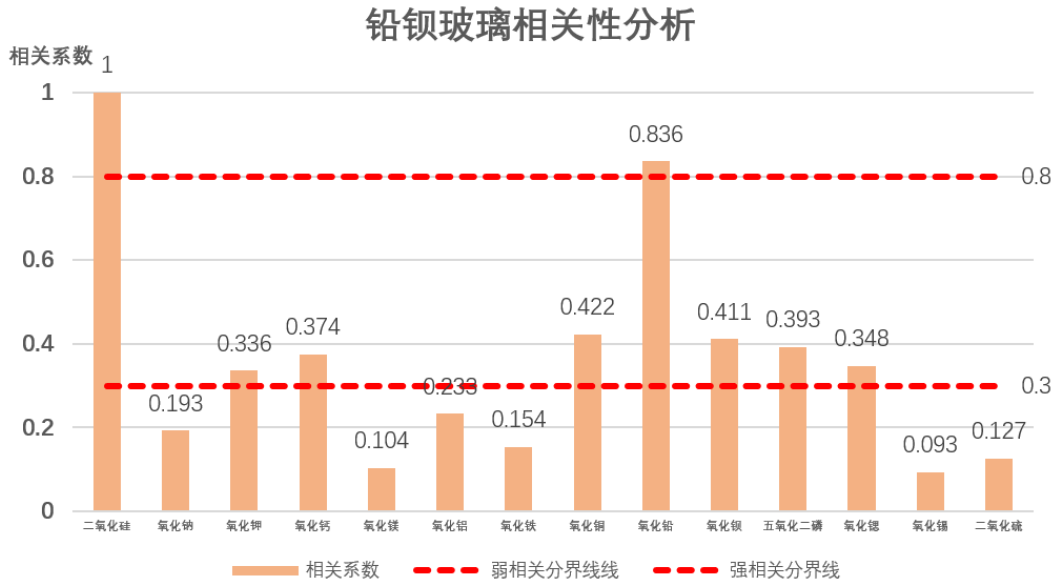


图 12　铅钡玻璃相关性分析图

由于数据已经过一次预处理，所以仅考虑无相关性的氧化物——Na_2O、MgO、Al_2O_3、Fe_2O_3、SnO 和 SO_2。下一步将含量过低的或几乎不怎么存在的氧化物 SnO 和 SO_2 剔除，又由于 Fe_2O_3 在玻璃制品中充当调色剂的作用，颜色已有体现，故剔除。Al 元素可以增加玻璃的化学稳定性，降低玻璃质变可能，结合铅钡玻璃可知，由于存在大量铅钡元素，因此该种类型的玻璃稳定性较好，无需再用 Al 元素进行细分类，因此可以剔除 Al_2O_3。

经以上筛选可得到 Na_2O，且 Na_2O 可以增加玻璃的活泼型以及降低玻璃的融化温度 [1]，所以 Na_2O 大于一定含量时可认为该玻璃的活泼型较好，视作为亚类。接下来用高钾的相同步骤作出下图：

我们将拟合曲线上方点对应的玻璃制品作为铅钡活泼型，共有 12 个玻璃制品，下方点对应的玻璃制品作为铅钡稳定型，共有 26 个玻璃制品。具体的文物数据表可参见附录。

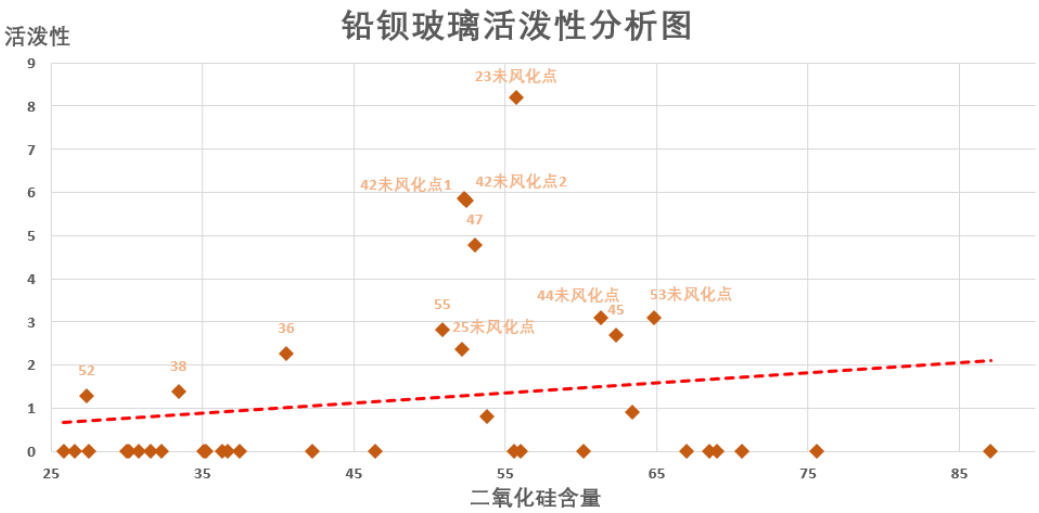


图 13　铅钡玻璃活泼性分析图

5.2.3 敏感性分析

对于高钾玻璃类别，由于纵坐标是对氧化镁和五氧化二磷进行赋值后作为纵坐标，所以我们将对氧化镁和五氧化二磷的比例进行敏感性分析。以氧化镁所占比例为横坐标，最终分类后属于透光型的玻璃文物个数作为纵坐标进行画图分析，如下：

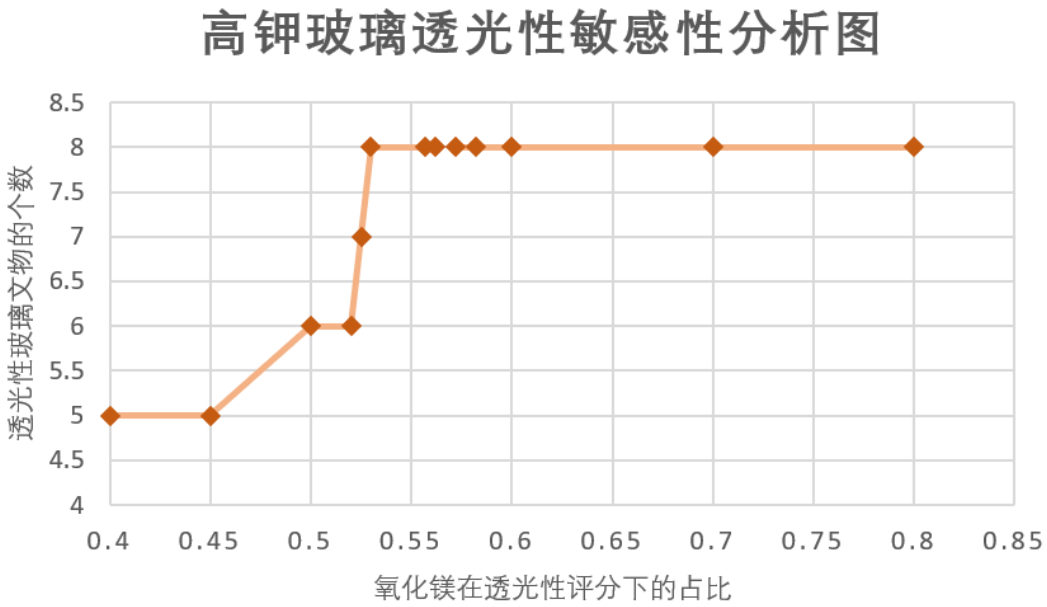


图 14　高钾玻璃透光性敏感性分析图

依图可以看到氧化镁占比处于 [0.52,0.53] 区间时，高钾玻璃文物的分类情况对氧化镁的评分占比非常敏感，而处于 [0.53,0.8] 区间时分类情况对氧化镁的评分占比非常不敏感，敏感性为 0。

对于铅钡玻璃类别，由于开始是将二氧化硅含量小于 25% 的文物判定为已风化并

剔除，现在我们对纳入计算的二氧化硅含量最低值进行敏感性分析。以纳入计算的二氧化硅含量最低值作横坐标，最终分类后属于透光型的玻璃文物个数作为纵坐标进行画图分析，如下：

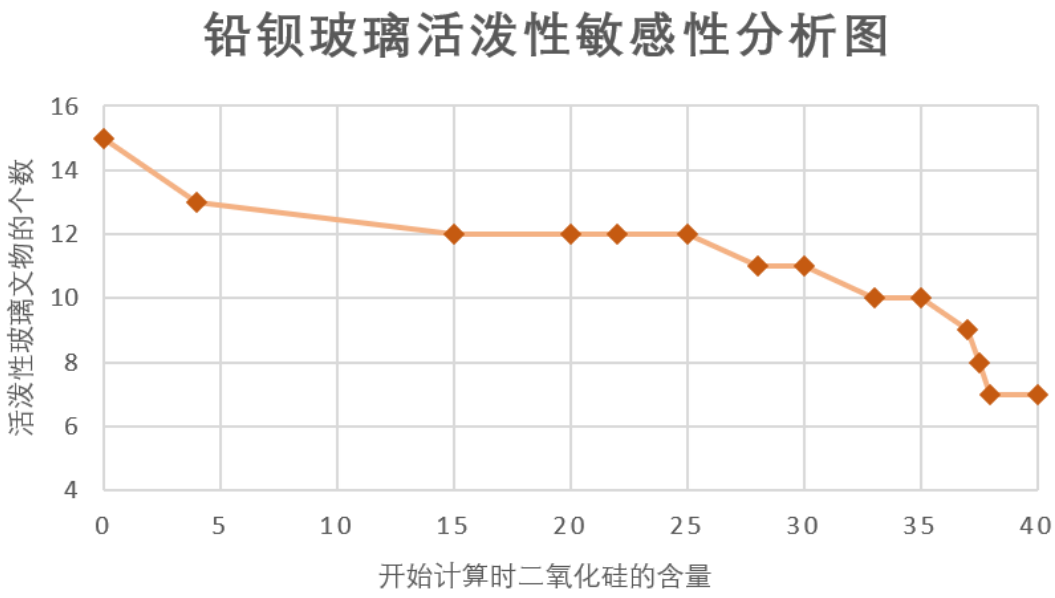


图 15　铅钡玻璃活泼性敏感性分析图

依图可以看到开始计算时二氧化硅的含量处于 [0,15] 区间时，铅钡玻璃文物的分类情况对开始计算的二氧化硅含量不太敏感，处于 [15,25] 区间时完全不敏感，处于 [25,35] 时较为敏感，处于 [35,37.5] 区间时十分敏感。

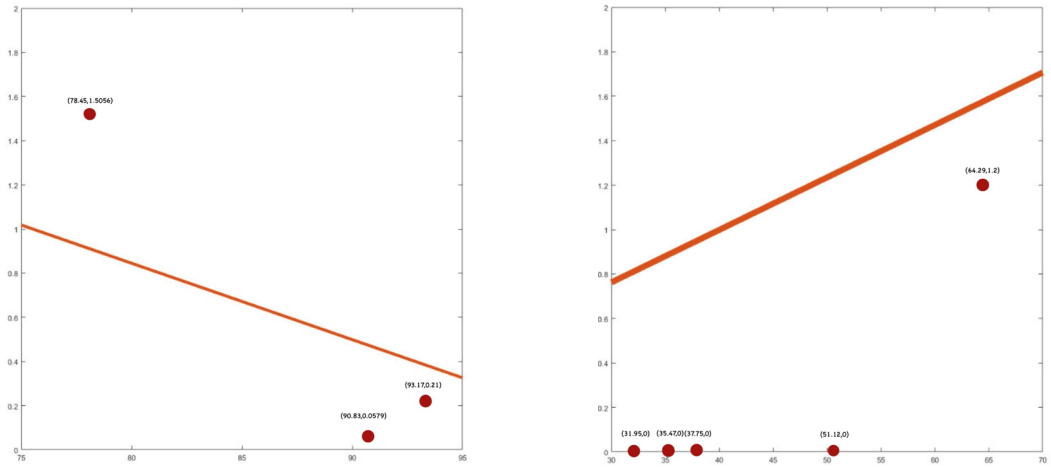
六、 问题三分析与求解

问题 3 首先是对已给的未知类别玻璃文物的化学成分进行分析并鉴别其类型。第一步是区分 A1-A8 这 8 个文物是属于高钾还是铅钡。模型二通过 CART 决策树分类模型给出的分类方法以及得出的分类结果，如下所示：

(a) 分类方法　　**(b) 分类结果**

图 16　决策树模型对文物的分类方法及分类结果

接下来分别判断细分类型，首先判断高钾类玻璃制品 A1,A6 和 A7。第一步求玻璃制品的氧化镁与五氧化二磷的加权平均值，使用了模型二给出的加权平均公式，计算得到 A1,A6 和 A7 的镁磷加权平均值分别为 1.5056，0.21 和 0.05759。再以 A1,A6 和 A7 的二氧化硅含量作为横坐标，镁磷加权平均值作为纵坐标在图中标点，与最小二乘分界函数 $y = -0.0346x + 3.6129$ 作比较如下：

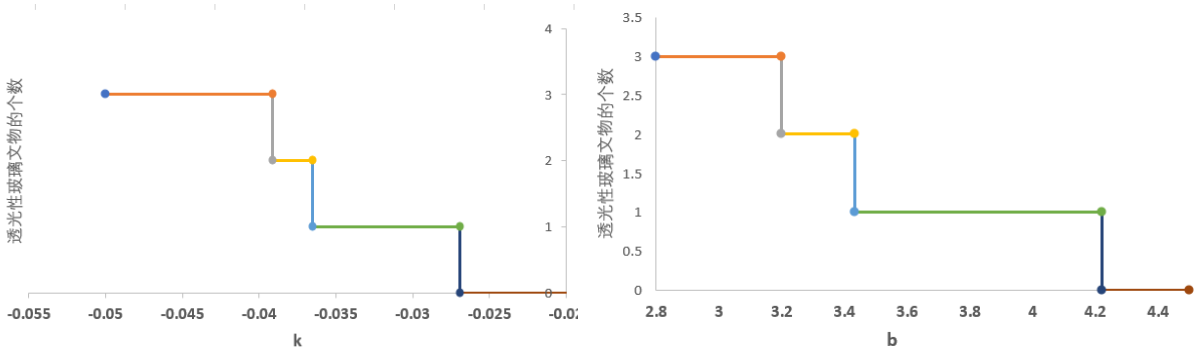


(a) 高钾玻璃的分类　　**(b) 铅钡玻璃的分类**

图 17　附件三文物的分类结果

依图可知 A2,A3,A4,A5 和 A8 均在分界函数以下，均为铅钡稳定型玻璃制品。

下面是对分类结果进行敏感性分析。由于我们分类的依据都是关于文物所表示的坐标轴在分别函数上下的位置进行判断的，不妨对该分界函数的参数进行调整，观察对分类结果的影响情况。由于分界函数都是一元线性函数，形式为 y=kx+b，不妨分别考虑固定 b 改变 k 和固定 k 改变 b 两种情况，同时观察分类情况。高钾玻璃制品如下：



(a) 高钾玻璃的敏感性　　**(b) 铅钡玻璃的敏感性**

图 18　对文物分类结果的敏感性分析

可以发现分界函数在 k=-0.0346 附近不敏感，同时在 b=3.6129 附近也不敏感。

同理可以发现该分界函数在 k=0.0236 附近不敏感，同时在 b=0.0546 附近也不敏感。

七、 问题四的分析与求解

7.1 问题四的分析

问题四总共分成了两个小问，且二者关系是承接关系，因此解题过程要循循渐进。我们首先要清楚题目第一问的要求，是让我们在同一类别对化学成分之间的关联关系进行分析。因此对同一类别中的不同化学成分之间分析关联关系应该采取用相关系数的方法进行量化分析，大于某一固定值时，则认为二者化学成分之间具有关联关系。将所有化学成分的相关系数进行判定后，最终得到了同一类型中各种化学成分之间是否具有关联关系。

对于第二小问，题目要求我们比较不同类别之间的化学成分关联关系的差异性，因此只用将第一小问分析出在一类玻璃中具有相关性的化学成分组合分别与另外三类玻璃具有相关性的化学成分组合进行对比。若某种化学组合在不同的类别之间相关性一致，则表明该种组合在此二类之间的关联关系无差异性；若某种化学组合在不同的类别之间相关性不一致，则表明该种组合在此二类之间的关联关系有差异性。

7.2 问题四的求解

结合第二问的分类可知文物玻璃一共分为四类，其分别是高钾不透光、高钾透光、铅钡活泼和铅钡稳定。我们首先先做出不同元素的含量图（除去 SiO_2）如下图所示：

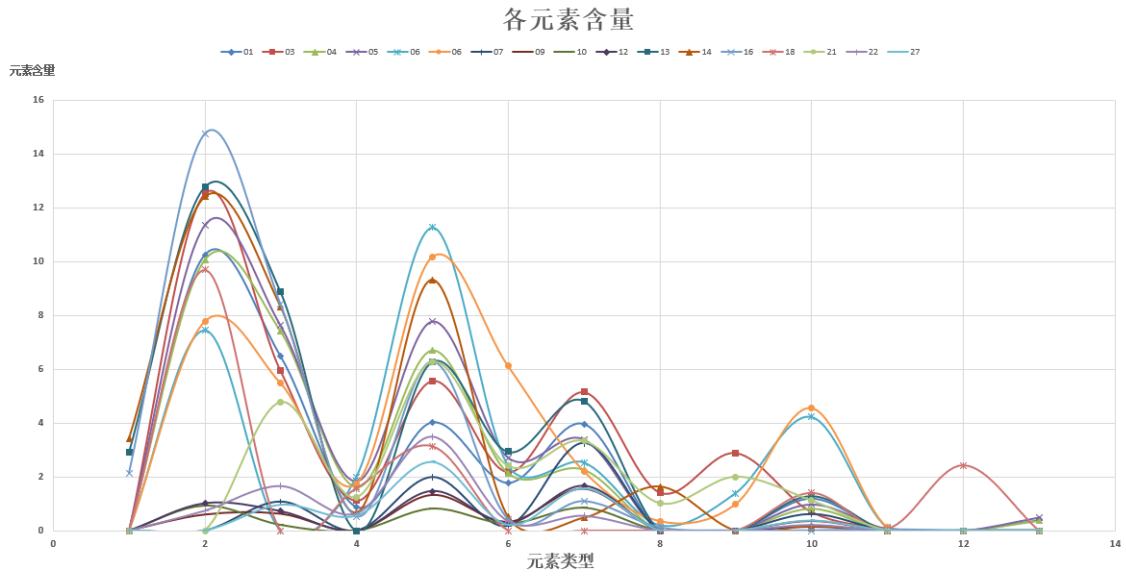


图 19　高钾不透光玻璃文物中各元素含量图

基于此图，可以定性分析出各个元素之间的确是存在着某种关联情况，因此可采用问题二中使用过的斯皮尔曼相关系数进行相关性分析来做定量判断，最终得到四种分类的“相关系数热力图”，即颜色越红正相关性越大、颜色越绿负相关性越大。

以下选取高钾不透光的数据进行展现，剩下的三组类别数据分析类似（附在附件 4 中）高钾不透光各元素之间的相关系数如图所示：

	二氧化硅 (SiO2)	氧化钠 (Na2O)	氧化钾 (K2O)	氧化钙 (CaO)	氧化镁 (MgO)	氧化铝 (Al2O3)	氧化铁 (Fe2O3)	氧化铜 (CuO)	氧化铅 (PbO)	氧化钡 (BaO)	五氧化二磷(P2O5)	氧化锶 (SrO)	氧化锡 (SnO2)	二氧化硫 (SO2)
二氧化硅 (SiO2)	1	-0.514	-0.762	-0.833	-0.862	-0.929	-0.762	-0.262	-0.846	-0.577	-0.371	-0.655	0	-0.082
氧化钠 (Na2O)	-0.514	1	0.592	0.733	0.282	0.764	0.265	-0.655	0.75	-0.216	-0.4	0.173	0	-0.216
氧化钾 (K2O)	-0.762	0.592	1	0.762	0.761	0.762	0.762	-0.024	0.764	0.412	-0.108	0.733	0	0.082
氧化钙 (CaO)	-0.833	0.733	0.762	1	0.685	0.952	0.595	0	0.682	0.082	0.132	0.452	0	0.247
氧化镁 (MgO)	-0.862	0.282	0.761	0.685	1	0.736	0.939	0.368	0.668	0.615	0.517	0.581	0	0.439
氧化铝 (Al2O3)	-0.929	0.764	0.762	0.952	0.736	1	0.643	-0.024	0.846	0.247	0.168	0.483	0	0.082
氧化铁 (Fe2O3)	-0.762	0.265	0.762	0.595	0.939	0.643	1	0.31	0.627	0.577	0.467	0.546	0	0.412
氧化铜 (CuO)	-0.262	-0.655	-0.024	0	0.368	-0.024	0.31	1	-0.218	0.577	0.778	0.327	0	0.412
氧化铅 (PbO)	-0.846	0.75	0.764	0.682	0.668	0.846	0.627	-0.218	1	0.472	-0.041	0.589	0	-0.283
氧化钡 (BaO)	-0.577	-0.216	0.412	0.082	0.615	0.247	0.577	0.577	0.472	1	0.415	0.756	0	-0.143
五氧化二磷(P2O5)	-0.371	-0.4	-0.108	0.132	0.517	0.168	0.467	0.778	-0.041	0.415	1	0.024	0	0.581
氧化锶 (SrO)	-0.655	0.173	0.733	0.452	0.581	0.483	0.546	0.327	0.589	0.756	0.024	1	0	-0.216
氧化锡 (SnO2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
二氧化硫 (SO2)	-0.082	-0.216	0.082	0.247	0.439	0.082	0.412	0.412	-0.283	-0.143	0.581	-0.216	0	1

图 20　高钾不透光玻璃各元素斯皮尔曼相关系数热力图

结合相关性范围指标，将相关系数 $r_s > 0.6$ 的二者元素视作为具有较强正相关性，将相关系数 $r_s < -0.6$ 的二者元素视作为具有较强负相关性。从中我们可初步得出个元素之间的相关性质，从而有针对性地进行以下的定量分析。

基于此可得到高钾不透光中具有相关性的二者元素的归类，具体的归类结果，我们放在了支撑材料当中。

依据上述表格可以得到，同种类型化学成分之间的关联关系，如 Na_2O 与 CaO 是强正相关的关联关系，SiO_2 与 K_2O 是强负相关的关联关系。

基于此种关联关系，分别将其放置于不同类别之间观察是否有相同的关联关系出现。若某一种关联关系在不同的类别文物之间相同，即表明该种关联关系在其两类别之间无差异性；若某一种关联关系在不同的类别文物之间不同，即表明该种关联关系在其两类别之间存在差异性。

将某一类文物之中的所有元素之间存在的关联关系与四种类别之间的任意二类进行比较即可得到其所有化学成分关联关系的差异性。

以下选取每一类中的关联关系对其他不同类分别进行一一对比，最终即可得到如下结论：

图 21　关联关系和差异性判定图

八、 模型的优缺点

8.1 模型的优点

（1）模型具有坚实可靠的数学基础且易于实现。

（2）模型进行分类前都会判定不同元之间的相关性，从而能够规避严重风化点对分类的干扰因素。

（3）模型能够尽可能的排除因风化造成元素的流失从而影响后续的分类工作。

（4）模型对敏感性分析制作成函数，均验证了模型的敏感性高。

8.2 模型的缺点

（1）研究样本数较少，直接拿频率充当概率。

（2）对高钾和铅钡的分类，未直接对 Ba 和 K 元素进行考虑。

（3）模型对风化过程中不同时期的预测采用均值估计，可能造成偏差。

参考文献

[1] “玻璃材料与玻璃技术”专题 [J]. 硅酸盐通报,2022,41(04):1483.

[2] 学兵, 张俊. 决策树算法及其核心技术 [J]. 计算机技术与发展,2007(01):43-45.

[3] 序平, 姚玉兰. 利用 Q-Q 图与 P-P 图快速检验数据的统计分布 [J]. 统计与决策, 2010, 000(020):151-152.

附录 A　支撑文件目录

- code_cart_decide_tree.py
- code_spearman.py
- fun_cskmeans.m
- fun_kfjy.m
- m_KSiO2.xls
- m_PbSiO2.xls
- run_kfjy.m
- fun_kmeans.m
- t1_1_ 卡方检验.xlsx
- t1_2_ 高钾玻璃 k-means 聚类.xlsx
- t1_2_ 铅钡玻璃 k-means 聚类.xlsx
- t1_3_ 数据预测.xlsx
- t2_1_ 决策树数据重要性分析.xlsx
- t2_2_ 高钾相关性.xlsx
- t2_2_ 铅钡相关性.xlsx
- t2_2_ 相关系数表.xlsx
- t3_ 敏感性分析.xlsx
- t4_ 相关性分析.xlsx

附录 B　模型计算代码

2.1　卡方检验-MatLab 程序

```
function [p, Q]= fun_kfjy(x)
   % 计算 = sum( (a-np*)^2/(np*(1-p*)) )
   s=   size(x, 1);
   r=   size(x, 2);
   np=     sum(x, 2)/sum(sum(x)) * sum(x);  % p=sum(x, 2)/sum(sum(x)) and n=sum(x)
   Q=   sum(sum((x-np).^2./(np)));
   % 计算卡方检验的概率p
   p=   1 - gammainc(Q/2, (r-1)*(s-1)/2);
end

%函数的使用部分
%卡方检验计算的matlab实现
%变量分析
%类型与纹饰
[p_1,Q_1] = fun_kfjy([15,10; 7,11; 9,7; 20,15; 26,21; 19,16]);
%类型与颜色
[p_2,Q_2] = fun_kfjy([12,3;4,16;0,4;1,6;1,1;0,3;0,2;0,1]);
%纹饰与颜色
[p_3,Q_3] = fun_kfjy([3,6,6;10,10,0;4,0,0;7,0,0;0,2,0;3,0,0;0,2,0;1,0,0]);
%%%%%%
%风化程度分析
%类型与风化程度
[pf_1,Qf_1] = fun_kfjy([12,12;6,24]);
%纹饰与风化程度
[pf_2,Qf_2] = fun_kfjy([13,11,0;15,9,6]);
%颜色与风化程度
[pf_3,Qf_3] = fun_kfjy([6,8,2,3,2,2,1,0;9,12,2,4,0,1,0,2]);
%备注:
%Q代表卡方的值，p代表的是概率，其中当p<0.05时，数据之间呈现显著性差异
```

2.2　k-means 均值聚类-matlab 源代码

```
%首先对k-means函数进行定义，分别使用x,k0,nc为其进行赋值
function [mat,nr,center] = cskmeans(x,k0,n2)
warning off
%首先取消警告
[n0,d] = size(x);
%进行安全性判定
if nargin < 3
dadj = ceil(n0*rand(1,k0));
%生成随机数
```

```
n2 = x(dd,:) + randn(k0,d);
end
mat = zeros(1,n0);
oldmat = ones(1,n0);
nr = zeros(1,k0);
%为矩阵复制
maxiter = 100;
temp = 1;
while ~isequal(mat,oldmat) & temp < maxiter
for i = 1:n0
%计算距离之和
dist = sum((rep(x(i,:),k0,1)-n2).^2,2);
[m,dd] = min(dist);
mat(i) = dd;
end
%进行循环
for i = 1:k0
dd = find(mat==i);
n2(i,:) = mean(x(dd,:));
%计算距离nr
nr(i) = length(dd);
end
temp = temp + 1;
end
maxiter = 2;
temp = 1;
move = 1;
%对聚类的循环进行判定
while temp < maxiter & move ~= 0
move = 0;
for i = 1:n0
%计算与聚类中心之间的距离
dist = sum((rep(x(i,:),k0,1)-n2).^2,2);
d = mat(i);
dadj = nr./(nr+1).*dist';
%求出其中的最小距离的情况
[m,dd] = min(dadj);
if dd ~= d
mat(i) = dd;
ic = find(mat == dd);
n2(dd,:) = mean(x(ic,:));
move = 1;
%如果存在最小距离，则对聚类中心进行移动
end
end
temp = temp+1;
end
```

```
center = n2;
if move == 0
warning on

%函数的调用部分
%读取文件
x1=xlsread('KSiO2.xls','sheet1','C2:C18');
x=x1';
k=3;
n=[1;1;1];
%根据题目条件设定相应矩阵
cskmeans(x,k,n);
%读取第二份文件
x2=xlsread('PbSiO2.xls','sheet1','C2:C18');
x0=x2';
k=3;
n=[1;1;1];%根据题目条件设定相应矩阵
cskmeans(x0,k,n);
```

2.3 斯皮尔曼相关性分析-python 代码

```
import pandas as pd
df=pd.DataFrame({open(data1.xlsx)})
print(df.corr('spearman'))
```

2.4 决策树-python 代码

```
from math import log

import csv
import os
def reader(filename):

    base_path=os.path.dirname(__file__)
    path=base_path.replace("func","test_data/"+filename)#
    list = []
    with open(path) as file:
        table=csv.reader(file)
        i=0
        for row in table:
            if i==0:
                pass
            else:
                list.append(row)
```

```
            i=i+1
    return list

def build_datainitialization():
    datainitialization = open()
    characteristic = ['age', 'work', 'house', 'credit']
    return datainitialization, characteristic


def countGini(datainitialization):

    long_of_lizi = len(datainitialization)
    tagcalc = {}

    for lizi in datainitializatipathon:

        nowtag = lizi[-1]

        if nowtag not in tagcalc.keys():
            tagcalc[nowtag] = 0
        tagcalc[nowtag] += 1

    for key in tagcalc:
        tagcalc[key] /= long_of_lizi
        tagcalc[key] = tagcalc[key] * tagcalc[key]

    Gini = 1 - sum(tagcalc.values())
    return Gini


def create_sub_datainitialization(datainitialization, index, value):
    sub_datainitialization = []
    for lizi in datainitialization:
        now_label = []
        if lizi[index] == value:
            now_label = lizi[:index]
            now_label.extend(lizi[index + 1 :])
            sub_datainitialization.append(now_label)
    return sub_datainitialization


def division_datainitialization(datainitialization, index, value):
    sub_datainitialization1 = []
    sub_datainitialization2 = []
    for lizi in datainitialization:
        now_label = []
        if lizi[index] == value:
```

```
            now_label = lizi[:index]
            now_label.extend(lizi[index + 1 :])
            sub_datainitialization1.append(now_label)
        else:
            now_label = lizi[:index]
            now_label.extend(lizi[index + 1 :])
            sub_datainitialization2.append(now_label)
    return sub_datainitialization1, sub_datainitialization2

def select_good_characteristic(datainitialization):

    mathnum = len(datainitialization[0]) - 1

    goodGini = 1

    index_of_good_charateristic = -1

    for i in range(mathnum):

        particularval = set(lizi[i] for lizi in datainitialization)

        Gini = {}

        for value in particularval:

            sub_datainitialization1, sub_datainitialization2 =
                division_datainitialization(datainitialization,i,value)

            hib1 = len(sub_datainitialization1) / float(len(datainitialization))
            hib2 = len(sub_datainitialization2) / float(len(datainitialization))

            Gini_of_sub_datainitialization1 = countGini(sub_datainitialization1)

            Gini_of_sub_datainitialization2 = countGini(sub_datainitialization2)

            Gini[value] = hib1 * Gini_of_sub_datainitialization1 + hib2 *
                Gini_of_sub_datainitialization2

            if Gini[value] < goodGini:
                goodGini = Gini[value]
                index_of_good_charateristic = i
                good_division_spot = value
    return index_of_good_charateristic, good_division_spot


def find_label(roomnum):
```

```
    tagcalc = {}
    for key in roomnum:
        if key not in tagcalc.keys():
            tagcalc[key] = 0
        tagcalc[key] += 1


    kind_tagcalc = sorted(tagcalc.items(), key = lambda a:a[1], reverse = True)

    return kind_tagcalc[0][0]


def build_determine_tree(datainitialization, characteristic):

    tag_num = [lizi[-1] for lizi in datainitialization]

    if tag_num.count(tag_num[0]) == len(tag_num):
        return tag_num[0]

    if len(datainitialization[0]) == 1:
        return find_label(tag_num)

    index_of_good_charateristic, good_division_spot =
         select_good_characteristic(datainitialization)

    good_charateristic = characteristic[index_of_good_charateristic]

    determin_tree = {good_charateristic: {}}
    del(characteristic[index_of_good_charateristic])

    sub_labels = characteristic[:]


    sub_datainitialization1, sub_datainitialization2 =
         division_datainitialization(datainitialization,index_of_good_charateristic,good_division_spot)

    determin_tree[good_charateristic][good_division_spot] =
         build_determine_tree(sub_datainitialization1, sub_labels)

    determin_tree[good_charateristic]['others'] = build_determine_tree(sub_datainitialization2,
         sub_labels)
    return determin_tree


def cluster(determin_tree, characteristic, try_lizi):

    one_characteristic = list(determin_tree.keys())[0]
```

```
    two_place = determin_tree[ one_characteristic]

    index_of_one_characterisitic = characteristic.index( one_characteristic)

    for key in two_place.keys():

        if key != 'others':
            if try_lizi[index_of_one_characterisitic] == key:

                if type(two_place[key]).__name__ == 'dict':

                    roomlist = cluster(two_place[key], characteristic, try_lizi)

                else:

                    roomlist = two_place[key]

            else:

                if isinstance(two_place['others'],str):
                    roomlist = two_place['others']

                else:
                    roomlist = cluster(two_place['others'], characteristic, try_lizi)
    return roomlist

if __name__ == '__main__':
    datainitialization, characteristic = build_datainitialization()
    determin_tree = build_determine_tree(datainitialization, characteristic)

    print(determin_tree)

    characteristic = ['age', 'work', 'house', 'credit']
    try_lizi = ['midlife', 'yes', 'no', 'great']
    print(cluster(determin_tree, characteristic, try_lizi))
```