基于统计与机器学习分析风化对古代玻璃的影响及对其成分的预测与鉴别

摘要

风化在古代文物中属于常见现象，然而由于风化会对文物外观及内部物质成分造成不同程度的伤害和影响，故研究风化的成因及鉴别风化后的文物都是十分重要的课题。本文将根据给定的 58 件古代玻璃文物的外形信息及化学成分信息，通过假设检验、统计值计算等统计方法，运用机器学习等分析手段，探究风化对古代玻璃的影响及对其成因的预测与鉴别这一综合问题。

对于问题一，我们首先使用卡方检验对风化状态及其玻璃类型、纹饰和颜色的关系进行分析，发现表面风化状态仅与玻璃类型有显著关联，与纹饰及颜色并无明显关联性；同时，通过对两种玻璃风化前后的化学成分含量及密度分布进行分析，发现其风化前后化学成分的变化不尽相同，风化过程对各自化学成分的改变也有着相反的趋势：高钾玻璃在风化后二氧化硅平均含量明显增加，而氧化钾、氧化钙、三氧化二铝的平均含量明显降低；铅钡玻璃在风化后二氧化硅平均含量明显减少，而氧化钙、氧化铅的含量明显增加。此外，通过使用均值分析和中位数分析，我们预测出已风化文物检测点在风化前的各化学成分含量。

对于问题二，我们先通过绘制比较两类玻璃各化学成分的密度图，对其分类规律产生一般认识，再使用随机森林算法，将学习训练得到决策树的分类结果与我们得到的一般认识加以验证，找到了高钾玻璃与铅钡玻璃的分类规律，及：高钾玻璃中二氧化硅、氧化钾含量高，氧化钡、氧化铅、氧化锶、五氧化二磷含量低，铅钡玻璃与之刚好相反。为对这两大类玻璃进行亚分类，我们使用了 Kmeans 聚类算法和以主成分分析法（PCA）为代表的四种降维方法对数据进行合理降维，进一步划分玻璃亚类，得到了 G1-G5 五种玻璃亚类，其中 G1、G2 属于高钾玻璃，G3-G5 属于铅钡玻璃。

对于问题三，我们基于问题二所得到的训练模型，得到了八种玻璃的鉴定结果：A1、A6-A8 为高钾玻璃，A2-A5 为铅钡玻璃；通过模拟实验中会产生的误差，我们通过对原数据做整体扰动来进行敏感性分析，发现测量误差稳定，鉴定结果准确。

对于问题四，通过绘制两类玻璃内两两成分间的线性相关热力图，我们发现高钾玻璃内部各化学成分间的正负相关性均强于铅钡类型玻璃内部的化学成分。此外，虽两类玻璃的二氧化硅的含量与其他成分含量均基本呈负相关，但其分别与两类玻璃的主要助燃剂成分的负相关性最强。

关键字：古代玻璃风化　随机森林算法　PCA 降维　Kmeans 聚类算法

一、 问题简述

1.1 问题背景

玻璃的制造在我国有着悠久的历史。自汉代丝绸之路开辟以来，我国与西方地区的文化交流日趋繁荣，其中就包括玻璃制品的制作工艺。在吸纳学习了西方的技术后，我国就地取材，发展了自己的玻璃制造术，也有许多精美的玻璃制品历经千年流传下来。通过对这些古代玻璃进行研究与分类，将推动我国考古、历史等领域的发展 [1]。

古代玻璃的主要成分是二氧化硅 (SiO_2)，根据在炼制时所加助熔剂的不同可将其分成不同的类型，如加入草木灰的高钾玻璃和以铅矿石为助熔剂的铅钡玻璃。其中一些化学元素受埋藏的环境影响，极易与外部环境元素进行交换形成风化层，为文物保护和其外型美观带来威胁，故研究风化与古代玻璃的化学成分是一个重要的课题 [2]。

1.2 题目重述

根据附件给出的 58 件文物的类型和外观信息，以及对其不同风化点和无风化点的化学成分鉴定，完成:

1. 分析玻璃文物表面风化与玻璃类型、纹饰和颜色的关系
2. 总结风化文物与无风化文物表面化学成分含量的统计规律，并预测已风化文物风化前的化学成分含量
3. 阐述高钾玻璃和铅钡玻璃的分类规律
4. 对高钾玻璃和铅钡玻璃进行亚类划分
5. 鉴别附件表单三中的 8 件文物所属类别
6. 分析不同类别的文物中化学成分之间的关联

二、 问题分析

2.1 利用卡方检验探索风化与其他因素的关系

问题一的第一部分需要求出风化与其他三个分类型变量的关系，满足卡方独立性检验的要求。首先，附件中提供了 58 个古代玻璃文物的数据，其数据量大于该检验方法所需的最小样本量；其次，通过性质相近的行列合并或直接省略该行/列的方法，可使每个因素的理论频数都达到该检验方法所需的最小理论频数。最后，通过设立零假设并利用四格表法求出两变量的卡方值、显著性水平，决定是否拒绝假设。

2.2 利用探索文物表面风化前后化学成分含量的统计规律并进行预测

问题一的第二部分需要分别找出两类玻璃风化前后表面化学成分含量的统计规律，并预测已风化玻璃在风化前化学成分的含量。基于两类玻璃在风化前后的样本量较小，选择用平均值与中位数相结合的方法揭示其统计规律，这样可减少因极端数值而造成平均值的不准确。其次，根据风化前后同一化学成分的分布特点选择对其进行按比例缩放无风化点的平均数，或以无风化点的中位数代替的方法对风化点进行预测。

2.3 利用随机森林探寻两种古代玻璃的分类规律

问题二的第一部分要分析高钾玻璃和铅钡玻璃的分类规律。可先通过绘制两类玻璃中各化学成分含量的密度对比图对其分类标准有一定基本认识，再通过随机森林算法得出较为准确的分类规律，以以两者互相加以验证。

2.4 利用聚类分析对两类玻璃进行亚分类

问题二的第二部分需要基于某种化学元素，将高钾玻璃和铅钡玻璃划分成更小的子类别。可通过 Kmeans 聚类算法及以主成分分析法（PCA）为代表的四种降维方法先对数据进行合理降维，再进一步进行玻璃亚类划分。

2.5 利用问题二中机器学习的结果将八种类型未知的古代玻璃分类

问题三需要对附录“表单三”中给出的八件玻璃文物进行分类。将未知文物的化学成分数据导入问题二中已经过学习和训练的决策树中即可得到鉴定结果。通过合理范围内地增大与减小各文物原化学成分含量来模拟实验过程中产生的误差，再通过观察鉴定结果是否发生改变来进行敏感性分析。

2.6 利用皮尔森相关系数衡量两种玻璃中化学成分间的关系

问题四需要分析高钾、铅钡玻璃中化学成分的关系。通过计算各成分两两间的皮尔森相关系数，绘制热图，即可看出化学成分间的正负相关性。

三、 基本假设与数据预处理

3.1 模型假设

- **假设 1：**本文所用数据均为样本的随机取样。
  ↪ **解释：**赛题中明确表明“文物采样点为该编号文物表面某部位的随机采样”。[2] 同时，本文所用的建模方法及检验方法，均建立在样本是随机变量的基础上。

- **假设 2：**将附件中同一文物的不同部位点、同一文物的风化点和无风化点当作多个颜色、纹饰和类型相同的不同文物处理。
  ↪ **解释：**据附件描述，文物的不同部位具有造型上的不同，其化学成分与含量可能存在差异，故可将其看作独立的不同文物。

3.2 数据预处理

按照题目的要求，仅保留附件中成分比例累加和介于 85% ∼ 105% 区间的数据。

四、 模型的建立与求解

4.1 问题一

4.1.1 分析表面风化与玻璃类型、纹饰和颜色的关系

卡方检验的一般步骤均可适用于分析风化与三个因素间的关系。此处进行基本步骤阐述。

- 假设两个事件 A 和 B, 分别对其进行划分后可得其分区数分别为 s,k.
- 计算自由度：$(s-1)(k-1)$
- 收集实际频数 o_{ij}，　$i=1,2,\cdots m,\cdots s; j=1,2,\cdots n,\cdots k$
- 计算理论频数 e：

$$e_{mn}=\frac{\sum_{i=1}^{s}N_{im}\sum_{j=1}^{k}N_{nj}}{\sum_{i=1}^{s}\sum_{j=1}^{k}N_{ij}} \tag{1}$$

其中，N_{ij} 表示同时满足 A_iB_j 的数量

- 计算卡方值：

$$\chi^2=\sum_{i=1}^{s}\sum_{j=1}^{k}\frac{(o_{ij}-e_{ij})^2}{e_{ij}} \tag{2}$$

- 取显著性水平 $\alpha=0.05$ 与 $\alpha=0.01$, 查阅卡方分布临界值表决定是否拒绝零假设

1. 风化与玻璃类型

   设立零假设：H_0：玻璃表面风化与玻璃类型无关。

   如表 1中的数据所示，风化与否与两种玻璃类型的理论频数均大于 5，且样本总数大于 40，故利用公式 (2) 进行计算，求得其卡方值。查阅卡方分布临界值表，可知其显著性水平大于 99%，故拒绝零假设，及风化与玻璃类型之间存在极显著的关联性。

原始数据				理论值			
有无风化	高钾	铅钡	行和	有无风化	高钾	铅钡	总和
风化	6	28	34	风化	10.55172414	23.44827586	34
无风化	12	12	24	无风化	7.448275862	16.55172414	24
列和	18	40	58	总和	18	40	58
自由度及卡方理论值				卡方计算值			
自由度	1			卡方值	6.880392157		
概率水平	0.05			0.01			
卡方理论值	3.84			6.63			
统计推断结论	0.05	风化与否与玻璃类型存在显著的关联性					
	0.01	风化与否与玻璃类型存在极显著的关联性					

表 1　玻璃表面风化与类型的卡方检验数据

2. 风化与纹饰

设立零假设 H_0：玻璃表面风化与纹饰无关。

表 2中汇总了附件中所给的 58 件文物的风化程度与纹饰信息。可发现，纹饰 B 的两个理论频数均未达到卡方检验所需的最小频数。

原始数据					理论值		
有无风化	纹饰 A	纹饰 B	纹饰 C	行和	纹饰 A	纹饰 B	纹饰 C
风化	11	6	17	34	12.89655172	3.517241379	17.5862069
无风化	11	0	13	24	9.103448276	2.482758621	12.4137931
列和	22	6	30	58	22	6	30

表 2　玻璃表面风化与纹饰的原始数据汇总和理论值计算

考虑其自由度为 2，无法使用一般的卡方修正公式进行修正，并且附件中并未提及纹饰之间的相似性，不可将纹饰 B 贸然合并到其他两种纹饰之间。由数据所示，拥有 B 纹饰的文物是以纹饰划分的三类文物中中唯一一组全部风化的文物，而其他两类纹饰中风化和无风化的文物数量基本相当，故可以将其认定为是 B 纹饰内部的特性，而不能代表整体的文物表面风化与纹饰的关系，决定省略该项。

省略后的信息和计算结果汇总在表 3中，可见其显著性水平小于 95% ，故无法拒绝原假设，表明风化与否与玻璃纹饰无显著性关联。

原始数据				理论值			
有无风化	纹饰 A	纹饰 C	行和	有无风化	纹饰 A	纹饰 C	总和
风化	11	17	28	风化	11.84615385	16.15384615	28
无风化	11	13	24	无风化	10.15384615	13.84615385	24
列和	22	30	52	总和	22	30	52
自由度及卡方理论值				卡方计算值			
自由度	1			卡方值	0.226984127		
概率水平	0.05			0.01			
卡方理论值	3.84			6.63			
统计推断结论	0.05	风化与否与玻璃纹饰不存在显著的关联性					
	0.01	风化与否与玻璃纹饰不存在极显著的关联性					

表 3　玻璃表面风化与纹饰的卡方检验数据

3. 风化与颜色

设立零假设 H_0：玻璃表面风化与颜色无关。

表 4中汇总了附件中给出的 58 件文物的风化程度与颜色数据。可以看出，除浅蓝的数据量达到卡方检验所需的最小频数外，其他值过于分散，故在计算卡方值前须进行处理。

有无风化	黑	蓝绿	绿	浅蓝	浅绿	深蓝	深绿	紫
风化	2	9	0	12	1	0	4	2
无风化	0	6	1	8	2	2	3	2

表 4　玻璃表面风化与颜色的原始数据汇总

玻璃颜色与着色剂与熔炼温度和炉焰性质有关。常见的着色剂及其对玻璃产生的着色包括：氧化铜（绿色）、高价态的铜（蓝色）、二氧化锰（紫色）、氧化亚铁（黑色）等 [3]。而颜色的深浅则是由于玻璃在煅烧过程中电子接受光照后跃迁的能量之差不同。故为使理论频数达到最低的数值，将着色剂相同的文物分在一组。黑色和紫色由于其着色剂与其他各不相同，且理论频数过小，故省略。如此处理的实验数据及计算结果汇总在表 5中，可见其显著性水平小于 95%，故无法拒绝零假设，表明玻璃表面风化与否与颜色无显著关联。

原始数据					理论值		
有无风化	蓝绿	深蓝, 浅蓝	深绿, 浅绿, 绿	行和	蓝绿	深蓝, 浅蓝	深绿, 浅绿, 绿
风化	9	12	5	26	8.125	11.91666667	5.958333333
无风化	6	10	6	22	6.875	10.08333333	5.041666667
列和	15	22	11	48	15	22	11
自由度及卡方理论值				卡方计算值			
自由度	2			卡方值	0.543165925		
概率水平	0.05				0.01		
卡方理论值	5.99				9.21		
统计推断结论	0.05	风化与否与玻璃颜色不存在显著的关联性					
	0.01	风化与否与玻璃颜色不存在极显著的关联性					

表 5　玻璃表面风化与颜色的卡方检验数据

4.1.2 分析玻璃表面风化前后化学成分含量的统计规律

由附件中的表单一可知，玻璃类型可分为高钾玻璃和铅钡玻璃，风化状态可以分为风化和无风化。因此根据题目描述我们将所有玻璃样本分为四类：高钾风化玻璃、高钾未风化玻璃、铅钡风化玻璃、铅钡未风化玻璃。我们以样本均值和中位数来展示其统计规律，经分别计算后得到如下表格（所有数据保留两位小数）

	SiO_2	Na_2O	K_2O	CaO	MgO	Al_2O_3	Fe_2O_3	CuO	PbO	BaO	P_2O_5	SrO	SnO_2	SO_2
无风化 (mean)	67.98	0.7	9.33	5.33	1.08	6.62	1.93	2.45	0.41	0.6	1.4	0.04	0.2	0.1
风化 (mean)	93.96	0	0.54	0.87	0.2	1.93	0.27	1.56	0	0	0.28	0	0	0
无风化 (50%)	65.53	0	7.6	4.04	0.63	5.14	0.48	1	0	0	0.69	0	0	0
风化 (50%)	93.41	0	0.67	0.83	0	1.72	0.28	1.55	0	0	0.28	0	0	0

表 6　高钾玻璃风化前后化学成分含量统计数据

	SiO_2	Na_2O	K_2O	CaO	MgO	Al_2O_3	Fe_2O_3	CuO	PbO	BaO	P_2O_5	SrO	SnO_2	SO_2
无风化 (mean)	55.44	0.77	0.26	13	0.49	3.19	0.93	1.56	23.59	10.5	0.94	0.29	0.06	0.28
风化 (mean)	34.44	0.98	0.14	35	0.72	3.92	0.57	1.96	37	9.91	4.05	0.36	0.06	0.59
无风化 (50%)	55.21	0	0.15	0.84	0.51	3.06	0	0.53	22.05	10.06	0.2	0.3	0	0
风化 (50%)	29.64	0	0	2.14	0.71	3.33	0.32	0.88	40.24	7.66	3.13	0.33	0	0

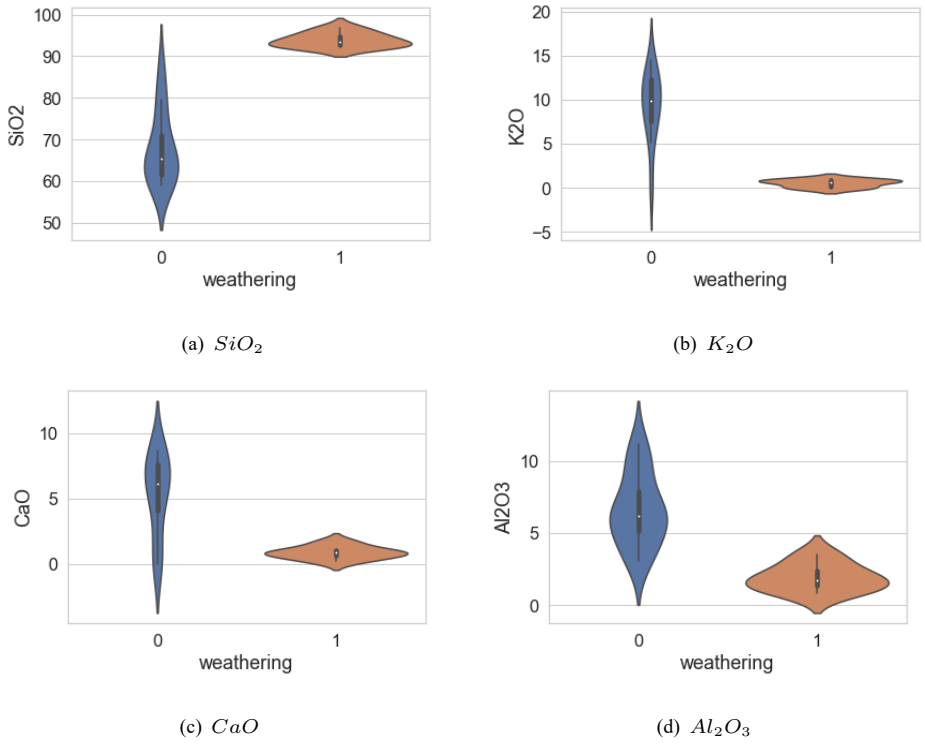
表 7　铅钡玻璃风化前后化学成分含量统计数据

由表 6可知，高钾玻璃在风化前后二氧化硅平均含量明显增加，而氧化钾、氧化钙、三氧化二铝的平均含量明显降低，其余成分含量基本保持不变；由表 7可知，铅钡玻璃

在风化前后二氧化硅平均含量明显减少，而氧化钙、氧化铅的平均含量明显增加，其余成分含量可看作保持不变。如此看来，两类玻璃的风化过程似乎有着相反的趋势。

4.1.3 预测玻璃风化前表面化学成分含量

为预测两种玻璃已风化表面在风化前的化学成分含量，我们根据表 6和表 7中的数据挑选变化量较大的成分绘制小提琴图，寻找其分布规律。在图 1 与图 2中,0 均表示风化前，1 均表示风化后。具体绘图代码见附录 B。



(a) SiO_2 (b) K_2O

(c) CaO (d) Al_2O_3

图 1　高钾玻璃风化前后变化量大的化学成分含量对比图

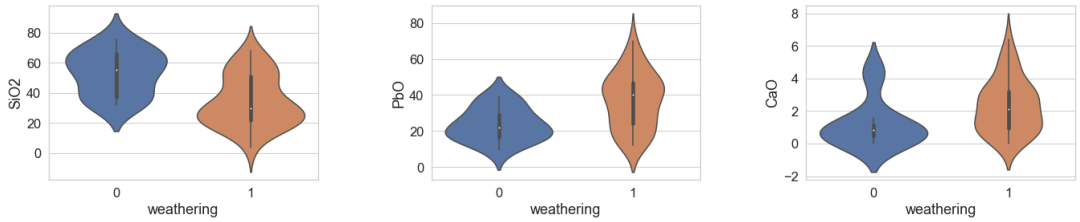
首先分析高钾玻璃。由上述四张小提琴图可以看出，图 1(a)风化前后含量集中且近似于正态分布，因此我们使用均值估计的方法预测各个风化样本风化前的百分含量:

$$\text{风化前预测成分含量} = \frac{\text{风化前均值}}{\text{风化后均值}} \times \text{风化后该物质百分含量}$$

相似地，我们可以用此方法分析氧化钙、氧化铝、氧化铜以及五氧化二磷。

对于其余物质，由表格和小提琴图可知，多数样本风化前不存在此物质，在这种情况下为了避免部分极端值对均值的影响，我们使用风化前的中位数作为这些物质风化前的预测值。

风化高钾玻璃的风化前成分预测表见附录 C。



(a) SiO_2 (b) PbO (c) CaO

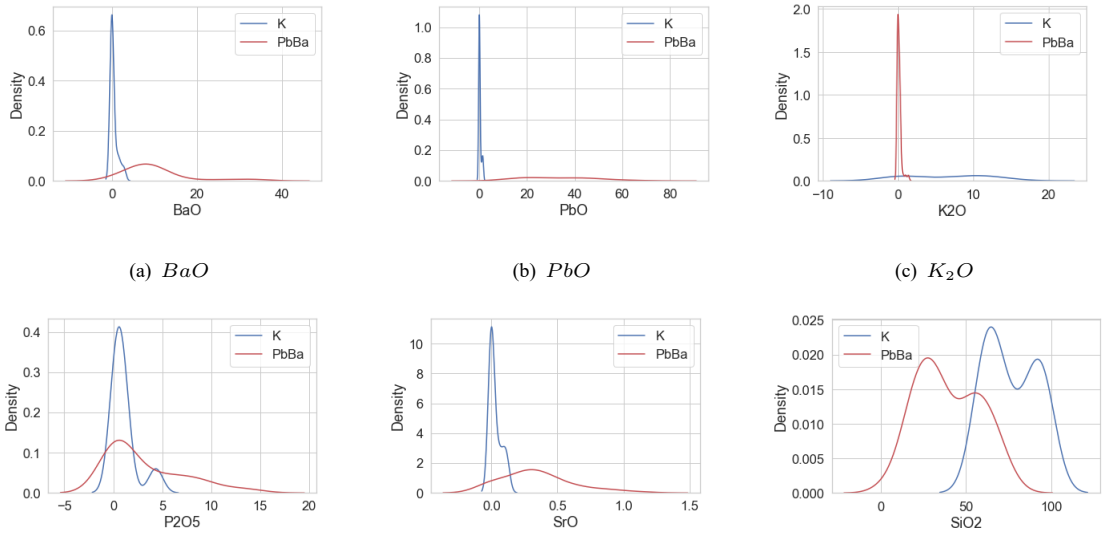
图 2　铅钡玻璃风化前后变化量大的化学成分含量对比图

与高钾玻璃类似，在铅钡玻璃中，我们对二氧化硅、氧化钙、氧化铝、氧化铜、氧化铅、氧化钡及五氧化二磷进行均值缩放，而对剩下的物质进行中位数估计。具体预测结果见附录 C。

4.2 问题二

4.2.1 对两类玻璃分类规律的数据探索

利用 *Python*，我们对高钾和铅钡玻璃各个样本的各类化学物质进行密度直方图绘制，图 3展示了两类玻璃中密度差距较明显的成分。具体代码见附录 B。



(a) BaO (b) PbO (c) K_2O

(d) P_2O_5 (e) SrO (f) SiO_2

图 3　两种玻璃各类化学物质的密度直方图

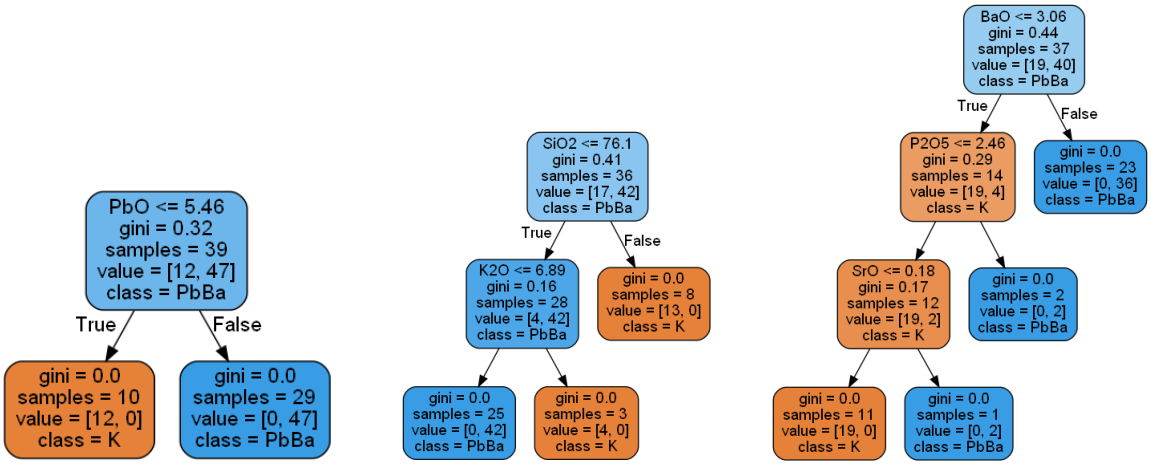
从图 3(a)、图 3(b)和图 3(c)中可以看到，在高钾玻璃中 BaO、PbO 两种化学物质分布集中在 0 附近，说明此两种化学物质在其含量极微，而其 K_2O 含量分布较广且峰值不为零；在铅钡玻璃中，我们可看到与此刚好相反的趋势。可见，正如它们的类型名称所指出的那样，铅、钾、钡元素是将某一玻璃归类到高钾型还是铅钡型最直接的分类依据。

除此之外，由图 3(d)和图 3(e)可知，P_2O_5 和 SrO 这两种物质也可帮助区分其是高钾玻璃还是铅钡玻璃，它们的含量均在高钾玻璃中小于铅钡玻璃。同时，图 3(f)体现出二氧化硅在两种玻璃中的分布也有明显的相似与不同：该成分在两种玻璃中的含量均呈双峰分布，但高钾玻璃的平均含量要高于铅钡玻璃。

4.2.2 基于随机森林算法的分类规律研究

通过绘制玻璃各化学成分的密度直方图，我们直观地探索了两类玻璃的分类规律，下面我们将使用随机森林算法对此做进一步验证。随机森林算法是一种常见的分类算法，通过集成多个基本单元 —— 决策树而形成。该算法本身具有准确率高、无需降维等优点，适合本题的分类问题。

随机森林算法首先在原始数据集中采取有放回的抽样，构造与原始数据量等大的子数据集。本文选择使用 sklearn 自带的划分函数，将原数据随机划分成训练集与验证集，为后面的机器学习过程和验证过程做准备。为找到精准度最高的模型，我们使用网格搜索法寻找超参数，即在一定的参数范围内，按步长依次调整决策树最大深度，利用树深的不断调整训练学习器，使其找到准确度最高的模型并将其确定为最终模型。通过训练，我们共得到 100 棵决策树，经数值分析与对比，我们选出图 4中所示的三棵。此后，我们使用验证集检验其学习成果，结果如图 5所示。其中，左对角线是训练和验证时，预测的类别不同于真实类别的数量。



(a) PbO 作为分裂特征　(b) K_2O 与 SiO_2 作为分裂特征　(c) 其他成分作为分裂特征

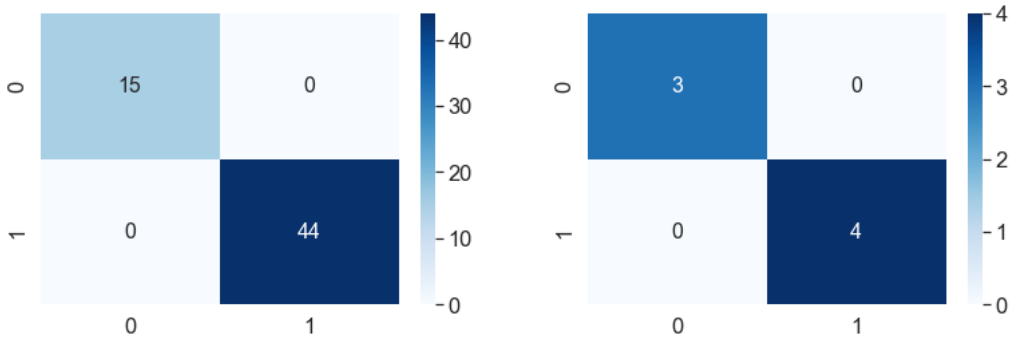
图 4　随机森林决策树

首先我们对图 4中出现的专业参数进行解释。基尼不纯度 (gini) 用来衡量划分的杂乱程度，其数值越小代表数据划分越准确。其计算公式为：

$$Gini(X) = \sum_{x \in \chi} p(x)(1 - p(x)) \tag{3}$$

其中 X 为一离散型随机变量，χ 是其取值空间，$p(x)$ 为其概率密度。samples 表示该节点处的样本总数；values 是一个标签，遵守 value = [判为真的数据量，判为假的数据量]。

基于以上知识，我们来进行决策树解读。由图 4(a)可看出，PbO 是一个有力的分裂特征，及若选择了该特征，则可以一步将高钾玻璃与铅钡玻璃区分出来。由图 4(b)和图 4(c)可知，SiO_2、K_2O、BaO、P_2O_5、SrO 也可快速将两类玻璃进行区分，一个直观的原因就是此三张图仅通过最多三步就对玻璃类别给出判断。值得注意的是，这些可用以区分两类玻璃的成分与我们在 4.2.1 中通过成分的密度直方图得到的结果一致。



(a) 训练成果　　(b) 验证结果

图 5　随机森林的训练与验证成果图

如图 5(a)和图 5(b)所示，在训练和验证中，决策树均具有 100% 的准确性，故认定这是一次成功的学习与鉴定。具体代码见附录 B。

总结来说，基于一块玻璃内硅、钾、钡、铅、磷和锶的含量，我们将其分成高钾玻璃与铅钡玻璃两类。由提供的 58 件文物的数据可得到如下分类规律：

类型	分类规律
高钾玻璃	SiO_2、K_2O 含量高，BaO、PbO、SrO、P_2O_5 含量低
铅钡玻璃	BaO、PbO、SrO、P_2O_5 含量高，SiO_2 含量较低，K_2O 含量低

表 8　两类玻璃的分类统计规律

4.2.3 亚类划分

Kmeans 算法是将 a 个数据按照定义的距离划分成 k 个类别的聚类算法。本文采用 Kmeans 聚类将高钾玻璃与铅钡玻璃分别进行亚类划分，其基本算法如下：

Algorithm 1 KMeans 聚类

Data: a 个 b 维数据（若采用余弦距离则需要标准化），k 个簇数，距离阈值 d 或者最大迭代次数 t

Result: a 个点到其所属中心点距离平方和最小的聚类方式

利用 Kmeans++ 算法或随机生成 k 个 b 维的中心点

while 新旧中心点之间距离大于设定阈值或未达到迭代次数 **do**

  for 每个点 **do**

    计算该点到 k 个中心点的距离（欧式距离等），并归入距离最小的中心点集合中

  end

  重新计算中心点坐标

end

由于聚类无监督算法，所以无法用准确度等指标进行衡量，故采用 Inertia 指标、轮廓系数和数据可视化进行综合考量。

Inertia 指标用来衡量簇内距离平方和，其公式为

$$\text{Inertia} = \sum_{i=0}^{n} \min_{\mu_j \in C}(\|x_i - \mu_i\|^2) \tag{4}$$

轮廓系数 $s(i)$ 范围为 (-1,1)。$s(i)$ 接近 1，说明样本 i 聚类合理；$s(i)$ 接近 -1，说明样本 i 更应该分类到另外的簇；$s(i)$ 近似为 0，则说明样本 i 在两个簇的边界上。其公式为

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad s(i) = \begin{cases} 1 - \frac{a(i)}{b(i)}, & a(i) < b(i) \\ 0, & a(i) = b(i) \\ \frac{b(i)}{a(i)} - 1, & a(i) > b(i) \end{cases} \tag{5}$$

其中，ai 是样本 i 到同簇其他样本的平均距离，称为样本 i 的簇内不相似度，ai 越小，说明样本 i 越应该被聚到该簇；bi 是样本 i 到其他某簇 C_j 的所有样本的平均距离 bij 的最小值，即 $bi = \min\{bi1, bi2, \cdots, bik\}$，称为样本 i 的簇间不相似度。

由于 Kmeans 聚类要求数值型变量之间有较高的独立性，故需要对原变量进行处理。本文采用了主成分分析法、不旋转的因子分析、基于最大方差旋转的因子分析以及利用专业知识分类共四种降维方法，并根据其 Inertia 指标、轮廓系数以及可视化最终选择了利用 PCA 降维的方法。

主成分分析法、不旋转的因子分析以及基于最大方差旋转的因子分析都是将具有较高相关性的原变量，通过一定变换，重新组合成较少数量的较为独立的新变量的降维方法。其数学原理如下：有数据集 $X = \{x_1, x_2, \cdots, x_n\}$，其连续潜变量模型为

$$x_i = Wh_i + \mu + \epsilon \tag{6}$$

其中，h_i 是潜变量，ϵ 是根据均值为 0 和协方差为 Ψ (i.e. $\epsilon \sim \mathcal{N}(0, \Psi)$) 的高斯分布而分布的噪声，$\mu$ 是偏移向量。如果把所有 x_i 当作列形成矩阵 X，所有 h_i 当作列形成矩阵 H，并且定义合适的 M 和 E，则有

$$\boldsymbol{X} = W\boldsymbol{H} + \boldsymbol{M} + \boldsymbol{E} \tag{7}$$

当 h_i 已知时，推出

$$p(x_i|h_i) = \mathcal{N}(Wh_i + \mu, \Psi) \tag{8}$$

对于一个完整的概率模型，我们还需要潜变量 h_i 的先验概率。基于高斯分布的特性，我们假设 $h \sim \mathcal{N}(0, \boldsymbol{I})$，则推出 x 的边际概率

$$p(x) = \mathcal{N}(\mu, WW^T + \Psi) \tag{9}$$

可对误差协方差 Ψ 做出不同假设：

1. 若 $\Psi = \sigma^2\boldsymbol{I}$，则是主成分分析法（PCA）

2. 若 $\Psi = \text{diag}(\psi_1, \psi_2, \cdots, \psi_n$，则是因子分析（不旋转的因子分析、基于最大方差旋转的因子分析等）[4]

通过查阅相关文献 [5]，我们将 12 个原物质（因对分类影响不大且难以归类，忽略 SnO_2 和 SO_2）分为 3 大类:

1. "SiO_2"（主要形成化合物）包括 SiO_2、P_2O_5、Al_2O_3、Fe_2O_3、PbO
2. "R_2O"（一价碱性化合物）包括 Na_2O、K_2O
3. "RO"（二价碱性化合物）包括 CaO、MgO、CuO、BaO、SrO

分别用四种降维方法对数据进行降维，对高钾玻璃和铅钡玻璃分别进行 Kmeans 聚类。以下以主成分分析为例进行展示，具体代码见附录 B：

Step 1. 得到主成分分析法降维后原变量组成新变量的系数在表 9:

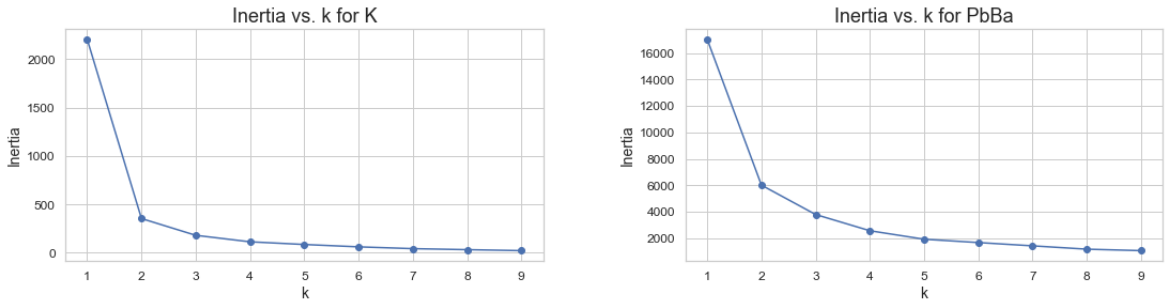
	高钾玻璃			铅钡玻璃		
	y1	y2	y3	y1	y2	y3
SiO2	-0.397523	-0.176119	0.084533	-0.469792	-0.207005	-0.037429
Na2O	0.097590	0.492189	0.135912	-0.198596	-0.261282	-0.108693
K2O	0.301581	0.275517	-0.182669	-0.143912	0.126047	0.387392
CaO	0.275585	0.434578	-0.074837	0.119469	0.458539	0.141778
MgO	0.310513	-0.270998	-0.223846	-0.182627	0.376514	0.075020
Al2O3	0.374468	0.009113	0.024356	-0.351596	0.171168	0.297678
Fe2O3	0.345623	-0.133723	-0.027290	-0.136125	0.318631	0.145767
CuO	0.213794	0.052121	0.114424	0.308137	-0.223594	0.291901
PbO	0.182719	0.138880	0.528848	0.333540	0.255274	-0.359632
BaO	0.230936	-0.228318	0.474403	0.309299	-0.293317	0.433307
P2O5	0.272730	-0.364237	-0.047212	0.240826	0.349588	0.005911
SrO	0.294615	-0.336585	0.019255	0.284794	0.134105	-0.037046
SnO2	-0.037854	-0.195060	-0.237227	-0.175790	0.185312	0.338686
SO2	0.120069	0.111441	-0.554312	0.239790	-0.120074	0.427447

表 9　主成分分析法降维后原变量组成新变量的系数

Step 2. 将高钾玻璃和铅钡玻璃的原数据与上表多列系数矩阵相乘，得到降维后数据。

Step 3. 利用 Inertia 指标和轮廓指数综合选取最佳簇数 k。Inertia 指标随着 k 增长而下降，故不采用 Inertia 指标最低的 k 作为最佳簇数，而是采用 k 前后斜率变化较大的点（拐点）作为最佳簇数。同时轮廓指数越靠近 1，聚类越合理，故选取轮廓指数最高的 k 作为最佳簇数。

Inertia 指标随着 k 变化如图所示



(a) 高钾玻璃的 Inertia 指标与聚类簇数关系　　(b) 铅钡玻璃的 Inertia 指标与聚类簇数关系

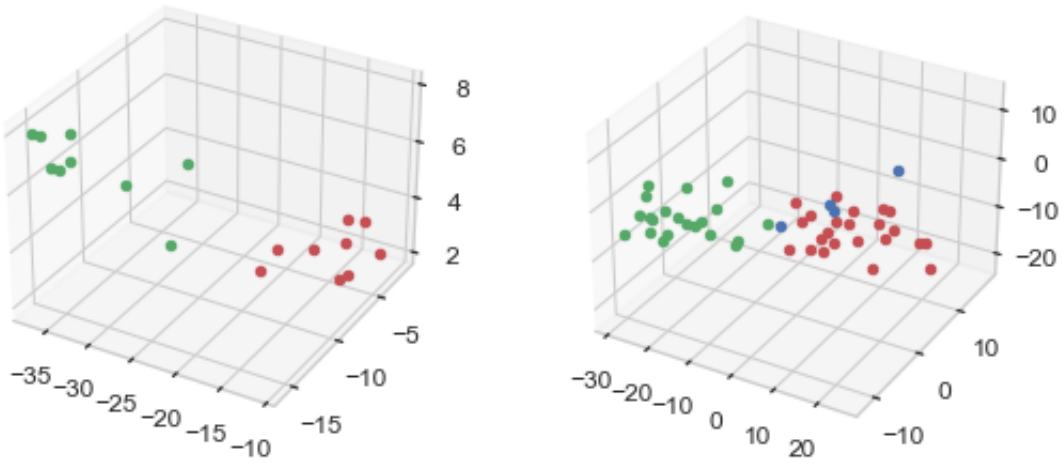
图 6　两种玻璃类型的 Inertia 指标与聚类簇数关系

不同簇数 k 的轮廓系数如表 10所示

簇数 k	高钾玻璃的轮廓系数	铅钡玻璃的轮廓系数
2	0.69403	0.55692
3	0.63841	0.574008
4	0.49687	0.48640857

表 10　两种玻璃类型的轮廓系数与簇数的关系

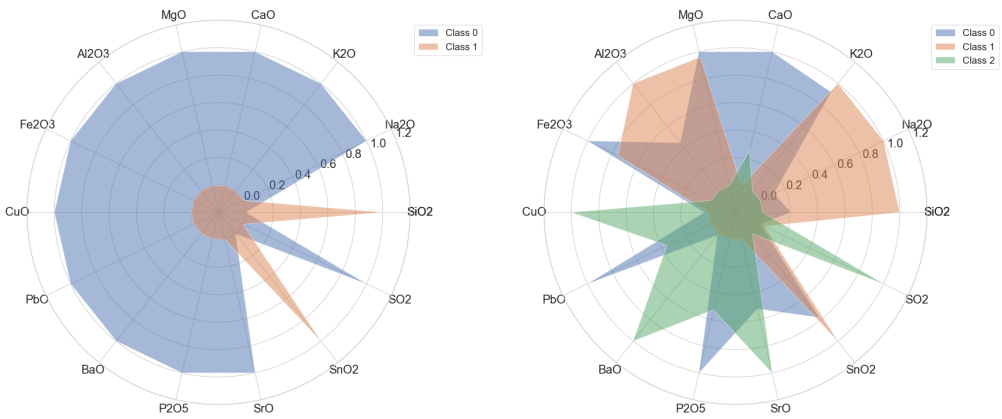
对于高钾玻璃来说，根据其 Inertia 指标与聚类簇数关系折线图和其轮廓系数与簇数的关系表格，得出最佳簇数为 2，其 Inertia 值为 355，轮廓系数为 0.69403。
对于铅钡玻璃来说，根据其 Inertia 指标与聚类簇数关系折线图和其轮廓系数与簇数的关系表格，得出最佳簇数为 3，其 Inertia 值为 3787，轮廓系数为 0.574008。
Step 4. 两种玻璃聚类结果的散点图如下



(a) 高钾玻璃的聚类结果散点图　　(b) 铅钡玻璃的聚类结果散点图

图 7　两种玻璃类型的聚类结果散点图

两种玻璃聚类结果的显著特征雷达图如下



(a) 高钾玻璃的聚类结果显著特征雷达图　　(b) 铅钡玻璃的聚类结果显著特征雷达图

图 8　两种玻璃类型的聚类结果显著特征雷达图

两种玻璃五种亚类各物质的均值表格如下（完整分类结果见附录 C）

	高钾玻璃		铅钡玻璃		
	G1	G2	G3	G4	G5
Counts	9.000	9.000	23.000000	21.0000	4.000000
Percentage	0.500	0.500	0.479167	0.4375	0.083333
SiO2	63.624	89.663	26.844000	57.4900	18.898000
Na2O	0.927	0.000	0.210000	1.8810	0.000000
K2O	10.818	1.986	0.181000	0.1880	0.100000
CaO	6.363	1.327	2.907000	1.1420	1.600000
MgO	1.133	0.437	0.733000	0.7040	0.000000
Al2O3	7.349	2.764	2.933000	5.0640	1.202000
Fe2O3	2.312	0.440	0.827000	0.6240	0.000000
CuO	2.819	1.492	1.367000	1.1650	8.260000
PbO	0.410	0.139	46.384000	19.8840	29.318000
BaO	0.579	0.219	8.294000	7.9750	31.290000
P2O5	1.523	0.533	5.096000	1.1280	3.225000
SrO	0.048	0.008	0.413000	0.2230	0.588000
SnO2	0.000	0.262	0.057000	0.0730	0.000000
SO2	0.136	0.000	0.000000	0.1740	5.122000

表 11　两种玻璃的五种亚类中各化学成分均值表

高钾玻璃分为两个亚类 $G1$ 和 $G2$，用下式表示（括号中是均值，wt% 是含量）：

$G1:\ K_2O(10.82wt\%) - CaO(6.36wt\%) - SiO_2(63.62wt\%)$

$G2:\ K_2O(1.99wt\%) - SiO_2(89.66wt\%)$

高钾玻璃是指以 SiO_2 为主要原料，以含钾量高的物质作为助熔剂，并添加石灰石作为稳定剂，石灰石煅烧以后转化为 CaO。在 G1 高钾玻璃亚类中，添加了较高含量的助熔剂和稳定剂，所以 SiO_2 含量相对较低；在 G2 高钾玻璃亚类中，助熔剂和稳定剂含量都较低，SiO_2 含量很高。

铅钡玻璃分为三个亚类 $G3$、$G4$ 和 $G5$，用下式表示（括号中是均值，wt% 是含量）：

$G3:\ PbO(46.38wt\%) - BaO(8.29wt\%) - CaO(2.91wt\%) - SiO_2(26.84wt\%)$

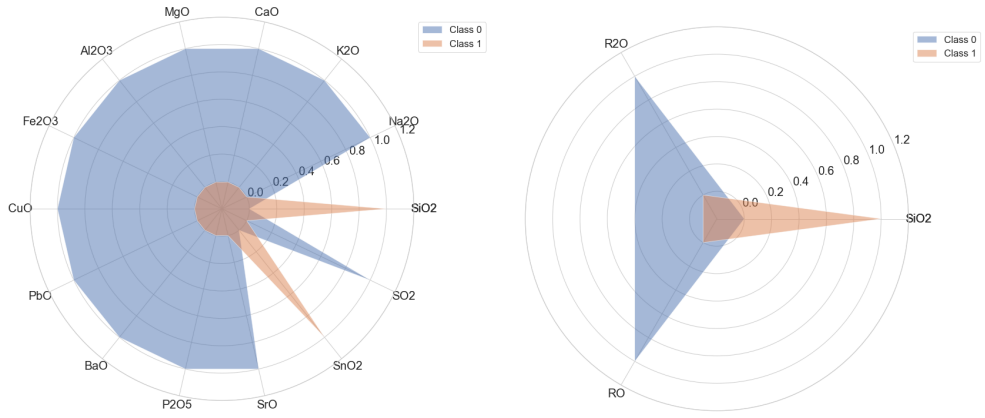
$G4:\ PbO(19.88wt\%) - BaO(7.98wt\%) - SiO_2(57.49wt\%)$

$G5:\ PbO(29.32wt\%) - BaO(31.29wt\%) - SiO_2(18.90wt\%)$

铅钡玻璃是指以 SiO_2 为主要原料，加入 PbO 和 BaO 含量较高的铅矿石作为助熔剂，并添加石灰石作为稳定剂，石灰石煅烧以后转化为 CaO。在 G3 铅钡玻璃亚类中，助熔剂 PbO 极高，甚至超过 SiO_2，也具有较高的稳定剂 CaO 含量；在 G4 铅钡玻璃亚类中，SiO_2 含量较高，是三个铅钡玻璃亚类中 SiO_2 含量最高的；在 G5 铅钡玻璃亚类中，虽然助熔剂 PbO 的含量没有 G3 铅钡玻璃亚类高，但是助熔剂 PbO 和 BaO 含量依然很高，SiO_2 含量是三个铅钡玻璃亚类中最低的。

4.2.4 可靠性分析

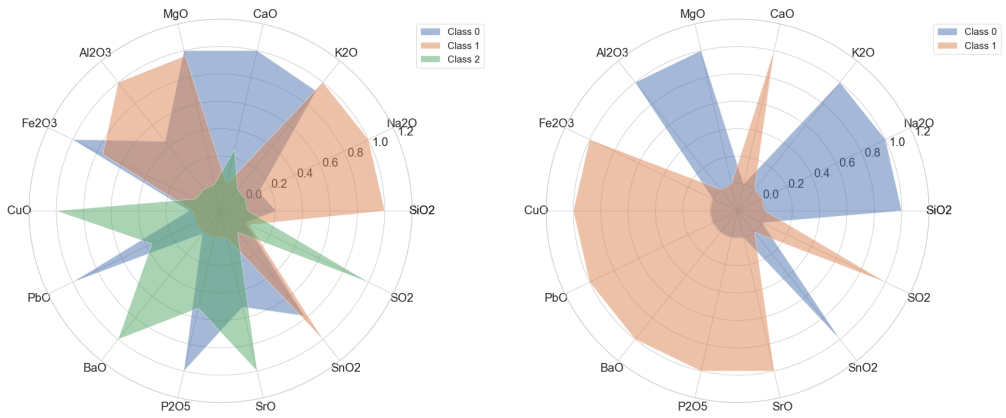
通过改变降维或者聚类模型，验证聚类结果变化是否明显。若不明显，则聚类结果具有广泛性，受主观选用模型影响小。一方面，改变聚类模型：对于高钾玻璃亚类模型和铅钡玻璃亚类模型，分别改变 Kmeans 的起始中心点计算方法（改成随机）和算法（改成 elkan、auto、full），发现聚类结果与原始一致。另一方面，改变降维模型：在高钾玻璃亚类模型中，采用不旋转的因子分析、基于最大方差旋转的因子分析和利用专业知识分类的方法，其中两种因子分析的显著特征雷达图与主成分分析方法一致，利用专业知识分类的显著特征雷达图与前三个模型表达的内涵相同，对比如下



(a) 高钾玻璃的主成分分析法降维后显著特征雷达图　　(b) 高钾玻璃的专业知识分类法降维后显著特征雷达图

图 9　高钾玻璃的主成分分析法与专业知识分类法降维后显著特征雷达图

在铅钡玻璃亚类模型中，采用不旋转的因子分析、基于最大方差旋转的因子分析虽然最优簇数是 $k=2$，但是由于跟主成分分析法有差异的一类仅有 4 个玻璃样本，所以总体上与主成分分析法非常一致（主成分分析法的橙色组对应因子分析的蓝色组，主成分分析法的蓝色和绿色组对应因子分析的橙色组），对比如下



(a) 高钾玻璃的主成分分析法降维后显著特征雷达图

(b) 高钾玻璃的因子分析（不旋转/旋转）降维后显著特征雷达图

图 10　高钾玻璃的主成分分析法与专业知识分类法降维后显著特征雷达图

综上可知，虽然聚类结果因为模型会有细微差别，但是总体分类可靠性较好，受主观因素影响小。

4.2.5 敏感性分析

通过对整体数据进行随机扰动，增大或减小原数据的值来模拟实验中会产生的误差。本文在已给的高钾玻璃所有化学成分中随机选择 50 个值放大和 50 个值缩小，在已给的铅钡玻璃所有化学成分中随机选择 200 个值放大和 200 个值缩小。扰动发现，在扰动增加到 5% 时鉴定结果仍保持不变，可见测量误差在一定范围内很稳定，鉴定结果较准确。

需要明确的是，扰动比例也不可有意增加过大，其原因有二：其一，扰动比例过大会超过一般误差产生的范围，及该结果将不再被认定为误差；其二，扰动比例过大会使其所有元素含量相加之和超过可被认定为有效数据的最大范围。

详细的扰动过程代码请参考附录 B。

4.3 问题三

4.3.1 鉴别未知文物所属玻璃类型

基于问题二所进行的机器学习（具体代码见附录 B），将附件表单三中给出的未知 8 件文物的数据导入后得到如下鉴定结果：

文物编号	A1	A2	A3	A4	A5	A6	A7	A8
鉴定类别	高钾	铅钡	铅钡	铅钡	铅钡	高钾	高钾	铅钡

表 12　未知玻璃文物类型鉴定结果

4.3.2 敏感性分析

通过对整体数据进行随机扰动，增大或减小原数据的值来模拟实验中会产生的误差。本文在已给的 8 件未知文物的所有化学成分中随机选择 50 个值放大，50 个值缩小。扰动发现，SiO_2 和 PbO 在扰动增加到 7% 时鉴定结果仍保持不变；其他元素在扰动增加到 20% 时鉴定结果仍保持不变，可见测量误差在一定范围内很稳定，鉴定结果较准确。

本次扰动仍遵循 4.2.5 中提到的合理比例，详细的扰动过程代码请参考附录 B。

4.4 问题四

4.4.1 Pearson 相关性检验

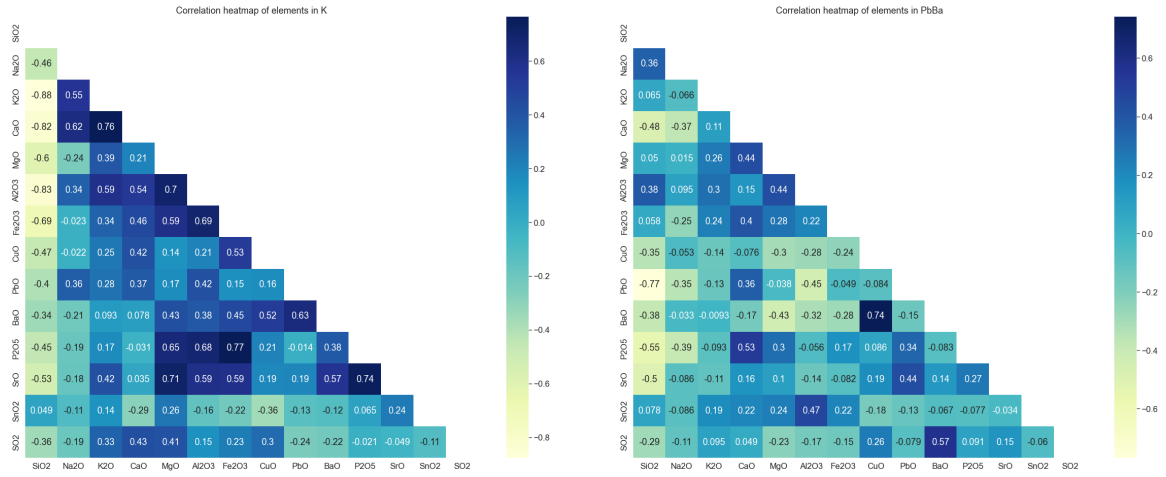
Pearson 相关系数用于度量两个变量之间的线性相关性，其值介于 -1 和 1 之间，越接近 1 表示正相关性越强，越接近 -1 表示负相关性越强。它的计算公式如下：

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{10}$$

其中，X, Y 表示两个随机变量。

4.4.2 两种类型的玻璃间化学成分的关联性分析

热力图可用来展示两事物间的关联程度。下图总结了两种玻璃中各化学成分间的关系，格子的颜色越偏蓝色代表两元素间的正相关性越强，颜色越偏黄色代表两元素间的负相关性越强，颜色介于浅蓝和浅绿则表明两元素间关联性较弱。具体代码见附录 B。



(a) 高钾玻璃内部化学成分关系热图　　(b) 铅钡玻璃内部化学成分关系热图

图 11　两种玻璃类型内部化学成分的关系

从图 11(a)中可见，高钾玻璃内部各化学成分间正相关性较强。其中 Fe_2O_3 与 P_2O_5 的正相关性最强，达到 0.77；此外，也有四组成分间的相关性均超过 0.7。在高钾玻璃中，K_2O 与 SiO_2 的负相关性最强，达到 -0.88。

从图 11(b)中可见，铅钡玻璃内部各化学成分间相关性较弱。其中 CuO 与 BaO 的正相关性最强，达到 0.74，但未有其他正相关性超过 0.6 的成分。在铅钡玻璃中，PbO 与 SiO_2 负相关性最强，达到 -0.77。

故总体来看，高钾玻璃内部中各化学元素间的正负相关性均强于铅钡类型玻璃内部的化学成分。此外，虽两类玻璃的 SiO_2 的含量与其他成分含量均基本呈负相关，但其分别与两类玻璃的主要助燃剂成分的负相关性最强。对两种玻璃内化学成分两两间的线性拟合图请见附录 B 中的图 12与图 13。

五、 总结

问题一中，通过卡方检验我们发现表面风化状态仅与玻璃类型有显著关联，与纹饰及颜色没有明显关联；对两种玻璃风化前后的统计数据分析表明，风化过程对于两种玻璃各自化学成分含量的改变有着相反的趋势，且两种玻璃风化前后化学成分的变化不尽相同：高钾玻璃在风化后二氧化硅平均含量明显增加，而氧化钾、氧化钙、三氧化二铝的平均含量明显降低；铅钡玻璃在风化后二氧化硅平均含量明显减少，而氧化钙、氧化铅的含量明显增加。

问题二中，我们通过对两种玻璃类别中的化学成分含量进行密度对比分析及应用随机森林算法加以验证，得到高钾玻璃与铅钡玻璃依照其钾、铅、钡、硅和磷含量进行分类。同时，我们将玻璃进一步划分得到 G1−G5 五种玻璃亚类，其中 G1、G2 属于高钾玻璃，G3−G5 属于铅钡玻璃。

问题三利用问题二的模型，得到了八种玻璃的鉴定结果：A1、A6−A8 为高钾玻璃，A2−A5 为铅钡玻璃。

问题四中，高钾玻璃内部中各化学元素间的正负相关性均强于铅钡类型玻璃内部的化学成分。此外，虽两类玻璃中二氧化硅的含量与其他成分含量均基本呈负相关，但其分别与两类玻璃的主要助燃剂成分的负相关性最强。

参考文献

[1] 陆春海. 防文物风化的材料设计与适用性研究 [D]. 成都理工大学,2009.

[2] 2022 年高教社杯全国大学生数学建模竞赛题目，C 题，“古代玻璃制品的成分分析与鉴别”.

[3] 杨伯达. 清代玻璃配方化学成分的研究 [J]. 故宫博物院院刊,1990(02):17-26+38.DOI:10.16319/j.cnki.0452-7402.1990.02.004.

[4] Scikit-learn 用户指南. Factor Analysis. https://scikit-learn.org/stable/modules/decomposition.htmlfa

[5] Cox, G. A., et al. “A STUDY OF THE WEATHERING BEHAVIOR OF MEDIEVAL GLASS FROM YORK MINSTER.” Journal of Glass Studies, vol. 21, 1979, pp. 54–75. JSTOR, https://www.jstor.org/stable/24190036. Accessed 18 Sep. 2022.

[6] 伏修锋, 干福熹. 基于多元统计分析方法对一批中国南方和西南地区的古玻璃成分的研究 [J]. 文物保护与考古科学,2006(04):6-13.DOI:10.16334/j.cnki.cn31-1652/k.2006.04.002.

附录 A　支撑材料文件列表

处理后附件.xlsx

结果

附件 A 问题 1 预测结果.xlsx

附件 B 问题 2 亚类划分结果.xlsx

附件 C 问题 3 鉴别类型结果.xlsx

问题 1 相关

问题 1 统计规律代码.ipynb

问题 1 表面风化关系（卡方检验）.xlsx

问题 1 风化前化学成分预测模型.xlsx

问题 2、3 相关

问题 2&3 随机森林分类模型代码.ipynb

问题 2 分类规律代码.ipynb

问题 2 铅钡玻璃亚类划分模型代码.ipynb

问题 2 高钾玻璃亚类划分模型代码.jpynb

问题 4 相关

问题 4 关联关系代码.ipynb

附录 B　代码

2.1　问题 1：统计规律可视化 *Python* 实现核心代码

```
# 画单个物质风化前后
fig = plt.figure(figsize=(15,100))
cols = list(a_K.columns)[1:]
for i in range(14):
    ax = fig.add_subplot(14, 1, i + 1)
    print(sns.violinplot(data = a_K, x = 'weathering', y = cols[i]))

# 画整体风化前后
```

```python
def combinate(a):
    com = a.copy()
    com['SiO2'] = com['SiO2']+com['P2O5']+com['Al2O3']+com['Fe2O3']+com['PbO']
    com['R2O'] = com['Na2O']+com['K2O']
    com['RO'] = com['CaO']+com['MgO']+com['CuO']+com['BaO']+com['SrO']
    com = com.drop(columns={'Na2O','K2O','CaO','MgO','Al2O3','Fe2O3',
    'CuO','PbO','BaO','P2O5','SrO','SnO2','SO2'})
    com['R2O+RO'] = com['R2O']+com['RO']
    com['R2O/RO'] = com['R2O']/com['RO']
    com['SiO2/(R2O+RO)'] = com['SiO2']/(com['R2O']+com['RO'])
    return com

com_K = combinate(a_K.copy())

fig = plt.figure(figsize=(15,50))
cols = list(com_K.columns)[1:]
for i in range(6):
    ax = fig.add_subplot(6, 1, i + 1)
    print(sns.violinplot(data = com_K, x = 'weathering', y = cols[i]))
```

2.2 问题 2：分类规律可视化 *Python* 实现核心代码

```python
# 画数值型变量
def distribution(col, data):
    plt_data = data.copy()
    sns.distplot(plt_data[plt_data['type'] == 'K'][col], hist = False, color = 'b')
    sns.distplot(plt_data[plt_data['type'] == 'PbBa'][col], hist = False, color = 'r')
    plt.legend(['K', 'PbBa'])
# 画离散型变量
def comparison(col, data):
    plt_data = data.copy()
    sns.barplot(x = 'type', y = col, data = plt_data)
```

2.3 问题 2 和 3：随机森林分类算法与敏感性分析 *Python* 实现核心代码

```python
# 划分训练集、测试集
from sklearn.model_selection import train_test_split
train, val = train_test_split(df4, test_size=0.1, random_state=42)
# 特征工程
def fill_missing_value(data):
    filled_data = data.copy()
    cols = list(filled_data.columns)
    for col in cols:
        filled_data[col] = filled_data[col].fillna(0)
```

```
    return filled_data
def filter_invalid_data(data):
    filtered_data = data.copy()
    cols = list(filtered_data.columns)
    filtered_data['Total'] = filtered_data[cols[1:15]].sum(axis=1)
    filtered_data = filtered_data[(filtered_data['Total'] >= 85) & (filtered_data['Total'] <=
        105)]
    return filtered_data
def rename_data(data):
    renamed_data = data.copy()
    renamed_data['表面风化'][renamed_data['表面风化'] == '无风化'] = 0
    renamed_data['表面风化'][renamed_data['表面风化'] == '风化'] = 1
    renamed_data = renamed_data.rename(columns={'表面风化':'weathering',
                            '二氧化硅(SiO2)':'SiO2',
                            '氧化钠(Na2O)':'Na2O',
                            '氧化钾(K2O)':'K2O',
                            '氧化钙(CaO)':'CaO',
                            '氧化镁(MgO)':'MgO',
                            '氧化铝(Al2O3)':'Al2O3',
                            '氧化铁(Fe2O3)':'Fe2O3',
                            '氧化铜(CuO)':'CuO',
                            '氧化铅(PbO)':'PbO',
                            '氧化钡(BaO)':'BaO',
                            '五氧化二磷(P2O5)':'P2O5',
                            '氧化锶(SrO)':'SrO',
                            '氧化锡(SnO2)':'SnO2',
                            '二氧化硫(SO2)':'SO2'})
    return renamed_data
def process_data_gm(data, pipeline_functions):
    """Process the data for a guided model."""
    for function, arguments, keyword_arguments in pipeline_functions:
        if keyword_arguments and (not arguments):
            data = data.pipe(function, **keyword_arguments)
        elif (not keyword_arguments) and (arguments):
            data = data.pipe(function, *arguments)
        else:
            data = data.pipe(function)
    return data
def select_columns(data, *columns):
    """Select only columns passed as arguments."""
    return data.loc[:, columns]
def process_data_fm(data):
    X = data
    X = process_data_gm(X, [
                            (fill_missing_value, None, None),
                            (filter_invalid_data, None, None),
                            (rename_data, None, None),
```

```
                        (select_columns, [# Binary Features
                                          'weathering',
                                          # Numerical Features
                                          'SiO2',
                                          'Na2O',
                                          'K2O',
                                          'CaO',
                                          'MgO',
                                          'Al2O3',
                                          'Fe2O3',
                                          'CuO',
                                          'PbO',
                                          'BaO',
                                          'P2O5',
                                          'SrO',
                                          'SnO2',
                                          'SO2'], None)])
    return X
# 网格搜索寻找超参
tr = train.copy()
tr['类型'][tr['类型'] == '高钾'] = 'K'
tr['类型'][tr['类型'] == '铅钡'] = 'PbBa'
tr = tr.rename(columns={'类型':'type'})
from sklearn.model_selection import GridSearchCV
def model_finder_function(phi_function, model, params):
    y_train = tr.copy()['type']
    X_train = phi_function(tr.copy())
    model_finder = GridSearchCV(estimator = model,
                                param_grid = params,
                                scoring = "accuracy",
                                cv = 5,
                                n_jobs = -1)
    model_finder.fit(X_train, y_train)
    print("Best Model: ", model_finder.best_estimator_)
    print("Mean Accuracy of Best Model: ", model_finder.best_score_)

from sklearn.ensemble import RandomForestClassifier
max_depth = [5, 10, 15, 20, 25, 30]
min_samples_split = [2, 5, 10]
min_samples_leaf = [1, 5, 10]

params_group = {'max_depth':max_depth,
                'min_samples_split':min_samples_split,
                'min_samples_leaf':min_samples_leaf}

random_forest_finded_model = model_finder_function(process_data_fm,
       RandomForestClassifier(random_state = 0), params_group)
```

```
# 评价函数
def evaluation_metrics(df):
    df_cp = df.copy()
    TP = len(df_cp[(df_cp['type'] == 'PbBa') & (df_cp['type_predict'] == 'PbBa')])
    FP = len(df_cp[(df_cp['type'] == 'K') & (df_cp['type_predict'] == 'PbBa')])
    FN = len(df_cp[(df_cp['type'] == 'PbBa') & (df_cp['type_predict'] == 'K')])
    TN = len(df_cp[(df_cp['type'] == 'K') & (df_cp['type_predict'] == 'K')])
    acc = (TP + TN) / (TP + FP + FN + TN)
    print("Accuracy: ", acc)
    precision = TP / (TP + FP)
    print("Precision: ", precision)
    recall = TP / (TP + FN)
    print("Recall: ", recall)
    False_alarm_rate = FP / (FP + TN)
    print("False Alarm Rate: ", False_alarm_rate)
from sklearn.metrics import confusion_matrix
def plot_confusion_matrix(df):
    df_cp = df.copy()
    cm = confusion_matrix(df['type'], df['type_predict'])
    sns.heatmap(cm, annot = True, fmt = 'd', cmap = 'Blues', annot_kws = {'size': 16})
# 最终模型
model_fi = RandomForestClassifier(max_depth=5, random_state=0)
train_fi = train.copy()
train_fi['类型'][train_fi['类型'] == '高钾'] = 'K'
train_fi['类型'][train_fi['类型'] == '铅钡'] = 'PbBa'
train_fi = train_fi.rename(columns={'类型':'type'})
y_train_fi = train_fi.copy()['type']
X_train_fi = process_data_fm(train_fi.copy())
model_fi.fit(X_train_fi, y_train_fi)
train_fi['type_predict'] = model_fi.predict(X_train_fi)
print("-------------- Training Evaluation Metrics --------------")
print('')
evaluation_metrics(train_fi)
print('')
print("-------------- Training Confusion Matrix --------------")
plot_confusion_matrix(train_fi)
# 可视化
from sklearn.tree import export_graphviz
model_fi = RandomForestClassifier(max_depth=5, random_state=0)
train_fi = train.copy()
train_fi['类型'][train_fi['类型'] == '高钾'] = 'K'
train_fi['类型'][train_fi['类型'] == '铅钡'] = 'PbBa'
train_fi = train_fi.rename(columns={'类型':'type'})
y_train_fi = train_fi.copy()['type']
X_train_fi = process_data_fm(train_fi.copy())
model_fi.fit(X_train_fi, y_train_fi)
for idx, estimator in enumerate(model_fi.estimators_):
```

```
    # 导出dot文件
    export_graphviz(estimator,
                    out_file='tree{}.dot'.format(idx),
                    feature_names=list(X_train_fi.columns),
                    class_names=['K','PbBa'],
                    rounded=True,
                    proportion=False,
                    precision=2,
                    filled=True)
    # 转换为png文件
    os.system('dot -Tpng tree{}.dot -o tree{}.png'.format(idx, idx))
# 验证
val_fi = val.copy()
val_fi['类型'][val_fi['类型'] == '高钾'] = 'K'
val_fi['类型'][val_fi['类型'] == '铅钡'] = 'PbBa'
val_fi = val_fi.rename(columns={'类型':'type'})
y_val_fi = val_fi.copy()['type']
X_val_fi = process_data_fm(val_fi.copy())
val_fi['type_predict'] = model_fi.predict(X_val_fi)
print("-------------- Validation Evaluation Metrics --------------")
print('')
evaluation_metrics(val_fi)
print('')
print("-------------- Validation Confusion Matrix --------------")
plot_confusion_matrix(val_fi)
# 测试【问题3】
test_predictions = model_fi.predict(process_data_fm(df3.copy()))
df3_with_prediction = df3.copy()
df3_with_prediction['预测类型'] = test_predictions
# 敏感性分析
df3_jittered = process_data_fm(df3.copy())
col_list_in = []
row_list_in = []
col_list_de = []
row_list_de = []
import random
for i in range(50):
    col_list_in.append(random.randint(1,14))
    row_list_in.append(random.randint(0,7))
for i in range(50):
    col_list_de.append(random.randint(1,14))
    row_list_de.append(random.randint(0,7))
tuple_list_in = list(zip(row_list_in, col_list_in))
tuple_list_de = list(zip(row_list_de, col_list_de))
for coordinate in tuple_list_in:
    if coordinate[1] == 1 or coordinate[1] == 9:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] * 1.07
```

```
    else:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] * 1.2
for coordinate in tuple_list_de:
    if coordinate[1] == 1 or coordinate[1] == 9:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] / 1.07
    else:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] / 1.2
cols = list(df3_jittered.columns)[1:15]
df3_jittered['total'] = df3_jittered[cols].sum(axis=1)
df3_jittered = df3_jittered.drop(columns={'total'})
df3_jittered_pred = model_fi.predict(df3_jittered)
```

2.4 问题 2：亚类划分降维聚类模型 *Python* 实现核心代码

```
# 原数据
from sklearn.decomposition import FactorAnalysis, PCA
from sklearn.preprocessing import StandardScaler
a_K = process_data_fm(df_K.copy())
a_K = a_K[list(a_K.columns)[1:15]]
# 标准化
X_K = StandardScaler().fit_transform(a_K.copy())
X_K_std = pd.DataFrame(X_K, columns=list(a_K.columns))
feature_names = list(X_K_std.columns)
n_comps = 3
# 三种降维
methods = [
    ("PCA", PCA()),
    ("Unrotated FA", FactorAnalysis()),
    ("Varimax FA", FactorAnalysis(rotation="varimax")),
]
fig, axes = plt.subplots(ncols=len(methods), figsize=(10, 12))

i = 0
for ax, (method, fa) in zip(axes, methods):
    fa.set_params(n_components=n_comps)
    fa.fit(X_K_std)

    components = fa.components_.T
    print("\n\n %s :\n" % method)
    if i == 0:
        components_pca = components
        print(components_pca)
    elif i == 1:
        components_fa = components
        print(components_fa)
```

```
    else:
        components_va = components
        print(components_va)

    i = i+1
    vmax = np.abs(components).max()
    ax.imshow(components, cmap="RdBu_r", vmax=vmax, vmin=-vmax)
    ax.set_yticks(np.arange(len(feature_names)))
    if ax.is_first_col():
        ax.set_yticklabels(feature_names)
    else:
        ax.set_yticklabels([])
    ax.set_title(str(method))
    ax.set_xticks([0, 1, 2])
fig.suptitle("Factors")
plt.tight_layout()
plt.show()
# 新变量
new_a_K_pca = np.dot(a_K, components_pca)
new_a_K_fa = np.dot(a_K, components_fa)
new_a_K_va = np.dot(a_K, components_va)
# 利用专业知识降维
def combine(a_K):
    com_K = a_K.copy()
    com_K['SiO2'] = com_K['SiO2']+com_K['P2O5']+com_K['Al2O3']+com_K['Fe2O3']+com_K['PbO']
    com_K['R2O'] = com_K['Na2O']+com_K['K2O']
    com_K['RO'] = com_K['CaO']+com_K['MgO']+com_K['CuO']+com_K['BaO']+com_K['SrO']
    com_K = com_K.drop(columns={'Na2O','K2O','CaO','MgO','Al2O3','Fe2O3',
    'CuO','PbO','BaO','P2O5','SrO','SnO2','SO2'})
    return com_K
com_K = combine(a_K)
# 聚类
from sklearn.cluster import KMeans
plt.rcParams['axes.labelsize'] = 14
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12
np.random.seed(42)
# 找最佳簇数
kmeans_per_k = [KMeans(n_clusters = k).fit(new_a_K_pca) for k in range(1,10)]
inertias = [model.inertia_ for model in kmeans_per_k]
plt.figure(figsize=(8,4))
plt.plot(range(1,10),inertias,'bo-')
plt.ylabel('Inertia')
plt.xlabel('k')
plt.title('Inertia vs. k for K')
plt.show()
from sklearn.metrics import silhouette_score
```

```
# 通过平均轮廓系数检验得到最佳KMeans聚类模型
score_list = list()
silhouette_int = -1
for n_clusters in range(2, 5):
    model_kmeans = KMeans(n_clusters=n_clusters)
    labels_tmp = model_kmeans.fit_predict(new_a_K_pca)
    silhouette_tmp = silhouette_score(new_a_K_pca, labels_tmp)
    if silhouette_tmp > silhouette_int:
        best_k = n_clusters
        silhouette_int = silhouette_tmp
        best_kmeans = model_kmeans
        cluster_labels_k = labels_tmp
    score_list.append([n_clusters, silhouette_tmp])
print('{:*^60}'.format('K值对应的轮廓系数:'))
print(np.array(score_list)) # 打印输出所有K下的详细得分
print('最优的K值是:{0} \n对应的轮廓系数是:{1}'.format(best_k, silhouette_int))
# 结果
k = 2
kmeans_K_2 = KMeans(n_clusters = k,random_state=42)
y_pred_K_2 = kmeans_K_2.fit_predict(new_a_K_pca)
# 中心
kmeans_K_2.cluster_centers_
# 评估
kmeans_K_2.inertia_
# 可视化
import csv
A = np.array(new_a_K_pca)
num_K_2, dim_K_2 = new_a_K_pca.shape
color = ['r', 'g', 'b', 'c', 'y', 'm', 'k']
ax = plt.subplot(111, projection='3d')
f = open('result_K_2.csv', 'w', encoding='utf-8', newline='')
csv_writer = csv.writer(f)
for p in range(0,num_K_2):
    y=y_pred_K_2[p]
    csv_writer.writerow([y])
    ax.scatter(int(A[p, 0]), int(A[p, 1]), int(A[p, 2]), c=color[int(y)])
f.close()
plt.show()
# 含标签数据
cluster_labels_K_2 = pd.DataFrame(y_pred_K_2, columns=['clusters'])
merge_data_K_2 = pd.concat((a_K, cluster_labels_K_2), axis=1)
merge_data_K_2.to_csv('主成分分析K.csv')
# 各类别均值
count_list = []
perce_list = []
for i in range(2):
    count = len(merge_data_K_2[merge_data_K_2['clusters'] == i])
```

```
    count_list.append(count)
    perce = len(merge_data_K_2[merge_data_K_2['clusters'] == i]) / len(merge_data_K_2)
    perce_list.append(perce)

number_pd_K_2 = pd.DataFrame({'counts':count_list,'percentage':perce_list}).T
cluster_features = []
for line in range(k):
    label_data = merge_data_K_2[merge_data_K_2['clusters'] == line]

    part_data = label_data[list(merge_data_K_2.columns)[0:14]]
    part_desc = part_data.describe().round(3)
    merge_data1 = part_desc.iloc[1, :]

    cluster_features.append(merge_data1)
cluster_pd_K_2 = pd.DataFrame(cluster_features, index=[0, 1]).T
all_cluster_set = pd.concat((number_pd_K_2, cluster_pd_K_2), axis=0)
print('{:*^60}'.format('每个类别主要的特征:'))
#各类别数据预处理
from sklearn.preprocessing import MinMaxScaler
model_scaler = MinMaxScaler()
num_sets = cluster_pd_K_2.iloc[:14, :].T.astype(np.float64)
num_sets_max_min = model_scaler.fit_transform(num_sets)
# 各类别显著特征
fig = plt.figure(figsize=(12,12))
ax = fig.add_subplot(111, polar=True)
labels = np.array(merge_data1.index)
cor_list = ['g', 'r', 'y', 'b','o','PuRd','YlGnBu']
angles = np.linspace(0, 2 * np.pi, len(labels), endpoint=False)
angles = np.concatenate((angles, [angles[0]]))
labels = np.concatenate((labels, [labels[0]]))
for i in range(len(num_sets)):
    data_tmp = num_sets_max_min[i, :]
    data = np.concatenate((data_tmp, [data_tmp[0]]))
    ax.fill(angles, data,alpha=0.5)
ax.set_thetagrids(angles * 180 / np.pi, labels)
ax.set_rlim(-0.2, 1.2) # 设置坐标轴尺度范围
plt.legend(['Class 0', 'Class 1'], loc="upper right" ,bbox_to_anchor=(1.2,1.0))
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
```

2.5 问题 3：对鉴定结果的敏感性分析 *Python* 代码

```
col_list_in = []
row_list_in = []
col_list_de = []
```

```
row_list_de = []
import random
for i in range(50):
    col_list_in.append(random.randint(1,14))
    row_list_in.append(random.randint(0,7))
for i in range(50):
    col_list_de.append(random.randint(1,14))
    row_list_de.append(random.randint(0,7))
tuple_list_in = list(zip(row_list_in, col_list_in))
tuple_list_de = list(zip(row_list_de, col_list_de))
```

```
for coordinate in tuple_list_in:
    if coordinate[1] == 1 or coordinate[1] == 9:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] * 1.07
    else:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] * 1.2
for coordinate in tuple_list_de:
    if coordinate[1] == 1 or coordinate[1] == 9:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] / 1.07
    else:
        df3_jittered.iloc[coordinate] = df3_jittered.iloc[coordinate] / 1.2
cols = list(df3_jittered.columns)[1:15]
df3_jittered['total'] = df3_jittered[cols].sum(axis=1)
df3_jittered
```

2.6 问题 4：两类玻璃内部化学成分关系热图的绘图 *Python* 代码

```
corr = df4[df4['type'] == 'K'].corr()
plt.figure(figsize = (20,16))
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(corr,annot=True, mask=mask, cmap="YlGnBu")
plt.title('Correlation heatmap of elements in K')
plt.show()
```

```
corr = df4[df4['type'] == 'PbBa'].corr()
plt.figure(figsize = (20,16))
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(corr,annot=True, mask=mask, cmap="YlGnBu")
plt.title('Correlation heatmap of elements in PbBa')
plt.show()
```

2.7 问题 4：高钾玻璃与铅钡玻璃内部各化学成分关系线性拟合图及 *Python* 代码

```
plt.figure(figsize=(10, 8), dpi=80)
sns.pairplot(df4[df4['type'] == 'K'][list(df4.columns)[1:16]], kind="reg", hue="type")
plt.title('Pair plot of elements in K')
plt.show()
```

```
plt.figure(figsize=(10, 8), dpi=80)
sns.pairplot(df4[df4['type'] == 'PbBa'][list(df4.columns)[1:16]], kind="reg", hue="type")
plt.title('Pair plot of elements in PbBa')
plt.show()
```

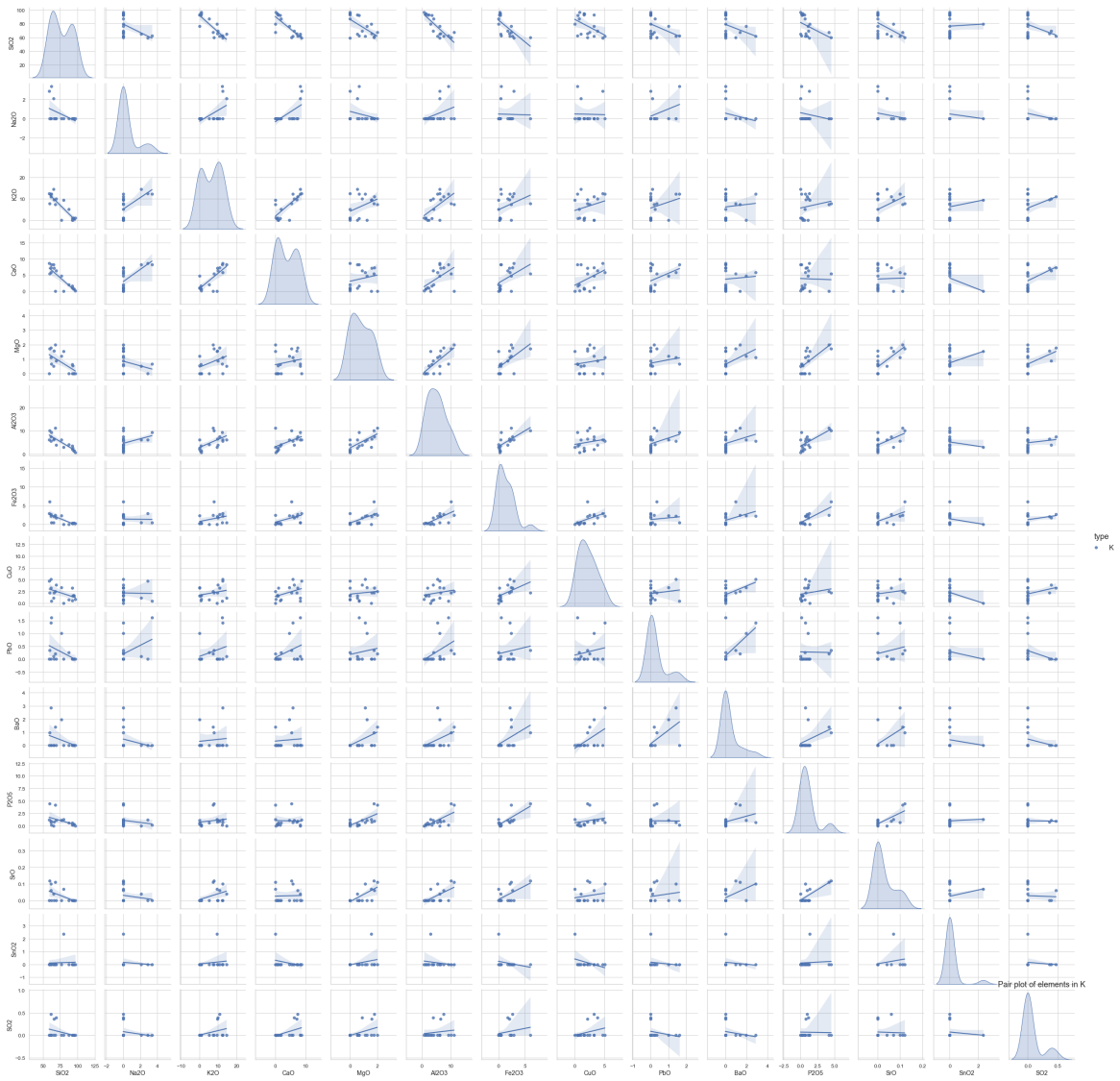


图 12　高钾玻璃化学元素线性关系表

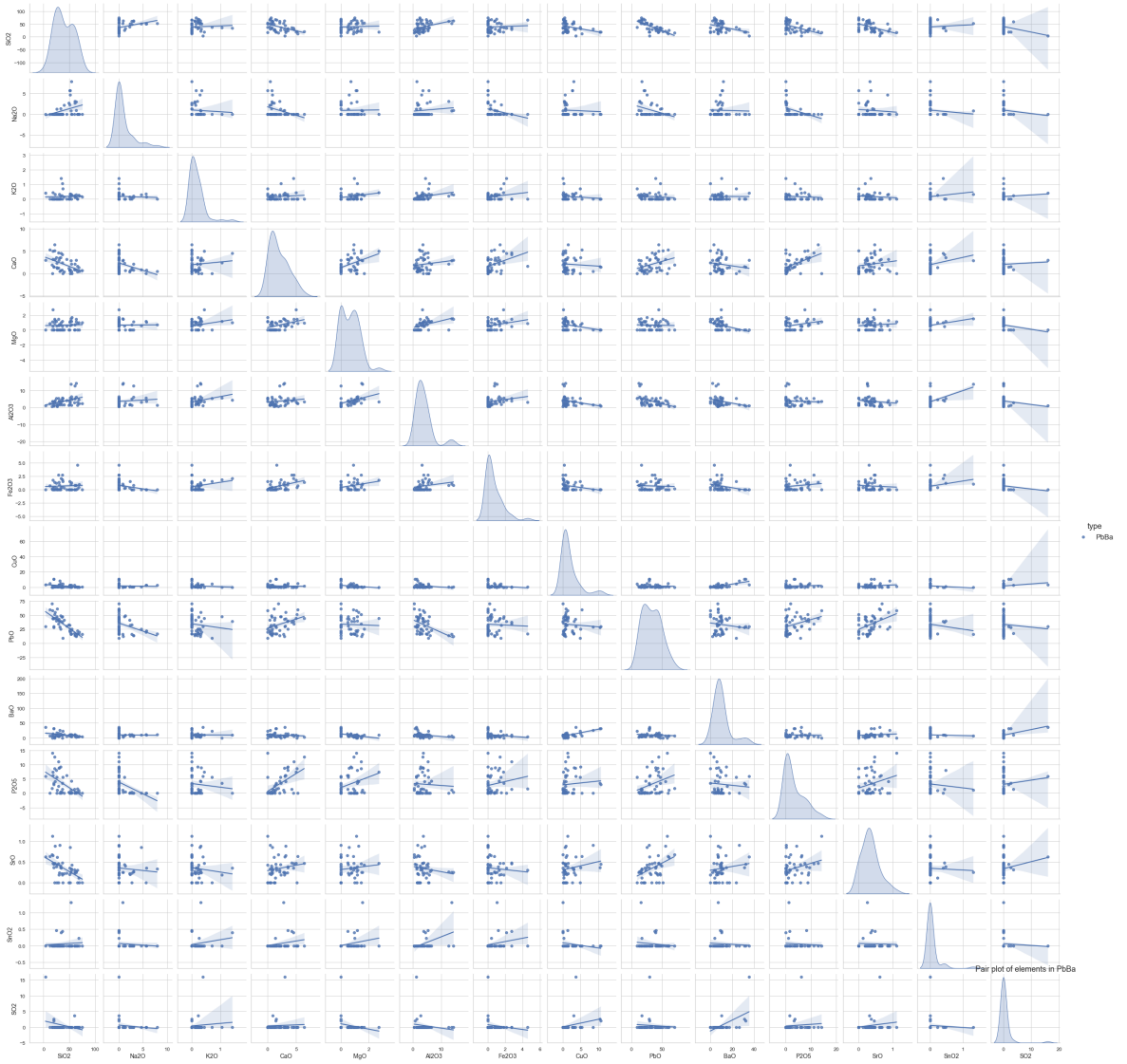


图 13　铅钡玻璃化学元素线性关系表

附录 C　表格

3.1　高钾玻璃与铅钡玻璃的风化前物质含量预测

物质类别	预测方法	样本 07	样本 09	样本 10	样本 12	样本 22	样本 27
SiO2	缩放	67.02	68.75	70.01	68.22	66.82	67.08
Na2O	中位数	0.00	0.00	0.00	0.00	0.00	0.00
K2O	中位数	9.83	9.83	9.83	9.83	9.83	9.83
CaO	缩放	6.56	3.80	1.29	4.41	10.17	5.76
MgO	中位数	1.17	1.17	1.17	1.17	1.17	1.17
Al2O3	缩放	6.79	4.53	2.78	5.01	12.01	8.61
Fe2O3	中位数	2.11	2.11	2.11	2.11	2.11	2.11
CuO	缩放	5.06	2.42	1.31	2.58	0.86	2.40
PbO	中位数	0.16	0.16	0.16	0.16	0.16	0.16
BaO	中位数	0.00	0.00	0.00	0.00	0.00	0.00
P2O5	缩放	3.06	1.75	1.02	0.75	1.05	1.80
SrO	中位数	0.02	0.02	0.02	0.02	0.02	0.02
SnO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00
SO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00

表 13　高钾玻璃风化前物质预测

物质类别	预测方法	02	08	11	19	26	26	34	36	38	39	40	41
SiO2	缩放	56.29	31.25	52.12	45.99	30.71	25.77	55.52	61.40	51.10	40.73	35.93	44.64
Na2O	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
K2O	中位数	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
CaO	缩放	1.24	0.79	1.86	1.55	0.76	1.60	0.41	0.20	0.36	0.59	0.99	2.63
MgO	中位数	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
Al2O3	缩放	3.06	1.09	2.19	2.91	0.57	0.96	1.32	1.31	2.10	0.41	0.37	2.72
Fe2O3	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CuO	缩放	0.21	8.25	3.91	2.78	8.38	2.85	1.20	0.54	0.58	0.70	0.00	0.15
PbO	缩放	30.25	18.29	16.19	27.31	18.83	19.08	29.68	26.53	31.45	38.92	44.77	28.14
BaO	缩放	0.00	33.08	15.48	5.67	34.16	37.55	10.59	11.47	10.37	7.65	7.09	10.34
P2O5	缩放	0.80	0.80	2.09	1.97	0.70	1.35	0.08	0.02	0.11	0.26	0.39	1.66
SrO	中位数	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
SnO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

表 14　铅钡玻璃风化前物质预测（1）

物质类别	预测方法	43	48	49	50	51_1	51_2	52	54	54	56	57	58
SiO2	缩放	19.26	62.75	49.67	42.90	38.19	33.13	39.94	34.57	41.55	45.23	39.44	47.15
Na2O	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
K2O	中位数	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
CaO	缩放	2.78	1.50	2.43	1.69	1.90	2.72	1.20	1.69	0.00	0.64	0.69	1.85
MgO	中位数	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
Al2O3	缩放	6.84	11.13	4.39	1.53	4.28	2.05	0.95	3.39	2.98	1.51	1.78	2.87
Fe2O3	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CuO	缩放	4.24	0.00	0.56	0.90	1.09	0.59	0.56	0.66	1.06	0.63	0.92	2.48
PbO	缩放	38.17	10.02	21.80	28.06	25.66	32.74	30.24	35.37	37.28	26.31	28.76	25.09
BaO	缩放	17.72	7.74	6.46	15.04	19.47	20.06	19.15	7.46	10.06	16.37	18.32	8.11
P2O5	缩放	0.00	0.25	2.47	1.41	1.80	1.95	1.27	0.94	3.15	0.57	0.00	2.00
SrO	中位数	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
SnO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SO2	中位数	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

表 15　铅钡玻璃风化前物质预测（2）

3.2　高钾玻璃与铅钡玻璃的主成分分析结果

	SiO2	Na2O	K2O	CaO	MgO	Al2O3	Fe2O3	CuO	PbO	BaO	P2O5	SrO	SnO2	SO2	clusters
0	69.33	0	9.99	6.32	0.87	3.93	1.74	3.87	0	0	1.17	0	0	0.39	0
1	87.05	0	5.19	2.01	0	4.06	0	0.78	0.25	0	0.66	0	0	0	1
2	61.71	0	12.37	5.87	1.11	5.5	2.16	5.09	1.41	2.86	0.7	0.1	0	0	0
3	65.88	0	9.67	7.12	1.56	6.44	2.06	2.18	0	0	0.79	0	0	0.36	0
4	61.58	0	10.95	7.35	1.77	7.5	2.62	3.27	0	0	0.94	0.06	0	0.47	0
5	67.65	0	7.37	0	1.98	11.15	2.39	2.51	0.2	1.38	4.18	0.11	0	0	0
6	59.81	0	7.68	5.41	1.73	10.05	6.04	2.18	0.35	0.97	4.5	0.12	0	0	0
7	59.01	2.86	12.53	8.7	0	6.16	2.88	4.73	0	0	1.27	0	0	0	0
8	62.47	3.38	12.28	8.23	0.66	9.23	0.5	0.47	1.62	0	0.16	0	0	0	0
9	65.18	2.1	14.52	8.27	0.52	6.18	0.42	1.07	0.11	0	0	0.04	0	0	0
10	79.46	0	9.42	0	1.53	3.05	0	0	0	0	1.36	0.07	2.36	0	1
11	76.68	0	0	4.71	1.22	6.19	2.37	3.28	1	1.97	1.1	0	0	0	1
12	92.63	0	0	1.07	0	1.98	0.17	3.24	0	0	0.61	0	0	0	1
13	95.02	0	0.59	0.62	0	1.32	0.32	1.55	0	0	0.35	0	0	0	1
14	96.77	0	0.92	0.21	0	0.81	0.26	0.84	0	0	0	0	0	0	1
15	94.29	0	1.01	0.72	0	1.46	0.29	1.65	0	0	0.15	0	0	0	1
16	92.35	0	0.74	1.66	0.64	3.5	0.35	0.55	0	0	0.21	0	0	0	1
17	92.72	0	0	0.94	0.54	2.51	0.2	1.54	0	0	0.36	0	0	0	1

表 16　高钾玻璃的主成分分析结果

	SiO2	Na2O	K2O	CaO	MgO	Al2O3	Fe2O3	CuO	PbO	BaO	P2O5	SrO	SnO2	SO2	clusters
0	37.36	0	0.71	0	0	5.45	1.51	4.78	9.3	23.55	5.75	0	0	0	1
1	31.94	0	0	0.47	0	1.59	0	8.46	29.14	26.23	0.14	0.91	0	0	2
2	34.34	0	1.41	4.49	0.98	4.35	2.12	0	39.22	10.29	0	0.35	0.4	0	0
3	36.93	0	0	4.24	0.51	3.86	2.74	0	37.74	10.35	1.41	0.48	0.44	0	0
4	65.91	0	0	1.6	0.89	3.11	4.59	0.44	16.55	3.42	1.62	0.3	0	0	1
5	69.71	0	0.21	0.46	0	2.36	1	0.11	19.76	4.88	0.17	0	0	0	1
6	75.51	0	0.15	0.64	1	2.35	0	0.47	16.16	3.55	0.13	0	0	0	1
7	65.91	0	0	0.38	0	1.44	0.17	0.16	22.05	5.68	0.42	0	0	0	1
8	60.12	0	0.23	0.89	0	2.72	0	3.01	17.24	10.34	1.46	0.31	0	3.66	1
9	61.28	2.66	0.11	0.84	0.74	5	0	0.53	15.99	10.96	0	0.23	0	0	1
10	55.21	0	0.25	0	1.67	4.79	0	0.77	25.25	10.06	0.2	0.43	0	0	1
11	51.54	4.66	0.29	0.87	0.61	3.06	0	0.65	25.4	9.23	0.1	0.85	0	0	1
12	49.01	2.71	0	1.13	0	1.45	0	0.86	32.92	7.95	0.35	0	0	0	1
13	36.28	0	1.05	2.34	1.18	5.73	1.86	0.26	47.43	0	3.57	0.19	0	0	0
14	20.14	0	0	1.48	0	1.34	0	10.41	28.68	31.23	3.59	0.37	0	2.58	2
15	33.59	0	0.21	3.51	0.71	2.69	0	4.93	25.39	14.61	9.38	0.37	0	0	0
16	29.64	0	0	2.93	0.59	3.57	1.33	3.51	42.82	5.35	8.83	0.19	0	0	0
17	53.79	7.92	0	0.5	0.71	1.42	0	2.99	16.98	11.86	0	0.33	0	0	1
18	50.61	2.31	0	0.63	0	1.9	1.55	1.12	31.9	6.65	0.19	0.2	0	0	1
19	19.79	0	0	1.44	0	0.7	0	10.57	29.53	32.25	3.13	0.45	0	1.96	2
20	3.72	0	0.4	3.01	0	1.18	0	3.6	29.92	35.45	6.04	0.62	0	15.95	2
21	68.08	0	0.26	1.34	1	4.7	0.41	0.33	17.14	4.04	1.04	0.12	0.23	0	1
22	63.3	0.92	0.3	2.98	1.49	14.34	0.81	0.74	12.31	2.03	0.41	0.25	0	0	1
23	35.78	0	0.25	0.78	0	1.62	0.47	1.51	46.55	10	0.34	0.22	0	0	0
24	39.57	2.22	0.14	0.37	0	1.6	0.32	0.68	41.61	10.83	0.07	0.22	0	0	0
25	32.93	1.38	0	0.68	0	2.57	0.29	0.73	49.31	9.79	0.48	0.41	0	0	0
26	26.25	0	0	1.11	0	0.5	0	0.88	61.03	7.22	1.16	0.61	0	0	0
27	16.71	0	0	1.87	0	0.45	0.19	0	70.21	6.69	1.77	0.68	0	0	0
28	18.46	0	0.44	4.96	2.73	3.33	1.79	0.19	44.12	9.76	7.46	0.47	0	0	0
29	51.26	5.74	0.15	0.79	1.09	3.53	0	2.67	21.88	10.47	0.08	0.35	0	0	1
30	51.33	5.68	0.35	0	1.16	5.66	0	2.72	20.12	10.88	0	0	0	0	1
31	12.41	0	0	5.24	0.89	2.25	0.76	5.35	59.85	7.29	0	0.64	0	0	0
32	21.7	0	0	6.4	0.95	3.41	1.39	1.51	44.75	3.26	12.83	0.47	0	0	0
33	60.74	3.06	0.2	2.14	0	12.69	0.77	0.43	13.61	5.22	0	0.26	0	0	1
34	53.33	0.8	0.32	2.82	1.54	13.65	1.03	0	15.71	7.31	1.1	0.25	1.31	0	1
35	28.79	0	0	4.58	1.47	5.38	2.74	0.7	34.18	6.1	11.1	0.46	0	0	0
36	54.61	0	0.3	2.08	1.2	6.5	1.27	0.45	23.02	4.19	4.32	0.3	0	0	1
37	17.98	0	0	3.19	0.47	1.87	0.33	1.13	44	14.2	6.34	0.66	0	0	0
38	45.02	0	0	3.12	0.54	4.16	0	0.7	30.61	6.22	6.34	0.23	0	0	1
39	24.61	0	0	3.58	1.19	5.25	1.19	1.37	40.24	8.94	8.1	0.39	0.47	0	0
40	21.35	0	0	5.13	1.45	2.51	0.42	0.75	51.34	0	8.75	0	0	0	0
41	25.74	1.22	0	2.27	0.55	1.16	0.23	0.7	47.42	8.64	5.71	0.44	0	0	0
42	63.66	3.04	0.11	0.78	1.14	6.06	0	0.54	13.66	8.99	0	0.27	0	0	1
43	22.28	0	0.32	3.19	1.28	4.15	0	0.83	55.46	7.04	4.24	0.88	0	0	0
44	17.11	0	0	0	1.11	3.65	0	1.34	58.46	0	14.13	1.12	0	0	0
45	29.15	0	0	1.21	0	1.85	0	0.79	41.25	15.45	2.54	0	0	0	0
46	25.42	0	0	1.31	0	2.18	0	1.16	45.1	17.3	0	0	0	0	0
47	30.39	0	0.34	3.49	0.79	3.52	0.86	3.13	39.35	7.66	8.99	0.24	0	0	0

表 18　铅钡玻璃的主成分分析结果