预测 WORLD 结果

摘要

Wordle 是纽约时报目前每天提供的流行拼图。玩家尝试通过在六次或更少的尝试中猜测一个五个字母的单词来解决这个难题，每次猜测都会收到反馈。对于这个版本，每个猜测都必须是一个实际的英文单词。不被比赛识别为单词的猜测是不允许的。Wordle 越来越受欢迎，该游戏的版本现已支持 60 多种语言。

纽约时报网站上的 Wordle 说明说明，在您提交单词后，图块的颜色会发生变化。黄色方块表示该方块中的字母在单词中，但位置错误。绿色方块表示该方块中的字母在单词中并且位于正确的位置。灰色方块表示该方块中的字母根本不包含在单词中（请参阅附件 2)。图 1 是一个示例解决方案，其中在三次尝试中找到了正确的结果。

对于问题一，报告结果的数量每天都在变化。开发一个模型来解释这种变化，并使用您的模型为 2023 年 3 月 1 日报告的结果数量创建一个预测区间。这个词的任何属性是否会影响报告的在困难模式下播放的分数的百分比？如果是这样，如何？如果不是，为什么不呢？

分析报告结果的统计规律，并根据记录的数量数据，预测其在 3 月 1 日的结果。首先我们要对提供的数据进行预处理，如异常值和多指标的去量纲处理，附件包括日期、比赛编号、当天的单词、当天报告分数的人数、困难模式下的玩家人数以及一次、两次、三次、四次猜中单词的百分比、五次尝试、六次尝试或无法解决难题（用 X 表示），研究各指标之间的变化规律可以绘制可视化的图表加以辅助。分析指标的统计规律，首先要借助最基本的统计量，如趋势、中位数、方差等等，接着还可以使用配对样本 t 检验处理连续数据，卡方独立性检验处理离散数据，灰色关联法也可以使用。预测短期内的数量，采用时间序列 ARIMA 和支持向量机进行预测，并对多种模式进行组合，以提高预测精度，可以进行一个对比。

对于问题二，对于未来日期的给定未来解决方案词，开发一个模型，使您可以预测报告结果的分布。换句话说，预测未来日期（1，2，3，4，5，6，X）的相关百分比。哪些不确定性与您的模型和预测相关？举一个你对 2023 年 3 月 1 日 EERIE 这个词的预测的具体例子。你对你的模型的预测有多自信？以实际例子预测其分布结果，采用神经网络进行多值预测，提高训练精度及维度。

对于问题三，开发并总结一个模型来按难度对解决方案单词进行分类。识别与每个分类关联的给定词的属性。使用您的模型，EERIE 这个词有多难？讨论分类模

型的准确性。这个题很明显是聚类分析问题，考虑常用的 K-MEANS 聚类分析问题，当然这个题大家都会考虑到这个算法，因此，需要改进创新。对算法进行优化，比如改进的 K-MEANS 聚类分析，K-means++；二分 k-means；k-medoids（k-中心聚类算法）；Kernel k-means；ISODATA；Mini Batch K-Means 等等，不同算法之间的好坏需要用指标去评估，如：误差平方和（SSE（The sum of squares due to error）；"肘"方法（Elbow method）-K 值确定；轮廓系数法（Silhouette Coefficient）；CH 系数（Calinski-Harabasz Index）等。研究多指标之间的关系，首先机器学习方法采用 XGBoost 为每个指标分配重要性分数构建因子分析-聚类模型，随后进行探索性因子分析对多变量进行降维和可视化，更直观地揭示各指标的差异性。在分类完成之后进行合理性与敏感性的分析，放上计算结果。对 EERIE 这个词属性，建立判别函数分析分类精度。

对于问题四，列出并描述该数据集的其他一些有趣的特征。

关键词：神经网络、预测、分类、支持向量机

一、问题的背景与重述

Wordle 是纽约时报当下每天提供的拼词游戏。 玩家尝试通过在六次或更少的尝试中猜测□个五个字母的单词来解决这个难题 ，每次猜测都会收到反馈。 对于这个版本，每个猜测都必须是□个实际的英文单词。 不被比赛识别为单词的猜测是不允许的。Wordle 越来越受欢迎 ，该游戏的版本现已支持 60 多种语言。纽约时报网站上的 Wordle 提示 ，在您提交单词后 ，图块的颜色会发生变化。 黄色方块表示该方块中的字母在单词中 ，但位置错误。绿色方块表示该方块中的字母在单词中并且位于正确的位置。灰色方块表示该方块中的字母根本不包含在单词中（见附件 2）。 图 1 是□个示例解决方案 ，其中在三次尝试中找到了正确的结果

玩家可以在常规模式或“困难模式”下玩。 Wordle 的困难模式要求玩家□旦在单词中找到正确的字母（方块为黄色或绿色），就必须在随后的猜测中使用这些字母 ，从而使游戏变得更加困难。 图 1 中的示例是在困难模式下运行的。

许多（但不是全部）用户在 Twitter 上分享了他们的得分。 针对这个问题 ，MCM 生成了□份 2022 年 1 月 7 日至 2022 年 12 月 31 日的日结果文件（见附件 1）。此文件包括日期、比赛编号、 当天的单词、 当天报告分数的□数、 困难模式下的玩家□数以及□次、两次、三次、 四次、五次、六次尝试猜中单词的百分比或无法解决难题（用 X 表示）。 例如图 2 中 2022 年 7 月 20 日的词是“TRITE” ，结果是通过挖掘推特得到的。尽管图 2 中的百分比总和为 100% ，但在某些情况下 ， 由于四舍五入的原因 ，这可能并非如此。

报告的在困难模式下播放的分数的百分比？ 如果是这样 ，是如何影响的？ 如果不是 ，请说明原因。对于未来日期的给定单词 ，开发□个模型 ，使您可以预测报告结果的分布。 换句话说 ，预测未来日期（1，2，3，4，5，6，X）的相关百分比。 哪些不确定性与您的模型和预测相关？ 举□个你对 2023 年 3 月 1 日 EERIE 这个词的预测的具体例子。你的模型可信度是多少？开发并总结□个模型来按难度对解决方案单词进行分类。 识别与每个分类关联的给定词的属性。 使用您的模型 ，EERIE 这个词有多难？ 讨论分类模型的准确性。

列出并描述该数据集的其他□些有趣的特征。

二、问题的分析

对于问题一，报告结果的数量每天都在变化。开发一个模型来解释这种变化，并使用您的模型为 2023 年 3 月 1 日报告的结果数量创建一个预测区间。这个词的任何属性是否会影响报告的在困难模式下播放的分数的百分比？如果是这样，如何？如果不是，为什么不呢？

分析报告结果的统计规律，并根据记录的数量数据，预测其在 3 月 1 日的结果。首先我们要对提供的数据进行预处理，如异常值和多指标的去量纲处理，附件包括日期、比赛编号、当天的单词、当天报告分数的人数、困难模式下的玩家人数以及一次、两次、三次、四次猜中单词的百分比、五次尝试、六次尝试或无法解决难题（用 X 表示），研究各指标之间的变化规律可以绘制可视化的图表加以辅助。分析指标的统计规律，首先要借助最基本的统计量，如趋势、中位数、方差等等，接着还可以使用配对样本 t 检验处理连续数据，卡方独立性检验处理离散数据，灰色关联法也可以使用。预测短期内的数量，采用时间序列 ARIMA 和支持向量机进行预测，并对多种模式进行组合，以提高预测精度，可以进行一个对比。

对于问题二，对于未来日期的给定未来解决方案词，开发一个模型，使您可以预测报告结果的分布。换句话说，预测未来日期（1，2，3，4，5，6，X）的相关百分比。哪些不确定性与您的模型和预测相关？举一个你对 2023 年 3 月 1 日 EERIE 这个词的预测的具体例子。你对你的模型的预测有多自信？以实际例子预测其分布结果，采用神经网络进行多值预测，提高训练精度及维度。

对于问题三，开发并总结一个模型来按难度对解决方案单词进行分类。识别与每个分类关联的给定词的属性。使用您的模型，EERIE 这个词有多难？讨论分类模型的准确性。这个题很明显是聚类分析问题，考虑常用的 K-MEANS 聚类分析问题，当然这个题大家都会考虑到这个算法，因此，需要改进创新。对算法进行优化，比如改进的 K-MEANS 聚类分析，K-means++；二分 k-means；k-medoids（k-中心聚类算法）；Kernel k-means；ISODATA；Mini Batch K-Means 等等，不同算法之间的好坏需要用指标去评估，如：误差平方和（SSE（The sum of squares due to error）；"肘"方法（Elbow method）-K 值确定；轮廓系数法（Silhouette Coefficient）；CH 系数（Calinski-Harabasz Index）等。研究多指标之间的关系，首先机器学习方法采用 XGBoost 为每个指标分配重要性分数构建因子分析-聚

类模型，随后进行探索性因子分析对多变量进行降维和可视化，更直观地揭示各指标的差异性。在分类完成之后进行合理性与敏感性的分析，放上计算结果。对 EERIE 这个词属性，建立判别函数分析分类精度。

对于问题四，列出并描述该数据集的其他一些有趣的特征。

三、问题的假设

1、假设附件提供的报告结果的数量的各项指标数量、尝试次数类型等的数据真实可信。

2、假设各玩家样本之间相互独立，在数量尝试次数、报告数量、困难模式数量之间不会相互影响。

四、符号说明

符号	说明	符号	说明
ρ	判别系数	y	困难模式与否
r_i	灰色关联度	x_1	玩家数量
x_i'	归一化后数据	x2	困难选择次数
θ_j	阈值	x3	单词个数
y_k	期望输出值	x4	报告数量
S	偏度	w_{ij}	尝试 i 次成功数
k	f 峰度	r_s	Spearman 相关系数

五、模型的建立与求解

首先对附件一和附件二的数据进行预处理，进行等精度测量，独立得到x_1，$x_2 \ldots$，x_n ，算出其算术平均值x及剩余误差v_i=x_i-x（i=1,2，...,n）,并按贝塞尔公式算出标准偏差δ，如果某个测量值x_b的剩余误差v_b（1≤b≤n）,满足下式：

$$|v_b|=|x_b\text{-x }|>3\delta$$

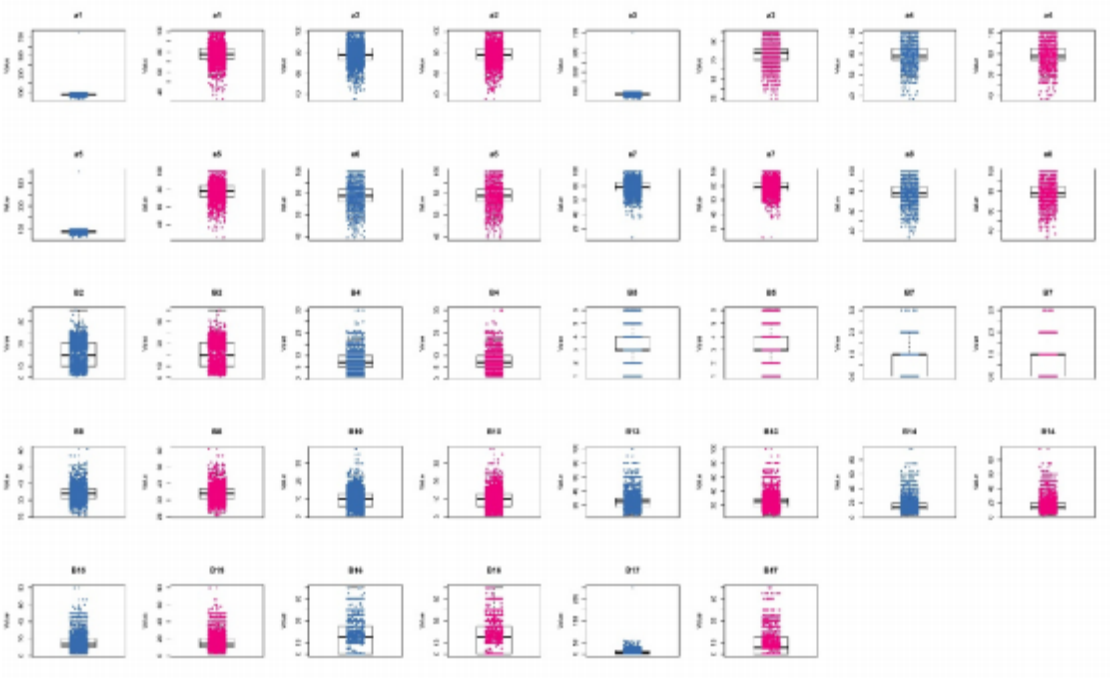
我们认为x_b是含有粗大误差值的坏值，应予以剔除。

对于一些不合常理的数据，我们假设这部分数据是因为手工写入差错或者抽样错误，并对这些数据进行适当处理，我们决定采取修改或者剔除的方法。我们假设给定指标的背景值范围限定在$x \pm 3\delta$区间内，因此我们认为在区间$(x - 3\delta, x + 3\delta)$内的数据为正常范围的值，同时我们相对应的剔除异常数据。然后运用箱线图准确判断极端异常值的值，对这一部分数据，我们将其剔除并插入相应的值。

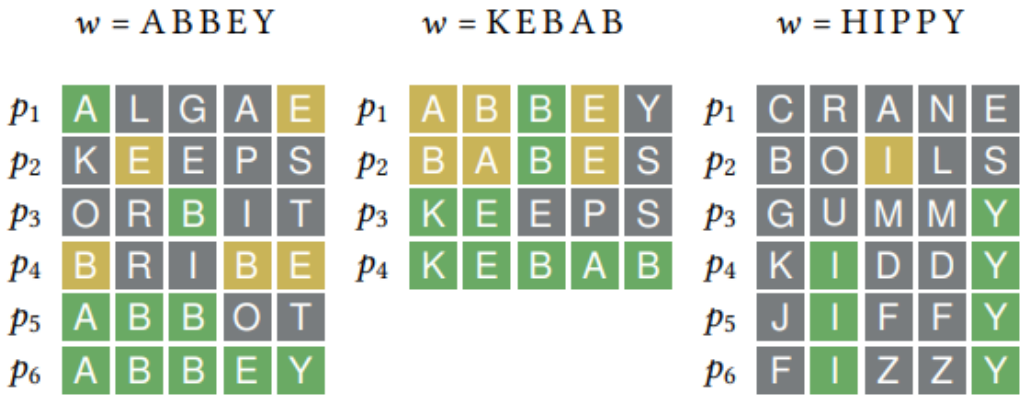
当数据当中存在异常值时， 尤其是存在着偏离较大的离群点时， 会对数据分析和建 模带来误差，因此，需要对异常值进行检测。常用的异常值检测方法包括 3σ 法则或 Z 分布方法， 而这一类方法是以正态分布为假设前提的。由于本文的一部分数值型特征的 分布并不符合正态分布， 即数据分布不均匀。故本文选择使用对数据分布没有要求的箱 线图，对数值型特性进行异常值检测。

使用箱线图对数据进行异常值检测的原理为：通过计算四分位数加减 1.5 倍四分位 距，即是计算 Q1－1.5IQR 和 Q3+1.5IQR 的值，规定落在这一区间之外的数据为异常点。 在箱线图中， 可以看出变量数据的中位数、上四分位数、下四分

位数、上下边缘和潜在 异常点。本文通过使用上四分位数代替数值大于 Q3+1.5IQR 的数据， 使用下四分位数 代替数值小于 Q1－1.5IQR 的数据，并绘制出了异常值处理前后的箱线图， 如图 1 所示。



游戏的内容是的一套单词D(通常称为字典)，所有的长度都是k，这是玩家完全知道的。形式上，对于一个给定的字母 Σ，$D \subseteq \Sigma k$． 玩家，我们将其标记为猜测者，希望通过一系列最多 ℓ 找到一个秘字$w \in D$猜猜$p1$，$p2$，……,$p\ell$，所有这些必须是属于D的单词。的猜测者据说如果它的猜测之一就会获胜pi正好等于w，如果不是她就输了这种匹配发生在 ℓ 猜测之后。每当做出p的猜测时，猜测者接收一些关于p和密字w之间关系的信息:她不知道每一个职位i，以致$p[i]=w[i]$，也不知道职位i使得字母$p[i]$出现在单词w中，而不在i的位置上。更多的准确地说，为了处理重复字母的情况，我们将使用标记的概念与颜色。在索引i上迭代，从 1 到k，我们说$w[i]$标记$p[i]$with 绿色 i$w[i]=p[i]$。如果$w[i] \neq p[i]$，则集合Si = { j : p [j] = w[i], p [j] is unmarked}



灰色关联分析

利用灰色系统的理论，对各个子系统进行了灰色关联度分析，并利用无量纲处理、求差值、算关联度等方法，以克服各个因子的量纲差别，并对各个子系统（或要素）的数字关系进行定量分析。其中，灰色关联度分析是基于序列的可比性和相似度，对系统发展的主要方向与各个影响因子的相关性进行了分析，从而为某一体系的演变趋势提供了定量的衡量指标。

在客观体系中，灰色系统分析能更真实、更全面地反映出人们对客观体系的认识，并能获得定性和定量的双重效果。其步骤如下：

（a）以玩家数量为母序列，$x_0(k)$为各指标影响因素，以与相邻的玩家数量指标为子序列。

（b）根据下式求差序列Δi（k），并找出计算结果中的最大绝对差值Δmax和最小绝对差值Δmin，$\Delta i(k) = x_i'(k) - x_0'(k)$.

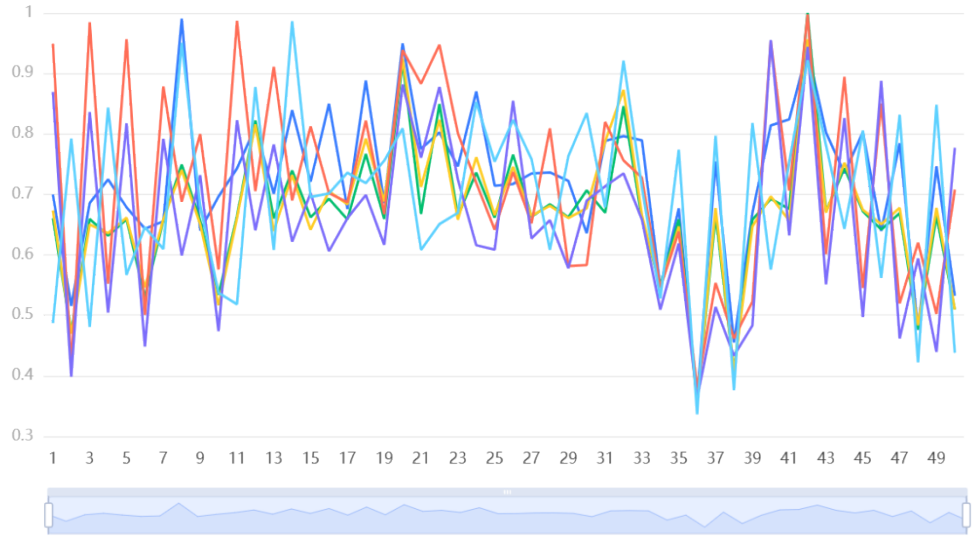
（c）求各站址对应的关联系数，$q_i(k)$，i＝1，2，3 …与关联度γ_i，i＝ 1，2，3… γ_1，γ_2、 γ_3.

$$q_i(k) = \frac{\Delta min + \rho \times \Delta max}{\Delta i(k) + \rho \times \Delta max}$$

$$\gamma_i = \frac{1}{n}\sum_{k=1}^{n} q_i(k)$$

其中ρ为分辨系数，一般取值为 0.5。我们认为γ_i值越接近于 1，该因素与系统发展主方向的关联性越好。运用热力学图和 EXCEL 软件对关联度进行线性求解，其结果如下。

1	0.92071553	-0.4730149	-0.8479046	-0.8383453	0.51344277
0.92071553	1	-0.3545402	-0.7726797	-0.788825	0.4691783
-0.4730149	-0.3545402	1	0.56887624	0.28267425	-0.387596
-0.8479046	-0.7726797	0.56887624	1	0.78944523	-0.6480228
-0.8383453	-0.788825	0.28267425	0.78944523	1	-0.7784353
0.51344277	0.4691783	-0.387596	-0.6480228	-0.7784353	1



玩家数量的可视化结果如上图所示。

基于对于题中十二个指标的降维，变成六种因素，可分别得出玩家数量变化与各指标之间的关系，由此，可以分析并得到不同属性的单词下的困难模式数量之间的差异性，例如对于 excel 单词的影响为正值 3.55，而对于 EERIE 来说则是负相关函数系数为-4.98，

评价项	关联度	排名
1	0.724	1
2	0.718	2
3	0.697	3
4	0.671	4
5	0.668	5
6	0.662	6

每个评估项目和“参考值”（母序列）的相似性，是通过相关系数的平均计算得到的。通过挖掘出的相关关系，可以从一种属性的信息中推导出另一种属性的信息，在一定程度上，该规则就能被证明是正确的。相关度愈大，则评估项目与“参考值”之间的关联度愈高，关联度愈高，则评估项目与“参考值”的关联度愈高。将各个评估指标与关联度相结合，得出各项指标的权重。

基于加权移动平均法的预测模型

根据历史记录检测数据，预测其在3月1日的变化区间。该问题中历史的数量在3月前构成时间序列，因而预测3月1日的各种各指标数量适用于时间序列模型。时间序列模型中常用的方法有移动平均法、指数平滑法和自适应滤波法，我们选择加权移动平均法进行预测。

进一步分析，由于玩家的游戏模式难度的不同，导致可能有部分数量未检测到，因此数据整体上出现“0”值，我们对数据进行加权求平均值处理，在权重计算部分使用正态分布曲线函数进行权重的分配，计算过程如下：

设定玩家数量，检测结果、困难模式结果数量等等指标12种分别为x1,x2，… x12，其权重分别为w1,w2 …,w12（某种指标的含量越高，其相应的权重越高），变化后后12种指标为x1′,x2′ ,… w14′ 权重为w1’ ，w2’，… ，w14’。在赋予权重的时候可采用标准正态分布对各数量的权重进行赋值，标准正态分布函数公式如下：

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{x^2}{2}}$$

将每一列数据与标准正态分布函数进行相乘，得到其加权平均数为：（加入公式）

$$\bar{x} = \frac{X_1W_1 + X_2W_2 + \ldots + X_{14}W_{14}}{W_1W_2 + \ldots + W_{14}}$$

$$\overline{x'} = \frac{x'_1w'_1 + x'_2\omega'_2 + \ldots + x'_{14}w'_{14}}{w'_1 + w'_2 + \ldots + w'_4}$$

规定 α_i 为第 i 个模式下的数量前后的加权占比，则

$$\alpha_i = \frac{\overline{x_i}}{x_i}$$

使用 Matlab 对该题进行编程求解，预测出不同困难模式下的数量在 3 月 1 日的区间范围。

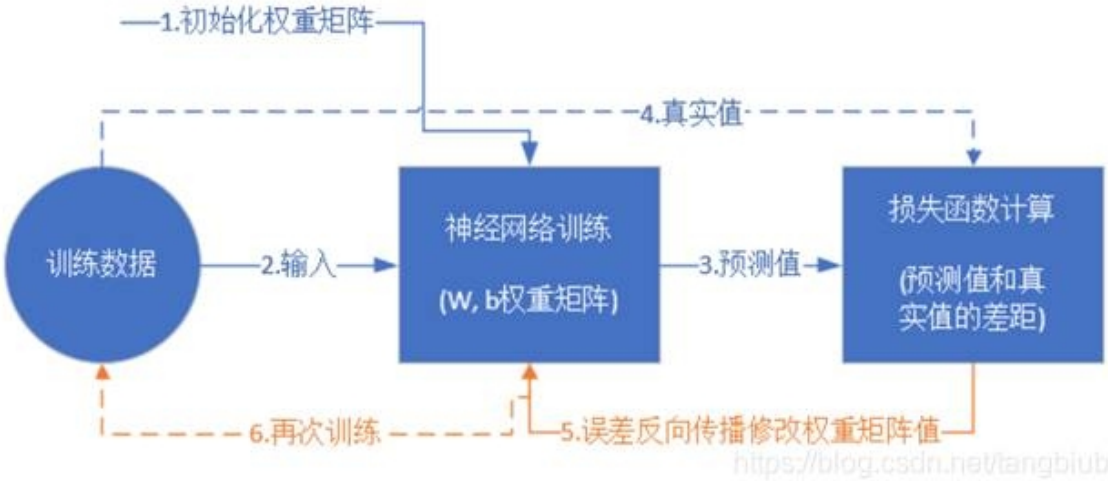
Contest number	报告数量	困难数量	x1	:::	x2	x3	:::	x6
533	32018	4733	49.30		37.44	0.00		0.00
532	31191	4835	55.76		25.56	2.99		0.00
531	35724	4809	57.46		30.71	0.00		0.00
530	31903	4906	56.75	:::	20.62	7.36	:::	0.00
529	35	4928	17 52		27 41	41 54		2 63

	3 4 3							
528	33 66 0	4973	13.76		24.59	46.80		2.71
527	34 28 1	5238	37.24		39.92	7.81		0.00
532	32 01 8	4733	75.58805		0	0		0

问题二的求解

利用 BP 神经网络预测结果分布情况

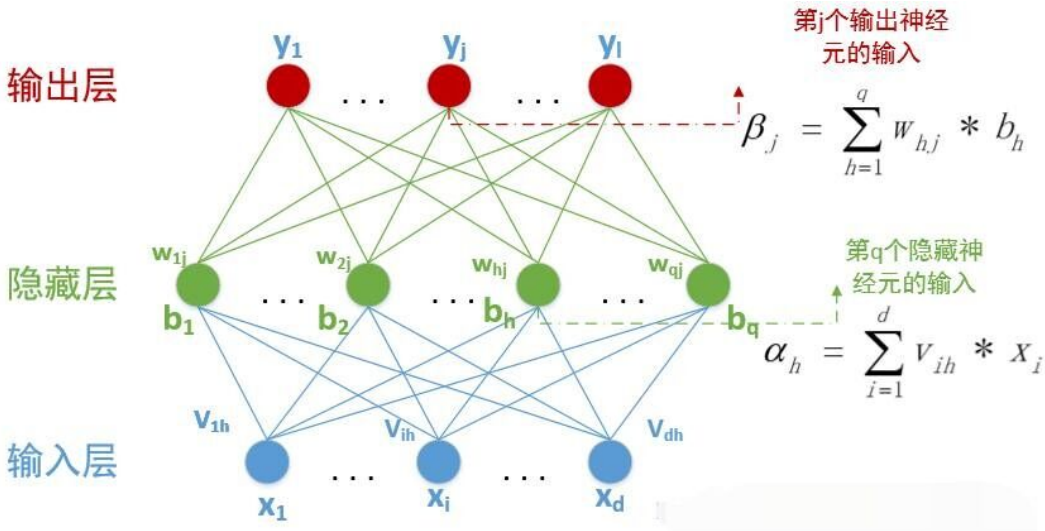
BP 神经网络是学习过程由误差进行反方向传播的一类 BP 网，利用 matlab 实现。每次根据训练得到的结果与预想结果进行误差分析，进而修改权值和阈值，一步一步得到能输出和预想结果一致的模型。具体流程如下图：



把输入定义为困难模式下的尝试次数，输出则为占据的百分比，问题就转化成了复杂函数的映射问题，采用的隐藏层为五层并做仿真处理。

它可以看作是一个复杂的函数映射问题，其中尝试次数数据是输入，占比数据预测值是输出。将该函数作为 BP 神经网络仿真的激活函数。将处理好的数值转化为(0,1)的范围，将一种将连续型的数据离散化为离散型数据。

Step1 训练集与测试集用异常值剔除与归一处理得到的各指标数据随机作为验证的样本以及训练集，得到乱序的 30 个数据的训练集与包括 5 个数据的验证集。　　Step2　BP 神经网络的构建将函数将神经元的输入映射到输出端，为了得到输入和输出数据之间的关系，我们在最后一个隐藏层和输出层之间使用一个函数，表达式如下。

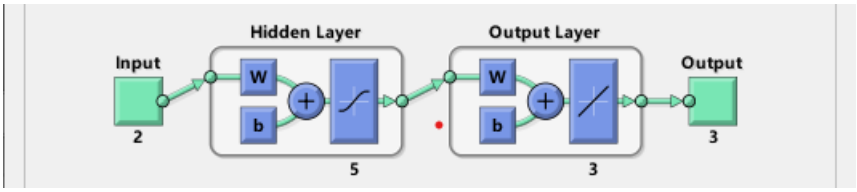


得到网络输入数据与输出数据之间的表达式

$$y_k \text{ ‘ } = \sum_{j=1}^{r} v_j \cdot \mathrm{f}[\sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{w_{ij}} \cdot \mathrm{p_i} + \theta_\mathrm{j}$$

(k = 1,2，…N)，为链路权值，为阈值，为期望输出值，为网络实际输出值。

神经网络结构示意图如下图所示:



Step3　BP 神经网络的参数在本次模型中设置的参数如下所示：

最大训练步数	训练结果间隔步数	学习速率	训练目标误差	训练次数
1000	1	0.0000001	0.000001	1000

下表为 BP 网络的预测结果

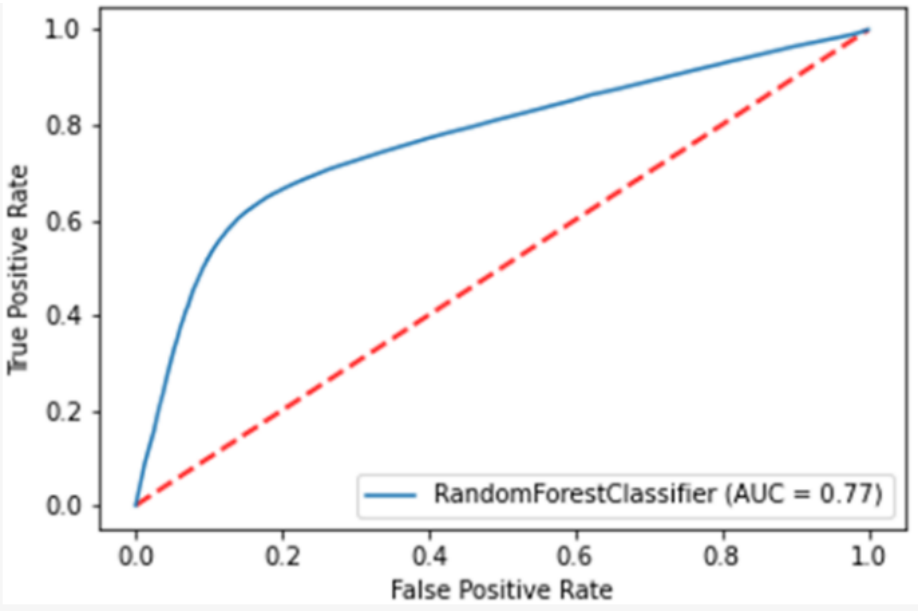
EERIE 不同尝试次数占比预测表(部分)

采样点	占比/%	方差	精度	准确度
1	2.28	10.23	7.98	2.15
2	6.31	14.52	3.15	1.23
3	14.12	2.59	6.1860	7.5895
4	35.16	2.39	5.8179	5.1191

5	30.89	3.43	6.76	3.77
6	10.21	2.96	9.55	4.31
X	74.86	2.22	5.39	8.17

粘性水平是在特征提取期间模拟游戏时间不平衡性的特征。它的计算方法是计算该时间段的前 30%和后 30%之间的事件计数之差。负值表示用户在观察到的时间间隔结束时玩得更多，因此不太可能尝试次数过多，与粘性水平为正的玩家相反。粘性向量提供了玩家活动分布的更详细视图。它由 7 个元素组成，每个元素代表特征提取周期总运行时间的 10%。元素的值是在相应的时间间隔内记录的整个期间内事件的百分比

各种组合的特征提取和流失期持续时间进行了广泛的测试，再加上几种机器学习算法。对于特征提取，考虑了 1,2,3 和 7 天的间隔，而预测了持续 1 到 7 天的周期，总共有 28 个组合。所得的剖面被用作逻辑回归、随机森林和 k 近邻的输入。以这种方式构建的模型使用先前描述的度量进行评估。仅根据精度判断，RF 算法的 72%将占上风，而 k 近邻算法排名最低，为 69%。而 F1 得分方面，logistic 回归的最高结果为 0.78。



问题三的建立与求解

对不同的单词属性基于难度进行分类，并判断 EERIE 这个单词属于一类，进一步在分类划分的基础上，针对分类模型的合理性和敏感性分析，我们建立了单词类型的系统聚类模型，在此基础上，根据两种划分通过标准差问题两种整体建模过程如下图所示

数据预处理及XGBoost算法

单词类型	分类特征二次筛选后的特征集
普通	玩家数量
困难	困难数量

使用 Excel 软件去除两种单词类型中差异明显的值，得到下表：

单词特征类型筛选结果表

类型	＞6	总精度范围
困难	是	99 81%- 100%
	否	97.25%- 100%
普通	是	90. 17%-99.89%
	否	88.41%-99.98%

再用基于机器学习方法的特征工程检验各类特征的重要程度。

对于不同类型分类问题的特征筛选，可以采用机器学习策略。采用 XGBoost 算法进行测试。该方法的核心思想就是不断的添加树去拟合上次的残差，以支持并行化和交叉验证提高划分效果。

$$IG\left(X,Y\right)=H\left(Y\right)-H\left(Y\middle|X\right)=\sum_{x,y}p\left(x\right)\cdot p\left(y\middle|x\right)-\sum_{y}p\left(y\right)\cdot\log p\left(y\right)$$

最后使用信息熵，根据信息增益最大的特征，将信息增益从小到大排序，得到最重要的特征。

分类及亚类划分

数据集中连续属性的配对样本 t 检验，不连续属性的卡方独立检验，结果分别如下：

单词类型	卡方检验得分阈值	分类特征一次筛选后的特征集
普通	1	1 、3 、6 、12
困难	10	3 、9 、11 、12

在基础分类上进行二次筛选，并采用最大信息系数法分析大样本数据集，将非线性的变量关系分析整合。公式如下：

$$I\left(X;Y\right)=\sum_{x\in X}\sum_{y\in Y}p\left(x,y\right)\log\frac{p\left(x,y\right)}{p\left(x\right)p\left(y\right)}$$

单词种类的系统聚类模型

系统聚类（层次聚类）算法：

Step1:将每个功能归类

Step2:计算类之间的距离矩阵，最近的两个类为新类

Step3:计算新班级与每个班级之间的距离。当类数量为 1 时，执行下一步，否则转到 Step2

Step4:绘制集群图

Step5:确定集群的数量和类别

支持向量机模型辅助分类

支持向量机模型

首先，需要从原始数据中提取训练集和测试集，然后进行预处理。然后利用训练集对支持向量机进行训练，最后利用得到的模型对测试集的分类标签进行预

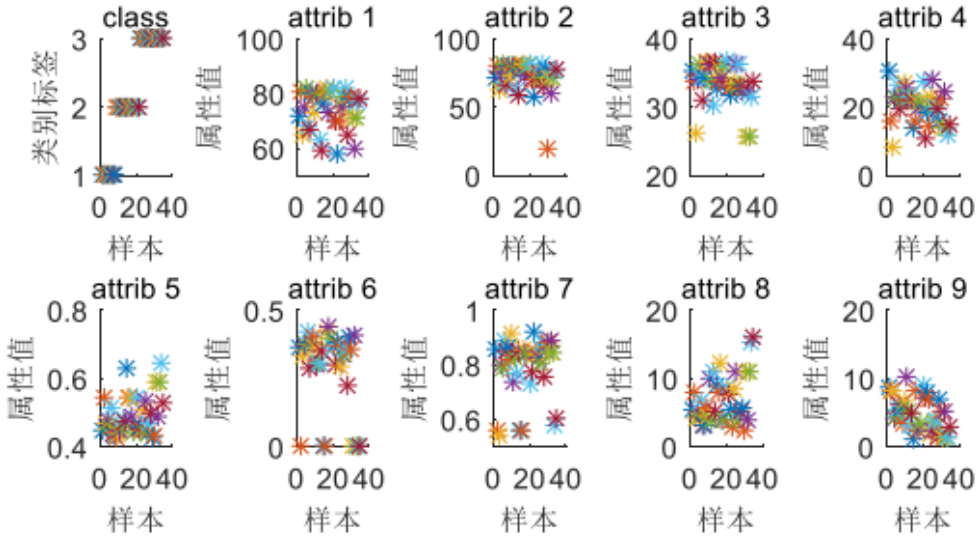
测，流程如下：

a)选定训练集和测试集

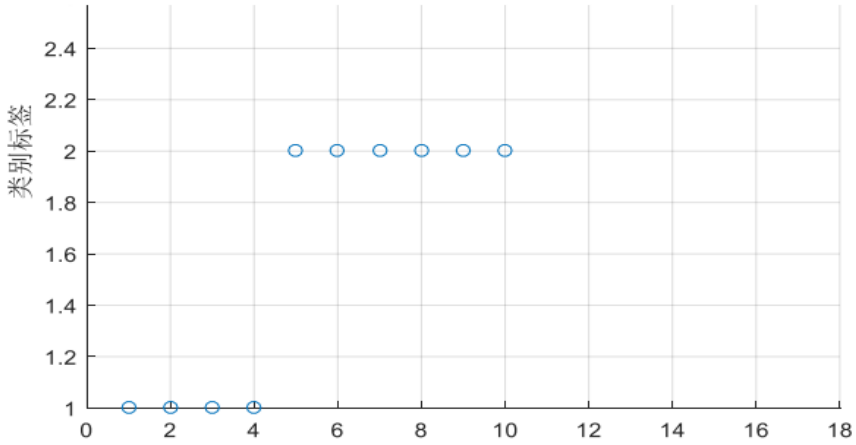
在 35 个样本中，样本 1-8 属于第一类(类别标签 1)，样本 9-21 属于第二类(类别标签 2)，样本 22-35 属于第三类(类别标签 3)。现在将每个类别分为两组并重新组合数据，一组作为训练集(train_wine)，另一组作为测试集(test.wine)。

b)训练与预测

使用训练集 train_Wine 对 SVM 分类器进行训练，得到的模型用于预测测试集的标签。最后，获得了分类精度。



可视化结果



数据分类

模型结果

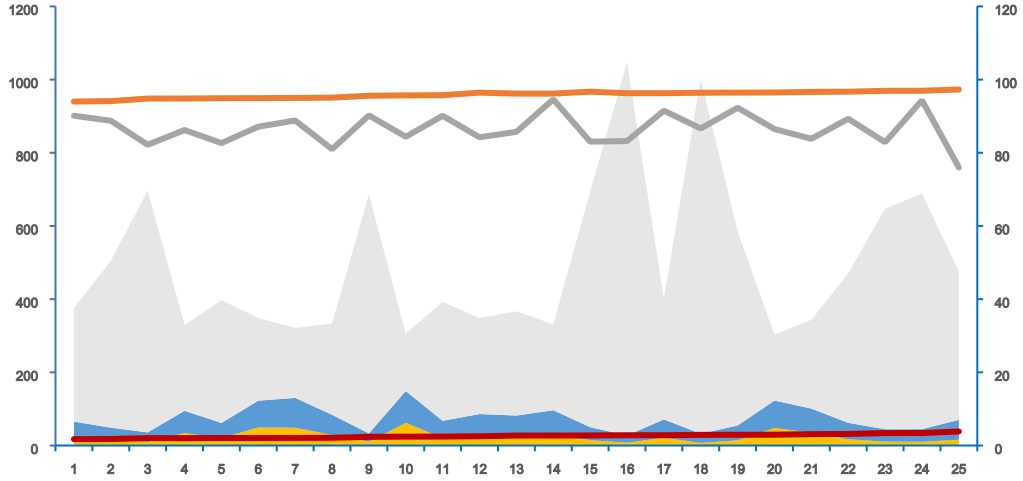
算法	类型数量	真实数量	正确率
支持向量机	**69**	**60**	**86.7%**

分类最终收敛，敏感性较好

得出聚类中心点与每一单词成分的分类

Calinski-harabasz 判据有时也被称为方差比判据(VRC)，它可以用来确定聚类的最佳 K 值，在各个变量之间绘制散点图，并标记聚类中心。聚类效果通过对角线上的直方图来观察，交叉部分越小聚类效果越好，反之聚类效果越差。

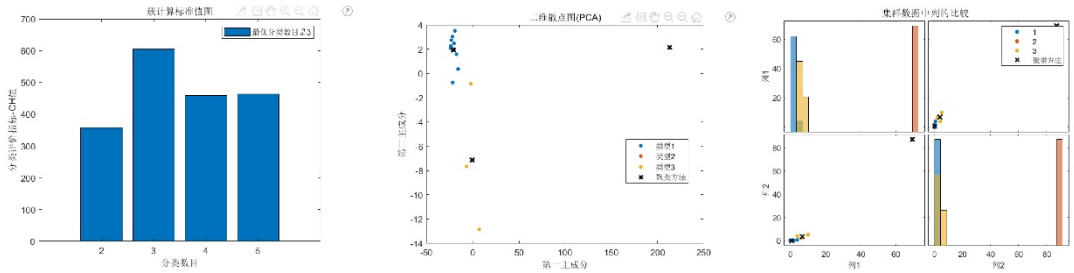
利用 Matlab 编程求解聚类中心点和各单词成分的分类，可视化分析普通单词和困难单词前后的对比，如下图所示:



K-Means++模型的评价指标选取与求解

Calinski-Harabasz 准可以用来确定聚类的最佳 K 值，对应具有较大的簇间方差和较小的簇内方差，最佳聚类数对应于具有最高 Calinski-Harabasz 指数值的解，在 Matlab 中设置范围为 2-5，在此范围内 C-H 值越大越好；

将每一个变量之间画出一个散点图，并标出聚类中心，通过对角线上的直方



图观察聚类效果，交叉部分越小说明聚类效果越好，反之则说明越差。最终使用 Matlab 进行编程求解得出了聚类中心点与各个单词成分的分类情况，通过对普通单词与困难单词在风化前后的对比进行可视化分析结果在附件中查看，普通单词的聚类情况分析图如下：

普通单词的聚类情况分析

求出每一个类别聚类中心点数据，由于数据量较大，我们将其放在附录 3 中体现，不同单词种类单词成分所属类别在风化前后的对比情况如下表所示：

不同类型单词所属类别的标签

所属类别	分类标签			
	普通 1	普通 2	困难 1	困难 2
movie	2	2	2	3
cater	1	1	1	2
tease	3	1	1	2
smelt	3	4	1	2
focus	1	1	1	2
today	3	3	5	2
watch	1	1	1	2
lapse	1	4	1	2
month	1	1	3	1
sweet	1	1	4	4
hoard	1	1	1	2
cloth	1	1	1	2
brine	1	1	1	2
ahead	1	1	1	2

不同类别之间的单词属性关联关系的差异性判断

在 K-Means++聚类结果的基础上选择一些合适的指标进行分类结果差异性的判断，本文我们选择 CHI、DBI、轮廓系数进行比较分析，三个指标的说明如下：

1)**CHI 指数：**CHI 指标是数据集的分离度与紧密度的比值，以各类中心点与数据集的中心点的距离平方和来度量数据集的分离度，以类内各点与其类中心的距离平方和来度量数据的紧密度。聚类效果越好，类间差距应该越大，类内差距越小，即类自身越紧密，类间越分散，CH 指标值越大聚类效果越好。

2)**DBI 指数：**Davies-Bouldin 指数（DBI）（由大卫 L●戴维斯和唐纳德●Bouldin 提出）是一种评估度量的聚类算法。

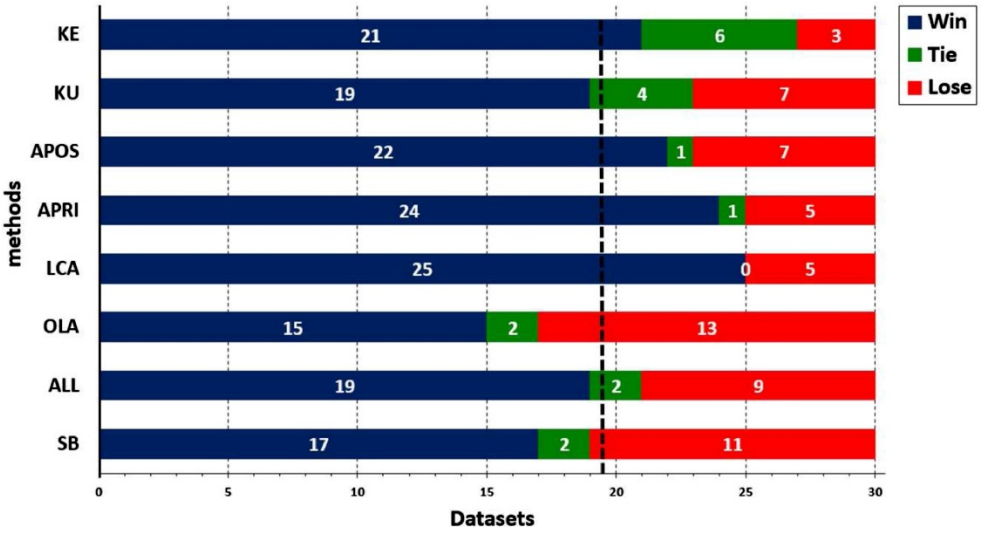
3)**轮廓系数：**轮廓系数（Silhouette Coefficient），是聚类效果好坏的一种评价方式。最佳值为 1，最差值为-1。接近 0 的值表示重叠的群集。负值通常表示样本已分配给错误的聚类，因为不同的聚类更为相似。

使用 Matlab 进行编程求解得出了不同单词类型风化前后的指标变化情况，具体分析如下表所示：

K-Means++聚类评价指标对比分析表

类别	最优分类数目	CHI	DBI	轮廓系数
普通 1	3	603.9374	0.41187	0.83888
普通 2	4	21478.3469	0.40376	0.80504
困难 1	5	387.4223	0.13902	0.88246
困难 2	4	156.951	0.11074	0.93844

由上表分析可得，普通类单词分类数目增加，困难的单词分类数目减少。当普通类单词 CHI 指数明显增加，困难类单词相对减少，通过 CHI 指数可以明显的判断出不同单词类型的聚类效果；针对 DBI 指数，普通类单词。DBI 指数增加千倍后减少；针对轮廓系数，普通单词风化后更远离+1，而困难单词风化后更接近于+1，说明困难单词成分关联关系较强，而普通单词成分关联关系相对减弱。



单词种类的系统聚类分析模型：

步骤 1：将每个样品看成一类，每类有且只有一个样品，共有 n 类。

步骤 2：计算 n 个样品两两间的距离，构造距离矩阵，并将距离最近的两类合并成为一个新类

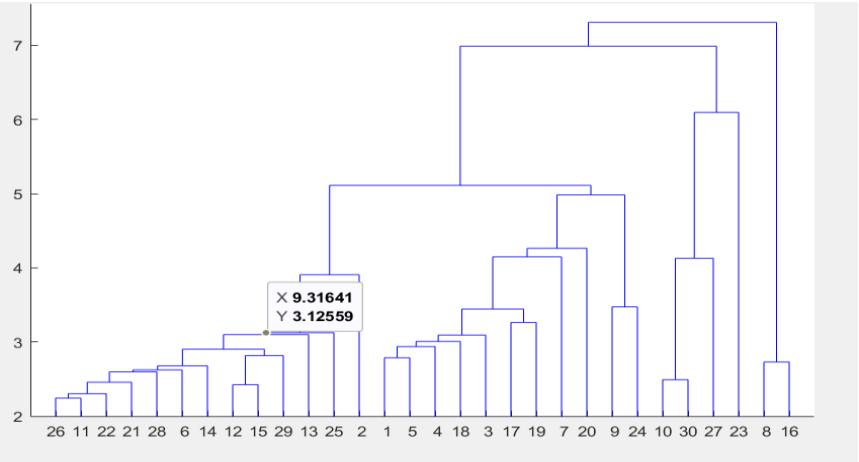
步骤 3：计算新类与当前各类之间的距离。若类的个数等于 1，则进行下一步， 否则转到步骤 2

步骤 4：画聚类图

步骤 5：决定类的个数和每类包含的样品数，并对类做出相应的解释。

使用 Matlab 进行编程求解,在编程求解中采用欧式距离计算样品两两间的距离，用最短距离法计算

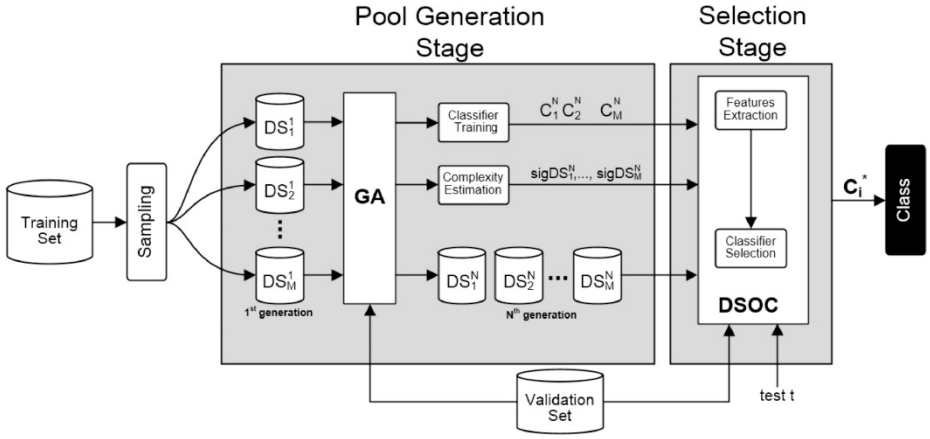
结果放入谱系图，如下图所示：



谱系图

以统计学分析和聚类方法为基础的模型：因子分析-聚类模型

考虑到前面的概念，我们现在可以定义问题复杂性空间。给定一个标准训练集 D，以及生成 M 个新的训练子集 DSi 的策略，每个子集只包含从 D 中获得的样本的一个百分比，通过统一采样和替换，对于每个 DSi，我们可以通过复杂度度量来估计难度。图 2 给出了给定分类问题的复杂度空间，用两个复杂度度量(F1 和 N2)表示。该空间中的每个元素对应于具有自己难度级别的子问题(数据子集，DSi)。应该提到的是，用于从 D 生成数据子集的策略在问题空间表示中起着重要作用。为了更清楚地解释，考虑给定测试实例在同一空间中的投影邻域，以找到类似的子问题。在图 2 的例子中，最相似的子问题是由 DSi 表示的。我们期望在 DSi 上训练的分类器可以提供必要的技能来处理测试实例 t。



我们选择区分聚类和画像描述来处理特征进而了解单词属性与各个因素的联系。我们选择对 K-Means 算法和层次聚类的对比。

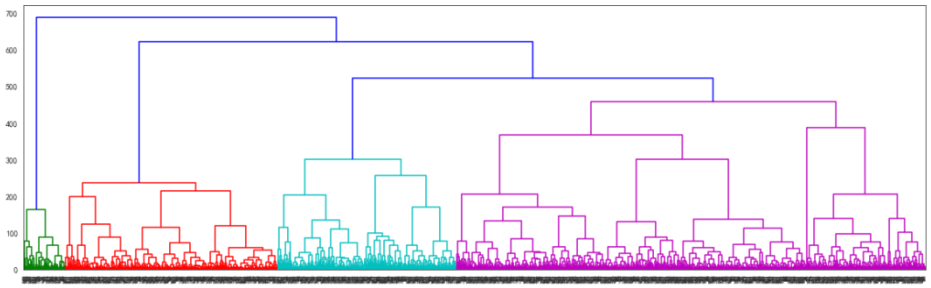
K-Means 算法属于无监督学习，它将相似的对象归到同一个簇中。它的中心思想为：设置一个 k 值随机给定 K 个初始的类簇中心点，把数据分配到离其最近的类簇中心点所含的类簇中，待所有点都具有自己所含的类簇后，再根据该类簇中的所有点的数值通过取平均值的方法再一次计算出新的类簇中心点，重复上述操作直到中心点的几乎无变化或者重复循环了足够的迭代次数。具有算法相对于来说快速简单，适合挖掘大规模的数据集，对于大规模的数据集拥有较高的效率而且还具有可伸缩性，整体的应用十分广泛。

层次聚类是非常的聚类算法的一种，其主要是以层次为基础，根据数据类别的不同但彼此之间的相似度来组成极具层次的一颗嵌套聚类树。从这颗树上，我们无论是自下而上进行合并又或者是自上而下的分类都可以获得到可以直接观察的可视图，使我们可以更加生动直接的观察。

轮廓系数也可以叫 Silhouette 分数，主要的思路是通过测量样本 i 到该类簇其他点的距离进而获得平均值 a，得到的 a 值越小就可以看出 i 越该属于该类簇。设 a 为 i 是否属于该类簇的归属度也可称为相似度。与此同时我们将测量出该样本距离其他类簇 C 中样本的距离平均值作为 b，我们就把 b 叫做该样本和类簇 C 的不归属度也称为不相似度。当得到的数值越靠近 1 说明该样本点的聚类越合适。

$$s(i)=\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}$$

通过上述操纵我们可以得出这两种方法下，效果最佳时都是聚类簇数量为 2 时。同时这两种方法的聚类结果大致上是相同的，下图为运用层次聚类法所得到的层次图。

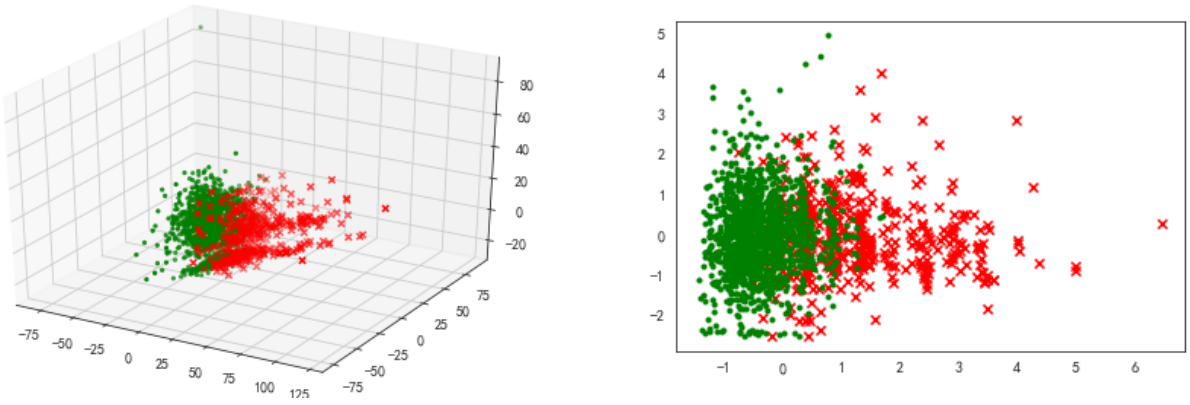


层次聚类的层次树

系统聚类法

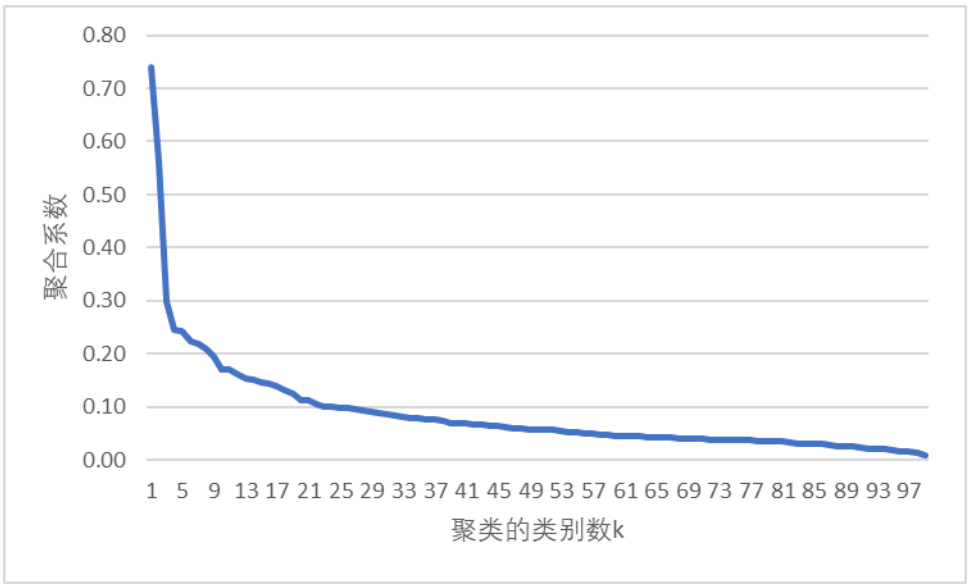
系统聚类法也被称为分层聚类法，是聚类分析中的常见方法。它的操作方法是将各个样品自身作为一类，之后将距离最小的群品首先聚合为一小类，然后把聚合到一起的小类根据各小类之间的距离再进行合并，以此不断下去，最终都合并至大类中。其好处是可得到从粗到细的诸多分类的情况。详细的操作流程如下图所示

将降维之后的聚类图与三维坐标系进行绘制，通过在因子 0 和因子 1 在二维平面上组成的投影制出因子分析-聚类模型



因子分析-聚类模型

下图所示是我们通过 SPSS 的聚类得到的相应结果，运用肘部法进一步得到 K(聚类的类别数)，以聚合系数为 y 轴，聚类的类别数 k 为 x 轴通过 SPSS 做出的肘部图如下图所示。



肘部图

根据观察图的整体趋势，可以看出k为5时聚合系数降低趋势明显减缓，所以我们认为聚类的类别数是5。

判别函数法

对于判别函数法，我们采用的是假设十个不同等级难度的单词，根据六个指标因素来得到，对于一个新的单词，我们只需要将其代入到我们的判别方程中去即可得到它所属于的难度级别。

十个级别的单词难度水平在 d 维特征空间里的样本均值量为：

$$M_i = \frac{1}{n_i}\sum y_k \in Y_i y_k \quad ,i = 1,2,\ldots,10$$

将 ω 变换映射到一维特征空间后，各类的平均值为:

$$m_i = \frac{1}{n_i}\sum y_k \in Y_i y_k \quad ,i = 1,2,\ldots,10$$

映射后，各类样本“类内离散度”定义为:

$$S_i^2 = \sum y_k \in Y_i (y_k - m_i)^2 \quad ,i = 1,2,\ldots,10$$

显然，我们希望在映射后，十个类的平均值之间的距离越大越好，每个类的类内发散越小越好。因此，Fisher 准则函数为:

$$J_F(\omega) = \frac{(m_1 - m_2)^2}{S_1^2 + S_2^2 + S_3^2}$$

所以最大解就是最佳解向量，也就是 Fisher 线性判别法。

$$\omega = S_\omega^{-1}(M_1 - M_2 - M_3)$$

其中，w是类内总离散度矩阵。

对十类判别函数进行求解，得到了决定系数和常数，结果如下表所示：

	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00	9.00
1	3.55 E-11	-4.98 E-12	-1.33 E-11	-1.46 E-11	-2.49 E-11	-4.93 E-13	-7.31 E-12	-5.35 E-11	-1.51 E-12
2	-8.51 E-11	2.26 E-11	7.09 E-11	5.10 E-11	8.96 E-11	1.92 E-11	3.36 E-11	1.04 E-10	8.69 E-11
3	0.00	0.00	0.00	0.00	-0.03	-5.37 E-5	0.00	0.00	-0.04
4	6.48 E-11	-9.39 E-11	-1.69 E-10	-1.69 E-10	-3.06 E-11	-8.58 E-11	-1.03 E-10	-1.54 E-10	-3.54 E-11
5	0.019	0.028	0.016	0.016	0.02	0.01	0.02	0.02	0.06

6	-2.66 E-11	4.10 E-11	3.59 E-11	4.49 E-11	7.52 E-11	1.01 E-11	2.20 E-11	1.03 E-10	3.45 E-11
常量	-3.33	-4.33	-3.47	-3.03	-4.41	-2.53	-2.78	-4.75	-11.2

$$F_1 = 3.55E^{-11}x_1 - 8.51E^{-11}x_2 + 6.48E^{-11}x_4 + 0.019x_5 - 2.66E^{-11}x_6 - 3.33$$

$$F_2 = -4.98E^{-12}x_1 + 2.26E^{-11}x_2 - 9.39E^{-11}x_4 + 0.028x_5 + 4.10E^{-11}x_6 - 14.33$$

……

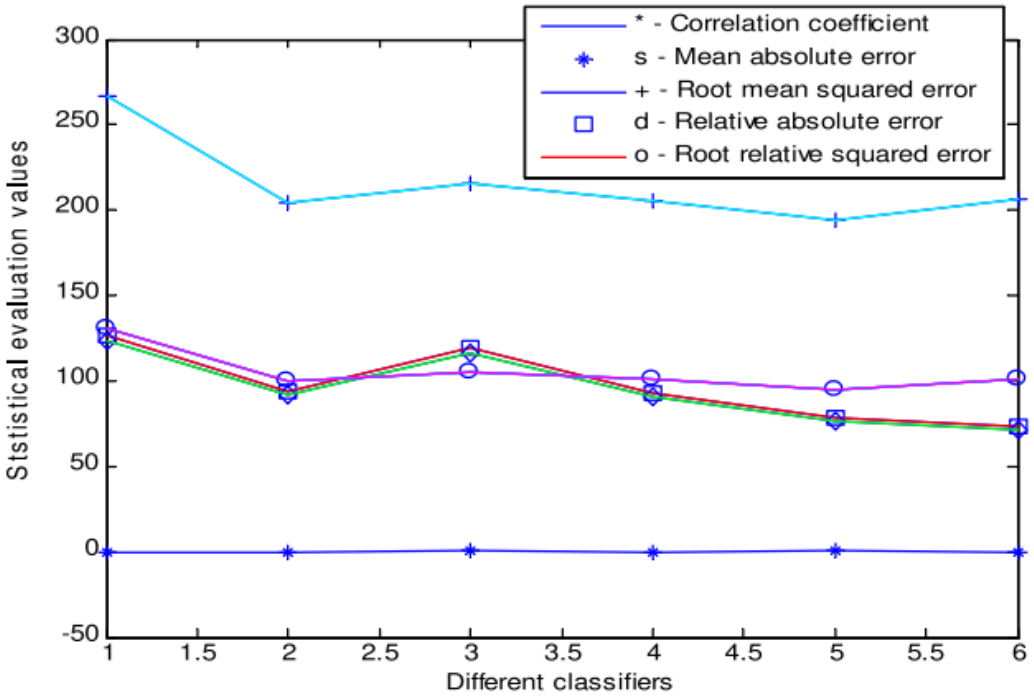
$$F_9 = -1.51E^{-12}x_1 + 8.69E^{-11}x_2 - 0.04x_3 - 3.54E^{-11}x_4 + 0.06x_5 + 3.45E^{-11}x_6$$

遍历的循环过程及收敛图像如下图所示。

我们将前面第二问预测得到的 EERIE 这个单词的各个指标带入求解，得到 F6 最大，表明其难度为中等偏上水平，这是通过判别函数的分类方法将定性的难度进行了量化处理，效果很好。

问题四的求解

特征选择是数据分析多样性领域中数据挖掘应用的关键部分。大多数特征选择方法都对分类进行特征分析，这不足以通过特征选择技术进行分类。传统类中涉及到分类，特征数据冗余，不利于数据库进一步分析。采用现有类对特征分析的影响可以忽略不计。因此，它提出了通过分析特征的微小数据，即传统类中对应实例的子特征（SF）数据来获得一个新的类或子类。这些数据涉及生成新类的有限数量的重要实例。从任何数据库中找到具有子特征的此类实例是一项具有挑战性的任务。因此，本文提出了基于拉格朗日乘数的优化模型，以找到此类数据并从传统类别中分析出新类别。已经采用了几种算法来解释子特征数据。使用基于域和基于方差的子特征以及子特征的收敛等理论方法来选择具有所提出模型的有效性的子特征。此外，几个具有搜索方法和统计方法（即局部和全局方差）的分类器通过子特征数据分析分类。在不同数据集上的实验结果分析表明，所提出的模型有利于基于所选子特征数据的新类。IEEE 使用基于域和基于方差的子特征以及子特征的收敛等理论方法来选择具有所提出模型的有效性的子特征。此外，几个具有搜索方法和统计方法（即局部和全局方差）的分类器通过子特征数据分析分类。在不同数据集上的实验结果分析表明，所提出的模型有利于基于所选子特征数据的新类。



模型的评价

参考文献

附录

```
val = [-5,1,8,10,15,30];
TrueResult = [2,1,1,2,1,1,2,2];
%生成扰动数据
for i=1:length(val)
    percent = val(i);
    dir = "rand" + percent;
    mkdir(dir);

    fname = "待预测数据.xlsx";
    %备份数据
    copyfile(fname,dir);
    fname = dir + "/" + fname;
    A = xlsread(fname);
    [n,m] = size(A);
    R = rand(n,m-2).*0.01*percent + 1-percent*0.005;%生成扰动矩阵
    A(:,1:m-2) = A(:,1:m-2).*R;%将扰动作用到原始数据上
    xlswrite(fname,A,"C2:R" + (n+1));%写进 excel 文件
end
pause(1);
```

```
%测试扰动数据
error = 0;
total = 0;
for i=1:length(val)
    percent = val(i);
    dir = "rand" + percent;
    fname = dir + "/待预测数据.xlsx";

    Result = Test(fname);
    for j=1:length(Result)
        if Result(j)~=TrueResult(j)
            error = error + 1;
        end
        total = total +1;
    end
    CorP(i) = (1 - error/total)*100;
end
CorP
function Result = Test(fileName)
T = xlsread("所有中心点.xlsx");%"所有中心点.xlsx"
P = [[4,5],[2,3,10],[7,8,9,13],[11]];
B = zeros(4,2,13);
B(1,:,:) = T(1:1:2,2:1:14);%普通类风化
B(2,:,:) = T(3:1:4,2:1:14);%普通类无风化
B(3,:,:) = T(5:1:6,2:1:14);%困难类风化
B(4,:,:) = T(10:1:11,2:1:14);%困难类无风化
error = 0;
total = 0;
A = xlsread(fileName);%"待预测数据.xlsx"
[n,m] = size(A);
for i =1:n
    flag = A(i,16);
    p = P(flag);
    t1 = GetNorm(p,A(i,2:14),B(flag,1,:));
    t2 = GetNorm(p,A(i,2:14),B(flag,2,:));
    if t1>t2
        Result(i) = 1;
    else
        Result(i) = 2;
    end
end

end
function val = GetNorm(p,a,b)
```

```
val = 0;
for i=1:length(p)
            val = val + (a(1,p(i))-b(1,1,p(i)))*(a(1,p(i))-b(1,1,p(i)));
      end
%         for i=1:13
%               val = val + (a(1,i)-b(1,1,i))*(a(1,i)-b(1,1,i));
%         end
end
```

2 K-Means++聚类以及指标评价 Matlab 程序

K-Means++聚类以及指标评价 Matlab 程序

```
clc;clear;
A=xlsread(' K 均值++聚类');
fh = @(X,K)(kmeans(X,K));
eva = evalclusters(A,fh,"CalinskiHarabasz","KList",2:5);
clear fh
K = eva.OptimalK;
clusterIndices = eva.OptimalY;

% 显示簇计算标准值
figure
bar(eva.InspectedK,eva.CriterionValues);
xticks(eva.InspectedK);
xlabel("分类数目");
ylabel("分类评价指标-CH 值");
legend("最优分类数目是" + num2str(K));
title("簇计算标准值图");
disp("最优分类数目是" + num2str(K));
% clear eva

% 计算质心
centroids = grpstats(A,clusterIndices,"mean");

% 显示结果

% 显示二维散点图(PCA)
figure
[~,score] = pca(A);
clusterMeans = grpstats(score,clusterIndices,"mean");
h = gscatter(score(:,1),score(:,2),clusterIndices,colormap("lines"));
for i = 1:numel(h)
      h(i).DisplayName = strcat("类型",h(i).DisplayName);
end
clear h i score
```

```
hold on
h = scatter(clusterMeans(:,1),clusterMeans(:,2),50,"kx","LineWidth",2);
hold off
h.DisplayName = "聚类方法";
clear h clusterMeans
legend;
title("二维散点图(PCA)");
xlabel("第一主成分");
ylabel("第二主成分");

% 矩阵图
figure
selectedCols = sort([1,2]);
[~,ax] =
gplotmatrix(A(:,selectedCols),[],clusterIndices,colormap("lines"),[],[],[],"grpbars");
title("集群数据中列的比较");
clear K
clusterMeans = grpstats(A,clusterIndices,"mean");
hold(ax,"on");
for i = 1 : size(selectedCols,2)
    for j = 1 : size(selectedCols,2)
        if i ~= j

scatter(ax(j,i),clusterMeans(:,selectedCols(i)),clusterMeans(:,selectedCols(j)), ...
                50,"kx","LineWidth",1.5,"DisplayName","聚类方法");
            xlabel(ax(size(selectedCols,2),i),("列" + selectedCols(i)));
            ylabel(ax(i,1),("列" + selectedCols(i)));
        end
    end
end
clear ax clusterMeans i j selectedCols

% clusterIndices = eva.OptimalY;% 聚类标签
% 计算其他评价标准
eva_CHI=max(eva.CriterionValues);%CHI 指数
eva2 = evalclusters(A,clusterIndices,"DaviesBouldin");%Davies-Bouldin 准则
eva_DBI=eva2.CriterionValues;% DBI 指数
eva3 = evalclusters(A,clusterIndices,"silhouette");%轮廓准则
eva_SC=eva3.CriterionValues;% 轮廓系数
%% *优化前后聚类效果指标输出*
disp(['----------------------','优化前评价指标','-------------------------'])
disp(['K-Means++聚类 CHI 指数：',num2str(eva_CHI)])
disp(['K-Means++聚类 DBI 指数：',num2str(eva_DBI)])
disp(['K-Means++聚类轮廓系数：',num2str(eva_SC)])
```

3 时间序列

```
Y=[42.79 42.79 42.79 42.76 42.58 43.01 54.9 61.25 55.29 54.69 55.02 55.02 54.96 54.08 54.08 62.91 60.95
55.29 52.83 52.75 52.76 52.76 52.76 52.75 52.56 52.24 53.89 55.34 49.35 45.27 44.44 44.52 44.52 44.52 44.52
44.52 44.62 44.45 43.57 46.3 47.79 47.5 47.6 48.39 45.51 43.44 44.77 45.86 45.86 45.86 45.81 45.61 45.5 44.29
43.26 42.71 42.53 42.64 42.65 42.62 42.49 42.36 45.46 57.18 60.56 62.59 63.66 63.68 63.41 62.31 60.77 57.97
52.05 48.82 47.08 47.06 47.07 46.43 54.24 68.52 76.04 80.84 82.25 82.44 82.44 82.44 82.43 81.94 79.86 76.28
85.27 85.64 87.32 92.52 93.43 93.45 93.45 93.45 93.44]
%
plot(Y)
figure
autocorr(Y)
figure
parcorr(Y)
%ACF 和 PACF 图
y_h_adf = adftest(Y)
y_h_kpss = kpsstest(Y)
%平滑性检验,yd1_h_adf =1，yd1_h_kpss =0，通过检验
Yd1 = diff(Y);
% 一阶差分，结果平稳。如果依旧不平稳的话，再次求差分，直至通过检验
yd1_h_adf = adftest(Yd1)
yd1_h_kpss = kpsstest(Yd1)
Yd1=Yd1'
Y=Y'
%Yd2 转换成列向量



LOGL = zeros(4,4); % Initialize
PQ = zeros(4,4);
for p = 1:4
    for q = 1:4
        Mdl = arima(p,1,q);
        [~,~,logL] = estimate(Mdl,Yd1,'Display','off');
        LOGL(p,q) = logL;
        PQ(p,q) = p + q;
     end
end

LOGL = reshape(LOGL,16,1);
PQ = reshape(PQ,16,1);
[~,bic] = aicbic(LOGL,PQ+1,100);

a=reshape(bic,4,4)
%reshape 重构数组
```

```
a_max=max(a(:));
[x,y]=find(a==min(a(:)));

%找最佳 lags 值 x=2，y=1，即对应 ARMA（2，1）模型

Mdl = arima(x, 1, y);  %第二个变量值为 1，即一阶差分
EstMdl = estimate(Mdl,Y);
[res,~,logL] = infer(EstMdl,Y);    %res 即残差

stdr = res/sqrt(EstMdl.Variance);
figure('Name','残差检验')
subplot(2,3,1)
plot(stdr)
title('Standardized Residuals')
subplot(2,3,2)
histogram(stdr,10)
title('Standardized Residuals')
subplot(2,3,3)
autocorr(stdr)
subplot(2,3,4)
parcorr(stdr)
subplot(2,3,5)
qqplot(stdr)
%上图为残差检验的结果图。
% Standardized Residuals 是查看残差是否接近正态分布，理想的残差要接近正态分布；
% ACF 和 PACF 检验残差的自相关和偏自相关，理想的结果应该在图中不存在超出蓝线的点；
% 最后一张 QQ 图是检验残差是否接近正太分布的，理想的结果中蓝点应该靠近红线。

% Durbin-Watson 统计是计量经济学分析中最常用的自相关度量
diffRes0 = diff(res);
SSE0 = res'*res;
DW0 = (diffRes0'*diffRes0)/SSE0 % Durbin-Watson statistic，
% 该值接近 2，则可以认为序列不存在一阶相关性。

%% 5.预测
step = 20; %预测步数为 20
[forData,YMSE] = forecast(EstMdl,step,'Y0',Y);
lower = forData - 1.96*sqrt(YMSE); %95 置信区间下限
upper = forData + 1.96*sqrt(YMSE); %95 置信区间上限

figure()
plot(Y,'Color',[.7,.7,.7]);
hold on
```

```
h1 = plot(length(Y):length(Y)+step,[Y(end);lower],'r:','LineWidth',2);
plot(length(Y):length(Y)+step,[Y(end);upper],'r:','LineWidth',2)
h2 = plot(length(Y):length(Y)+step,[Y(end);forData],'k','LineWidth',2);
legend([h1 h2],'95% 置信区间','预测值',...
        'Location','NorthWest')
title('Forecast')
hold off
```

附录 C（深度学习算法）

```
import os
from matplotlib import pyplot as plt
import numpy as np
from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop



data_dir = './jena_climate_2009_2016.csv/jena_climate_2009_2016.csv' #数据存放位置



f = open(data_dir) #打开文件

data = f.read() #读取文件

f.close() #关闭



lines = data.split('\n') #按行切分

header = lines[0].split(',') #每行按，切分

lines = lines[1:] #去除第 0 行，第 0 行为标记


print(header)
print(len(lines))


#将所有的数据转为 float 型数组

float_data = np.zeros((len(lines), len(header)-1))
for i, line in enumerate(lines):
    values = [float(x) for x in line.split(',')[1:]]
    float_data[i, :] = values
```

```
'''
temp = float_data[:, 1]
plt.figure()
plt.plot(range(len(temp)), temp)
plt.legend()

plt.figure()
plt.plot(range(1440), temp[:1440])
plt.show()
'''

#数据标准化，减去平均值，除以标准值

mean = float_data[:200000].mean(axis = 0)
float_data -= mean
std = float_data[:200000].std(axis = 0)
float_data /= std

#数据生成器

def generator(data, lookback, delay, min_index, max_index, shuffle=False, batch_size=128, step=6):
    if max_index is None:
        max_index = len(data) - delay - 1
    i = min_index = lookback
    while 1:

        if shuffle: #打乱顺序

            rows = np.random.randint(min_index + lookback, max_index, size=batch_size)
        else:
            if i + batch_size >= max_index:

                i = min_index + lookback #超过记录序号时，从头开始

            rows = np.arange(i, min(i+batch_size, max_index))
            i += len(rows)
        samples = np.zeros((len(rows), lookback // step, data.shape[-1]))
        targets = np.zeros((len(rows), ))
        for j, row in enumerate(rows):
            indices = range(rows[j] - lookback, rows[j], step)
            samples[j] = data[indices]
            targets[j] = data[rows[j] + delay][1]
        yield samples, targets


lookback = 1440 #给定 10 天的观测数据

step = 6 #每 6 个采样一次，即每小时一个数据点
```

```
delay = 144 #目标是未来 24 小时之后的数据

batch_size = 128


train_gen = generator(float_data, lookback=lookback, delay=delay,
                                  min_index=0,max_index=200000, shuffle=True,
                                  step=step, batch_size = batch_size)
val_gen = generator(float_data, lookback=lookback, delay=delay,
                                min_index=200001, max_index=300000, step=step,
                                batch_size=batch_size)
test_gen = generator(float_data, lookback=lookback, delay=delay,
                                  min_index=300001, max_index=None, step=step,
                                  batch_size=batch_size)
val_steps = (300000 - 200001- lookback) // batch_size
test_steps = (len(float_data) - 300001 - lookback) // batch_size

#计算 mae

def evaluate_naive_method():
    batch_maes = []
    for step in range(val_steps):
        samples, targets = next(val_gen)
        preds = samples[:, -1, 1]
        mae = np.mean(np.abs(preds - targets))
        batch_maes.append(mae)
    print('mae=',np.mean(batch_maes))


#evaluate_naive_method()
'''

#密集连接模型 DNN

model = Sequential()
model.add(layers.Flatten(input_shape=[lookback // step, float_data.shape[-1]]))
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(1))
model.compile(optimizer=RMSprop(), loss = 'mae')
history = model.fit_generator(train_gen, steps_per_epoch=500, epochs=20,
                                        validation_data=val_gen, validation_steps=val_steps)
'''


'''

#基于 GRU 的模型

model    = Sequential()
model.add(layers.GRU(32, dropout = 0.2, recurrent_dropout = 0.2, input_shape=(None, float_data.shape[-1])))
```

```
    model.add(layers.Dense(1))

    model.compile(optimizer=RMSprop(), loss='mae')
    history = model.fit_generator(train_gen, steps_per_epoch=500, epochs=20,
                                  validation_data=val_gen, validation_steps=val_steps)
    '''
    '''

    #循环差堆叠

    model  = Sequential()
    model.add(layers.GRU(32, dropout = 0.1, recurrent_dropout = 0.5, return_sequences=True,
input_shape=(None, float_data.shape[-1])))
    model.add(layers.GRU(64, activation='relu',dropout = 0.1, recurrent_dropout = 0.5))
    model.add(layers.Dense(1))

    model.compile(optimizer=RMSprop(), loss='mae')
    history = model.fit_generator(train_gen, steps_per_epoch=500, epochs=20,
                                  validation_data=val_gen, validation_steps=val_steps)
    '''

    '''

    #使用双向 GRU

    model = Sequential()
    model.add(layers.Bidirectional(layers.GRU(32), input_shape=(None, float_data.shape[-1])))
    model.add(layers.Dense(1))
    model.compile(optimizer=RMSprop(), loss = 'mae')
    history = model.fit_generator(train_gen, steps_per_epoch=500, epochs=40,
                                  validation_data=val_gen, validation_steps=val_steps)
    '''
    '''

    #使用一维 CNN

    model = Sequential()
    model.add(layers.Conv1D(32, 5, activation='relu', input_shape=(None, float_data.shape[-1])))
    model.add(layers.MaxPooling1D(3))
    model.add(layers.Conv1D(32, 5, activation='relu'))
    model.add(layers.MaxPooling1D(3))
    model.add(layers.Conv1D(32, 5, activation='relu'))
    model.add(layers.GlobalMaxPool1D())
    model.add(layers.Dense(1))
    model.compile(optimizer=RMSprop(), loss = 'mae')
    history = model.fit_generator(train_gen, steps_per_epoch=500, epochs = 20,
                                  validation_data=val_gen, validation_steps=val_steps)
```

```
'''


'''

#一维卷积基与 GRU 融合

model    = Sequential()
model.add(layers.Conv1D(32, 5, activation='relu', input_shape=(None, float_data.shape[-1])))
model.add(layers.MaxPooling1D(3))
model.add(layers.Conv1D(32, 5, activation='relu'))
model.add(layers.GRU(32, dropout = 0.1, recurrent_dropout = 0.5))
model.add(layers.Dense(1))

model.compile(optimizer=RMSprop(), loss='mae')
history = model.fit_generator(train_gen, steps_per_epoch=500, epochs=20,
                                          validation_data=val_gen, validation_steps=val_steps)
'''
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1,len(loss) + 1)
plt.figure()
plt.plot(epochs, loss, 'bo', label = 'training loss')
plt.plot(epochs, val_loss, 'b', label = 'Validation loss')
plt.title('Training and Validation loss')
plt.legend()
plt.show()
```