深圳杯 B 题-ATM 交易状态特征分析
与异常检测

参赛队员：杜林
        张笃振
        刘翠阁
        邵雪莹

指导老师：刘保东

山东大学
摘要

在对题目及数据在进行详细分析后，设计了一套基于交易状态特征分析与异常检测的预警系统，能够实时分析监测数据，采用概率统计模型、ARIMA模型、小波分析和Isolation–Forest算法协同检测。基于完整的规则库的建立，设计出的交易状态异常检测方案，在增加准确性和及时报警的同时，能尽量减少虚警误报，为系统监测人员提供帮助。

针对问题一：运用数据科学的基本方法，先对所给数据进行探索性分析，并结合实际，发现并验证数据集中特征的联系，并在此基础上对数据进行预处理工作（数据清洗、归一化处理），提取出主要特征，以便后续模型的构建。通过作图观察推理，并进行相关性分析，发现交易量、成功率与响应时间和时间存在关联，同时三者存在一定相关性。首先将三个指标分别进行处理，计算每一时间段的方差与平均值作为特征参数。用各时间段各参数的平均值代表工作稳态参数，用标准差代表工作动态参数用于评价工作状态的稳定性。通过日总交易量图的分析，发现交易量在1~2月之间相较于其他时间有显著的波动，推测是受春节影响，因此分析时将其单独分段处理。结合每日交易量图，工作日和非工作日的日总交易量和每日交易量散点图并没有明显区别，取其平均值和方差作为特征参数，用Z-score刻画偏离程度。成功率和响应时间与交易量及时间皆有关联，对其分布进行时间段划分后，再进行概率分布和拟合检验。

针对问题二：从三个角度出发构建预警模型，检验特征对异常情况的区分度，综合评价模型的可行性和进一步提升的可能性。依靠所提取的特征参数，构建机理模型，记录所有能发现的规则作为规则库，以此描述ATM交易的正常或异常状态。在时间序列建模中，结合小波分析、卡尔曼滤波等方法，对各个时间序列进行ARIMA乘积模型拟合，再使用Box-Ljung统计量检验残差序列的自相关性，对残差序列进行白噪声检验，最后通过参数显著性检验判断ARIMA乘积模型的准确性。其中，小波分析能在很大程度上弥补统计模型的不足，对数据的实时监测能力具有较大的提升效果。对故障二三四的判断结合了Isolation–Forest算法，在概率统计模型的基础上，与算法的结果进行综合以做出报警。

针对问题三：设想所有可能获取的新特征和原有特征下数据的增加，从数据周期、机器学习算法迭代步骤、统计学原理等方面分析论证，确定可以提升的精确程度和基于新特征可能发现的新异常现象。分别设想从横向和纵向扩展数据将对对模型的准确性和灵敏性带来怎样的改进。横向表示其他特征数据，例如各分行ATM机参数配置、进程参数、CPU占用率等，通过它们可以发现新的异常状况。纵向表示对已知特征获取更多数据，从而探究其随月份变化趋势，或同时获取多个分行数据，提高对前段或后端故障判断的准确性。

通过对历史数据回测等方法，对检测方案的准确性和及时性进行检验。结果显示，本方案对规则库中已收集的异常情况都能较为及时准确的检测出，同时对未知的异常状况也能及时发现，且误报率较低。

关键词：相关性分析 方差分析 概率统计模型 时间序列ARIMA乘积模型 小波分析 Isolation–Forest算法
1 引论

1.1 问题背景与分析

本文主要基于 ATM 系统实时数据，进行状态分析及故障诊断。为了建立合理模型解决状态分析与故障原因诊断问题，在本文中就如下问题进行重点分析：

1）基于大数据分析，选择、提取 ATM 交易状态的特征参数并对 ATM 工作状态进行分类；

2）建立数学模型，对 ATM 工作时出现的异常交易状态进行检测并根据所提取的特征参数对故障进行检测产生原因进行分析，在建立模型时充分考虑模型的稳定性，尽量提高故障预报的准确性；

3）查阅资料，新增影响 ATM 工作的特征参数，应用新增数据对前两问建立的模型进行优化。

异常检测问题的解决方案多种多样，由于异常检测这类问题的本质特性，目前业界皆使用阈值作为判定标准，并通过一个衡量指标与阈值进行比较作为判断标准，阈值的设置需要权衡准确率、及时性，一般为业界人员根据经验所设定。不同的方法和模型差别在于如何刻画衡量指标，以提高准确率并降低延误时间。它们的性能在很大程度上取决于数据集和参数，比较不同数据集和参数时，各种方法与其他方法相比的优势并不大。因此，本文认为，选取最适合本问题的模型、最贴近所给数据集的参数是解决该问题的最好方法。

题目所给问题属于多元变量异常检测，针对此类问题，一般分两类方法解决：

1）对每个变量独立检测；

2）对变量的联合分布检测。

由于题目所给异常情况绝大部分对应单个变量，只有一个对应二元变量。因此，对交易量进行独立检测，对响应时间和成功率分别进行独立和联合分布检测。

1.2 国内外研究现状总述

通过文献检索，研究该问题的方法主要有如下几类：文献在《一种基于自适应监测
的云计算系统故障检测方法》[1]一文中，采用的是余弦相似度，并使用主成分分析（PCA）提取特征。本题中数据的维度很小，PCA并不适用于特征的提取。

《基于小波分析的可疑金融交易时间序列研究》[2]采用了小波分析的方法。小波分析在数学上具有严格意义上的突变点诊断能力，不依赖于经验模型。针对本问题，如果采用小波分析可以精确识别突变点，但是交易数据存在周期性，单纯从波形分析会增大伪阳性比例，需要加入已知的规则和模式，也就是和经验结合，才能提高准确性。

《基于数据挖掘的异常交易检测方法》[3]运用了K-means聚类和贝叶斯信念网络分类器，判断交易属于正常交易的概率，再提取该用户的历史操作序列，通过BLAST-SSHA算法将序列与异常序列进行比对，以此检测异常。可以借鉴这篇论文的算法和思想，特别是聚类和贝叶斯信念网络分类器，但是该文是针对单个用户异常交易的检测，而本问题的数据是按分钟统计的总体数据，其数据本质上存在差异，因此不能完全照搬其模型。

1.3 异常点定义：

通常定义异常用的是二分类：异常和正常。但大多数时候，异常的定义是相对的，某些数据对象比其他对象更异常。换句话说，稀有性越强，异常程度越强，可到底要多稀有到什么程度才符合要求呢，这个问题必须看具体应用场景。

针对本问题，本文对异常点的定义基于以下假设：

（1）ATM系统的系统设计是鲁棒的，即在绝大多数情况下能满足业务需要，异常情况的出现是稀少的。

（2）由于异常和正常数据间无明显的界限，这里认为凡是有规律或以特定频率出现的状况都为正常。（此假设基于（1），即系统设计者不会让异常状况持续发生。）

对于特征参数的一项值，如果过去没有或者只有很少的类似值，那么会把这项值标记为异常。反过来说，如果经常出现类似的值，那么检测策略就会忽略掉它。

1.4 问题的特征分析

本题所给数据为时间序列型数据，同时有三个变量：交易量、成功率、响应时间。其分布在时间上有规律性，而时间又可按分钟、小时、天、星期、月份等不同的周期划分。针对多元变量数据，首先将其分解成三个一元变量时间序列，探索不同特征在时间周期上的分布及规律，其次在划分后的时间段上探索变量之间的关系。这里初步选择的
特征参数有:

1) 描述集中程度：均值、中位数；
2) 描述离散程度：标准差；
3) 描述偏离程度：Z-分数、分位数；
4) 描述多元变量相关程度：协方差、相关系数；
5) 通过分析上述特征参数在不同时间段上对变量的描述情况，最终将提取能最大程度提供检测异常所需信息的作为特征参数。

进一步地，应用前面所提取的特征参数，刻画与正常情况的偏离程度，由系统监测人员设定阈值，从而对异常状况进行报警。异常状况的严重程度可以从两个方面进行衡量：即

1）每分钟测量值与预期值的偏差
2）出现偏差的持续时间和发生次数。

若增加可采集的数据，对于 ATM 交易系统，除附件中所给材料及通过对其处理所得数据外，还可以增加对于交易类型及交易规模的统计。

2 基本假设

1) 排除人为操作失误及 ATM 应用系统前端发生故障对题中所给数据的影响。
2) 假设题目中给出的四种可能情况为最主要影响因素。
3) 假设只考虑题目中所给数据是 2017 年，工作日和节假日按照 2017 年划分。
4) 假设各种可能的异常独立分布，并且认为异常出现的概率很小。
5) 假设所增添数据均能有效科学收集。
3 主要符号、记号及其说明

<table>
<thead>
<tr>
<th>符号</th>
<th>说明</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu$</td>
<td>均值</td>
</tr>
<tr>
<td>$Med$</td>
<td>中位数</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>中位数</td>
</tr>
<tr>
<td>$z-score$</td>
<td>$z$ 分数</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>分位数</td>
</tr>
<tr>
<td>$Cov$</td>
<td>协方差</td>
</tr>
<tr>
<td>$Corr$</td>
<td>相关系数</td>
</tr>
</tbody>
</table>

4 模型的建立与求解

4.1 问题一的模型建立与求解

将四个月的数据制图显示，先对单一变量与时间的关系进行初步分析。

4.1.1 交易量分析

从图 1 中可以看出，四个月的趋势为先上升后急剧下降，再后逐渐上升，最终趋于平稳。2、3、4 这三个月份大体相同，但是一月份的数据有很大的不同，推测大量交易处于春节前，属于年前结算阶段，而除夕过后交易量明显降低，且低于周期平均值，这
是由时间的特殊性导致，可归为节假日因素。因此，将数据集分为一月份和二月份以后两部分。

从工作日与非工作日区别来看（上图绿色为工作日，红色为非工作日），观察二月以后数据，非工作日既有高于工作日，也有和工作日持平的情况，大多数情况下略低于工作日。对于是不是工作日对交易量变化的影响，这里使用方差分析法进行检验。

![图2 星期-交易量方差分析](image)

![图3 （一天中）时间-交易量方差分析](image)

*R-squared* 在统计学里叫判定系数，或决定系数，也称拟合优度，值在 0 到 1 之间，值越大，表示这个模型拟合的越好。对比图 3 中 *R-squared* 为 0.165，而图 2 中 *R-squared* 仅为 0.001，表示星期对交易量几乎没有可解释性。

一天不同时刻交易量有明显的波动，可分成交易高峰期和交易低谷期。下面选取 3 月 8 日到 9 日的数据进行展示。
每天交易量与图中大体相似，在深夜至清晨为低谷期，6：00 开始到 9：00 为上升区间，9：00 到 18：00 为高峰期，(其中 12：00-14：00 存在高峰期内微小低谷期)，18：00 到 22：00 下区间。以上结论均可通过人类经济活动规律做出合理解释。其特有的规律性将作为判断异常的重要标准。

将数据集绘制成直方图，并拟合其概率分布函数。利用核密度估计 (Kernal - Density - Estimation) [4]，是一种用于估计概率密度函数的非参数方法，\((x_1, x_2, \ldots, x_n)\) 为独立同分布 \(F\) 的 \(n\) 个样本点，设其概率密度函数为 \(f\)，核密度估计为以下:

\[
\hat{f}_h(x) = \frac{1}{n} \sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh} \sum_{i=1}^{n} K \left( \frac{x - x_i}{h} \right) \tag{1}
\]

\(K(.)\) 为核函数（非负、积分为 1，符合概率密度性质，并且均值为 0），\(h > 0\) 为一个平滑参数，称作带宽 (bandwidth)。\(K_h(x) = x/h K(x/h)\) 为缩核函数 (scaled- kernel)。
首先以分钟为单位划分数据集，选择10:00、13:00、18:00、23:00的交易量绘制直方图，并用公式拟合其最佳概率分布（核函数），发现其大致符合正态分布，接下来将尝试以更大时间间隔划分数据集。

以小时为单位划分数据集，可看出其核函数拟合的较好，同时也接近正态分布。但是，以小时为单位精度不高，会丢失部分信息量，同时只能用在变量对于时间变化不大的情况下。
为说明这一规律的普遍性，再以介于1min和1h之间的10min为单位划分数据集，可发现它也十分接近于正态分布。

此时，可以得出结论：不同日期相同时间的交易量数据符合正态分布，并且可以根据样本估算正态函数的μ和σ。

然而，还存在另外一个问题，这个时间段的均值和方差是否应该用所有已知数据计算，若数据中存在趋势，那么用更多数据并不会带来更好的效果，因为据当前时间非常远的数据并不能很好的代表当下的趋势。因此，用HP滤波方法把交易量分解为趋势部分和周期部分，然后对趋势部分和周期部分继续分解。

**HP滤波原理** [5]：

HP滤波方法是一种时间序列在状态空间的分解方法，相当于极小化波动方差的线性滤波，常被用于分解时间序列中的趋势要素和周期波动要素。

对于时间序列 $y_i, i = 1,2,...,T$，HP滤波分解是满足下式的趋势成分 $T_y$：

$$\min_T (\sum_{i=1}^{T} (y_i - T_y)^2 + \lambda \sum_{i=2}^{T} (T_{y_{i-1}} - T_y - (T_{y_{i-2}} - T_{y_{i-1}}))^2)$$

(2)

其中 $\lambda$ 是趋势中各种波动程度权重，对于年度数据常取经验值 $\lambda = 100$。HP滤波分解后可获得周期成分 $C_y = y_i - T_y$。

图7 10:00-10:09 和 16:00-16:09 交易量分布图

图8 交易量分解为趋势部分和周期部分
可以看出趋势中基本不存在额外的信息，而原本分解出的趋势围绕一条直线上下波动，可以说明不存在明显的趋势性。

因此，在不同日期相同时间（以 1min 为间隔）上可以用三个月份的数据来计算其 $\mu$ 和 $\sigma$，并将其作为交易量的特征参数。使用公式如下:

正态分布函数 $[6]$

$$
\phi(x) = \frac{1}{\sqrt{2\pi\sigma}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \, dt
$$

(3)

$\mu$ 为均值，$\sigma$ 为标准差

正态分布检验 $[6]$

$\chi^2$ 检验:

第一步：确定原假设与备择假设。原假设为 $H_0$: $F(x) = F_0(x)$，备择假设为 $H_0': F(x) \neq F_0(x)$。其中 $F_0(x)$ 为已知函数，在此题中为正态分布函数。

第二步：选取检验统计量: $\chi^2 = \sum_{i=1}^{k} \frac{(N_i - np_i)^2}{np_i}$。当 $H_0$ 为真时，检验统计量的极限分布是自由度为 $k-1$ 的 $\chi^2$ 分布。

第三步：确定拒绝域。给定显著性水平 $\alpha$，使 $P(\chi^2 \geq \chi^2_{a}(k-1)) = \alpha$，则拒绝域为 $[\chi^2_{a}(k-1), +\infty)$
第四步：计算检验统计量的观测值并根据上述步骤做出判断。

Z-Score\(^7\):

标准分数（Z Score）也叫做 Z 分数，Z 值。作为统计测量，它考察的是一个分数与一组分数的平均数的关系，这里把 Z 分数也作为交易量的特征参数。它可能是正值，也可能是负值。计算公式如下：

\[
Z = \frac{x - \mu}{\sigma}
\]  \hspace{1cm} (4)

\(x\) 为标准化的随机变量，\(\mu\) 为样本均值，\(\sigma\) 为样本标准差

图 11 3 月 1 日交易量每分钟 Z 分数

4.1.2 成功率分析

图 12 为一到四月成功率散点图，可明显看出成功率大致分布范围在 0.8 到 1.0 之间，三月下旬和四月下旬明显出现异常情况。
在对每天的成功率进行细致观察后，发现其呈现出以天为周期的变化趋势。在 8:00-21:00 较为平稳，其均值和方差近似成沿直线分布。相比之下，深夜至清晨成功率分布大不相同，可归为两类：即成功率等于 1，或不等于 1。且后者离散程度较高，并呈现拖尾的形状。同时通过折线图的绘制，发现成功率为 1 的情况并不是连续发生的，因此成功率将按一定比率随机出现在两类情况中。

图 14 中可看出，当交易量较少时，成功率 1 的可能性较大，当交易量较大时，成功率几乎不会为 1。

根据以上的结论分析，成功率离散程度高是因为交易量较低而导致的，推测原因是银行在交易量较少时减少服务器或进程数量以减少开支，与 ATM 机发生的异常故障没有直接的关系。
图 15 深夜至清晨成功率分布（右图排除为 1 情况，并进行分布函数拟合）

在去掉成功率为 1 的数据之后，成功率在深夜至清晨的分布情况如图 15 所示。因此，在之后探测异常情况时，要考虑每天在两种情况中的分布概率统计，将其区分开来计算，而不能放在一个集合中进行统计。

图 16 白天成功率分布及 $\alpha = 0.05$ 线

白天成功率较为稳定，其均值和方差均保持在恒定范围内，因此可假设服从同一概率分布。从图中可以看出，在 0.925 至 1.0 间近似服从正态分布，从 0.925 到 0 呈现指数型递减。可以用分位数 $\alpha$ 及其概率衡量异常情况的可能性大小，并将分位数作为成功率的特征参数。

分位数[8]:

设连续随机变量 $x$ 的分布函数为 $F(x)$，密度函数为 $p(x)$。那么，对任意 $0 < p < 1$ 的 $p$，称 $F(x) = p$ 的 $x$ 为此分布的分位数，或者下侧分位数。简单的说，分位数指的就是连续分布函数中的一个点，这个点对应概率 $p$。

这里取 $x = p(x < \alpha)$，$\alpha = 0.05$ 时，直线 $x = 0.9234$。可以理解为，发生产率小于 0.9234 的情况的概率只有 5%。

最后，图 17 为计算后的一天中成功率均值和方差。可看出与之前观察得出的结论
图 17 每分钟成功率的均值和标准差

4.1.3 响应时间分析

图 18 一到四月响应时间

图 19 02-16 至 02-17 响应时间
观察四个月的响应时间，同样发现存在以天为周期的变化趋势，同时还观察到每一天的数据，发现其规律与成功率近似，即夜晚至清晨响应时间较高，其出现极高值的概率也较大，而白天一般维持在较低水平，基本恒定在80ms左右，同时方差也很小。因此，与成功率类似同样把分位数作为响应时间的特征参数。在早晚和傍晚分别下降到低值区间和上升到高值区间。相比于交易量和成功率，响应时间的分布特征较为简单，同时其偏离正常区间时也较为明显。

图 20 二月和三月份响应时间与交易量散点图

由图 20 可看出当交易量增大时，响应时间会下降，据此推断，由于白天交易量大，银行开设的服务器及处理器较多，因此导致了响应时间的下降。所以，不会存在因交易量过大而出现响应时间离奇偏高的情况。

图 21 每分钟响应时间均值及标准差

从图 21 中，发现偏离正常区间的点出现的次数较多，可归结为响应时间数据中离群值很多，会显著拉大同一分钟里的平均值和标准差。因此，数据的预处理非常重要，若
处理恰当，将很大程度降低数据中的噪声，从而获取更多有用信息。

中位数与平均数相比，更不易受极端离群点的影响，但是精确度不如平均数。由于这里只需确定响应时间的大致范围，因此尝试使用滑动中位数方法。

滑动中位数原理

中值滤波是一种非线性滤波器，其原理是用一个奇数点的滑动窗口，将窗口内各点的中值作为基准，当前值和中值比较差别较大的认为是噪声进行滤除，否则予以保留。如果取窗口点数为一奇数，它的中值滤波就是从输入数据中抽取个数，并将这个点的值按其数值大小进行排序，取其序号为中心点的那个数作为滤波输出，中值运算可表示:

\[ X_m = \text{Med} \left\{ x_{i-v} \ldots x_i \ldots x_{i+v} \right\}, v = \frac{m-1}{2} \]  (5)

式（5）中：\( x_i \) 为数据序列，\( X_m \) 为中值，\( A \) 为中值滤波窗口，\( v \) 为滤波半径，\( \text{Med} \) 为取中值运算。中值滤波算法可表示:

\[
x_i = \begin{cases} 
  x_i & \text{if } x_i \leq x_m \\
  x_m & \text{else}
\end{cases}
\]  (6)

从图 22 中看出当窗口大小为 5 或 10 时，还是易受离群点的影响，而当窗口大
30 时，有偏离原数据的趋势，当窗口为 20 时，能较好的反应数据的集中趋势。

另外，发现在 3 月 19 到 3 月 22 日，和 4 月 16 到 4 月 19 日，整个时间段的响应时间都偏离正常区间，且与周围数据无连续性，呈断崖式凸起。由于发生次数稀少，且并不存在可解释的因素，因此再考虑正常响应时间范围时，将这两段时间的数据排除在外。

图 23 三四月份响应时间异常段

先前已经探究过成功率和响应时间与交易量的关系，再对三个变量做相关性矩阵后，发现成功率与响应时间可能存在关系，成功率较高时，响应时间较少，这符合前面对数据集的探索结果和实际情况。

表 1: Pearson 相关性分析

<table>
<thead>
<tr>
<th>相关性</th>
<th>业务量</th>
<th>成功率</th>
<th>响应时间</th>
</tr>
</thead>
<tbody>
<tr>
<td>业务量</td>
<td>1.0</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>成功率</td>
<td>-0.08</td>
<td>1</td>
<td>-0.38</td>
</tr>
<tr>
<td>响应时间</td>
<td>-0.03</td>
<td>-0.38</td>
<td>1</td>
</tr>
</tbody>
</table>

下面做出成功率和响应时间的联合分布图，观测两个变量的大致分布。
4.1.4 特征参数的选择

经过上述对变量的分析，结合数据集本身的属性，选取以下特征参数，作为异常检测的依据。

<table>
<thead>
<tr>
<th>特征参数</th>
<th>( \mu )</th>
<th>( \sigma )</th>
<th>( z - score )</th>
<th>( q )</th>
<th>( t )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>平均值</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>标准差</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z-分数</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>分位数</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>异常持续时间</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>持续时间内发生次数</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

数据集原有的三个变量能分别从不同维度反应异常情况，并且三者之间不存在重叠。

因此，以上特征参数分别在选取的时间段上应用于每个变量，如交易量每分钟的 Z-分数，成功率和响应时间每小时的分位数。

当检测到的变量在分位数 \( q \) 以外或超过某一 \( z \- score \) 时，则有可能发生故障，并根据贝叶斯推理，当连续多次发生时，即 \( t \) 或 \( c \) 都较大时，可以使异常的判定更加明确。

4.2 问题二的模型建立与求解

4.2.1 工作流程分析：

银行的 ATM 应用系统包括前端和后端两个部分。银行总行数据中心监控系统通过汇总统计每家分行的业务量、交易成功率、交易响应时间，来做出数据分析，从而捕捉整
个前端和后端整体应用系统运行情况以及时发现异常或故障。可以绘制出 ATM 应用系统图如图 25 所示，并标注出各项障碍对应位置。

![图 25 ATM 应用系统图](image)

对于题目中对四种故障造成的相应指标变化可以归纳如表 3 所示:

<table>
<thead>
<tr>
<th>故障及相应指标变化</th>
<th>指标变化</th>
</tr>
</thead>
<tbody>
<tr>
<td>故障一业务量陡降</td>
<td></td>
</tr>
<tr>
<td>故障二成功率降低</td>
<td></td>
</tr>
<tr>
<td>故障三响应时间陡升</td>
<td></td>
</tr>
<tr>
<td>故障四成功率下降、响应时间增加</td>
<td></td>
</tr>
</tbody>
</table>

4.2.2 交易状态异常检测方案的设计

异常情况出现的情况是稀少的，特别是在商业应用系统的设计里，因为重大错误引起的代价巨大，因此设计人员在进行系统设计时，会将异常情况发生的可能性降到最低，以保证系统平稳运行。另一方面，异常与正常状况并无明确的界限，它们之间的区别可能是模糊的，因此对大多数情况很难明确区分它是正常还是异常。

针对异常现象的本质，采用偏离程度与阈值比较法，即采用之前选取的特征参数，刻画与预期正常状况的偏离程度，具体区分异常状况时，由系统监测人员设定阈值进行报警。
图 26 按星期划分箱线图

箱形图中提供了一种只用 5 个点对数据分析简单总结的方式。这 5 个点包括下界，
Q₁（下四分位数）、Q₂（中点）、Q₃（上四分位数）和上界[5]。

在 Q₁−1.5(Q₃−Q₁) 与 Q₁+1.5(Q₃−Q₁)Q 处画两条与中位线一样的线段，这两条线段
为异常值截断点，称其为内限；在 Q₁−3(Q₃−Q₁) 和 Q₃+3(Q₃−Q₁) 处画两条线段，称其
为外限。处于内限以外位置的点表示的数据都是异常值，其中在内限与外限之间的异常
值为温和的异常值（mild outliers），在外限以外的为极端的异常值（extreme outliers）。括号内的 Q₁−Q₁ 表示四分位距。

上图使用传统的离群点标准对数据集中数据进行衡量，其中异常点以空心圆圈表
示。可以看出，在成功率和响应时间两幅图中，存在大量空心圆圈，然而因其在一星期
中每天都有分布，且较为密集，可以认定大部分是伪异常点。由于该 ATM 系统中异常
情况极少，且数据量巨大，因此传统方法在此应用的价值不大。

4.2.3 交易量异常状况检测

根据历史同期值预期同一时刻的变量的大致区间。当显著偏离该区间时，即是可能
存在的异常情况。该方法在对时间上有记忆性，但对一天之内的变化灵敏度不高。

4.2.3.1 基于统计的交易量异常检测

数据预处理：

在实际计算中，滑动窗口比按固定区间划分有着更好的效果，且通过比较各种数据
平滑方法，滑动中位数和卡尔曼滤波都能较好的降噪并对数据进行平滑处理。

卡尔曼滤波原理

卡尔曼滤波是以最小均方误差为估计的最佳准则，寻求一套递推估计的算法，其基本思想是采用信号与噪声的状态空间模型，利用前一时刻地估计值和现时刻的观测值来更新对状态变量的估计，得到现时刻的估计值。

一个线性动态系统可用线性差分方程组描述：

$$
X_k = \phi_{k,k-1} X_{k-1} + W_{k-1}, \quad k = 1, 2, \ldots 
$$

$$
Y_k = H_k X_k + V_k
$$

式（7）中：

- $X_k$ 为 $n$ 维向量，表示系统第 $k$ 时刻状态；
- $Y_k$ 为 $m$ 维系统观测向量；
- $\phi_k$，$k-1$ 是一个 $n \times n$ 阶矩阵，它反应了系统从 $k-1$ 时刻的状态到第 $k$ 时刻状态的变换；
- $W_k$ 表示第 $k$ 时刻作用于系统的随机干扰，称为模型噪声，一般假设为高斯白噪声序列，具有已知的零均值和协方差矩阵 $Q_k$；
- $H_k$ 为 $m \times n$ 阶观测矩阵，表示了状态向量 $X_k$ 到观测向量 $Y_k$ 的转换；
- $V_k$ 为 $m$ 维观测噪声，同样假设其为高斯白噪声序列，具有已知的零均值和协方差矩阵 $R_k$。

经推导（过程略），可得如下卡尔曼滤波递推公式：

$$
G_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1} 
$$

$$
\tilde{X}_k = F_{k,k-1} \tilde{X}_{k-1} + G_k [Y_k - H_k F_{k,k-1} \tilde{X}_{k-1}] 
$$

$$
C_k = (I - G_k H_k) P_k 
$$

$$
P_{k+1} = F_{k+1} C_k F_{k+1}^T + Q_k 
$$

式（8）中：

- $Q_k$ 为 $n \times n$ 阶模型噪声 $W_k$ 的协方差阵；
- $R_k$ 为 $m \times m$ 阶观测噪声 $V_k$ 的协方差阵；
- $G_k$ 为 $n \times m$ 阶增益矩阵；
- $\tilde{X}_k$ 为 $n$ 维向量，表示第 $k$ 时刻经滤波后的估计值；
- $C_k$ 为 $k$ 时刻 $n \times n$ 阶的估计协方差矩阵；
- $P_{k+1}$ 是 $k+1$ 时刻 $n \times n$ 阶估计协方差矩阵。

由可从给定的初值 $\tilde{X}_k = E\{X_0\}$，$P_0$ 出发，利用已知的矩阵 $Q_k, R_k, H_k, \phi_k, k-1$ 以及 $k$ 时刻的观测值 $Y_k$，递推计算出每个时刻的状态估计值 $\tilde{X}_k (k = 1, 2, \ldots)$。
一般情况下，对于一个线性定常系统，$\phi_k$, $k - 1 = \phi$, $H_k = H_k$，即转移矩阵均为常阵；而如果模型噪声 $W_k$ 和观测噪声 $V_k$ 都是平稳随机序列，则 $Q_k$ 和 $R_k$ 也都是常阵。此时，常增益的离散卡尔曼滤波是渐近稳定的。

图27 滑动中位数进行平滑处理
当选取窗口大小为 10 时，能较好的反映数据的集中程度，当窗口大小为 30 时，在一些部分开始偏离数据的中心，而当窗口大小为 60 时，已严重偏离真实数据。

图28 卡尔曼滤波器平滑数据

一天有 1440 分钟，基于历史交易量数据（排除一月份数据），每分钟有大约 80 次记录，这些数据刻画每分钟交易量的均值和方差。当有新数据到来时，以 $z$ - 分数刻画其与均值的偏离程度，当偏离程度超过设定阈值时，系统记录下此时间点，并统计此后 $t$ 分钟内超过阈值的次数，若次数达到设定阈值，系统发出异常报警以提示监控人员。

因交易量陡降无定量刻画，这里选取不同偏离程度和持续时间阈值进行比较。选取 3 月份的数据作为回测数据。
这里选取的 $z$-score 为 2，$t$ 和 $c$ 皆为 10，即连续十分钟未达到历史同期 95% 以上的水平。发现在 3 月 12 日和 3 月 22 日的下午高峰期，交易量均未达到预期水平，与周围日期交易量相比也可明显看出，这两日的交易量未达到高点便开始下滑。
图 30 不同阈值检测结果对比

表 4 上图所选取阈值

<table>
<thead>
<tr>
<th>颜色</th>
<th>Z-分数</th>
<th>持续时间</th>
<th>发生次数</th>
</tr>
</thead>
<tbody>
<tr>
<td>红色</td>
<td>2</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>深红色</td>
<td>2</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

模型检验：

选取 4 月份 3 天的交易量数据做测试，绘制散点图，计算每分钟对应的均值和标准差，并将 $\mu \pm 3\sigma$ 的范围画入图中。

图 31 异常检测模型测试
显然，基于统计的异常检测模型，只能反映出交易量应该有的大致走向，在对异常数据进行检测时不够灵敏，如图32的情况则检测不出来。

![Volume 30](image)

**图32 设想交易量陡降情况**

**方案改进：**

接下来，对基于统计的交易量异常检测进行优化和改进，引入小波分析法，将原始交易量数据用小波函数进行去燥，得到高频和低频信号，对高频和低频数据进行重构，得到重构数据序列，再将原始交易量数据和重构数据做差形成残差数据序列，该残差序列和原始交易量数据有相同的特点，即不同日期相同时间的数据大致符合正态分布，因此，用残差数据替换交易量数据，进行异常检测。

对替换后的残差数据做与之前交易量数据的相同的处理，即对2-4月不同日期相同时间的残差数据分别求均值μ和标准差σ。在对新数据进行检测时，首先对新数据利用小波分析得到它的残差序列，对应于不同日期相同时间的残差数据，找到它所对应的均值μ和标准差σ。若该残差数据在μ±3σ的范围内，则认为该残差数据是正常的，其所对应的原始交易量数据同样是正常的。

小波分析检测奇异信号：

假设原始数据序列信号的频率空间为s，经过m层分解可得子空间（d₁，d₂，...,dₘ，aₘ），小波分析的子空间具有如下特性:

\[ S=d₁ ⊕ a₁ = d₁ ⊕ d₂ ⊕ a₂ = ⋯ = d₁ ⊕ d₂ ⊕ ⋯ ⊕ dₘ ⊕ aₘ \]  \hspace{1cm} (9)

式中：dᵢ i = (1,2,...,m)为信号分解的高频部分；aₘ为信号分解的低频部分。
在数据序列中，受各种人为因素和机器因素影响的异常数据一般存在高频数据中。因此对分解后得到的高频部分进行处理，然后将处理后的高频部分和低频部分重构，生成重构数据序列。原始数据序列和重构数据序列相减可得到一个残差数据序列，对残差数据序列进行异常状况分析，若某个残差数据异常，则认为其对应数据为异常数据。

残差序列特点：
残差数据序列分布并非杂乱无章，它有两个特点：
1）同一日期不同时间的残差数据，在残差的平均处具有最大的频率值，处于两侧的数据频率依次减小，大致符合正态分布，因此可根据正态分布的拉伊达准则来查找异常数据。
2）不同日期的相同时间的残差数据，在残差的平均值处也具有最大频率值，处于两侧的数据频率依次减小，也可根据正态分布的拉伊达准则来查找异常数据。

根据残差序列的特点可以得到相应的两种异常检测方案。
1）第一种方案对数据进行异常检测时不依赖于先前数据，对同一日期不同时间的数据利用小波分析得到其残差序列，对其求均值$\mu$和标准差$\sigma$，在$\mu \pm 3\sigma$范围外的残差数据认为是异常数据，其所对应的原始数据也为异常数据。这种方案适用于原始数据波动范围较小，分段较明显的情况下。
2）第二种方案对数据进行异常检测时依赖于先前数据，对不同日期相同时间的残差数据分别求均值$\mu$和标准差$\sigma$。在对新数据进行检测时，首先对新数据利用小波分析得到它的残差序列，对属于不同日期相同时间的残差数据，找到它所对应的均值$\mu$和标准差$\sigma$，若该残差数据在$\mu \pm 3\sigma$的范围内，则认为该残差数据是正常的，其所对应的原始交易量数据同样是正常的。

显然，对交易量的异常检测应该选用第二种方案。原因在于交易量数据的波动范围很大，且对一天中的交易量数据分段较为复杂。

拉伊达准则$^{[10]}$；

设数据的残差数据序列为$\{\Delta q(x)\}(x = 1, 2, ..., M)$，其平均值为：

$$
\Delta q = \frac{1}{M} \sum_{x=1}^{M} \Delta q(x)
$$

按下式计算各残差数据与$\Delta q$的差值：
按贝塞尔公式计算标准差:

$$\sigma = \sqrt{\frac{1}{M} \sum_{x=1}^{M} [\delta q(x)]^2}$$

(12)

按照拉伊达准则，随机误差超出 ±3σ 的概率只有 0.0026，概率很小，一旦出现，则认为是异常数据。因此，若某个残差数据与其平均值的差值满足式（5），则认为 δq(x) 所对应的数据 q(x) 为异常数据，应当予以剔除。

$$|\delta q(x)| = |\Delta q(x) - \Delta q| > 3\sigma$$

(13)

至此，确定了基于统计的交易量异常检测的改进方案，即先用小波函数对原始的交易量数据进行处理，得到高频和低频信号，将高频分量置 0 再与低频信号重构，原始数据和重构数据做差得到残差数据，接着采用方案二。

由于 Daubechies 小波函数具有良好的光滑性和分析特性，因此选用 db4 小波对原始交易量数据进行 3 层去噪处理。

在图 33 中，运用改进后的方案对 4 月 15 到 17 日交易量数据的残差数据进行了检测，有了明显的改善，能检测出一些偏离程度较大的残差数据，其对应原始交易量数据中的陡降数据。

图 33 交易量残差的检测
4.2.3.2 基于 ARIMA 模型的交易量异常检测

一、时间序列分析及模型识别

在统计学研究中，常用按时间顺序排列的一组随机变量 $X_1, X_2, X_3, \ldots, X_t$ 来表示一个随机事件的时间序列，而时间序列分析则认为当前时刻的观测值与之前的观测值是有关联的，同时也可能与当前时刻出现某些新情况有关。

1 时间序列模型[11]

1）拟合平稳序列的模型

拟合平稳序列时常用的模型有：自回归（Auto Regression）模型，$p$ 阶自回归模型简记为 $AR(p)$；移动平均（Moving Average）模型，$q$ 阶移动平均模型简记为 $MA(q)$；自回归移动平均（auto regression moving average）模型，常记为 $ARMA(p,q)$ 模型。

2）拟合非平稳序列的模型

拟合非平稳序列时常用的模型有和自回归移动平均（autoregressive integrated moving average）模型，乘积季节模型，残差自回归（auto-regressive）模型。

2 模型拟合

对于一个时间序列，首先要通过图形检验或单位根检验来判断其平稳性。若为平稳序列，需要根据序列的自相关图以及偏自相关图对模型进行识别，判别模型的类别，并用 $AIC$ 准则或 $SBC$ 准则对模型进行优化，确定最终的 $p,q$ 值。若为非平稳时间序列，首先需要进行合适的差分运算，再对差分后的平稳序列确定合适的模型并为模型定阶，同时模型参数的估计也是关键。

二、ARIMA 模型建立

运用 python 程序检验发现所给数据时间并不连贯，将时间补全后再运用 spss 软件进行交易量变量中的缺失值替换，本文中采用临近的 5 个非缺失值的平均值替换相应的缺失值，再进行时间变量的定义。在以每分钟为观测值进行 ARIMA 模型拟合的过程中，发现该序列周期太大，达到 1440，而且 $p,q$ 值也很难判断。所以转而计算对每小时中交易量的平均值，选取每小时的交易量平均值来进行分析拟合。
(一) 对一至四月所给数据的每小时交易量平均值做时间序列分析并建立模型

1 绘制交易量的序列时序图:

![图34 一至四月交易量的时序图](image)

纵轴为每个小时内交易量的均值，横轴为时间。

2 判断序列的平稳性

对序列的平稳性检验有两种方法，一种是根据时序图和自相关图显示的特征做出检验的图检验方法；另一种是构造检验统计量进行假设检验的方法。

图检验方法操作简便、运用广泛，对于有明显趋势性或者周期性的序列，可以直接使用图检验。而对于不太明显的序列，需要采用更为精确的平稳性统计检验方法。本题中采用图检验法。

a) 通过时序图可以看出最初(1月份)数据与其他的具有较为明显的差异，而且一月之后的数据则有明显的周期性，故而可以判断该序列为非平稳序列。

b) 为更加准确的判断其平稳性，进一步采取自相关图检验

自相关系数^{[1]}：时间序列 \{x_t\}的自相关程度由自相关系数衡量。利用延迟 k 自协方差函数:

\[ \gamma(k) = \gamma(t, t + k), \quad (14) \]

可以得到基于全体观察样本计算出来的延迟 k 自协方差函数的估计值:

\[ \hat{\gamma}_k = \frac{1}{n-k} \sum_{t=1}^{n-k} (x_t - \bar{x})(x_{t-k} - \bar{x}) , \quad \forall 0 < k < n. \quad (15) \]
从而可以得到当延迟阶数 \( k \) 远远小于样本容量 \( n \) 时，延迟 \( k \) 自相关系数的估计值为

\[
\hat{\rho}_k = \frac{\sum_{i=1}^{n-k} (x_i - \overline{x})(x_{i+k} - \overline{x})}{\sum_{i=1}^{n} (x_i - \overline{x})^2}, \forall 0 < k < n. \tag{16}
\]

做出自相关图如图：

![自相关图](image)

**图 35 一至四月交易量的自相关图**

通过自相关图 35 可以看出，自相关系数衰减向零的速度很慢，而且具有正弦波动规律，所以可以进一步确认该序列为非平稳序列。同样可以进一步利用单位根检验证明其非平稳。由于白噪声序列一定是平稳序列，所以在此无需进行纯随机性检验即可断定其为非白噪声序列。

3 利用频谱分析确认周期

由时序图 34 以及自相关图 35 可以看出原序列蕴含明显的周期性，具有稳定的季节波动。为进一步确定周期，采用频谱分析。
由图 36 可以看出，在小于 0.05 的最低频率处有最高的峰值，因此可以怀疑数据中包含一个日度的周期成分，日度成分的贡献组成了周期图。在此题的时间序列中每个数据点表示一个小时，因此一个日度周期对应当前数据周期的 24。又因为周期与频率互为倒数，周期 24 对应频率为 1/24（0.0417），所以日度成分暗示在周期图中 0.0417 处的一个峰值，正好与图像峰值出现处一致，所以确认该序列周期为 24。

4 对原序列进行周期差分运算

K 步差分：相距 k 期的两个序列值之间的减法运算。记 $\nabla_k x_t$ 为 $x_t$ 的 k 步差分

$$\nabla_k x_t = x_t - x_{t-k}$$

对该序列进行以天为周期的差分（即进行 24 步差分），提取季节波动信息。周期差分后序列 $\{\nabla_{24} x_t\}$ 的时序图如图：

图 36 一至四月交易量的频谱分析周期图
对一至四月交易量进行 24 步差分后的时序图

时序图 37 显示，周期差分后序列呈现随机波动特征。为进一步确定平稳性，考察差分后序列的自相关图，如图 38 所示：

图 38 24 步差分后序列的自相关图

自相关图显示周期差分后的序列自相关系数仍旧衰减缓慢，且有一定的波动特性，所以周期差分后的序列仍然不是平稳序列。所以，进一步尝试对周期差分后的序列提取趋势信息。

对 24 步差分后的序列再进行一阶差分

24 步差分后的序列自相关图与偏自相关图均显示出拖尾的性质，且自相关图显示自相关系数长期位于零轴的一边，这是具有单调趋势序列的典型特征。尝试对该序列再进行一阶差分以提取其趋势性，差分后序列图如图 39：
再做出其自相关图，图 40

自相关图显示延迟 24 阶自相关系数显著大于二倍标准差，这说明差分后序列中仍蕴含非常显著的季节效应。延迟 1 阶、2 阶的自相关系数也显著大于二倍标准差，这说明差分后序列还具有短期相关性，需要结合偏自相关图确定其短期相关模型。
偏自相关系数：滞后 k 自相关系数得到的并不是 $x_t$ 与 $x_{t-k}$ 之间单纯的相关关系，
其受到中间 $k-1$ 个变量的影响。所谓滞后 k 偏自相关系数，是在给定中间 $k-1$ 个随机变量的条件下，即剔除中间 $k-1$ 个随机变量的干扰之后，$x_{t-k}$ 对 $x_t$ 相关影响的度量。

偏自相关系数：
\[
\phi_k = \frac{E[(x_t - \hat{E}_t)(x_{t-k} - \hat{E}_{t-k})]}{E[(x_{t-k} - \hat{E}_{t-k})^2]}
\] (18)

根据偏自相关系数做出偏自相关图如图 41。

模型类别判别方法如下：

<table>
<thead>
<tr>
<th>模型</th>
<th>AR (p)</th>
<th>MA(q)</th>
<th>ARMA(p,q)</th>
<th>ARIMA(p,d,q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>自相关系数（ACF）</td>
<td>拖尾</td>
<td>截尾</td>
<td>拖尾</td>
<td>拖尾</td>
</tr>
<tr>
<td>偏自相关系数（PACF）</td>
<td>截尾</td>
<td>拖尾</td>
<td>拖尾</td>
<td>拖尾</td>
</tr>
</tbody>
</table>

由表可知自相关系数与偏自相关系数均拖尾，而且序列为平稳序列，尝试对其拟合 $ARMA(p,q)$ 模型。

$ARMA(p,q)$ 模型简介：

对于某个被观察的时间序列 $\{x_t\}$ 而言，若 $t$ 时刻值 $x_t$ 不仅与之前时间段的观测值 $x_{t-1}, x_{t-2}, \ldots, x_{t-p}$ 有关，而与以前时刻 $x_{t-1}, x_{t-2}, \ldots, x_{t-q}$ 进入序列的随机扰动项 $\varepsilon, \varepsilon_{t-1}, \varepsilon_{t-2}, \ldots, \varepsilon_{t-q}$ 有关，则该序列被称为自回归移动平均序列，其满足的模型为 $ARMA(p,q)$ 模型。

\[
\begin{align*}
  &x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + \varepsilon - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} \\
  &\phi_i \neq 0, \theta_i \neq 0 \\
  &E(\varepsilon) = 0, \text{Var}(\varepsilon) = \sigma^2, E(\varepsilon_\omega) = 0, s \neq t \\
  &E(x_t \varepsilon_t) = 0, \forall s < t
\end{align*}
\] (19)
根据以上步骤选择合适的模型后，仅凭借直觉建立某一阶数的模型是不可取的，需要对不同的 p, q 进行检验，以便选出最佳模型作为最终预测模型。

一下为几种定阶与优化准则，在实际中应当综合运用：
a) 相关性判定
根据自相关图以及偏自相关图推断模型的阶数。
b) 残差自相关性检验法
选取残差方差最小的模型。
c) 信息准则
选择 p, q 的阶数标准通常还需要使用 AIC 准则与 SBC 准则，SBC 准则是对 AIC 准则的改进，其表达形式如下：
\[ AIC = -2 \ln(l) + 2k \] （模型的极大似然函数值）
\[ SBC = -2 \ln(l) + \ln(n) \ln(k) \] （模型中未知参数的个数）

在所有通过检验的模型中使得 AIC 或 SBC 函数达到最小的模型为相对最优模型，可以使用这两种信息准则进行模型优化。

通过自相关图与偏自相关图判断模型可能的 p, q 值，并计算其相应的 SBC 值与平稳的 R 方值。

<table>
<thead>
<tr>
<th>模型</th>
<th>SBC 值</th>
<th>平稳的 R 方</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMA(2,6)</td>
<td>7.340</td>
<td>0.245</td>
</tr>
<tr>
<td>ARMA(2,7)</td>
<td>7.293</td>
<td>0.283</td>
</tr>
<tr>
<td>ARMA(1,7)</td>
<td>7.331</td>
<td>0.237</td>
</tr>
</tbody>
</table>

综合考虑 SBC 值以及平稳的 R 方，p=2, q=7 最优。所以尝试使用 ARMA(2,7) 模型提取差分后序列的短期自相关信息。

7. 结合季节自相关性拟合 ARIMA乘积模型

ARIMA乘积模型简介：

当序列具有短期相关性的时候，通常使用低阶 ARMA \((p, q)\) 模型提取。当序列具有季节效应，季节效应本身还具有自相关性时，季节相关性可以使用以周期步长为单位的 ARMA \((P, Q)\) 模型提取。综合 \(d\) 阶趋势差分和 \(D\) 阶以周期 \(S\) 为步长的季节差分运算，拟合模型为 ARIMA \((p, d, q)(P, D, Q)_s\) 为：

\[
\nabla^d \nabla^D_{s} x_i = \frac{\Theta(B)\Theta_s(B)}{\Phi(B)\Phi_s(B)} \epsilon_i
\]

(20)
\[ \nabla^d \nabla^\gamma \nabla x_t = (1 - B)^d (1 - B^\gamma)^\gamma x_t \]  

\[ \begin{align*}
\Theta(B) &= 1 - \theta_1 B - \cdots - \theta_q B^q \\
\Phi(B) &= 1 - \phi_1 B - \cdots - \phi_p B^p \\
\Theta_s(B) &= 1 - \theta_1 B^s - \cdots - \theta_q B^{qs} \\
\Phi_s(B) &= 1 - \phi_1 B^s - \cdots - \phi_p B^{ps}
\end{align*} \]  

(22)

B 为延迟算子 \( B^d x_t = x_{t-d} \)

考虑序列的季节自相关特征，根据图 39 与图 40，这时要考察延迟 24 阶、48 阶等以周期长度为单位的自相关系数和偏自相关系数的特征，二者都显示出拖尾性质。所以尝试使用以 24 步为周期的 ARMA(1,1),24 模型提取差分后序列的季节自相关信息。

综合上面信息，要拟合的乘积模型为 ARIMA(2,1,7)(1,1,1),24

\[ \nabla \nabla_{24} x_t = \frac{(1 - \theta_1 B - \cdots - \theta_7 B^7)(1 - \theta B^{24})}{(1 - \phi_1 B^2 - \phi_2 B^4)(1 - \phi B^{24})} \varepsilon_t \]  

(23)

\( x_t \) 为第 t 个小时对应的预测值，B 为延迟算子，\( \nabla \) 为一阶差分，\( \nabla_{24} \) 为 24 步差分，\( \theta_1, \ldots, \theta_7, \phi_1, \phi_2 \) 为参数，\( \varepsilon_t \) 为第 t 个小时对应的随即干扰。

8 参数估计

选择好拟合模型之后，下一步就需要利用序列的观察值确定该模型的口径，即计算模型中未知参数的值。对参数的估计方法有三种：矩估计、极大似然估计、最小二乘估计。

在此使用最小二乘估计计算参数值：
对方程做一个转化，并对相同的变量进行合并同类项，并进行回归分析，得到表格：

<table>
<thead>
<tr>
<th>表 7：不同变量对应的系数及其显著性</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>係數</strong></td>
</tr>
<tr>
<td>模型</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>1 LAGS(小时交易量均值,1)</td>
</tr>
</tbody>
</table>
根据表7中给出的系数值反解出需要的参数$\theta_1, \cdots, \theta_7, \phi_1, \phi_2, \theta_1', \phi_1'$，得到

<table>
<thead>
<tr>
<th>LAGS(小时交易量均值, 1,1)</th>
<th>-0.088</th>
<th>0.009</th>
<th>-0.088</th>
<th>-9.433</th>
<th>0.000</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAGS(小时交易量均值, 2,1)</td>
<td>-0.403</td>
<td>0.006</td>
<td>-0.402</td>
<td>-62.785</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 24)</td>
<td>1.315</td>
<td>0.007</td>
<td>1.333</td>
<td>187.496</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 24,1)</td>
<td>-1.917</td>
<td>0.011</td>
<td>-1.944</td>
<td>-181.000</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 26,1)</td>
<td>0.629</td>
<td>0.006</td>
<td>0.638</td>
<td>102.644</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 48)</td>
<td>-0.319</td>
<td>0.007</td>
<td>-0.327</td>
<td>-46.379</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 48,1)</td>
<td>0.459</td>
<td>0.011</td>
<td>0.471</td>
<td>40.351</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 49,1)</td>
<td>0.080</td>
<td>0.009</td>
<td>0.082</td>
<td>8.789</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(小时交易量均值, 50,1)</td>
<td>-0.224</td>
<td>0.005</td>
<td>-0.229</td>
<td>-45.886</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResidual 小时交易量均值_模型_1,1)</td>
<td>0.084</td>
<td>0.007</td>
<td>0.004</td>
<td>11.551</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_1,1)</td>
<td>-0.584</td>
<td>0.009</td>
<td>-0.026</td>
<td>-67.904</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_2,1)</td>
<td>-0.302</td>
<td>0.006</td>
<td>-0.014</td>
<td>-51.949</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_3,1)</td>
<td>-0.089</td>
<td>0.004</td>
<td>-0.004</td>
<td>-21.549</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_4,1)</td>
<td>0.079</td>
<td>0.004</td>
<td>0.004</td>
<td>19.823</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_5,1)</td>
<td>0.072</td>
<td>0.004</td>
<td>0.003</td>
<td>18.155</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_6,1)</td>
<td>-0.014</td>
<td>0.004</td>
<td>-0.001</td>
<td>-3.617</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResidual 小时交易量均值_模型_1,24)</td>
<td>-0.772</td>
<td>0.008</td>
<td>-0.035</td>
<td>-99.385</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_24,1)</td>
<td>-0.082</td>
<td>0.006</td>
<td>-0.004</td>
<td>-13.337</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_25,1)</td>
<td>0.547</td>
<td>0.008</td>
<td>0.025</td>
<td>71.412</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_26,1)</td>
<td>0.279</td>
<td>0.005</td>
<td>0.013</td>
<td>57.567</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_27,1)</td>
<td>0.068</td>
<td>0.004</td>
<td>0.003</td>
<td>17.268</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_28,1)</td>
<td>-0.067</td>
<td>0.004</td>
<td>-0.003</td>
<td>-16.921</td>
<td>0.000</td>
</tr>
<tr>
<td>LAGS(NResid_29,1)</td>
<td>-0.065</td>
<td>0.004</td>
<td>-0.003</td>
<td>-16.679</td>
<td>0.000</td>
</tr>
</tbody>
</table>

a. 應變數: 值减残差
b. 透過原點的線性迴歸
模型检验

首先进行残差白噪声检验，其有以下几种常用方法

a) DW 检验: $DW = 2 \left[ 1 - \frac{\sum_{t=2}^{n} \varepsilon_t \varepsilon_{t-1}}{\sum_{t=1}^{n} \varepsilon_t^2} \right] \approx 2(1 - \rho)$, $0 \leq DW \leq 4$, $\rho$ 为自相关系数。根据 $\rho$ 的值与序列正负相关性的关系，可以确定两个临界值 $d_u, d_l$。当 $DW < d_l$ 时，序列显著正相关。当 $DW > 4 - d_u$ 的时候，序列显著负相关。当 $d_u \leq DW \leq 4 - d_l$ 时，序列显著不相关。

b) 可以采用 LB 统计量或 Box-Ljung 统计量或 $\chi^2$ 统计量进行检验，其 $P$ 值（显著性水平）大于 0.05 时，说明残差序列纯随机，为白噪声序列。

c) 更准确的做法是采用拉格朗日乘子检验（La-grange multiplier test），简记为 LM 检验。当随机概率大于 0.05 时，表明该残差序列为白噪声序列；若随机概率小于 0.05，则说明残差序列不是白噪声序列，模型还需要重新构建。

在此选择 Box-Ljung 统计量对残差序列进行检验

| 序列(E): 小时交易量均值-模型_1 中的噪声残值 | Box-Ljung 统计量 |
|---|---|---|---|---|
| 延迟 | 自相关 | 标准错误 | 值 | 自由度 | 显著性 |
| 1 | .002 | .022 | .012 | 1 | .912 |

模型方程

$$
\phi_1 = 0.467, \phi_2 = 0.403;
\phi_1' = 0.315;
\theta_1 = -0.084, \theta_2 = 0.584, \theta_3 = 0.302, \theta_4 = 0.089, \theta_5 = -0.079, \theta_6 = -0.072, \theta_7 = 0.014;
\theta_1' = 0.772
$$

$$
\nabla \nabla_{2 \mu}^{\nu} = \frac{(1 + 0.084B - 0.584B^2 - 0.302B^3 - 0.089B^4 + 0.079B^5 + 0.072B^6 - 0.014B^7)(1 - 0.772B^{24})}{(1 - 0.467B - 0.403B^2)(1 - 0.315B^{24})}
$$
表 8

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-0.003</td>
<td>0.022</td>
<td>0.033</td>
<td>2</td>
<td>0.984</td>
</tr>
<tr>
<td>3</td>
<td>-0.008</td>
<td>0.021</td>
<td>0.158</td>
<td>3</td>
<td>0.984</td>
</tr>
<tr>
<td>4</td>
<td>-0.005</td>
<td>0.021</td>
<td>0.204</td>
<td>4</td>
<td>0.995</td>
</tr>
<tr>
<td>5</td>
<td>0.022</td>
<td>0.021</td>
<td>1.248</td>
<td>5</td>
<td>0.940</td>
</tr>
<tr>
<td>6</td>
<td>0.028</td>
<td>0.021</td>
<td>2.888</td>
<td>6</td>
<td>0.823</td>
</tr>
<tr>
<td>7</td>
<td>0.007</td>
<td>0.021</td>
<td>2.995</td>
<td>7</td>
<td>0.885</td>
</tr>
<tr>
<td>8</td>
<td>-0.010</td>
<td>0.021</td>
<td>3.202</td>
<td>8</td>
<td>0.921</td>
</tr>
</tbody>
</table>

a. 假定的基本过程为独立性（白噪声）。
b. 基于渐近卡方近似值。

由表 8 可以得出延迟 1 至 8 阶检验统计量相应的 P 值均显著大于 0.05，所以认为该残差序列未白噪声序列。再进行参数显著性检验，由表 7 即可看出参数具有显著性。

10 拟合图形
图 42 为拟合后图形，图 43 为放大后图形，蓝色折线为真实值，绿色折线为预测值

图 42 乘积模型拟合后的图形

图 43 拟合后图形的放大图形

11 根据 95%的置信区间可以得到图 44
图 44 增加置信区间上下限所组成的图形
图 45 图 44 的放大图

蓝色为真实值，绿色为预测值，黄色为下限，紫色为上限。

又因为最初 25 个小时的值没有预测值，所以一共得到 2184-25=2159 个预测值，进一步根据置信区间得到共有 2039 个值位于置信区间范围之内。所以可得异常点共有 120 个。

（二）对二至四月的每小时交易量平均值做时间序列分析与模型建立
从第一步的时序图中不难看出，最初九天左右的观测值（即 1 月份的交易量）与其他月份的交易量差异较大。为了更进一步的看出 1 月份与其他月份的区别，将 2,3,4 月作为一个整体与 1 月份进行比较，利用 spss 中的独立样本 T 检验可得

表 9 组统计

<table>
<thead>
<tr>
<th></th>
<th>月份</th>
<th>数字</th>
<th>平均值(E)</th>
<th>标准偏差</th>
<th>标准误差平均值</th>
</tr>
</thead>
<tbody>
<tr>
<td>交易量</td>
<td>1</td>
<td>12954</td>
<td>740.52</td>
<td>815.172</td>
<td>7.162</td>
</tr>
<tr>
<td></td>
<td>234</td>
<td>118059</td>
<td>573.41</td>
<td>454.189</td>
<td>1.322</td>
</tr>
<tr>
<td>响应时间</td>
<td>1</td>
<td>12954</td>
<td>89.94</td>
<td>21.606</td>
<td>.190</td>
</tr>
<tr>
<td></td>
<td>234</td>
<td>118059</td>
<td>101.95</td>
<td>633.569</td>
<td>1.844</td>
</tr>
</tbody>
</table>

由表格可以更精确的得出，1 月份的交易量与响应时间与其他月份有很大差异。所以，在此选择二至四月份的值重新建立 ARIMA 乘积模型。
1、绘制二至四月交易量的序列时序图:
2、判断序列的平稳性

a) 通过时序图可以看出数据具有较为明显的周期性，故而可以判断该序列为非平稳序列。

b) 为更加准确的判断其平稳性，进一步采取自相关图检验，自相关图如图 46:

通过自相关图可以看出，自相关系数衰减向零的速度很慢，而且具有正弦波动规律，所以可以进一步确认该序列为非平稳序列。由于白噪声序列一定是平稳序列，所以在此无需进行纯随机性检验即可断定其为非白噪声序列。

3、利用频谱分析确认周期

由时序图以及自相关图可以看出原序列蕴含明显的周期性，具有稳定的季节波动。为进一步确定周期，采用频谱分析。
由图48可以看出，在小于0.05的最低频率处有最高的峰值，因此可以怀疑数据中包含一个日度的周期成分，日度成分的贡献组成了周期图。在此题的时间序列中每个数据点表示一个小时，因此一个日度周期对应当前数据周期的24。又因为周期与频率互为倒数，周期24对应频率为1/24（0.0417），所以日度成分暗示在周期图中0.0417处的一个峰值，正好与图像峰值出现处一致，所以确认该序列周期为24。

4. 对原序列进行周期差分运算

对该序列进行以天为周期的差分（即进行24步的周期差分），提取季节波动信息。

周期差分后序列 \{\n_{t}x_{24}\} 的时序图如图：
时序图显示，周期差分后序列呈现随机波动特征。为进一步确定平稳性，考察差分后序列的自相关图，如图 50 所示：

图 50 24 步差分后序列的自相关图

图 50 显示周期差分后的序列的自相关系数仍然衰减缓慢，在很长延迟期里一直为正，而后又一直为负，这说明序列具有单调趋势。进一步采用一阶差分提取趋势信息。

6. 对序列再进行一阶差分
   对 24 步差分后的序列再进行一阶差分以提取线性趋势。
   序列图如图 51

自相关图如图 52
图 52 24 步 1 阶差分后的自相关图

7、拟合 ARIMA 乘积模型

自相关图显示延迟 24 阶自相关系数显著大于二倍标准差，这说明差分后序列中仍蕴含非常显著的季节效应。延迟短期阶的自相关系数也显著大于二倍标准差，这说明差分后序列还具有短期相关性。

先结合偏自相关图确定其短期相关模型。偏相关图如图 53

图 53 24 步 1 阶差分后的偏自相关图

通过自相关图与偏自相关图判断模型可能的 p,q 值，并运用 SBC 准则定阶，得到表 10。
表 10 SBC 和平稳的 R 方值

<table>
<thead>
<tr>
<th>模型</th>
<th>SBC 值</th>
<th>平稳的 R 方</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMA(1,9)</td>
<td>7.105</td>
<td>0.166</td>
</tr>
<tr>
<td>ARMA(1,7)</td>
<td>7.005</td>
<td>0.167</td>
</tr>
<tr>
<td>ARMA(2,6)</td>
<td>7.007</td>
<td>0.165</td>
</tr>
</tbody>
</table>

综合考虑 SBC 值与平稳的 R 方值之后，选择 p=1, q=7 时的模型。所以尝试使用 ARMA(1,7) 模型提取差分后序列的短期自相关信息。

再考虑季节自相关特征，根据自相关图与偏自相关图，这时要考察延迟 24 阶、48 阶等以周期长度为单位的自相关系数和偏自相关系数的特征，二者都显示出拖尾性质。所以尝试使用以 24 步为周期的 ARMA(1,1)24 模型提取差分后序列的季节自相关信息。

综合上面信息，要拟合的乘积模型为 ARIMA(1,1,7)(1,1,1)24

对应的参数方程为：

$$

\nabla \nabla_{24} x_t = \frac{(1-\theta_1 B-\cdots-\theta_7 B^7)(1-\theta_1 B^{24})}{(1-\phi_1 B)(1-\phi_1 B^{24})} \varepsilon_t

$$

(19)

$x_t$ 为第 $t$ 个小时对应的预测值，$B$ 为延迟算子，$
abla$ 为一阶差分，$
abla^{24}$ 为 24 步差分，$\theta_1, \cdots, \theta_7, \phi_1$ 为参数，$\varepsilon_t$ 为第 $t$ 个小时对应的随机干扰。

8、参数估计
在此使用最小二乘估计计算参数值。
对方程做一个转化，并对相同的变量进行合并同类项，利用 spss 中的线性回归并对其进行分析。
得到表格：

表 11 不同变量对应的系数及其显著性
根据表中给出的系数值反解出需要的参数 \( \theta_i, \phi_i, \phi'_i, \phi''_i \)，得到

\[ \phi_i = 0.208; \]
\[ \phi'_i = -0.579; \]
\[ \theta_i = -0.468, \theta_2 = -0.261, \theta_3 = -0.348, \theta_4 = -0.291, \theta_5 = -0.304, \theta_6 = -0.232, \theta_7 = -0.152; \]
\[ \theta'_i = 0.806 \]

\[ \nabla \nabla_{24} r = \frac{(1 + 0.468B + 0.261B^2 + 0.348B^3 + 0.291B^4 + 0.304B^5 + 0.232B^6 + 0.152B^7)(1 - 0.806B^{24})}{(1 - 0.208B)(1 + 0.579B^{24})} \epsilon_t \]

\[ (20) \]

\[ (21) \]
9、模型检验

对残差序列进行白噪声检验

表 12 残差序列检验

自相关(A)

<table>
<thead>
<tr>
<th>延迟</th>
<th>自相关</th>
<th>标准错误</th>
<th>Box-Ljung 统计</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>值</td>
</tr>
<tr>
<td>1</td>
<td>-0.02</td>
<td>0.023</td>
<td>0.09</td>
</tr>
<tr>
<td>2</td>
<td>-0.04</td>
<td>0.023</td>
<td>0.042</td>
</tr>
<tr>
<td>3</td>
<td>-0.04</td>
<td>0.023</td>
<td>0.079</td>
</tr>
<tr>
<td>4</td>
<td>-0.04</td>
<td>0.023</td>
<td>0.107</td>
</tr>
<tr>
<td>5</td>
<td>-0.01</td>
<td>0.023</td>
<td>0.109</td>
</tr>
<tr>
<td>6</td>
<td>-0.06</td>
<td>0.023</td>
<td>0.189</td>
</tr>
<tr>
<td>7</td>
<td>0.03</td>
<td>0.023</td>
<td>0.209</td>
</tr>
<tr>
<td>8</td>
<td>-0.06</td>
<td>0.023</td>
<td>0.290</td>
</tr>
<tr>
<td>9</td>
<td>-0.046</td>
<td>0.023</td>
<td>4.436</td>
</tr>
<tr>
<td>10</td>
<td>0.056</td>
<td>0.023</td>
<td>10.667</td>
</tr>
<tr>
<td>11</td>
<td>-0.01</td>
<td>0.023</td>
<td>10.670</td>
</tr>
<tr>
<td>12</td>
<td>0.045</td>
<td>0.023</td>
<td>14.598</td>
</tr>
<tr>
<td>13</td>
<td>0.020</td>
<td>0.023</td>
<td>15.384</td>
</tr>
</tbody>
</table>

a. 假定的基本过程为独立性（白噪声）。
b. 基于渐近卡方近似值。

检验结果显示延迟 1 至 13 阶的检验统计量相应的 P 值都显著大于 0.05，即说明残差序列为白噪声序列。再进行参数显著性检验，由表 11 即可看出参数具有显著性。

10 拟合图形如图 54，放大后如图 55。

蓝色为真实值，绿色为预测值
图 54 拟合图形
由图可以直观看出对于模型的拟合程度较高。
根据 95% 的置信区间，可得到下面左图，放大一部分得到右图

图 55 拟合图形放大图

图 56 含置信区间上下限的拟合图形
图 57 图 56 放大后的图形

蓝色为真实值，绿色为预测值，黄色为置信区间下限，紫色为置信区间上限。

又因为最初 25 个小时的值没有预测值，一共得到 1968-25=1943 个预测值，进一步根据置信区间得到共有 1811 个值位于置信区间范围之内。所以可得异常点共有 132 个。
4.2.4 成功率和响应时间异常检测

4.2.4.1 基于统计的成功率和响应时间异常检测

成功率和响应时间的异常检测同样遵循本文开头提到的每分钟测量值与预期值的偏差以及出现偏差的持续时间和发生次数。当 $z$-score 越大时，其与预期值的偏差越大，出现的概率越小。尽管如此，不能仅因为出现一次概率小的时间就定义其为异常情况，但当小概率事件连续发生时，此时这个整体事件在这个时间段发生的概率将趋近于 0。对于一个事件来说，它发生的可能性越趋近于 0，越有把握认定其为异常事件。然而，在同时要求报警及时性的前提下，不能等待它无限减小，一旦整个事件的概率达到某一界限，检测系统就会发出报警信号。

由于成功率和响应时间存在极大偏离正常值，首先尝试的思路是对数据进行预处理。

图 58 数据预处理，$\alpha = 0.0001$ 和 $\alpha = 0.001$

对于极端异常值的处理方法有多种，可以将其直接剔除，或以平均值取代，或以划定极值取代。这里，由于假设 $ATM$ 统计信息是准确无误的，因此，所有数据都能体现系统运行的状况，不存在观测噪声。若直接剔除或以均值代替都将影响此时刻的方差，因此，以划定极值取代极端异常值。

<table>
<thead>
<tr>
<th>表 11 极端值替换公式</th>
</tr>
</thead>
<tbody>
<tr>
<td>成功率 $s$</td>
</tr>
<tr>
<td>响应时间 $r$</td>
</tr>
</tbody>
</table>

可以看出，当 $\alpha$ 取值越大时，所丢失的信息越多。由于题目未给出异常明确界限，这里认为通过主观方法对数据进行修改是不严谨的，对于有价值的信息都应保留。接下来尝试用多种概率分布函数拟合数据的最佳分布。选取 SSE 最小的拟合函数作为最佳拟合曲线。

SSE$^{[12]}$ （和方差）：
该统计参数计算的是拟合数据和原始数据对应点的误差的平方和，计算公式如下：

\[
SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2
\]  \hspace{1cm} (22)

\(SSE\) 越接近于 0，说明模型选择和拟合更好，数据预测也越成功。

图 59 夜间成功率多种分布函数拟合

第二种方法是直接用分位数来设定阈值，而不必关心数据具体分布。因为要报警的异常分别是成功率偏低和响应时间偏高，这两部分都位于呈现长尾分布，只需确定 \(\alpha\) 即可。同时，检测异常的持续时间和发生次数，以减少虚警误报的发生。

表 14 各时间段 \(\alpha\) 取值

<table>
<thead>
<tr>
<th>时间段</th>
<th>(\alpha)</th>
</tr>
</thead>
<tbody>
<tr>
<td>白天</td>
<td>0.01</td>
</tr>
<tr>
<td>夜间</td>
<td>0.0001</td>
</tr>
<tr>
<td>早晨、傍晚</td>
<td>0.01</td>
</tr>
</tbody>
</table>
4.2.4.2 基于小波分析的响应时间和成功率异常检测

在此之前，曾使用过小波分析对交易量的异常检测进行改进，由于交易量的数据波动范围大且分段较为复杂，因此，选择了方案二进行检测。在对响应时间和成功率进行检测时，由于二者的数据波动整体不是很大，从数据分布特征来看，深夜至凌晨阶段数据较为离散，白天则较为平稳，因此很容易对这两个量进行分段处理。响应时间和成功率的分布特点决定了在用小波分析进行异常检测时，选择方案一，即利用相同日期不同时间的残差数据符合正态分布这一特点。

1 试验分析

第一问中曾对 1-4 月份的响应时间和成功率进行整体分析，发现一月份的响应时间
整体来说低于其他月份，而成功率整体高于其他月份。因此，在对二者进行异常检测时，对一月份单独处理，2-4月份统一处理，由于2-4月份整体分布大致相同，这里选取4月份的响应时间和成功率数据进行演示。

2 异常数据判别

首先，对4月份的响应时间和成功率数据进行分段处理，把4月份分布比较离散的所有深夜至凌晨的数据统一进行检测，把分布比较平稳的白天段的数据统一进行检测。

然后，对分段后的数据利用db4小波进行3层降噪处理，得到高频以及低频数据，将高频数据置0与低频数据形成重构序列，原始数据与重构数据作差得到残差数据，进而画出残差图。

最后，对所有得到的残差数据求平均值和标准差，并将3倍标准差线加到残差图中，根据前面所提的拉伊达法则，在标准线之外的残差数据认为是异常数据，其所对应的原始数据也为异常数据。

3 模型检验

3.1 响应时间的检验

![](image1)

图 62 4月6:00-20:00响应时间原始数据和去噪后数据的残差图

![](image2)

图 63 4月20:00-次日6:00响应时间原始数据和去噪后数据的残差图
以上是对 4 月份响应时间分段后运用上述模型得到的残差图以及异常检测情况，发现这两张图中存在很大的问题。由于响应时间数据不论是在平稳段还是离散段，都存在数值极大的点，在计算残差数据标准差时会使得标准差数据过高，从而只能检测出极端异常点，而对于一般的异常点则检测不出来。因此，先对分段后的响应时间去除极端异常值，采用前面所述的极端值替换法，具体的极端值替换公式见表 13。

对除去极端异常值的分段数据运用上述模型。得到图 64。

图 64 处理后的 4 月 6：00–20：00 响应时间原始数据和去噪后数据的残差图

在去除极端值影响后，发现检测效果比较好，能够检测出大部分的异常数据。图中上方红线外的残差数据认为是异常数据，其所对应的原始响应时间数据为响应时间增加的数据，也是异常的。
3.2 成功率的检验

图 65 4 月 8:00-19:00 成功率原始数据和去噪后数据的残差图

图 66 4 月 19:00-次日 8:00 成功率原始数据和去噪后数据的残差图

以上是对 4 月份成功率分段后运用上述模型得到的残差图以及异常检测情况，和响应时间不同的是，这两张图并未明显出现残差数据标准差过大从而导致异常检测不灵敏的问题，虽然和响应时间类似，成功率在离散段和平稳段也会有数值较低的点，但由于成功率的取值在 0-1 之间，因此对残差数据标准差的影响比较小。但是，为了是模型更加准确，仍然对数值非常低的极端点进行了处理，依旧采用上述极端值替换法，具体替换公式见表 13。
图 67 处理后的 4 月 8 日 00:00-19:00 成功率原始数据和去噪后数据的残差图

虽然成功率在去除极端值影响后，检测效果变化并没有响应时间那么明显，但这仍然可以在一定程度上使检测更为精准，呈现出比较好的检测效果。图中下方红线外的残差数据认为是异常数据，其对应的原始成功率数据为成功率下降的数据，也是异常的。

4.2.5 基于 Isolation-Forest（孤立森林）\[13\] 的响应时间与成功率的联合检测模型

孤立森林是一个基于 Ensemble 的快速异常检测方法，具有线性时间复杂度和高精度。其策略为用一个随机超平面切割数据空间直至每个子空间里只有一个数据点，那么直观来讲，那些密度很高簇要被切割多次才停止，而密度低的簇很早就停止切割了。对于一组数据，多次切割使其内部点孤立并生成二叉树，树的节点为生成一个孤立节点的切割所需次数，反复切割生成多棵树，求这些树的节点的平均值，节点越小切割所需次数越少，离集群越远。

由于不同时间段交易量不同，对响应时间和交易成功率有较大影响，所以在判断响应时间和交易成功率时以每小时为单位把一天的数据切割成段，分段考察每段的响应时间和成功率异常情况。
1）以每天上午 9 点到 10 点为例，对这段时间应用孤立森林算法，得到结果如图 69-72

图 69 上午 9-10 点的成功率和响应时间的联合检测

图 70 上午 9-10 点的成功率和交易量的联合检测
上午 9-10 点的交易量和响应时间的联合检测

上午 9-10 点的成功率、响应时间和交易量的联合检测

2) 以每天上午 8 点到 9 点为例，对这段时间应用孤立森林算法，得到结果如图 73-76
图 73 上午 9-10 点的成功率和响应时间的联合检测

图 74 上午 9-10 点的成功率和交易量的联合检测

图 75 上午 9-10 点的交易量和响应时间的联合检测
3) 以每天晚上 22 点到 23 点为例，对这段时间应用孤立森林算法，得到结果如图 77-80

图 76 上午 9-10 点的成功率、响应时间和交易量的联合检测

图 77 下午 10-11 点的成功率和响应时间的联合检测
图 78 下午 10-11 点的成功率和交易量的联合检测

图 79 下午 10-11 点的交易量和响应时间的联合检测
图 80 下午 10-11 点的成功率、响应时间和交易量的联合检测

其中，红色代表异常点，绿色代表正常点。
从效果上来看，孤立森林不仅有效的把响应时间和成功率有异常的点找了出来，这些点对应响应时间增加或成功率下降的点，还能把交易量、响应时间，成功率任意二者联合以及三者联合的异常点检测出来，这些点对应交易量陡降，响应时间增加或成功率下降的点。

4.2.6 模型优化

在上述过程中，分别对交易量，成功率和响应时间数据通过使用不同的模型进行了检测，并对模型进行了检验，取得了不错的检测效果，对一些不太灵敏的模型也进行了一定改进和优化，使之能有效的检测出大部分的异常数据。

不过，虽然能对异常数据做到及时报警，但在降低误报率方面还有一定的优化空间，因此引入如下的检测模型：

$$\tau = \frac{x - q}{\sigma} \frac{c}{t}$$  \hspace{1cm} (23)

其中，$$\tau$$ 为异常指数（$$\tau$$ 值越大，异常程度越严重），$$x$$ 为数据观测值，$$q$$ 为异常检测的阈值，$$t$$ 为异常持续时间，$$c$$ 为持续时间内异常点的个数。

在这里，$$q$$ 一般取 3 $$\sigma$$，$$t$$ 一般取 10min，即若在当前阈值下，检测到一个异常数据，再检测该数据之后的 10min 内异常数据出现的频率，以及计算该异常数据超过阈值部分占其观测值的比重，二者作积，若计算结果小于 0.05 则认为现在检测到的异常数、据极有可能是误报的，从而予以忽略，降低误报率。

在实际应用中，检测者可以根据实际情况设置和调整阈值，异常持续时间，从而降低误报率，使检测结果更为精准。

4.3 问题三的模型建立与求解

1. 所需数据

（1）各分行 ATM 机参数配置：

ATM 机的参数数据变更或者配置错误，数据中心后端处理失败率增加，影响交易成功率指标。可用数字 1 表示参数配置合适，数字 0 表示参数配置错误。

（2）ATM 机的工作温度、工作电压：

温度、电压是电子产品工作状态影响很明显的两个指标。ATM 机在其能够正常工作的温度、电压范围内交易成功率较高。根据已有数据可知，夜间交易量较小，成功率
较高。而温度、电压也存在白天与夜晚的周期性变化。当 ATM 机长时间运行，或者随外界温度变化，自身温度超出正常工作温度变化范围时，可能造成前端故障。因白天用电人数增多、用电时段集中，早中晚各个时段电压会有差别，也可能造成 ATM 机交易状态异常。

3) 数据后端处理系统 CPU 占用率：

处理系统 CPU 负责处理每一笔交易，当 CPU 占用率过高时，会造成系统交易响应时间延长。利用 CPU 占用率与交易响应时间之间的正相关关系对报警方案进行矫正，可大大提高因响应时间过长引起报警的准确率。

4) 数据中心后端处理系统应用进程参数：

交易信息由各个分行 ATM 机前端传送至数据中心后端处理系统，当后端处理系统应用进程参数错误时，交易成功率降为零。可利用该指标的此特点对报警方案进行优化，提高其准确度。

5) 若检测到网络负载率达到较大值或者满载值，响应时间较大，成功率较低或交易量骤减就很可能不是前端或后端的故障问题导致，而是数据传输过程出现阻塞导致，此数据的采集也能一定程度上减少误报率。需要采取更多的数据进行 ATM 机异常或故障的分析，ATM 机的交易无非就是各个银行卡账户上的钱数目交易，因此通过对业务量、交易成功率、响应时间、交易数额和网络负载率五者的全面分析，更多的特征能让模型更加精确，从而使系统 ATM 机异常或故障的更精确报警。

2. 基于数据的扩展

1) 由于本文方法主要基于统计学，根据统计学原理所表述的思想，要想增加模型预测的准确度，同时降低误报的概率，增加数据量是唯一的解决方案，这同样也使用与机器学习模型的训练。如果训练 Isolation 模型的数据量加大一倍，预计能提升当前准确度的 10% 左右，（右下图非本题数据参数）

当然如果训练数据过多，可能会导致模型对于数据的过拟合，在预测新数据时可能会导致错误的增多。但是根据经验判定，当前数据容量远没有达到过拟合的情况，离最佳预测结果还有提升的可能。

2) 另外如果知道全年其他月份或其他年份的数据，便可以从更大的霎时间周期来探索数据特征的规律，在已有的探索中，一月份与其他月份已显示出明显的不同。随着时间推移，数据必定会发生变化的趋势，如果不能把握这种趋势，的模型可能会对新出现的
的形势做出错误的报警。

5 模型的创新点和特色

1. 探索方法全面多样，力图从不同角度发现特征的联系。应用了数据科学的基本方法，从不同角度全面的对数据进行探索，包括数据特征随时间变化的规律，特征与特征之间的联系，同时不满足于定性分析，使用相关性度量，时间序列平稳性检验，变量间函数关系拟合的方式，以定量的具体揭示出特征间的关联，并对其中一些特殊现象利用知识推理判断其相应的原因。

2. 运用概率分布的原理，加入已知的知识规则，对异常进行分类。

3. ARMA 模型是分析时间序列的常用方法，从最基本的步骤开始，结合问题，理解数据，尝试不同模型，并逐步提升模型的复杂度。

4. 小波分析是异常检测的利器，其应用极具创新性，已被业界证明检测效果突出。由于小波分析原理艰深难懂，从最简单的知识，边学习边实践，最后取得良好结果，弥补统计模型方法上的不足。

5. 机器学习方法的应用带来一种全新的思路，采用以聚类先分类后训练的原则得出模型，并于前面的结果作比较。

6 参考文献

[1] 王焘, 顾泽宇, 张文博, 徐继伟, 魏峻, 钟华, . 一种基于自适应监测的云计算系统的故障检测方法
7 附录

1. 准备数据. py

```python
# coding: utf-8

def pre_process():
    # 添加时间戳，Dayofweek, Hour (默认不添加)，转换成功率类型，列名转换成英文

    name=['Date','Time','Volume','Success Rate','Response Time','Timestamp']
    for r in range(1,2):
        # m=pd.read_excel(r'D:\B 题附件数据(4.24 附件)\'+str(r)+'月.xls')
        # 添加时间戳属性
        m = pd.read_excel(r'D:\2017B 题\B 题测试数据0511-0624 (8.11 附件)\测试数据1 (0511-0624).xls')

        m['Timestamp']='
        for i in range(len(m)):
            m.loc[i,'Timestamp'] =pd.Timestamp('2017-0'+ 
            str(m.at[i,'日期'][:1])+':'+str(m.at[i,'日期'][:1])+':'+ 
            str(m['时间'][i]).zfill(4)[:2]+':'+str(m['时间' 
            '][i]).zfill(4)[2:])
            m.loc[i,'成功率'] = float(m.at[i,'成功率'][:-1])/100
        print(i)
```
```python
m.columns = name
m['Success Rate'] = m['Success Rate'].convert_objects(convert_dates=False, convert_numeric=True, convert_timedeltas=False)
m['Dayofweek'] = m['Timestamp'].dt.dayofweek
#m['Hour'] = m['Timestamp'].dt.hour
m.to_pickle(r'D:\题附件数据(4.24附件)\'+str(r) + '月.pickle')
m.to_pickle(r'D:\#2017B题B题测试数据0511-0624(8.11附件)\'+ '5-6月.pickle')

#---------------------------------------------------------------

def wavelet(dataset):
    
    return smo 拟合后结果， delta 残差

    import pywt

    vol = dataset

cA, cD4, cD3, cD2, cD1 = pywt.wavedec(vol, 'db4', level=4)  # 13
cD1 = np.zeros(np.size(cD1))
cD2 = np.zeros(np.size(cD2))
cD3 = np.zeros(np.size(cD3))
cD4 = np.zeros(np.size(cD4))
coeff = [cA, cD4, cD3, cD2, cD1]
smo = pywt.waverec(coeff, 'db4')

if len(smo) > len(vol):
    smo = np.delete(pywt.waverec(coeff, 'db4'), len(vol))

res = vol - smo
aver = np.sum(res) / len(smo)
delta = res - aver
cigama = np.sqrt(np.sum(delta**2)/len(smo))

    
plt.figure()
plt.grid(True)
plt.plot(smo)
plt.plot(m4['Volume'], '.', c='g', label='real value')
```
return smo, delta

#---------------------------------------------------------

def res_pp(train_data):
    #响应时间预处理, 去掉 3 月 19 到 3 月 22 日，和 4 月 16 到 4 月 19 日

data = train_data.copy()

pp_data = data.loc[(data.index < 66240) |
    ((data.index >= 66240 + 1440*4) & (data.index < 106557)) |
    (data.index >= 106557 + 1440*4)]
return pp_data

#---------------------------------------------------------

if __name__ == '__main__':

    #m5_6=pd.read_pickle(r'D:\#2017B题\B题测试数据 0511-0624（8.11 附件）\\' + '5-6月.pickle')

    train_data = m5_6.copy()
    vol_mean_per_minute = train_data[\['Time','Volume'\]].groupby([\'Time\']).mean()
    vol_std_per_minute = train_data[\['Time','Volume'\]].groupby([\'Time\']).std()
    vol_median_per_minute = train_data[\['Time','Volume'\]].groupby([\'Time\']).median()

    train_data = three_month.copy()
    train_data = res_pp(p_three_month)

    mean_per_hour = train_data.groupby([\'Hour\']).mean()
    vol_mean_per_hour = mean_per_hour[\'Volume\']
    suc_mean_per_hour = mean_per_hour[\'Success Rate\']
    res_mean_per_hour = mean_per_hour[\'Response Time\']

    std_per_hour = train_data.groupby([\'Hour\']).std()
    vol_std_per_hour = std_per_hour[\'Volume\']
    suc_std_per_hour = std_per_hour[\'Success Rate\']
    res_std_per_hour = std_per_hour[\'Response Time\']
小波

smo, delta = wavelet(m5_6['Volume'])
wave_three_month = m5_6.copy()
wave_three_month['Smo'] = smo
wave_three_month['Delta'] = delta

wav_delta_mean = 
wave_three_month[['Time', 'Delta']].groupby(['Time']).mean()
wav_delta_std = 
wave_three_month[['Time', 'Delta']].groupby(['Time']).std()

for i in range(0, 2):
    wav_delta_mean = 
    wav_delta_mean.append(wave_three_month[['Time', 'Delta']].groupby(['Time']).mean())
    wav_delta_std = 
    wav_delta_std.append(wave_three_month[['Time', 'Delta']].groupby(['Time']).std())

delta_mean_per_minute, _ = wavelet(wav_delta_mean['Delta'])
delta_mean_per_minute = delta_mean_per_minute[1440:1440*2]
temp = vol_mean_per_minute.copy()
temp.columns=['Delta']
temp['Delta'] = delta_mean_per_minute
delta_mean_per_minute = temp.copy()

delta_std_per_minute, _ = wavelet(wav_delta_std['Delta'])
delta_std_per_minute = delta_std_per_minute[1440:1440*2]
temp['Delta'] = delta_std_per_minute
delta_std_per_minute = temp.copy()

2. 检测方案.py
# -*- coding: utf-8 -*-

print_flag = False

#寻找空值点
def search_zero(test_data):
    data = test_data
    list_zero = []

    for index, row in data.iterrows():
        if index < len(data) - 1 and data.ix[index + 1]['Time'] % 100 - data.ix[index]['Time'] % 100 > 1:
            minute = data.ix[index + 1]['Time'] - data.ix[index]['Time']
            if minute == 2:
                li = [row['Timestamp'] + pd.to_timedelta(1, unit='m')]
            else:
                li = [row['Timestamp'] + i * pd.to_timedelta(1, unit='m') for i in range(1, minute)]
            list_zero.append(li)
    return list_zero

# print(list_zero)

# 基于交易量每分钟均值和方差
# 单次 sigma 比较，参数: Z_score = 2 z 分数, duration = 10 持续时间, occurrence = 10 发生次数

def vol_ano_det(test_data, Z_score=2, duration=1, occurrence=1, column='Volume'):
    list_anomaly = []
    count = 0
    flag = False
    timer = duration

    for index, row in test_data.iterrows():
        t = row[1]
        if timer == 0:
            if count >= occurrence:
                li = [row['Timestamp'] - (duration - 1 - i) * pd.to_timedelta(1, unit='m') for i in range(duration)]
            list_anomaly.append(li)
    if print_flag:
        print(li)
```python
def delta_ano_det(test_data, Z_score=3, duration=3, occurrence=3, column='Delta'):
    list_anomaly = []
    count = 0
    flag = False
    timer = duration

    for index, row in test_data.iterrows():
        t = row[1]
        if timer == 0:
            if count >= occurrence:
                li = [row['Timestamp'] - (duration - i - 1) * pd.to_timedelta(1, unit='m')
                     for i in range(duration)]
                list_anomaly.append(li)
                if print_flag:
                    print(li)
            else:
                pass
        count = 0
        timer = duration
        if row[-1] <= delta_mean_per_minute.loc[t][column] - \
```
delta_std_per_minute.loc[t][column]*Z_score:
count += 1
flag = True
if flag:
timer -= 1

return list_anomaly

#---

#成功率检测  success anomaly detection

参数： 白天敏感度：s_day  夜间敏感度：s_night  傍晚、清晨敏感度：
s_morning_evening
9:00 - 19:00 低均值  低方差
6:00 - 9:00 下降  19:00 - 23:00 上升
23:00 - 次日6:00

def suc_ano_det(test_data,duration = 10,occurrence = 2,column = 'Success Rate'):
    list_anomaly = []
count = 0
flag = False
timer = duration

train_data = p_three_month
expectation = train_data.groupby(["Hour"])["Success Rate"]
s_day = expectation.quantile(0.005)
s_night = expectation.quantile(0.001)
s_morning_evening = expectation.quantile(0.005)

for index,row in test_data.iterrows():
t=row[1]
h=row[5].hour
suc=row[3]

    if timer == 0:
        if count >= occurrence:
            li = [row['Timestamp'] - (duration - 1 - i)*pd.to_timedelta(1,
unit = 'm') \n                  for i in range(duration)]
            list_anomaly.append(li)
        if print_flag:
```python
print(li)
else:
count = 0
timer = duration

if t >= 900 and t <= 1900:
    if suc <= s_day.loc[h]:
        count += 1
        flag = True

if t >= 2300 or t <= 600:
    if suc <= s_night.loc[h]:
        count += 1
        flag = True

if (t > 1900 and t < 2300) or (t > 600 and t < 900):
    if suc <= s_morning_evening.loc[h]:
        count += 1
        flag = True

if flag:
    timer -= 1

return list_anomaly

#---------------------------------------------------------------

# 应时间检测 response anomaly detection

参数： 白天敏感度：s_day 夜间敏感度：s_night 傍晚、清晨敏感度：s_morning_evening
9:00 - 19:00 低均值 低方差
6:00 - 9:00 下降 19:00 - 23:00 上升
23:00 - 次日6:00

def res_ano_det(test_data, duration=1, occurrence=1, column='Response Time'):
    list_anomaly = []
count = 0
flag = False
timer = duration
```
train_data = res_pp(p_three_month)
expectation = train_data.groupby(['Hour'])['Response Time']
s_day = expectation.quantile(0.995)
s_night = expectation.quantile(0.9995)
s_morning_evening = expectation.quantile(0.995)

for index, row in test_data.iterrows():
    t = row[1]
    h = row[5].hour
    res = row[4]

    if timer == 0:
        if count >= occurrence:
            li = [row['Timestamp'] - (duration - 1 - i)*pd.to_timedelta(1, unit = 'm') \
                for i in range(duration)]
            list_anomaly.append(li)
        else:
            pass
        count = 0
        timer = duration

    if t >= 900 and t <= 1900: #白天
        if res > s_day.loc[h]:
            count += 1
            flag = True

    if t >= 2300 or t <= 600: #夜间
        if res > s_night.loc[h]:
            count += 1
            flag = True

    if (t > 1900 and t < 2300) | (t > 600 and t < 900): #清晨和傍晚
        if res > s_morning_evening.loc[h]:
            count += 1
            flag = True

    if flag:
        timer -= 1

return list_anomaly

#------------------------------------------------------------
#故障四检测  suc & response anomaly detection

参数： 白天敏感度: s_day  夜间敏感度: s_night  傍晚、清晨敏感度: s_morning_evening
9:00 - 19:00 低均值 低方差
6:00 - 9:00 下降  19:00 - 23:00 上升
23:00 - 次日 6:00

```python
def suc_and_res_ano_det(test_data, duration = 1, occurrence = 1):
    list_anomaly = []
    count1 = 0
    count2 = 0
    flag = False
    timer = duration
    column1 = 'Success Rate'
    column2 = 'Response Time'

    train_data = p_three_month
    expectation1 = train_data.groupby(['Hour'])['Success Rate']
    s_day1 = expectation1.quantile(0.05)
    s_night1 = expectation1.quantile(0.01)
    s_morning_evening1 = expectation1.quantile(0.05)

    train_data = res_pp(p_three_month)
    expectation2 = train_data.groupby(['Hour'])['Response Time']
    s_day2 = expectation2.quantile(0.95)
    s_night2 = expectation2.quantile(0.99)
    s_morning_evening2 = expectation2.quantile(0.95)

    for index, row in test_data.iterrows():
        t = row[1]
        h = row[5].hour
        suc = row[3]
        res = row[4]

        if timer == 0:
            if count1 >= occurrence and count2 >= occurrence:
                li = [row['Timestamp'] - (duration - 1 - i)*pd.to_timedelta(1, unit = 'm') 
                     for i in range(duration)]
                list_anomaly.append(li)
            else:
                pass
            count1 = 0
```

count2 = 0
timer = duration

if t >= 900 and t <= 1900:  #白天
    if suc<s_day1.loc[h]:
        count1 +=1
        flag=True
    if res > s_day2.loc[h]:
        count1 +=1
        flag=True

if t >= 2300 or t <= 600:  #夜问
    if suc<s_night1.loc[h]:
        count1 +=1
        flag=True
    if res > s_night2.loc[h]:
        count2 +=1
        flag=True

if (t>1900 and t<2300) | (t>600 and t<900):  #清晨和傍晚
    if suc < s_morning_evening1.loc[h]:
        count1 +=1
        flag=True
    if res > s_morning_evening2.loc[h]:
        count2 +=1
        flag=True

if flag:
    timer -= 1

return list_anomaly

#----------------------------------------------------------------------------------
if __name__ == '__main__':
    #小波检测用 wave_three_month
    test_data = wave_three_month.copy()

    vol_zero = search_zero(test_data)

    #vol_anomaly = vol_ano_det(test_data,2,1,1)
delta_anomaly = delta_ano_det(test_data,3,10,2)
suc_anomaly = suc_ano_det(test_data,60,3)
res_anomaly = res_ano_det(test_data,60,5)
suc_and_res_anomaly = suc_and_res_ano_det(test_data,60,3)

#要求程序输出报告包含：异常个数，每个异常时间窗口和类型描述。
print('异常个数为: {}',format(len(delta_anomaly)+len(suc_anomaly)+len(res_anomaly)+len(suc_and_res_anomaly)))
li_anomaly = [delta_anomaly,suc_anomaly,res_anomaly,suc_and_res_anomaly]
li_name = ['故障一','故障二','故障三','故障四']
for i in range(0,4):
    ano = li_anomaly[i]
    for j in ano:
        print(j,li_name[i])

3. 孤立森林.py
#-*- coding: utf-8 -*-
from mpl_toolkits.mplot3d import Axes3D
from sklearn.ensemble import IsolationForest
def res_pp(train_data):
    
    #响应时间预处理,去掉3月19到3月22日，和4月16到4月19日
    
    data=train_data.copy()
    pp_data = data.loc[(data.index<66240) | ((data.index >= 66240+1440*4) &
    
                     (data.index<106557)) | (data.index >= 106557+1440*4)]
    .reset_index()
    return pp_data

pp_data = res_pp(three_month)

train_data = pp_data.set_index('Timestamp')
train_data = train_data.loc[(train_data.index.hour >= 8) &
    (train_data.index.hour <= 19)]
suc = train_data['Success Rate']
vol = train_data['Volume']
res = train_data['Response Time']
timestamp = train_data.index
time_list = []

rs = np.c_[res, suc]
clf = IsolationForest(contamination=0.005)
clf.fit(rs[len(rs)/100:))
y = clf.predict(rs)

inline = np.zeros([1, 2])
outline = np.zeros([1, 2])

mid_s = np.min(rs[:, 0])+(np.max(rs[:, 0])-np.min(rs[:, 0]))/2
mid_r = np.min(rs[:, 1])+(np.max(rs[:, 1])-np.min(rs[:, 1]))/2

for i in range(len(y)):
    if y[i] == 1:
        inline = np.insert(inline, 1, rs[i], axis=0)
    else:
        outline = np.insert(outline, 1, rs[i], axis=0)
time_list.append(timestamp[i])

inline = np.delete(inline, 0, axis=0)
outline = np.delete(outline, 0, axis=0)

plt.figure()
plt.scatter(inline[:, 0], inline[:, 1], c='green', label='Normal')
plt.scatter(outline[:, 0], outline[:, 1], c='red', label='Anomaly')
plt.legend()
plt.xlabel('Response Time')
plt.ylabel('Success Rate')
plt.title('Isolation Forest 8:00-20:00 (co=0.005)')
plt.show()

plt.figure(2)
ax = plt.subplot(111, projection='3d')
ax.scatter(inline[:, 2], inline[:, 0], inline[:, 1], color='green')
ax.scatter(outline[:, 2], outline[:, 0], outline[:, 1], color='red')
ax.set_xlabel('Date')
4. 小波变换
# -*- coding: utf-8 -*-

def fft_tran():
    
    快速傅里叶变换
    
    from scipy.fftpack import fft, ifft

    vf = fft(Vol)
    ex_vol = ifft(vf)
    diff = vol_301 - ex_vol
    plt.figure()
    plt.plot(Time, diff)
    plt.show()

#------------------------------------------------------------------------------------------

def kalman(data):

    #data=np.array((delta_mean_per_minute+3*delta_std_per_minute).reset_index()["Delta"]).flatten()

    # 可调参数 q, r
    vkk = np.zeros(np.size(data))
    vkk[0] = data[0]
    pkk = np.zeros(np.size(data))
    pkk[0] = 100
    kg = np.zeros(np.size(data))
    q = 1e-1
    r = 5
for i in range(len(data)-1):
    z = pkk[i] + q
    kg[i] = z/(z+r)
    vkk[i+1] = vkk[i] + kg[i]*(data[i+1]-vkk[i])
    pkk[i+1] = (1-kg[i])*z

plt.figure()
plt.plot(vkk,'-',c='r',label='after filtering')
plt.plot(data,'.',c='g',label='real value')
plt.xlabel('Timestamp')
plt.ylabel('Volume')
plt.title('Kalman Filter')
plt.legend()

plt.show()

return vkk

#---------------------------------------------------------------------------------------------

def wavelet_original(dataset):
    import pywt

    d=m4.set_index('Timestamp')
    d=d[(d.index.hour>=0) | ( d.index.hour<=23)].reset_index()
    vol=d['Volume']

    vol=dataset

    cA, cD4, cD3, cD2, cD1 = pywt.wavedec(vol, 'db4', level=4)  # 13
    cD1 = np.zeros(np.size(cD1))
    cD2 = np.zeros(np.size(cD2))
    cD3 = np.zeros(np.size(cD3))
    cD4 = np.zeros(np.size(cD4))

    coeff = [cA, cD4, cD3, cD2, cD1]
    smo = pywt.waverec(coeff, 'db4')

    print(len(d),len(vol),len(smo))

    if len(smo)>len(vol):
        smo = np.delete(pywt.waverec(coeff, 'db4'), len(vol))
res = vol - smo
aver = np.sum(res) / len(smo)
delta = res - aver
cigama = np.sqrt(np.sum(delta**2)/len(smo))

d['Delta']=delta

plt.figure()
plt.grid(True)
plt.plot(smo)
plt.plot(m4['Volume'],'.',c='g',label='real value')

return d