ATM 交易状态特征分析与异常检测

董天文 俞周瑜 叶勇超 杨嘉欣

宁波大学 理学院

摘要

随着 ATM 机器的普及，ATM 机具有无需人员管理自动完成交易的特点，ATM 机的自动故障识别与交易状态异常检测成为关键问题。本文针对商业银行总行数据信息进行多维度建模分析，设计了一套异常识别的智能算法。

首先，我们对交易数据进行预处理。结合日期与交易量的变化情况，我们将数据分为三个时间区间，即 1 月 23 日至 1 月 27 日，1 月 28 日至 2 月 1 日以及 2 月 2 日至 4 月 23 日。同时我们分析了工作日与非工作日数据,发现其无显著差异，在下文处理中不对其区分处理。

对成功率与响应时间两个指标，我们发现具有明显的集簇现象。由于题中并未给出故障点标记，我们采用无监督机器学习对成功率与响应时间的样本点进行特征提取。因为本题的异常点稀疏、正常点集中，所以采用传统的 K-means 或层次聚类算法效果较差。为此，我们采用了改进的 K-means 聚类算法，根据数据点的分布范围情况来均匀生成 K 个质心。利用 80%的现有数据聚类后，我们利用决策树对特征值进行提取和阈值划分，利用剩余 20%数据对阈值进行检验。我们将数据点划分为正常点、疑似异常点、明确异常点三大类。对于疑似的异常点，我们再根据其时间序列周围点的分布情况确定其是否确实为异常点。

对于交易量指标，由于数据随时间序列的分布特点，我们较难用线性或非线性回归来描述其分布；又因为其分布特点也较难用聚类算法来进行异常点识别，为此我们设计了基于 LOF 离群因子的三道筛选流程来判断交易量异常点。考虑到交易量在正常情况下与时间序列有一定的联系，我们首先通过 LOF 局部离群因子对离群点进行识别，结合交易量随时间的移动均线及标准差加以辅助筛选得到初步的异常点。我们将每天的交易量指标标准化，通过将疑似异常点与不同天同一时刻数据进行比较，进行了进一步筛选，最终确定是否为异常点。

以上识别过程中，我们还设置了橙色预警、红色预警、黑色预警三种警报等级。在黑色预警（需要维修点）的判定过程中，我们分别对橙色预警与红色预警设定了权重，一段时间内总得分超过某个阈值就确定为黑色预警。综合以上的识别过程，我们建立了一套综合识别的系统算法。通过我们的算法识别，我们一共发现橙色预警 759 处，红色预警 889 处，黑色预警 37 处，分行侧网络传输节点故障情况 858 处，数据中心后端处理系统异常情况 13 处，数据中心后端处理系统应用进程异常情况 18 处。除此之外，我们还基于识别出来的异常点提出了一种提前发现重大故障的方案作为预防的参考。

对于增加采集的数据，我们一方面考虑了增加数据的维度，引入每分钟交易金额与每分钟的网络负载率，构成了“交易量+交易金额”的新模型以及“响应时间+网络负载率”的新模型。另一方面，我们考虑引入专家参考意见，对我们的一部分阈值进行确定，增加模型的可靠性。最后我们考虑引入一部分故障样本集，结合无监督机器学习与有监督机器学习来优化模型效果，同时对算法进行了一定的描述。

在模型的检验方面，我们利用随机选取的 20%数据对阈值进行检验。通过检验，我们发现采用决策树提取的阈值划分，在时间区间 1 月 23 日至 1 月 27 日以及 1 月 28 日至 2 月 1 日效果都较好；在时间区间 2 月 2 日至 4 月 23 日，决策树在异常区域与正常区域提取的阈值划分效果良好，对于在疑似异常区域与正常区域所表现出来的模糊性，可进一步采集专家意见对阈值的合理性进行评价。

本文采用的思想可以推广用于电子商务犯罪和信用卡欺诈的侦查、网络入侵检测、生态系统失调检测、公共卫生、医疗和天文学上稀有的未知种类的天体发现等领域中，具有一定的启发作用。

关键词：无监督机器学习；特征提取；时间序列分析；决策树； LOF 离群因子

目录

一、 问题重述

商业银行的 ATM 应用系统包括前端和后端两个部分。前端是部署在银行营业部和各自助服务点的 ATM 机（系统），后端是总行数据中心的处理系统。商业银行总行数据中心监控系统为了实时掌握全行的业务状态，每分钟对各分行的交易信息进行汇总统计。汇总信息包括业务量、交易成功率、交易响应时间三个指标，各指标解释如下：

1、业务量：每分钟总共发生的交易总笔数；

2、交易成功率：每分钟交易成功笔数和业务量的比率；

3、交易响应时间：一分钟内每笔交易在后端处理的平均耗时(单位：毫秒)。

交易数据分布存在以下特征：工作日和非工作日的交易量存在差别；一天内，交易量也存在业务低谷时间段和正常业务时间段。当无交易发生时，交易成功率和交易响应时间指标为空。

商业银行总行数据中心监控系统通过对每家分行的汇总统计信息做数据分析，来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。常见的故障场景包括但不限于如下情形：

1、分行侧网络传输节点故障，前端交易无法上送请求，导致业务量陡降；

2、分行侧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；

3、数据中心后端处理系统异常（如操作系统 CPU 负荷过大）引起交易处理缓慢，影响交易响应时间指标；

4、数据中心后端处理系统应用进程异常，导致交易失败或响应缓慢。

附件是某商业银行 ATM 应用系统某分行的交易统计数据。你的任务是：

（1）选择、提取和分析 ATM 交易状态的特征参数；

（2）设计一套交易状态异常检测方案，在对该交易系统的应用可用性异常情况下能做到及时报警，同时尽量减少虚警误报；

（3）设想可增加采集的数据。基于扩展数据，你能如何提升任务（1）（2）中你达到的目标？

二、 问题分析

对于问题一，我们需要选择提取和分析 ATM 交易状态的特征参数，我们可以先对日期、时刻、交易量、成功率、响应时间的数据特点进行分析。利用 spss 软件对参数的相关性进行分析，同时利用 matlab 软件分析交易量时间序列随着日期的变化规律，并对问题进行分段研究；分析工作日与非工作日的数据特点，对其是否分开处理进行研究。对于特征参数提取，我们选择利用聚类与决策树提取成功率、响应时间的阈值，对于交易量指标，我们选择提取每一时刻的离群因子作为其特征参数。

对于问题二，根据问题一提取的特征参数，我们利用 80%数据聚类与决策树划分的阈值作为指标，同时用 20%数据对决策树结果进行检验。进而判断日期分段以后的每一

段数据的成功率与响应时间是否异常。同时根据阈值将警报等级划分为橙色预警、红色预警、需要修理的异常点等不同的预警等级。对交易量我们采用的参数为离群因子。利用离群因子对同一天不同时刻以及不同天同一时刻的交易量随时间变化的数据点进行刻画，筛选交易量比正常点低，同时对应的数据点离群因子比正常点高的数据点作为疑似异常点，结合交易量均线设定的曲线下界对疑似的异常点进行筛选，筛选出的点再代入同一时刻不同日期的交易量图像进行预警等级划分，最终判断出异常点以及对应的预警等级。

对于问题三，我们考虑增加的采集数据有两方面，一方面增加数据的维度，我们考虑到每分钟交易金额与网络负载率指标。另一方面增加故障样本数据集，可以通过专家意见采集等方式，理论上能够从本文的无监督机器学习转换为有监督机器学习。

三、 基本假设

1、所有的数据无误，无非正常因素对数据产生扰动影响。

2、不考虑银行的运营情况对 ATM 系统的影响。

四、 符号说明

符号	说明
J	数据点标签（0 为正常点，1 为疑似异常点，2 为异常点）
T	响应时间
R	成功率
D	日期
g_1	红色预警得分
g_2	橙色预警得分
C	得分阈值（需要修理的异常点阈值）
k	近邻对象数
$LOF_k(p)$	对象 p 的局部异常因子

五、 模型建立与求解

1. 数据预处理与分析

（1） 数据预处理

题中数据有一定的缺失，题中提到：当无交易发生时，交易成功率和交易响应时间指标为空。因此，首先我们需要补全数据，为之后的分析做准备。我们对缺失的时间段的数据补 0 操作。

（2） 数据分析

- 数据分段处理

为了更准确的选择、提取和分析 ATM 交易状态的特征参数，我们对交易量数据进

行分段预处理，避免出现较大或较小的交易量对平稳交易量的影响，我们首先根据交易量对数据进行分段，交易量与日期的关系如下图所示：

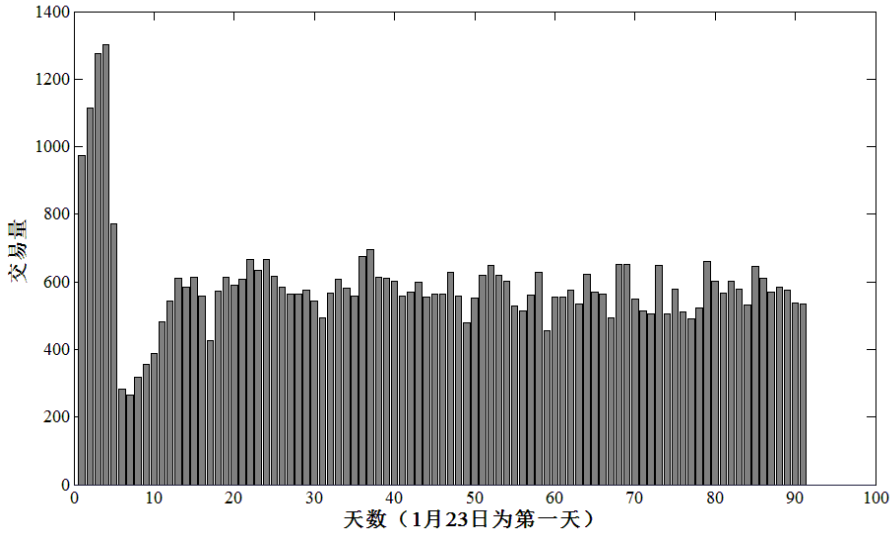

图 1 日平均交易量与日期关系图

从上图中，我们直观上分析，在 1 月 23 日至 1 月 27 日、1 月 28 日至 2 月 1 日的日平均交易量存在明显差别，为了更加明显表示其区别，我们在一张图上表示出这三段时间的平均交易量随时间的变化。

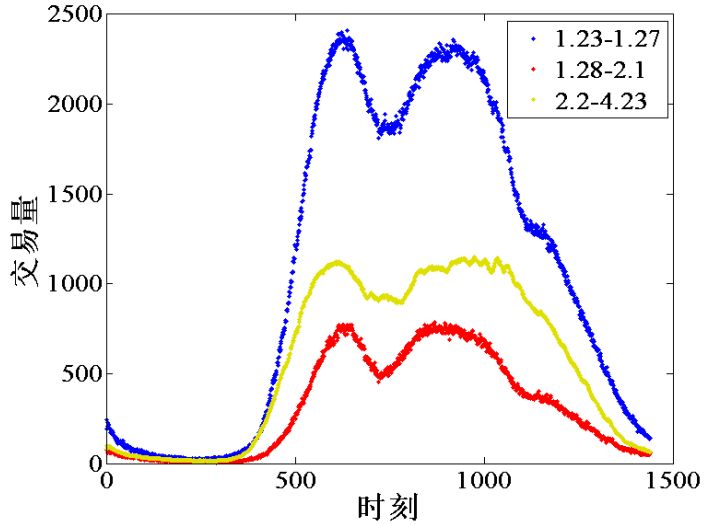

图 2 三段日期的交易量分布曲线

从实际因素分析，我们推断可能是除夕前人们需要准备送礼或准备压岁钱 ATM 机导致交易量增加，春节期间人们走亲访友导致交易量减少，因此决定分段处理日总交易量和时间的曲线，即 1 月 23 日至 1 月 27 日、1 月 28 日至 2 月 1 日、2 月 2 日至 4 月 23 日三段。

- 工作日与非工作日数据处理

题目中提到：工作日和非工作日的交易量存在差别，我们绘制工作日的日总交易量与非工作日每一分钟交易量平均值的图像，来判断差异性如下所示：

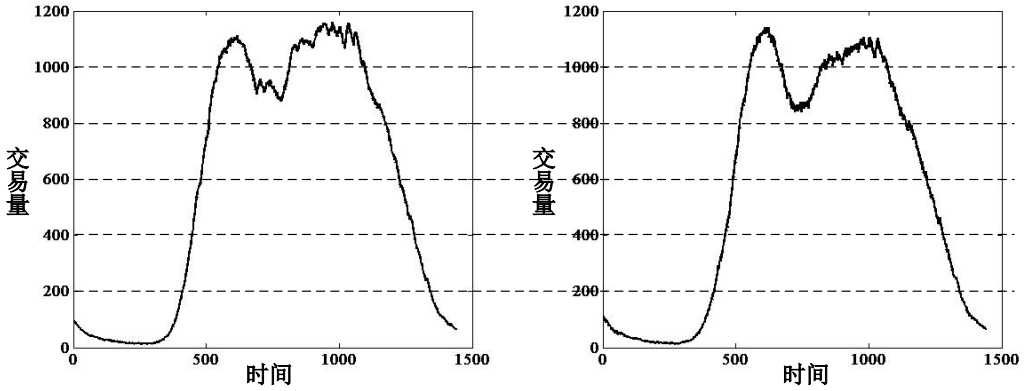

图 3 非工作日（左）与工作日（右）的时间-交易量散点图

通过上图初步分析，我们发现工作日与非工作日的每一时刻交易量基本一致，仅有第 600 分钟到第 900 分钟的交易量波谷区域存在一些差异，并且观察得出工作日和非工作日的单日交易量平均值不存在明显区别。为了方便统计与参数提取，我们在接下去的论述过程中，不对工作日与非工作日的数据进行分段处理。

（3） 相关性分析

首先我们对题中所给的日期、时刻、交易量、成功率、响应时间进行相关性分析，其目的是进行初步的数据相关性观察。为下一步提取特征参数打下基础。

得到以下相关性矩阵：

表 1 相关性矩阵

		日期	时刻	交易量	成功率	响应时间
相关	日期	1.000	0.000	-0.064	-0.003	0.012
	时刻	0.000	1.000	0.374**	-0.088	-0.028
	交易量	-0.064	0.374**	1.000	-0.075	-0.033
	成功率	-0.003	-0.088	-0.075	1.000	-0.365**
	响应时间	0.012	-0.028	-0.033	-0.365**	1.000

**相关性在 0.01 水平显著（双尾）

由上表我们可以大致发现数据之间的关系：

- 时刻与交易量存在相关性；
- 成功率与响应时间存在负相关性；

2. 问题一模型建立与求解

（1） 成功率与响应时间特征参数提取

由于题目未给出故障样本，我们提取特征参数的时候需要参考一系列无监督机器学习的办法。根据我们上文数据处理过程中发现的数据特点，结合成功率与响应时间的图像进一步分析数据特点如下：

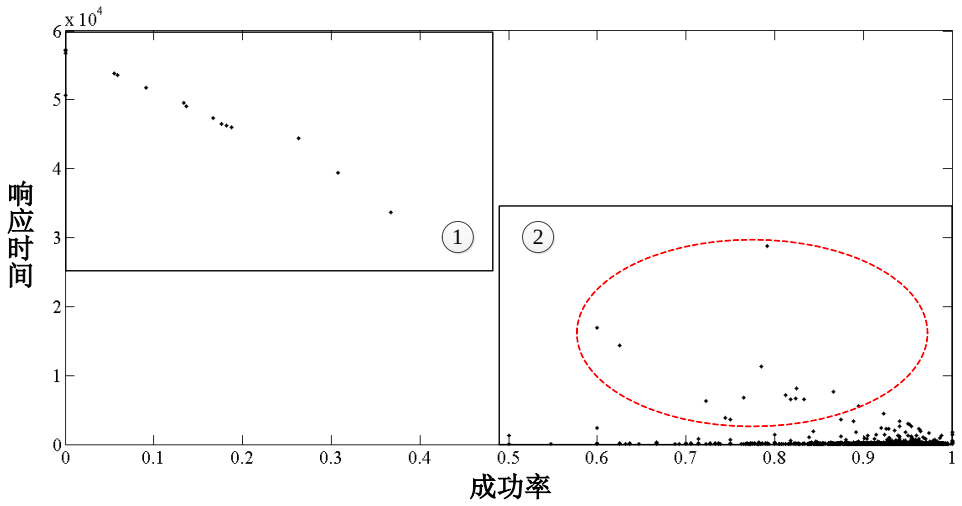

图 4 成功率与响应时间相关图像

由上图，我们可以发现，①区域成功率低，响应时间高，呈现的可以确认是故障点情况。在②区域的数据点相对较离散，我们可以看到大部分的交易点在靠近右下角的位置，即正常参数区域，但是图中红色圆部分的数据点存在远离正常点的趋势，我们称其为疑似异常点。于是我们有了正常点、疑似异常点、明确异常点三类点，在进行参数提取的时候，我们决定对成功率-响应时间两个指标的数据点先进行聚类，再通过决策树确定类的界限，从而确定参数。

- 传统 K-means 聚类

传统 K-Means 算法的基本思想是初始随机给定 K 个簇中心，按照最邻近原则把待分类样本点分到各个簇。然后按平均法重新计算各个簇的质心(这个点可以不是样本点)，从而确定新的簇心。一直迭代，直到簇心的移动距离小于某个给定的值。

传统的 K-Means 聚类由于其聚类中心的随机选取，在本题的数据异常点较少、正常点较多的情况下表现不稳定。因此下文我们根据点的分布来确定其聚类中心，增加其稳定性。

- 改进的 K-means 聚类

我们在传统的 K-means 聚类基础上，根据 X 的分布范围均匀的随机生成 K 个质心。即当图 4 中的①区域偏离点较少时，也能根据其分布范围产生偏离的质心，从而使其结果较为稳定，我们对成功率-响应时间的分三段改进聚类结果如下（每个时间段**用 80%数据聚类，剩余 20%数据用来后续检验**，如：其中第三段时间段有 82 天，我们随机抽取 66 天进行聚类，用 16 天进行检验）：

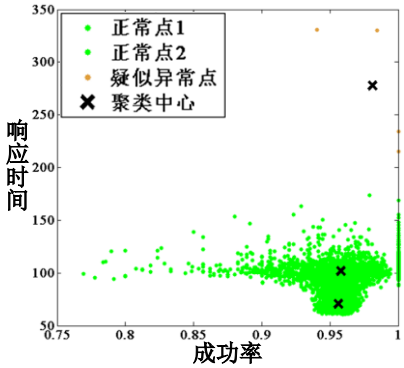

图 5 1 月 23 日-1 月 26 日聚类

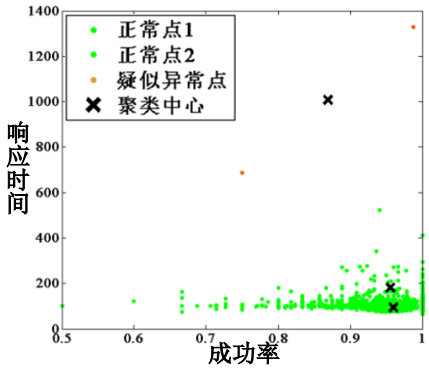

图 6 1 月 28 日-1 月 31 日聚类

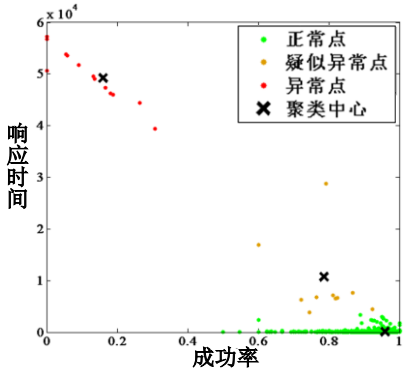

图 7 第三段随机 66 天数据聚类

根据我们上面三段时间段的聚类结果，我们做出以下解释：

- 春节前阶段（1 月 23 日-1 月 27 日）我们取了四天的数据作为训练集。数据整体的响应时间与成功率都在较理想的状态，即成功率整体较高，响应时间整体较低，我们聚类之后发现第一类与第二类都属于正常点，第三类作为疑似异常点进行下一步的判断。
- 春节阶段（1 月 28 日-2 月 1 日）我们取了四天的数据作为训练集。数据特点是平均交易量偏低，从成功率与响应时间来看，第一类与第二类都属于正常点范围，第三类属于异常点。
- 春节后阶段（2 月 2 日-4 月 23 日数据）一共有 82 天，我们随机抽取 66 天（80%样本）作为训练集。数据是较为明显分为了三类，我们将其归纳为正常点、疑似异常点、异常点，对疑似异常点进行下一步的判断。
- 对于第一段与第二段数据，其中的疑似异常点放在第三段中就会变成正常点。但是因为交易量的区别较为明显，我们判断其为独立的时间段，即相对于每一段的数据点情况区别讨论参数情况，故障判定也独立。

下面我们对每一段时间段（成功率-响应时间）进行参数提取，具体采取的方法是对用 80%样本聚类完成的数据用决策树提取阈值。同时对每一时间段的阈值，我们利用 20%未使用的数据作为测试集对其结果进行检验，如下所示：

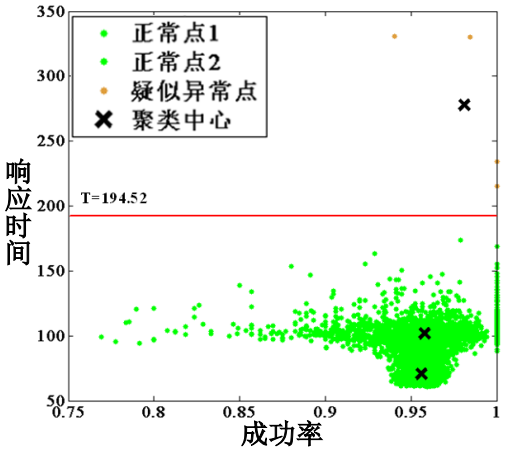

图 8 第一个时段决策树结果

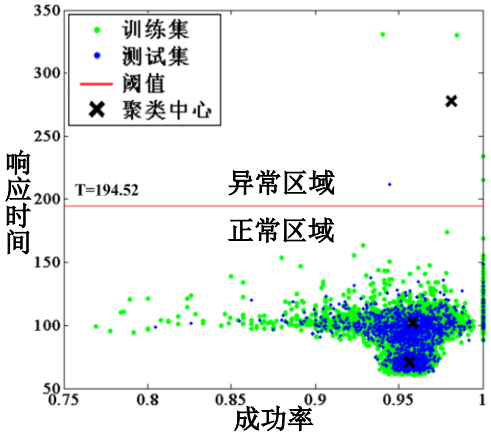

图 9 第一个时段测试集测试结果

我们可以看到春节前数据经过决策树划分后，红线阈值上方暂定为异常区域，红色阈值下方暂定为正常区域。我们利用 1 月 27 日数据进行检验，如图 9 所示，我们可以发现阈值划分效果非常好，即第一时段决策树划分成功。

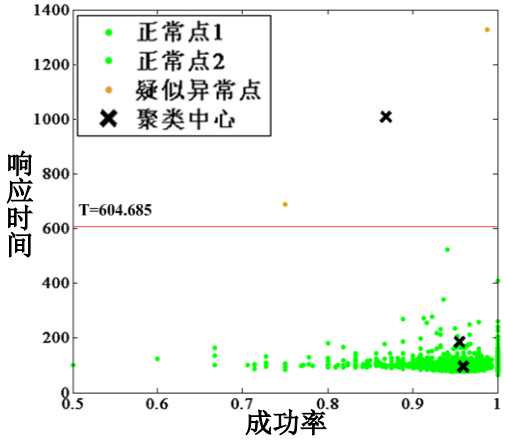

图 10 第二个时段决策树结果

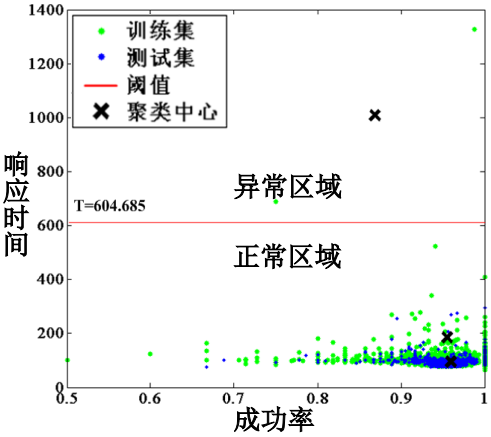

图 11 第二个时段测试集测试结果

同理如上图所示，春节中的数据也被分成上下两个区域。同样我们用 2 月 1 日数据进行测试，发现该日数据全部分布在正常区域，划分情况也较好。

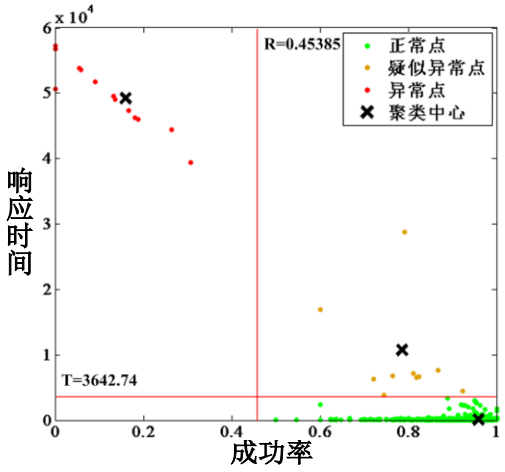

图 12 第三个时段决策树结果

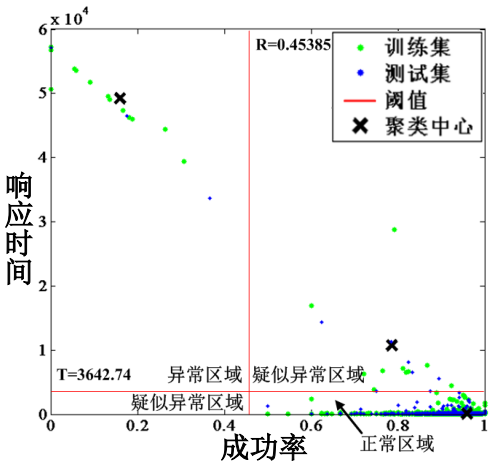

图 13 第三个时段测试集测试结果

对春节以后的数据点，我们通过聚类发现能够分成三类。分别利用决策树提取了成功率与响应时间的阈值。第三段数据一共 82 天，我们随机选取了 20%的天数作为测试集，发现决策树在异常区域与正常区域表现良好。只有疑似异常区域与正常区域交界处较为模糊，我们将会在第二文中给出解决方案，来进一步判断疑似异常点与正常点。

综合上面三段时间，我们再对以上过程做简单说明如下：

- 成功率与响应时间的特征参数我们直接采取聚类后的决策树确定，将点分为正常点、疑似异常点、异常点三类。
- 对于疑似异常点的进一步判断将在下一问进一步判断，这里仅给出参数范围。
- 图 13 的右上和左下区域被认为是疑似异常点区域，虽然所给数据中没有点落在左下区域，我们暂且假设该区域为疑似异常区域。

对以上提取的参数用公式表示如下:

$$\begin{cases} J=0, & (T<194.52 \,\&\, 1/23 \le D \le 1/27) \\ J=1, & (T>194.52 \,\&\, 1/23 \le D \le 1/27) \\ J=0, & (T<604.685 \,\&\, 1/28 \le D \le 2/1) \\ J=1, & (T>604.685 \,\&\, 1/28 \le D \le 2/1) \\ J=0, & (T<3642.74 \,\&\, R>0.45385 \,\&\, 2/2 \le D \le 4/23) \\ J=1, & (T>3642.74 \,\&\, R>0.45385 \,\&\, 2/2 \le D \le 4/23) \\ J=1, & (T<3642.74 \,\&\, R<0.45385 \,\&\, 2/2 \le D \le 4/23) \\ J=2, & (T>3642.74 \,\&\, R<0.45385 \,\&\, 2/2 \le D \le 4/23) \end{cases} \tag{1}$$

上式中， J 表示判定，0、1、2 分别对应正常点、疑似异常点、异常点。 T 表示响应时间， R 表示成功率， D 表示日期。

（2） 交易量特征参数提取

由于交易量是一个随时间序列变化的变量，且其图像特点较难用以上的聚类、决策树办法进行描述，我们进一步通过图像分析其数据特点，我们做出了日期-时间-交易量三个变量的三维图像，如下所示：

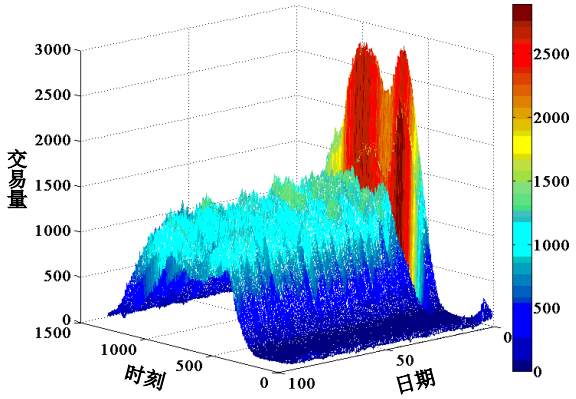

图 14 日期-时间-业务量三维图形

从上图我们可以得出初步结果：业务量与时间序列存在明显的相关性，曲线呈 M 型，且从图中可以看出同一段时间内不同日期的曲线形状相似度很高。由于业务量与时间序列的数据特点，我们决定提取每个数据点的局部离群因子（LOF）作为特征参数。

LOF(Local Outlier Factor，对象的局部异常因子) 算法中每个数据都被分配一个局部异常因子，局部异常因子愈大，就认为它更可能是一种异常；反之则可能性小．计算局部异常因子，先产生所有数据点的 k -邻域(同时得到 k -距离)，并计算到其中每个点的距离。算法流程如下：

1） 计算对象 p 的 k -距离

对任意的自然数 k，定义 p 的 k -距离(k - $distance(p)$)，为 p 和某个对象 o 之间的距离,这里的 o 满足:至少存在 k 个对象 $o' \in D\setminus\{p\}$，使得 $d(p,o') \le d(p,o)$，并且至多存在 $k-1$个对象 $o' \in D\setminus\{p\}$，使得 $d(p,o') < d(p,o)$.

2） 计算对象 p 的 k -距离邻域($N_{k\text{-}distance}$)

$$N_{k\text{-}distance}(p) = \{q \mid d(p,q) \le k\text{-}distance(p)\} \tag{2}$$

3） 计算对象 p 相对于对象 o 的可达距离

给定自然数 k，对象 p 相对于对象 o 的可达距离为:

$$reach-dist_k(p,o) = \max\{k\text{-}distance(o), d(p,o)\} \tag{3}$$

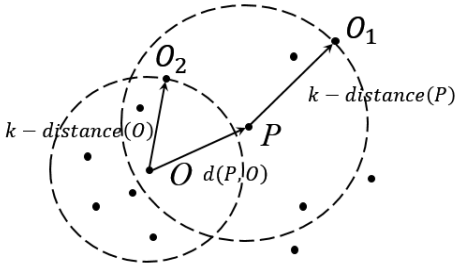

图 15 可达距离示意图

4） 计算对象 p 的局部可达密度(Local Reachable Distance)

对象 p 的局部可达密度为对象 p 与它的 k -邻域的平均可达距离的倒数.

$$lrd_k(p)=1/\left[\frac{\sum reach-dist_k(p,o)}{|N_k(p)|}\right] \tag{4}$$

5） 计算对象 p 的局部异常因子

$$LOF_k(p)=\sum_{o\in N_k(p)}\frac{lrd_k(o)}{lrd_k(p)}\Big/|N_k(p)| \tag{5}$$

LOF 提取后，对于下一问的故障判断，我们还会采用移动平均线结合标准差的办法加强模型稳定性，这点将会在第二问故障检验具体描述。

3. 问题二模型建立与求解

题中所给四个故障反映在指标上的关系图如下所示：

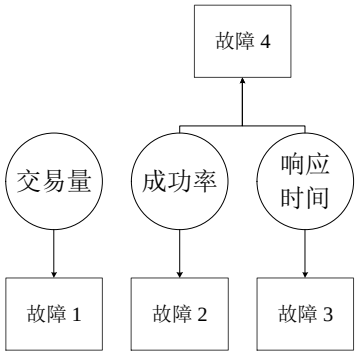

图 16 指标反映对应故障

由于故障判断需要通过三个指标数据的异常得出，因此我们先对三个指标的异常数据判断分别建立模型，然后建立一个综合检验体系。

（1） 交易量异常故障检测模型

在交易量分析中，我们上文已将交易量分段为 1 月 23 日至 1 月 27 日、1 月 28 日至 2 月 1 日以及 2 月 2 日至 4 月 23 日。

我们要检验的异常点示意如下图三段时间段中的红圈所示，也即我们需要将交易量突然下降或者明显远离数据的时间序列分布的点找出，将其作为疑似异常点。

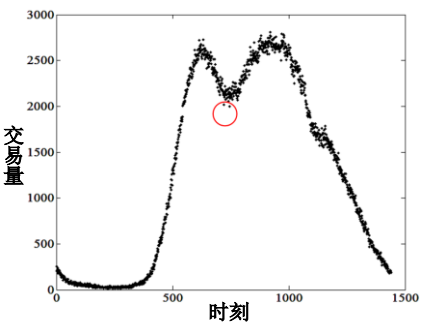

图 17 第一段（1 月 25 日）数据

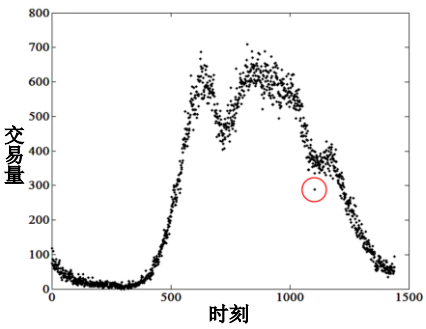

图 18 第二段（1 月 28 日）数据

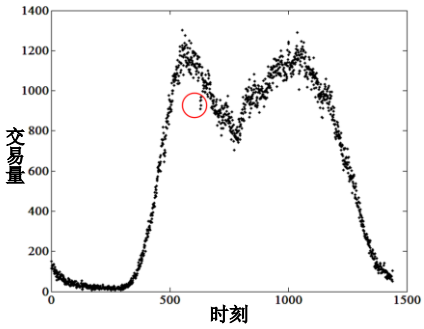

图 19 第三段（4 月 18 日）数据

LOF 检验离群点需要确定 k 值，k 值取太小会使得平缓部分的一部分疑似异常点被当成正常点，如图 20 所示。k 值取太大又会使得上升下降阶段时 LOF 值较敏感，如图 21 所示：

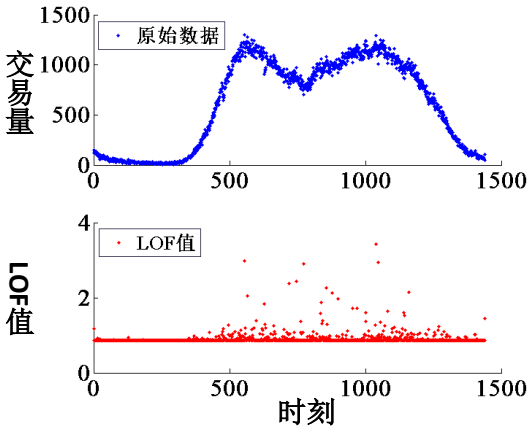

图 20 k =2 时 LOF 表现情况

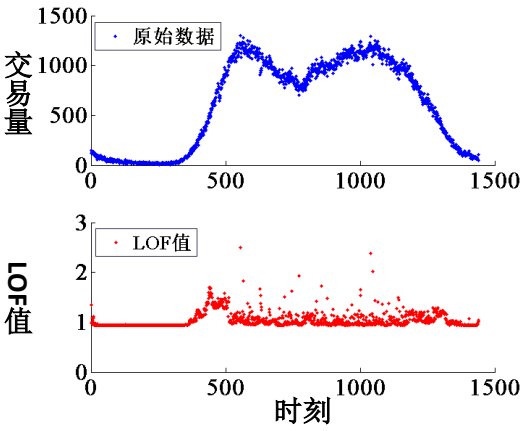

图 21 k =20 时 LOF 表现情况

经过 k 取值的分析对比，我们采用 k =5 作为邻近对象。

一般情况下，我们认为异常因子接近 1 的点，说明它和周围点的密度一致，判定为正常；异常因子越小说明它和周围点的密度相差越大，因此，成为异常点的可能性也就越大。由于本题的数据分布特点，我们发现不能直接定下一个 LOF 阈值（如 LOF>2 算作疑似异常点）作为判断依据，如下图 k =5，我们选取 LOF 值等于 2 作为阈值观察情况。

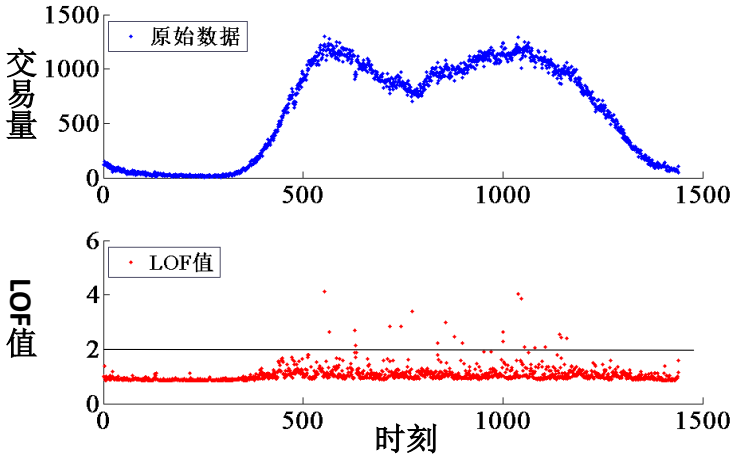

图 22 LOF 阈值划分判断离散点的不足

从上图中我们可以看到，虽然在前 500 分钟时间段也有一部分离群点，但是对于 500-1000 分钟的离群点来说 LOF 值较小。我们如果直接利用阈值划分来判断是否为离群点，会将第一段时间的部分离群点直接忽略。因此我们引入**移动平均+标准差**的阈值划分方法，针对不同时间段的分布特点能够分别进行判断。

- 移动平均法(moving average method)是根据时间序列,逐项推移,依次计算包含一定项数的序时平均数,以此进行预测的方法。

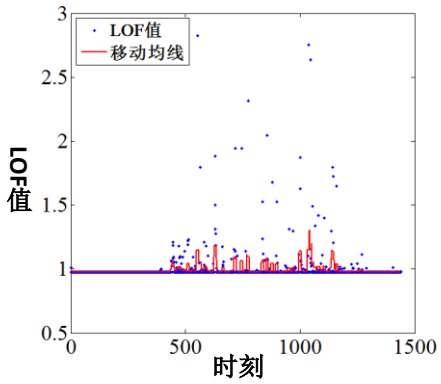

图 23 LOF 移动平均处理后的图像

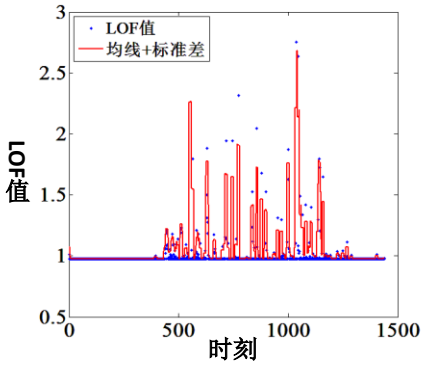

图 24 移动标准差后的图像

我们还需要考虑到，离群有向上离群和向下离群两种情况，在判断交易量是否异常时，我们只考虑交易量突然下降的情况为异常点。为此我们还需要考虑交易量随时间序列变化的图像。我们发现用函数拟合的效果不理想，因此我们还是采用移动均线+标准差来筛选离群点，如下所示：

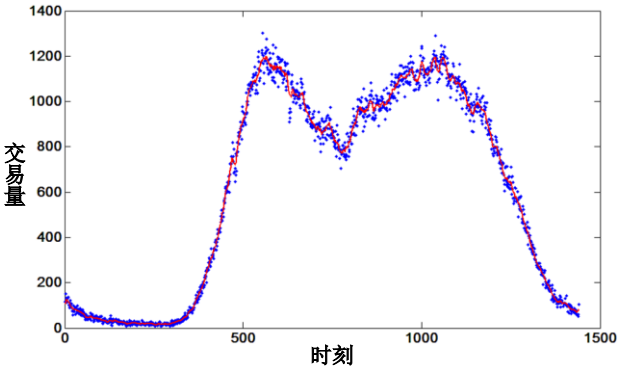

图 25 采取 10 分钟移动平均线图像

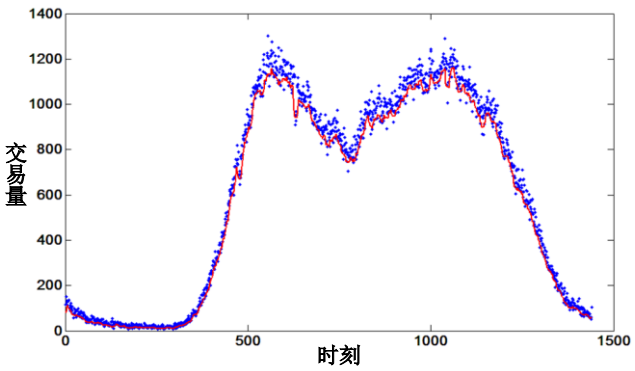

图 26 移动平均线向移动 10 分钟标准差

同时位于 LOF 移动均线标准差偏移图像（图 24）上方以及交易量时间序列移动平均线下方（图 26）的数据点即为疑似异常点。为了更清楚表现算法实现的过程，给出该天筛选过程如下：选取 4 月 18 日当天数据，位于交易量时间序列移动平均线标准差偏移后的曲线下面的点如图 27 红色点所示，结合 LOF 离群因子与交易量移动均线，最终得到的异常点如图 28 所示。

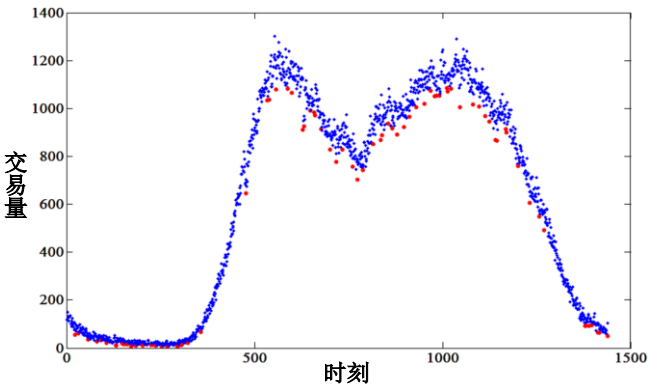

图 27 交易量移动均线偏移后筛选异常点（红色点）

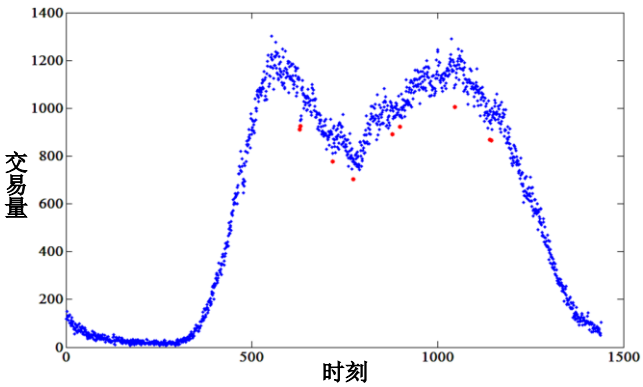

图 28 交易量移动均线与 LOF 结合后筛选的异常点

对于异常点，我们进一步对其预警等级进行划分，操作的方法如下。

我们选择用同一时刻不同日期的数据进一步对疑似异常点进行筛选，区分预警等级。我们首先对不同日期的交易量数据进行标准化，以排除日平均交易量偏低对筛选的影响，标准化方法如下：

$$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)} (i = 0...1440) \tag{6}$$

其中，x_i 为原交易量数值，x_i' 为标准化后的交易量数值，$\min(x)$ 为当天交易量最小值，$\max(x)$ 为当天交易量最大值。对于以上筛选过程，我们做以下解释，便于理解：

对于图 28 被箭头标记的点（第 50 天，528 分钟），我们将其放于图二的标准化以后的 528 分钟不同日期的图中。其中的红线为所有点交易量的平均值向下移动一个标准差所得。我们可以发现，图中标记点在该红线上方，即我们认为该点并非最高等级预警，定其为橙色预警。如果在该线下方，则定为红色预警。

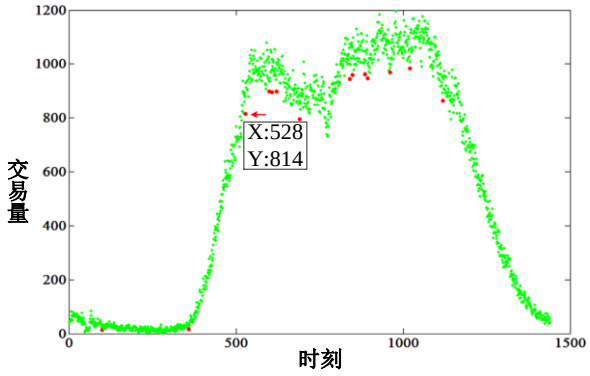

图 29　3 月 23 日的异常点（初筛）

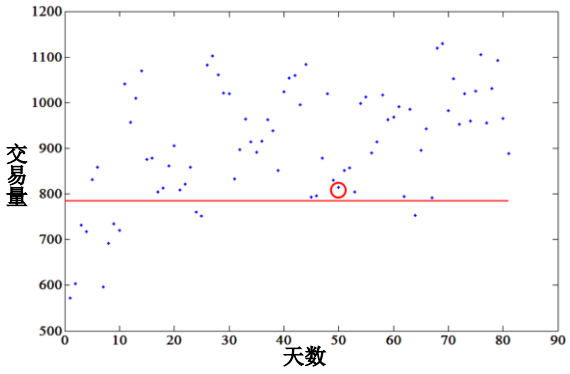

图 30　均线-标准差判断预警等级

为了展现该方案的可行性，我们在三个时间段的每段时间段随机抽取一天，筛选情况如下所示。

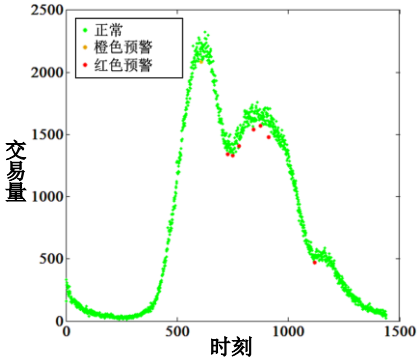

图 31　1 月 27 日异常点判断

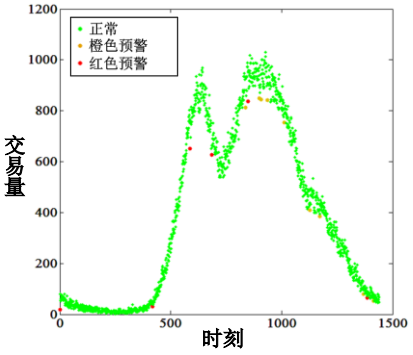

图 32　2 月 1 日异常点判断

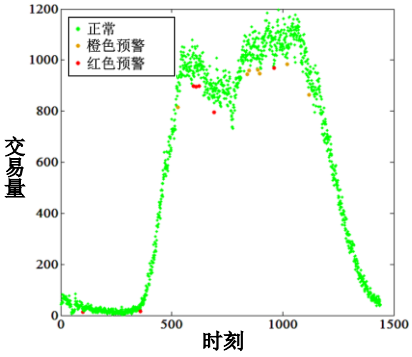

图 33　3 月 13 日异常点判断

如上所示，我们抽取了 1 月 27 日、2 月 1 日、3 月 13 日，对预警情况进行判断，得出的结果较好。

对以上筛选过程进行总结，得出以下流程图：

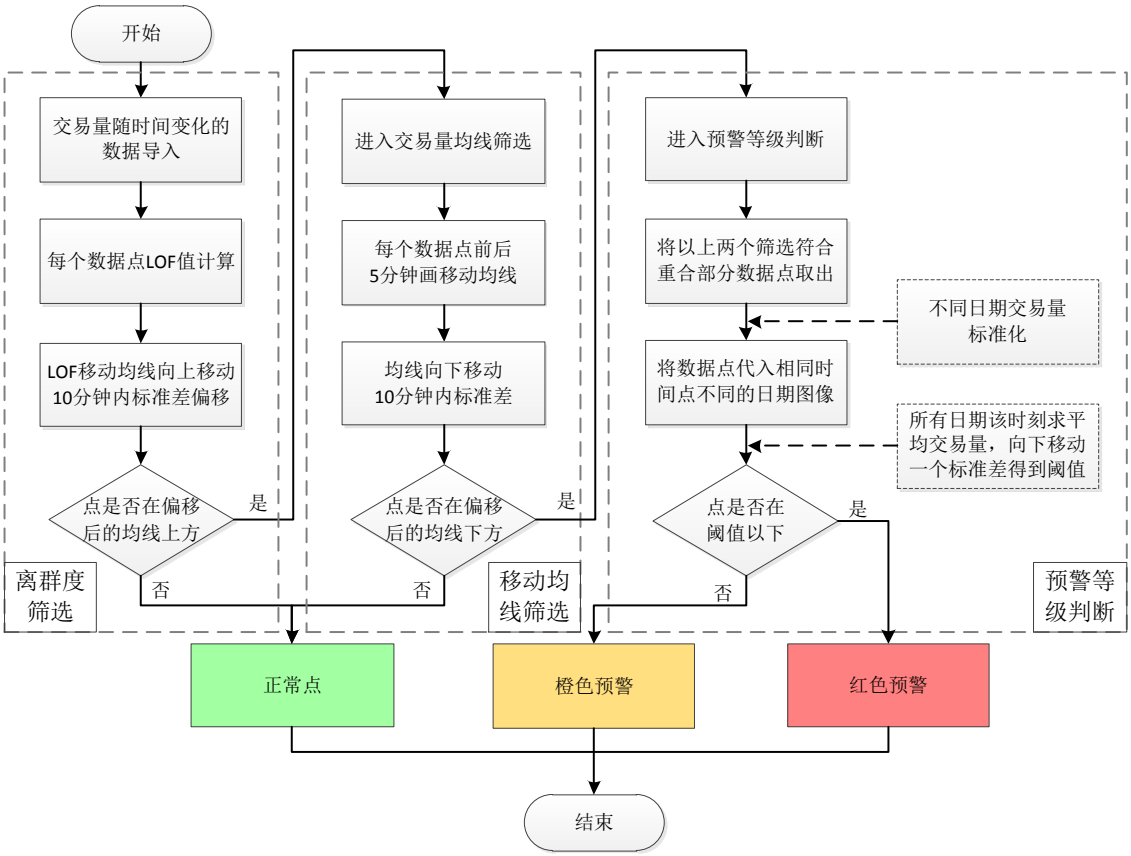

图 34　算法实现过程

通过以上流程，我们在数据点中共检测到红色预警 858 处，橙色预警 751 处。我们将识别出的异常点标记在图上反映每天发生的故障次数如下图所示：

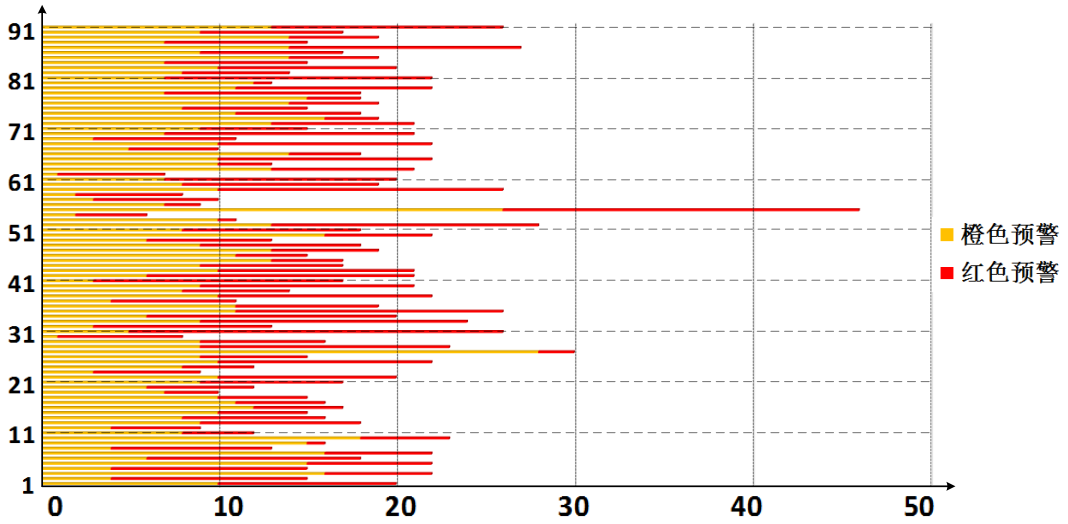

图 35　交易量指标判断的异常点识别结果

其中共发现 27 处数据缺失，也即无交易的情况。

我们对其前后的交易量进行检测判断，前 9 个数据缺失是正常情况，4 月 16 日的 6 时 4 分至 6 时 21 分的 18 分钟数据缺失，我们判断该断时间发生了严重的故障，也即红色预警。数据缺失的情况如下表所示:

表 2 无交易量时刻

月	日	时	分	持续时间
1	28	4	51	1 分钟
1	28	5	2	1 分钟
1	29	4	44	1 分钟
1	29	5	1	1 分钟
1	30	4	48	1 分钟
1	31	4	41	1 分钟
3	19	4	38	2 分钟
3	30	3	28	1 分钟
4	16	6	4	18 分钟

我们进一步研究中思考，判断异常点的目的就是对有需要修理的点进行识别，尽早完成修理工作。而不是所有的异常点都需要修理，如：某一分钟网络卡顿导致交易量指标陡降，但是下一分钟恢复正常，这种情况就不需要修理。我们对于需要修理的点（黑色预警）进一步进行判断，给出下面的判断方案：

设定橙色预警点的权值为g_1 ，红色预警点的权值为g_2（ $g_2 > g_1$），黑色预警得分阈值为C。n分钟内，发生橙色预警的次数为x_1，发生红色预警的次数为x_2，则满足下述条件的被判定为需要修理的点：

$$x_1 \bullet g_1 + x_2 \bullet g_2 \geq C \tag{7}$$

这里的参数我们没办法得知，需要下一步专家确定，这里我们给出一种我们假定的参数来测试效果。设$g_1 = 3$，$g_2 = 4$，$C = 8$，间隔$n = 10$，x_1与x_2分别为 10 分钟间隔内发生橙色、红色预警的次数。即当满足以下条件时，异常点被判定为需要修理：

$$x_1 \bullet 3 + x_2 \bullet 4 \geq 8, \left(x_1 = 0,1,2... , x_2 = 0,1,2...\right) \tag{8}$$

在我们的参数设定情况下，1609 处交易量异常点中有 29 处需要修理，比例约为 1.8%，我们列出这部分数据如下所示：

表 3 需要修理的异常点（共 29 处）

时间段	月	日	时	分
第一段 1.23-1.27	1	25	1	10
	1	25	12	26
	1	26	19	18
	1	27	4	37
	1	27	12	5
第二段 1.28-2.1	无			
第三段 2.2-4.23	2	5	12	15
	2	5	15	10
	2	8	15	46
	2	11	12	39
	2	11	15	28
	2	13	14	47
	2	16	20	7
	2	19	12	1
	2	21	17	1
	2	23	12	15
	2	24	14	49
	2	25	12	21
	3	3	11	35
	3	5	14	8
	3	15	18	35

	3	18	10	12
	3	18	13	12
	3	20	14	8
	3	20	14	55
	3	25	8	56
	3	28	9	17
	3	31	17	33
	4	10	15	2
	4	11	18	47
	4	19	19	47
	4	23	19	47

（2） 成功率与响应时间故障点检测

对于成功率与响应时间故障点检测，我们已经在第一问详细给出了聚类与决策树的判断方案，并且对决策树结果进行了检验。

算法执行流程如下所示：

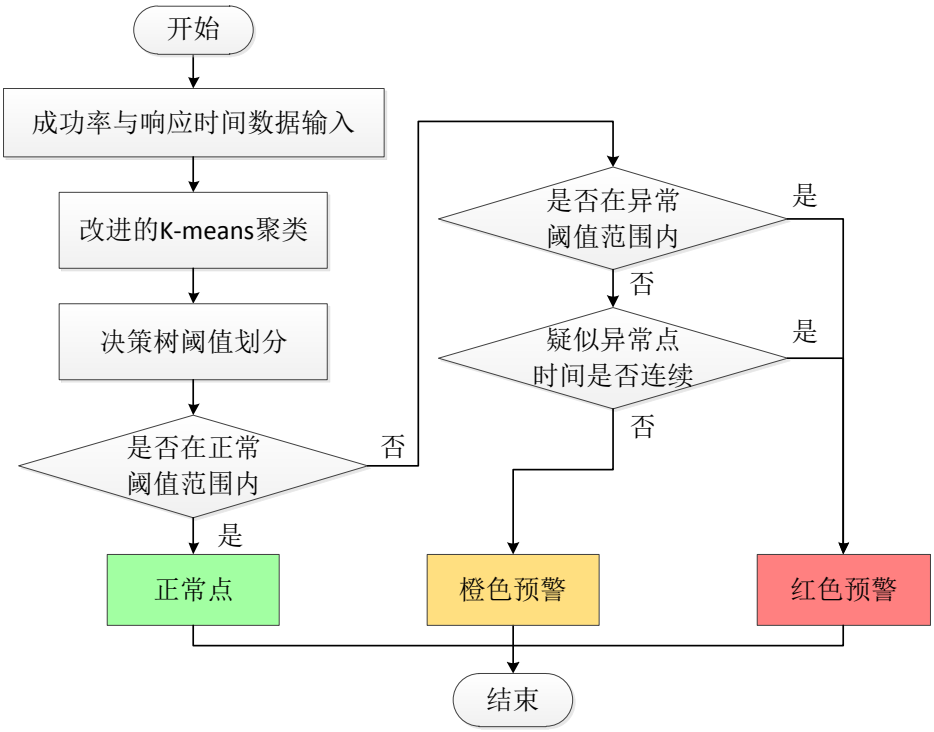

图 36 成功率与响应时间算法流程

对于上文的疑似异常点，我们用异常点的持续时间来判断其警报等级。对疑似异常点用表格展示如下：

表 4 疑似异常点数据

时间段	月	日	时	分	是否连续
第一段 1.23-1.27	1	23	3	39	否
	1	24	4	12	否
	1	25	1	9	是
	1	25	1	10	
	1	27	4	37	否
第二段 1.28-2.1	1	28	4	47	否
	1	28	22	44	否
第三段 2.2-4.23	2	9	2	17	否
	2	9	2	19	是
	2	9	2	20	
	2	9	2	27	是
	2	9	2	28	

	2	9	2	29	是
	2	9	2	30	
	3	23	1	1	是
	3	23	1	2	
	4	14	17	33	是
	4	14	17	34	
	4	14	17	35	
	4	16	4	2	否
	4	16	4	47	否

以上算法结合成功率与响应时间检测到的故障点预警如下：

- 红色预警18+13=31处，其中 18 处直接通过决策树划分区域得到，13 处通过筛选疑似的异常点是否有时间连续性得到；
- 橙色预警 8 处，过筛选疑似的异常点是否有时间连续性得到。

题中给出了故障可能发生的原因与对应的指标如下：

成功率指标：分行侧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；对应下图③区域。

响应时间指标：数据中心后端处理系统异常（如操作系统 CPU 负荷过大）引起交易处理缓慢；对应下图②区域。

成功率与响应时间指标共同作用：数据中心后端处理系统应用进程异常。对应下图①区域。

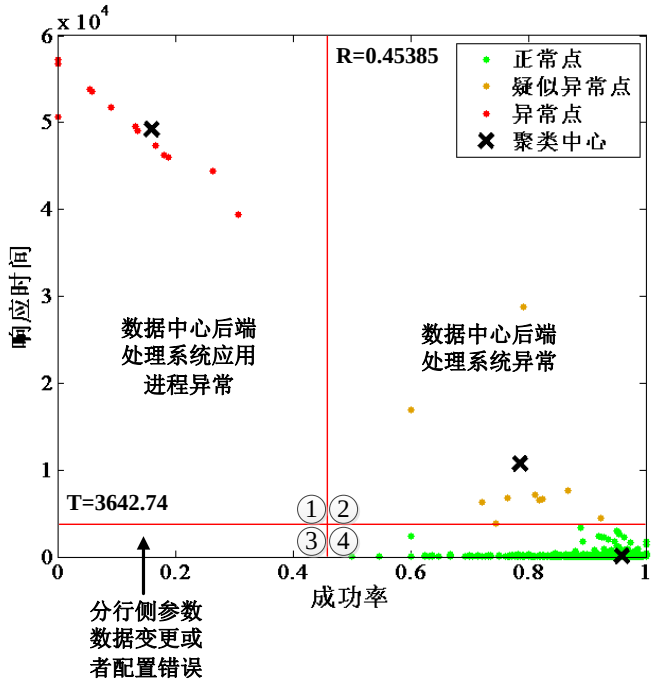

图 37 故障类型判断

对于需要维修的故障点，不同于交易量维修的故障点判断，成功率与响应时间红色预警的点都是需要维修的点，即有 31 处需要维修。

（3） 综合指标检测

结合交易量、响应时间、成功率三者，我们建立故障点判断的最终模型描述流程图如下图所示。

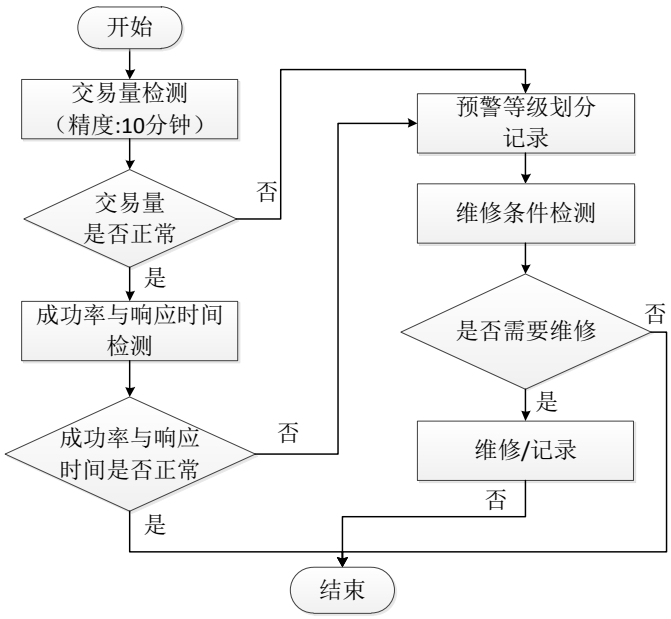

图 38 综合指标故障点检验流程

我们根据以上流程编写程序，结合三个指标判断故障，寻找到橙色预警点 759 处，其中，交易量橙色预警 751 处，成功率与响应时间橙色预警 8 处；寻找到红色预警点 889 处，其中，交易量红色预警 858 处，成功率与响应时间红色预警 31 处。最终寻找出综合指标下 62 处需要维修的异常点，合并在一段时间内的点，一共有 37 个需要修理的点，给出需要维修的点如下：

表 5 综合指标判断的需要维修的点

时间段	月	日	时	分
第一段 1.23-1.27	1	25	1	10
	1	25	12	26
	1	26	19	18
	1	27	4	37
	1	27	12	5
第二段 1.28-2.1	无			
第三段 2.2-4.23	2	5	12	15
	2	5	15	10
	2	8	15	46
	2	9	2	19
	2	9	2	27
	2	11	12	39
	2	11	15	28
	2	13	14	47
	2	16	20	7
	2	19	12	1
	2	21	17	1
	2	23	12	15
	2	24	14	49
	2	25	12	21
	3	3	11	35
	3	5	14	8
	3	15	18	35
	3	18	10	12
	3	18	13	12
	3	20	14	8
	3	20	14	55
	3	23	0	48

	3	25	8	56
	3	28	9	17
	3	31	17	33
	4	10	15	2
	4	11	18	47
	4	14	17	33
	4	16	4	1
	4	16	6	0
	4	19	19	47
	4	23	19	47

即第二问故障检测的目标达成。

（4） 提前发现重大故障方案

在缺失数据中，我们发现，在 4 月 16 日的 6 时 4 分至 6 时 23 分长达 18 分钟数据缺失，可以说这是一次较为严重的故障或是系统维护。我们可以在这天的数据进行分析，查看是否能提前预警这种重大故障，从而尽可能避免损失。

按照我们的预警等级，我们发现当天的 4 时 1 分、2 分、3 分连续出现三次红色预警；同时在当天 6 时 0 分、1 分、2 分、3 分出现连续四个红色预警。像这样一天里连续出现的红色预警情况，应该引起重视，可能会有重大的事故发生。

我们对想法进行验证，如 3 月 23 日的 0 时 48 分至 1 时 0 分共有 13 分钟的数据点被检测到 13 处红色预警，这也是一起较为严重的故障。我们观察其之前的数据，发现在 3 月 22 日的 23 时 49 分至 3 月 23 日 0 时 31 分共出现三次红色预警，高出正常水平较多。由于本文给的数据量有限，且重大故障情况不常出现，这里也只给出了我们的推测，在连续多个红色预警出现时应该引起重视。如果检测为需要修理的点应该及时进行检查与维修，避免出现更大面积的损失。

4. 问题三模型建立与求解

我们可以增加采集的数据，来使得我们在特征参数提取以及故障检测的时候各个参数的正常区间划分更为可靠，提升故障异常发现率同时减少误报率。同时增加某些指标理论上能将无监督机器学习转换为有监督机器学习，进一步增加模型的可靠性。增加的数据有两方面：一方面增加数据的维度，进一步提高不同故障的检测精度，如网络负载率指标等；另一方面增加专家参考意见采集相关数据，如：何时需要修理的C阈值，一部分已知故障点数据等，能够根据这些数据进行训练，提高模型表现。下面对这两种采集的数据分别进行说明。

（1） 增加数据维度

我们考虑增加以下数据维度来改进我们的模型：

- 每分钟交易金额指标

若每分钟交易金额增大，每笔交易的平均时间会有一定的增加，间接的导致交易量的下降。若发现交易量的突降的同时交易金额的突增，可以不进行交易量的预警，降低交易量指标的误报率。交易金额在交易量绝对数量较大时对交易量影响较大，在 ATM 闲时影响会比较小。我们可以在交易量与交易金额两者之间建立综合指标，然后再采用

我们上文的建模方式进行模型改进，建立交易量+交易金额的综合模型。

- 网络负载率指标

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。基于网络负载率指标，我们可以建立响应时间+网络负载率的综合模型。

（2） 专家参考意见及故障样本相关数据

- 专家参考意见

本文在识别维修点时，对于红色预警、橙色预警的得分判断以及 C 阈值的选定都是未知的。本文做了一些参数的假设，取得了较好的参考结果。但是关于这些参数的选定，还需要专家商量后决定，能够使得模型对需要维修的故障点检测更加完善。同时，专家对于响应时间应该有一定的自己的判断，比如专家能对响应时间的大致阈值进行划分，即响应时间达到多大为异常点，这样也能对我们阈值选择起到一定的帮助作用。

- 故障样本

故障样本对于从无监督转换为有监督的机器学习尤其关键。本文由于缺少故障样本，全程采用无监督机器学习的算法。只能够人为判断一些特别明显的异常点（成功率极低以及响应时间极高），对于一些边界值无法判断。如果有了一部分训练样本，我们会换一种方式处理异常点检验的问题。对于无监督转换为有监督的机器学习，我们将详细讲解其过程。

（3） 无监督机器学习到有监督机器学习

给出的异常点样本数量不需要太多，该样本数据也可以通过部分已有的数据询问专家来判断是否是异常点获得样本集。如下图所示，我们假设类别 1 为待检测的数据点，类别 2 为经过采集以后得到的样本点集合（假设采取了 100 个异常点样本）。

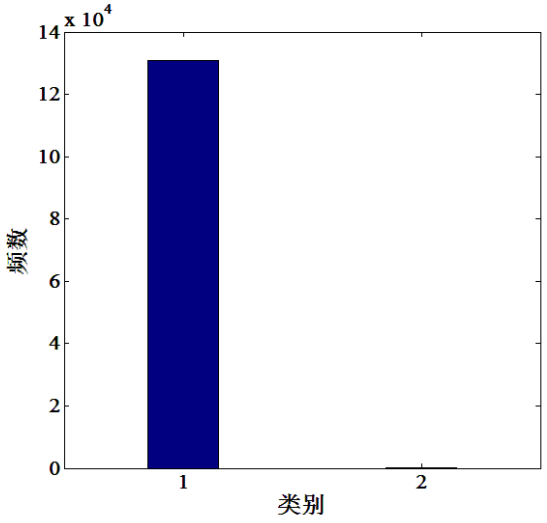

图 39 假定样本数与待检测集合比值图

首先，我们来解决样本不均衡的问题。一般情况下，有两种解决办法：下采样与过采样。这里采用过采样来完成数据的处理。即我们基于取得的 100 个异常点样本生成一部分新的异常点。具体的生成算法如下：

SMOTE 算法

1）对于类别 2（少数样本类）中每一个样本 x，以欧氏距离为标准计算它到少数类样本集中所有样本的距离，得到其 k 近邻。

2）根据样本不平衡的比例设置一个采样比例以确定采样倍率 N，对于每一个少数类样本 x，从其 k 近邻中随机选择若干个样本，假设选择的近邻为 x_n。

3）对于每一个随机选出的近邻 x_n，分别于原样本按照如下的公式构建新的样本。

$$x_{new} = x + \text{rand}(0,1) \times (\tilde{x} - x) \tag{9}$$

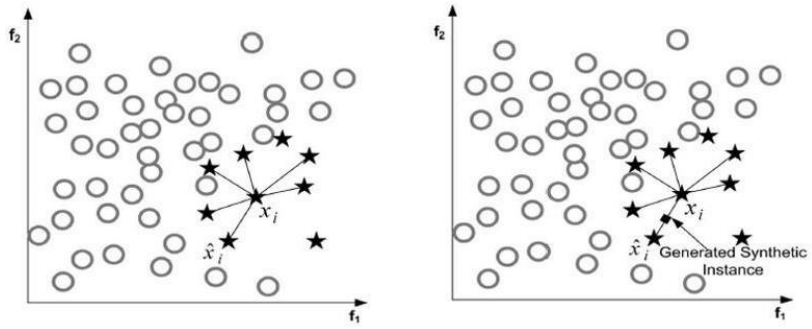

图 40 SMOTE 算法流程

对于逻辑回归的模型这里不再具体介绍，在有监督的机器学习中，很重要的一部分是参数的调节。我们使用交叉验证对不同的参数进行测试，进而提高模型的准确度。我们会采用 recall 来计算模型的好坏，也就是说那些异常的样本我们的检测到了多少，正常样本误判为异常样本的有多少。有监督机器学习流程如下：

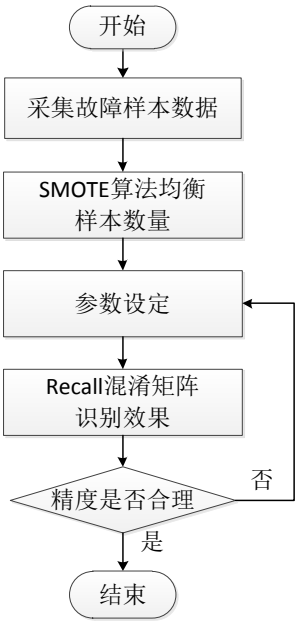

图 41 有监督学习流程

我们并不是传统意义上直接拿样本数据作为训练集建立模型，而是仍然在无监督学习的情况下，利用样本集对模型精度进行完善。这样一方面使得模型操作简便，另一方面也保证了模型精度。

六、 模型的评价

1. 模型的优点

- 本文对时间段、工作日与双休日数据进行了较细致的研究，并且对日期进行分段处理，使得每段日期段内的判定独立，增加模型的精度。
- 对于成功率与响应时间指标，本文利用 80%的数据聚类后的决策树划定阈值，同时利用剩余 20%的数据对结果进行检验，得出结论：划分阈值表现较好。
- 对于交易量指标指标，我们采用了类似化学提纯过程的三道筛选流程，以 LOF 离群因子为主导，结合交易量时间序列的移动均线偏移以及同一时刻不同日期的均线/标准差筛选，最终找到了异常点，算法表现较为稳定。
- 本文提出了橙色预警、红色预警、需要修理的异常点三个预警等级，能够较好地区分异常点的故障严重程度，便于下一步处理。
- 本文最后提出了一种增加采集数据使无监督机器学习转换为有监督机器学习的方法，能使得模型进一步被完善。

2. 模型的缺点与改进

- 在没有专家指导意见的情况下，为了展现模型的效果，本文假设了一定量的参数。如在判断需要修理的预警点时，假设了得分阈值、时间间隔等。这些参数的设定还需要专家的意见，使得模型更加可靠。
- 决策树阈值划分对于边界值的处理表现不是特别理想，即很难区分阈值附件的正常数据与异常数据。可以结合专家意见对如响应时间的阈值进行调整，使得模型的精度尽可能提高。

七、 模型的推广

1. 离群点检验算法推广

本文采用的离群点检验算法可以广泛地应用在电子商务犯罪和信用卡欺诈的侦查、网络入侵检测、生态系统失调检测、公共卫生、医疗和天文学上稀有的未知种类的天体发现等领域中。

本文的离群点检验算法不同于传统的单指标 LOF 离群因子检验，而是设置了三道流程，能够很大程度上减少误判虚警，也可以应用于以上的领域减少虚报，提高模型精度。

2. 改进的聚类+决策树算法推广

在很多领域里，没有异常的样本，通常都是采用无监督的机器学习。本文提到的 80%

数据用来模型建立，20%数据用来检验模型效果的思想对于无监督机器学习的模型检验有一定的启发意义。即在无训练集的情况下能够用自身的数据来检验模型的效果，能很大程度上增加模型的可靠性和结果的可信度。

八、 参考文献

[1] Goldstein M. FastLOF: An Expectation-Maximization based Local Outlier detection algorithm[C]// International Conference on Pattern Recognition. IEEE, 2012:2282-2285.

[2] Yuwono M, Moulton B D, Su S W, et al. Unsupervised machine-learning method for improving the performance of ambulatory fall-detection systems.[J]. Biomedical Engineering Online, 2012, 11(1):9.

[3] Claus M. Zimmermann, Robert S. Bridger. Effects of dialogue design on automatic teller machine (ATM) usability: Transaction times and card loss[J]. Behaviour & Information Technology, 2000, 19(6):441-449.

[4] Yu H K, Lee K W. A Study on the Efficient Operation of Automated Teller Machine(ATM) Maintenance Using Simulation[C]// IEEE International Conference on Emergency Management & Management Sciences. 2011:520-524.

[5] Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: synthetic minority over-sampling technique[J]. Journal of Artificial Intelligence Research, 2002, 16(1):321-357.

[6] Ben-Haim Y, Tom-Tov E. A Streaming Parallel Decision Tree Algorithm.[J]. Journal of Machine Learning Research, 2008, 11(11):849-872.

[7] Aggarwal C C, Yu P S. Outlier detection for high dimensional data[C]// Acm Sigmod International Conference on Management of Data. ACM, 2001:37-46..

[8] Chan P K, Fan W, Prodromidis A L, et al. Distributed Data Mining in Credit Card Fraud Detection[J]. IEEE Intelligent Systems & Their Applications, 1999, 14(6):67-74..

[9] Choi D W, Ahn Y Y, Kim J C, et al. Rate-based traffic scheduling algorithm based on load measurement in ATM network[C]// International Conference on Communication Technology Proceedings. IEEE Xplore, 1998:456-461 vol.1

[10]Roll R. A Mean/Variance Analysis of Tracking Error[J]. Journal of Portfolio Management, 2009, 18(4):13-22.

[11]William H. Kruskal, W. Allen Wallis. Errata: Use of Ranks in One-Criterion Variance Analysis[J]. Journal of the American Statistical Association, 1952, 47(260):583-621.

[12]MM BREUNIG, H-P KRIEGEL. LOF: identifying density-based local outliers[C]. Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston: ACM Press, 2009, 93-104. 39.

九、 附录

附录一 聚类与决策树

```
rate=xlsread('date.xlsx','F2:F131014');
time=xlsread('date.xlsx','G2:G131014');
X=[rate,time];
opts = statset('Display','final');
% [Idx,Ctrs,SumD,D] = kmeans(X,3,'Replicates',3,'Options',opts);
[Idx,Ctrs,SumD,D] = kmeans(X,3,'start','uniform','Replicates',3,'Options',opts);
figure
%画出聚类为 1 的点。X(Idx==1,1),为第一类的样本的第一个坐标；X(Idx==1,2)为第二类的样本的第二个坐标
plot(X(Idx==1,1),X(Idx==1,2),'b.','MarkerSize',14)
hold on
plot(X(Idx==2,1),X(Idx==2,2),'g.','MarkerSize',14)
hold on
plot(X(Idx==3,1),X(Idx==3,2),'r.','MarkerSize',14)
hold on
%绘出聚类中心点,kx 表示是圆形
plot(Ctrs(:,1),Ctrs(:,2),'kx','MarkerSize',14,'LineWidth',4)
plot(Ctrs(:,1),Ctrs(:,2),'kx','MarkerSize',14,'LineWidth',4)
plot(Ctrs(:,1),Ctrs(:,2),'kx','MarkerSize',14,'LineWidth',4)
legend('Cluster 1','Cluster 2','Cluster 3','Centroids','Location','NW')
%%
%决策树
t = classregtree(X,Idx,'names',{'rate' 'time' });
treetype = type(t);
view(t)
```

附录二 LOF 判定

```
function [ lof_judge ] = method_lof( day,range_minute,D_volume,k)
%% lof 方法判断
%    此处显示详细说明
% 输入参数 day--第 i 天
%          range_minute--移动均线取前后分钟数
%          D_volume--交易量按天划分的数据
%          k-lof 中设定的 k 值
% 输出参数 lof_judge--移动平均判断结果
%% 构造移动平均的数据
%加入前一天夜里与后一天凌晨的数据
if day==91          %数据的最后一天，取该天的凌晨代替后一天的凌晨
    A=[D_volume(1440-range_minute+1:1440,day-1);D_volume(:,day);D_volume(1:range_minute,day)];
else if day==1   %数据的第一天，取该天的夜里代替前一天的夜里
A=[D_volume(1440-range_minute+1:1440,day);D_volume(:,day);D_volume(1:range_minute,day)];
    else            %加入前一天夜里与后一天凌晨的数据

A=[D_volume(1440-range_minute+1:1440,day-1);D_volume(:,day);D_volume(1:range_minute,day+1)];
    end
end
%计算 lof 值
[lof ] = LOF(A,k);
for i=1+range_minute:length(lof)-range_minute
    lof_ave(i-range_minute)=mean(lof(i-range_minute:i+range_minute)); %计算平均值
    lof_std(i-range_minute)=std(lof(i-range_minute:i+range_minute));   %计算标准差
end
%去除前一天夜里与后一天凌晨的数据
lof=lof(1+range_minute:length(lof)-range_minute);
lof_judge=zeros(length(lof),1);
for i=1:length(lof)
    if lof(i)>(lof_ave(i)+2*lof_std(i))&&lof(i)>1
        lof_judge(i)=1;
    end
end
%去除前一天夜里与后一天凌晨的数据
A=A(1+range_minute:length(A)-range_minute);
%% 画图
%移动均线
plot(lof,'.')
hold on
plot(lof_ave,'-')
figure
plot(lof,'.')
hold on
```

```
plot(lof_ave+lof_std,'-')
%标记找出的点
figure
for i=1:length(lof_judge)
    if    lof_judge(i)==1
          %record=[record i];
          plot(i,A(i),'r.','MarkerSize',15)
          hold on
    else
          plot(i,A(i),'b.')
          hold on
    end
end
输出 lof 判断的天数
disp('---lof 平均找出---')
sum(lof_judge)
end
```

附录三 LOF 移动均线

```
function [ ave_judge ] = method_ave( day,range_minute,D_volume )
%% 移动均线判断
%    此处显示详细说明
% 输入参数 day--第 i 天
%          range_minute--移动均线取前后分钟数
%          D_volume--交易量按天划分的数据
% 输出参数 ave_judge--移动平均判断结果
%% 构造移动平均的数据
if day==91 %数据的最后一天，取该天的凌晨代替后一天的凌晨
    A=[D_volume(1440-range_minute+1:1440,day-1);D_volume(:,day);D_volume(1:range_minute,day)];
else if day==1 %数据的第一天，取该天的夜里代替前一天的夜里

A=[D_volume(1440-range_minute+1:1440,day);D_volume(:,day);D_volume(1:range_minute,day)];
    else          %加入前一天夜里与后一天凌晨的数据

A=[D_volume(1440-range_minute+1:1440,day-1);D_volume(:,day);D_volume(1:range_minute,day+1)];
    end
end
%
A=[D_volume(1440-range_minute+1:1440,day-1);D_volume(:,day);D_volume(1:range_minute,day+1)];
for i=1+range_minute:length(A)-range_minute
    volume_ave(i-range_minute)=mean(A(i-range_minute:i+range_minute)); %计算平均值
    volume_std(i-range_minute)=std(A(i-range_minute:i+range_minute));  %计算标准差
end
volume_ave_new=volume_ave-volume_std;
%%
%去除前一天夜里与后一天凌晨的数据
A=A(1+range_minute:length(A)-range_minute);
%与移动平均范围内的值比较
ave_judge=zeros(length(A),1);
for i=1:1440
    if i-5<1
        if  all(A(i)<volume_ave_new(1:i+5))
            ave_judge(i)=1;
        end
    else if i+5>1440
            if  all(A(i)<volume_ave_new(i-5:1440))
                ave_judge(i)=1;
            end
        else
            if  all(A(i)<volume_ave_new(i-5:i+5))
                ave_judge(i)=1;
            end
```

```
            end
        end
end
%% 画图
% 移动均线
plot(A,'.')
hold on
plot(volume_ave,'-')
%移动均线向下偏移一个标准差
figure
plot(A,'.')
hold on
plot(volume_ave_new,'-')
%标记出找到的点
figure
for i=1:length(ave_judge)
    if  ave_judge(i)==1
        %record=[record i];
        plot(i,A(i),'r.','MarkerSize',15)
        hold on
    else
        plot(i,A(i),'b.')
        hold on
    end
end
% 输出移动平均判断的天数
disp('---移动平均找出---')
sum(ave_judge)
end
```

附录四 同一时刻不同天筛选

```
function [ final_judge ] = method_time( day,std_volume,all_judge,record,A)
%% 同一时刻不同天的比较
%仅对移动均线与 lof 找出的疑似异常分钟数进行判断！！
% 输入参数 day--第 i 天
%          all_judge--移动均线与 lof 的判断结果
%          std_volume--标准化后交易量按天划分的数据
%           record--移动均线与 lof 找出的疑似异常分钟数
%            A--第 i 天的交易量原始数据
% 输出参数 final_judge--三种方法判断结果
final_judge=all_judge;
%%
if day>10
    for i=1:length(record)
        B=std_volume(record(i),11:91);        %第三段同一时刻的数据（除去前十天）
        ave=mean(B);
        std_=std(B);
        if B(day-10)<ave-std_                 %判断是否低于平均-标准差
            final_judge(record(i))=all_judge(record(i))+1;
        end
    end
else if day>5
        for i=1:length(record)
            B=std_volume(record(i),6:10);     %第二段同一时刻的数据（除去前十天）
            ave=mean(B);
            std_=std(B);
            if B(day-5)<ave-std_              %判断是否低于平均-标准差
                final_judge(record(i))=all_judge(record(i))+1;
            end
        end
    else
        for i=1:length(record)
            B=std_volume(record(i),1:5);      %同一时刻的数据（除去前十天）
            ave=mean(B);
            std_=std(B);
            if B(day)<ave-std_                %判断是否低于平均-标准差
                final_judge(record(i))=all_judge(record(i))+1;
            end
        end
    end
end
%% 画图 标记异常点
figure
for i=1:length(all_judge)
```

```
    if final_judge(i)==3    %三种方法均包含  预警等级最高
        %record=[record i];
        plot(i,A(i),'r.','MarkerSize',15)
        hold on
    else if final_judge(i)==2    %移动均线与 lof 包含
            %record=[record i];
            plot(i,A(i),'b.','MarkerSize',15)
            hold on
        else
            plot(i,A(i),'g.')
            hold on
        end
%       end
end
end
```

附录五 警报等级判定

```
%%
load D_volume
load D_success_rate
load D_work_time
range_minute=5;
k=5;
red_record=[];
orange_record=[];
for i=1:91
    [ red_alert,orange_alert ] = main_judge( i,range_minute,k );
    red_record=[red_record;red_alert];
    orange_record=[orange_record;orange_alert];
end
red_alert=zeros(length(red_record),1);
orange_alert=zeros(length(orange_record),1);
day_count=22;
for i=1:length(red_record)-1
    if 1+day_count>90
        red_alert(i,2)=1+day_count-28-31-31;
        red_alert(i,1)=4;
    else if 1+day_count>59
            red_alert(i,2)=1+day_count-59;
            red_alert(i,1)=3;
        else if 1+day_count>31
                red_alert(i,1)=2;
                red_alert(i,2)=1+day_count-31;
            else
                red_alert(i,1)=1;     %从 1 月 23 日开始
                red_alert(i,2)=1+day_count;
            end
        end
    end
    red_alert(i,3)=floor(red_record(i)./60);
    red_alert(i,4)=mod(red_record(i),60);
    red_alert(i,5)=D_volume(red_record(i),day_count+1-22);
    red_alert(i,6)=D_success_rate(red_record(i),day_count+1-22);
    red_alert(i,7)=D_work_time(red_record(i),day_count+1-22);
    if red_record(i)>red_record(i+1)
        day_count=day_count+1;
    end
end
day_count=22;
for i=1:length(orange_record)
```

```
        if 1+day_count>90
            orange_alert(i,2)=1+day_count-28-31-31;
            orange_alert(i,1)=4;
        else if 1+day_count>59
                orange_alert(i,2)=1+day_count-28-31;
                orange_alert(i,1)=3;
            else if 1+day_count>31
                    orange_alert(i,1)=2;
                    orange_alert(i,2)=1+day_count-31;
                else
                    orange_alert(i,1)=1;
                    orange_alert(i,2)=1+day_count;
                end
            end
        end
        orange_alert(i,3)=floor(orange_record(i)./60);
        orange_alert(i,4)=mod(orange_record(i),60);
        orange_alert(i,5)=D_volume(orange_record(i),day_count+1-22);
        orange_alert(i,6)=D_success_rate(orange_record(i),day_count+1-22);
        orange_alert(i,7)=D_work_time(orange_record(i),day_count+1-22);
        if orange_record(i)>orange_record(i+1)
            day_count=day_count+1;
        end
    end
    xlswrite('red_alert.xlsx',red_alert);
    xlswrite('orange_alert.xlsx',orange_alert);
```

附录六 需要维修的点判定

```
%% 初始化
clc;clear
%% 读入数据
red_sign=xlsread('orange_red_alert.xlsx');
%% 计算标准时间
red_time=red_sign(:,3)*60+red_sign(:,4);%转化为分钟
red_sign(:,10)=0;
[b]=find(red_sign(:,8)==1);%找出红色警告点
[c]=find(red_sign(:,8)==2);%找出橙色警告点
red_sign(b,10)=4;%红色警告点赋值 4
red_sign(c,10)=3;%橙色警告点赋值 3
i=10%时间跨度 10 分钟
diff_r=diff(red_time);%求差分
cou=(diff_r<i).*(diff_r>0);%找到两个时间差小于时间跨度的点
a=find(cou==1);
a2=a+1;%错位向量
counts=sum(cou);%记录出现的次数
a1=cou.*[cou(2:end);0];%判断是否出现三个相邻时间小于时间跨度
e=find(a1==1);
es=diff(e);
e(es==1,2)=1;%对出现三个相邻时间小于时间跨度的位置进行标记
time_poor=red_sign(e(es==1,1)+2,9)-red_sign(e(es==1,1),9);%计算标准时间误差
count_2=find(e(:,2)==1);
count_3=find(time_poor<i);
for j=1:counts
dd(j,1)=sum(red_sign([a(j),a2(j)],10));%求警报值
end
gg=sum(red_sign(e(count_2(count_3)):e(count_2(count_3))+2,10));%计算相邻三个之间的警报数值和
qq=find(a==e(count_2(count_3)));
for u=1:length(qq)
dd(qq(u))=gg(u);%将原先个别未分开的警报值替换
end
z=find(dd>=8);
error_time=red_sign(a2(z),1:4);%求得出错时间表
xlswrite('需维修故障点',error_time)
```