摘要

本文针对生产企业在生产过程中生产安全的问题，要求最大程度的做到保证生产的安全、防范意外风险，而生产过程中产生的数据又能实时的反应可能存在的生产风险，所以我们可以根据数据筛选出风险性异常数据，对此进行量化评价，并且建立风险性预警模型来防范生产过程中可能存在的安全隐患，以达到生产安全的要求。

针对问题一，我们先根据附件一中脱敏处理后的数据，根据数据绘制出利用箱型图筛选出传感器编号 1-100 的所有异常数据，通过 MDS 多维标度法把数据转化为二维数据，再利用 Isolation Forest 异常点检测算法检测出风险性异常数据，由此我们可以判定非风险性异常数据和风险性异常数据。

针对问题二，结合问题一的结果，根据已经求得的异常数据，来量化评价风险性异常数据异常程度。此问题与可看作问题一的后续处理，我们首先构建孤立森林中的随机二叉树，之后对数据进行归一化处理，最后得到的结果为[0,1]，越接近 1 表示异常程度越高，越接近 0 则表示异常程度越低，再利用代码求出最佳转化百分制的函数，筛选出得分最高的 5 个，一一对比后找到最高的 5 个时刻及这 5 个时刻对应的异常传感器编号填入表中。

针对问题三，为了提前发现未来生产过程中可能存在的风险隐患，我们利用 ARIMA 模型结合问题二的处理结果来建立风险性异常预警模型，得出 23:00:00-23:59:59 中四个时间段的的最高异常分值及对应的异常传感器编号，将结果填入表中。

关键词： python　MDS 多维标度法　孤立森林算法　ARIMA 模型

1.问题重述

1.1 背景知识

1. 背景介绍

近年来，全国生产安全事故逐年下降，安全生产状况总体稳定、趋于好转，但形势依然十分严峻，事故总量仍然很大，非法违法生产现象严重，重特大事故多发频发，给人民群众生命财产安全造成重大损失，暴露出一些企业重生产轻安全、安全管理薄弱等突出问题。为进一步加强安全生产工作，全面提高企业安全生产水平，来根据生产过程中产生的数据用以建立模型最大化减少风险事故发生情况。

2. 问题的产生

在企业生产过程中，可能有某些潜在的风险无法被轻易发现，而生产过程中产生的数据能够实时反映潜在的风险，但是这些收集到的生产数据或多或少都会有数据波动，这些数据波动有些是随着外界温度或者产量变化的正常波动，并不能产生安全风险，不需要人为干预；有些异常性波动的出现是生产过程中的不稳定因素造成的，预示着可能存在安全隐患，视为风险性异常，因此生产安全的问题也就由此产生。

1.2 具体问题

1.问题一

根据附件中经过脱敏处理后的数据，筛选出异常数据，并且给出判定非风险性异常数据和风险性异常数据的方法。

2.问题二

结合问题一的处理结果，按百分制进行异常程度评价，找到数据中异常分值最高的 5 个时刻及这 5 个时刻对应的异常传感器编号并填入表中。

3. 问题三

结合问题 2 中给出的风险性异常程度量化评价方法，建立风险性异常预警模

型，预测当日 23:00:00-23:59:59 可能产生的风险性异常。填写每个时间段内的最高异常分值及对应的异常传感器编号。

4.问题四

根据问题 2 和问题 3 中的结果，在 00:00:00-23:59:59 内每隔 30 分钟，用百分制进行安全性评分，并用适当的方法对所给评分的结果进行评价和敏感性分析。

2.问题的分析

2.1 研究现象综述

虽然在近几年来，企业生产安全事故逐年下降，但是事故总量仍然很大，非法违法生产现象严重，重特大事故多发频发，给人民群众生命财产安全造成重大损失，暴露出一些企业重生产轻安全、安全管理薄弱等突出问题。为进一步加强安全生产工作，全面提高企业安全生产水平，来研究企业生产数据以尽可能避免事故发生就是十分必要的。

2.2 对问题的具体分析

1.对问题一的分析

由于附件 1 所给出的已经进行数据脱敏的时间序列数据，因此我们选择利用箱型图对附件 1 中的数据进行异常值筛选，将得到的所有非风险性异常数据和风险性异常数据导出，并且利用 MDS 多维标度法进行数据处理，通过 python 语言来编程处理，转化为二维数据并绘制成图，最后通过异常点检测算法 Isolation Forest 来判定非风险性异常数据和风险性异常数据。

2.对问题二的分析

结合问题一的结果，来构建孤立森林中的随机二叉树，对数据进行归一化处理后，得到的结果为[0,1]，分析处理后越接近 1 表示异常程度越高，越接近 0 则

表示异常程度越低，再将其最佳转化百分制表示，筛选出得分最高的 5 个，对比后找到最高的 5 个时刻及这 5 个时刻对应的异常传感器编号填入表中。

3.对问题三的分析

为了发现生产过程中可能存在的风险隐患，我们利用 SARIMAX 模型结合问题二的处理结果来建立风险性异常预警模型，得出 23:00:00-23:59:59 中四个时间段的的最高异常分值及对应的异常传感器编号，填入表中。

4.对问题四的分析

根据问题 2 和问题 3 中的结果，在 00:00:00-23:59:59 内每隔 30 分钟，用百分制进行安全性评分，并用适当的方法对所给评分的结果进行评价和敏感性分析。

3.模型的假设

1.假设在生产过程中设备不会出现故障

4.名词解释和符号说明

4.1 名词解释

1.箱型图

箱形图（Box-plot）又称为盒须图、盒式图或箱线图，是一种用作显示一组数据分散情况资料的统计图。它主要用于反映原始数据分布的特征，还可以进行多组数据分布特征的比较。

2.多维标度法（MDS）

多维标度法(multidimensional scaling，MDS)是一种在低维空间展示“距离”数据结构的多元数据分析技术,是一种将多维空间的研究对象(样本或变量)简化到低维空间进行定位、分析和归类，同时又保留对象间原始关系的数据分析方法。

3.孤立森林算法

孤立森林算法适用于连续数据的异常检测，并且将异常定义为“容易被孤立的离群点”，也可以理解为分布稀疏且离密度高的群体较远的点，由此可以区分出异常点。

4.2 符号说明

符号	说明
Q_3	箱型图中的上四分位数
X_m	箱型图中的数据均值
Q_1	箱型图中的下四分位数
D_{ij}	原始空间下的距离阵
X_i	空间中第i个点
$H(x)$	叶子节点到根节点的路径长度
$S_{(x,n)}$	记录x在n个样本中的训练数据构成的孤立树的异常指数
Y_{SC}	原始数据中的 Score

5.模型的建立和求解

5.1 问题一的分析和求解

1.对问题一的分析

本问题要求我们为附件一中的数据来进行异常值数据筛选，并且区分出非风险性异常数据和风险性异常数据。我们首先从原始数据中筛选出所有异常数据，在此基础上进行数据处理来转化为二维数据并绘制成图，最后通过异常点检测算法来判定非风险性异常数据和风险性异常数据。

2.对问题一的求解

首先我们需要从原始数据中识别出所有的异常数据，箱型图可以帮我们做到这一点。箱型图（Box-plot）又称盒式图，是一种用作显示一组数据分散情况资料的统计图，图中主要包括六个数据节点，将一组数据从大到小排列，分别计算出他的上边缘，上四分位数Q_3，中位数X_m，下四分位数Q_1，下边缘，和该数据的异常值。箱形图为我们提供了识别异常值的一个标准：异常值被定义为小于Q_1 $-1.5IQR$或大于$Q_3+1.5IQR$的值，我们依据此定义绘制出箱型图并且识别出异常数据。

箱型图的示意图如下图 5.1 所示：

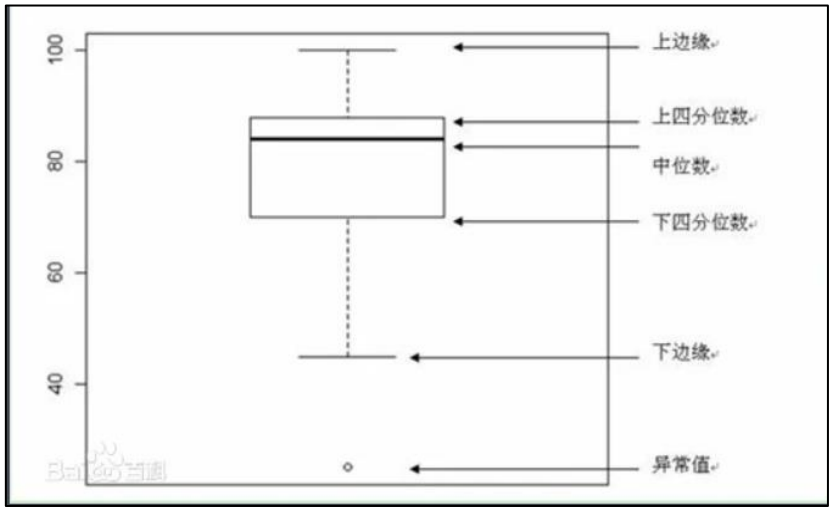

图 5.1 箱型图示意图

由此，我们先对题设所给数据进行异常值筛选，由于数据项编号共有 100 项，列举全部箱型图过于繁琐，我们只展示出编号为 1 的数据的箱型图，后续余项箱型图不再一一列举。

以编号 1 数据为例，绘制出编号一的箱型图如下图 5.2 所示：

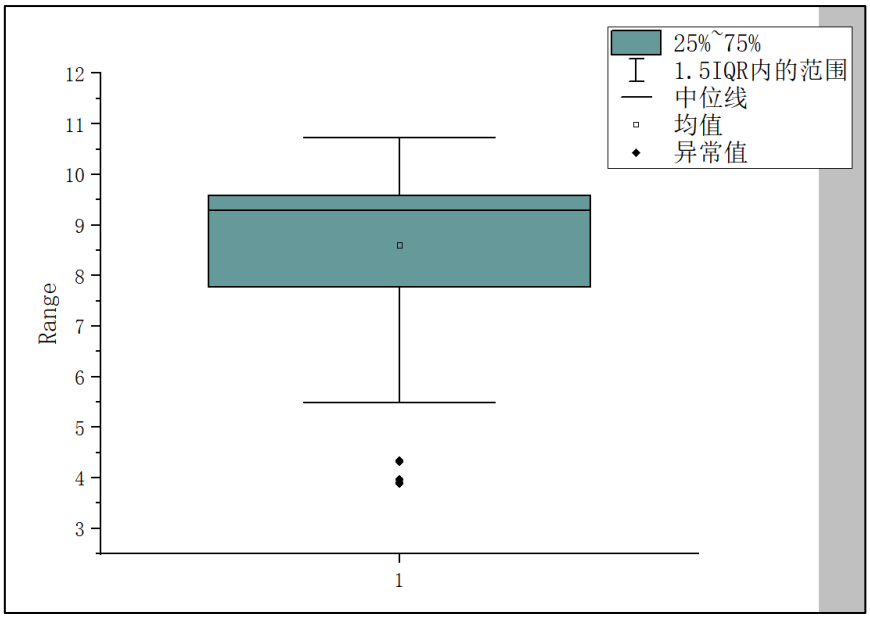

图 5.2 编号 1 数据的箱型图

在图 5.2 中我们可以看出编号为 1 数据的上四分位数Q_3、中位数X_m、下四分位数Q_1、下边缘、和该数据的异常值，我们记录该数据所有的异常值。以此类推，画出编号为 1-100 的所有数据的箱型图，记录所有的异常数据。

在得到所有的异常数据后，第二步我们通过 MDS 多维标度法把所有异常数据数据转化为二维数据。MDS 多维标度法是一种在低维空间展示“距离”数据结构的多元数据分析技术，是一种将多维空间的研究对象(样本或变量) 简化到低维空间进行定位、分析和归类，同时又保留对象间原始关系的数据分析方法；也即当n 个对象中各对对象之间的相似性（或距离）给定时，确定这些对象在低维(欧式)空间中的表示，并使其尽可能与原先的相似性（或距离）“大体匹配”，使得由降维所引起的任何变形达到最小。

由于原始空间下的距离阵和低维空间下的距离阵都采用欧式距离阵，距离阵D为欧式的, 即存在某个正整数p 以及R_p空间的n个点$x_1, x_2, \dots, x_{100}$，使得有：

$$d_{ij}^2 = \left\|\chi_i - x_j\right\|^2, i, j = 1,2,3 \dots 100$$

该公式称为 Classical MDS 算法公式，作用是用来寻找 D 的（拟合）构图。经过 MDS 多维标度法处理后的编号 1-100 的数据如下图所示：

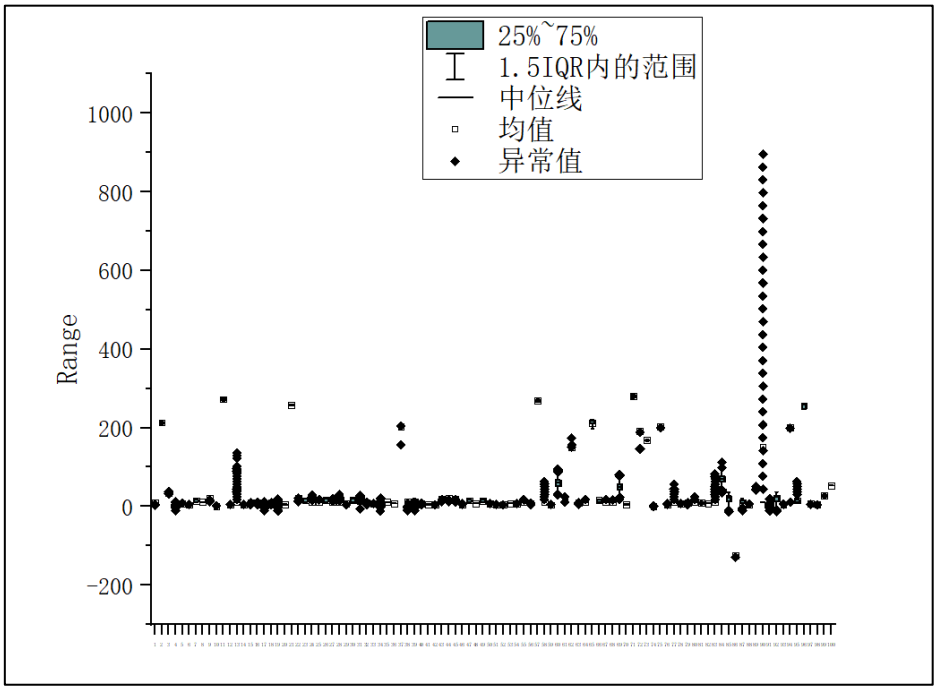

图 5.3 编号 1-100 数据的异常数据

由此我们得到所有的异常数据，最后一步我们选择利用异常点检测算法，也即孤立森林算法（Isolation Forest），用以区分所有异常数据中的风险性异常数据。孤立森林异常点检测算法的目的是找到数据集中和大多数数据不同的数据，一般可用于做数据预处理的时候需要对异常的数据做过滤，防止对归一化等处理的结果，也可用于对没有标记输出的特征数据做筛选，来找出异常的数据。

随机森林算法适用于连续数据（Continuous numerical data）的异常检测，也可用于挖掘数据，将异常定义为“容易被孤立的离群点（more likely to be separated）”，也可理解为分布稀疏且离密度高的群体较远的点。用统计学来解释，在数据空间里面，分布稀疏的区域表示数据发生在此区域的概率很低，因此可以认为落在这些区域里的数据是异常的。

据此，我们可以知道 Isolation Forest 算法通过随机选择一个特征，然后随机选择所选特征的最大值和最小值之间的分割值来“隔离”观察由于递归分区可以表示为一个树结构，分裂需要隔离一个样本的数量相当于从根节点到终止节点路径长度，随机划分为异常生产明显更短的路径，因此，当一个随机森林产生短路径长度，他们极有可能是异常点。

在此之后，我们利用 python 对数据进行编写处理，数据处理代码（部分）如下图 5.4 所示：

```
X_train = a
outliers_fraction = 0.05
n_samples = len(a)
# 构造模型并拟合
clf = IsolationForest(max_samples=n_samples,
                      random_state=rng,
                      contamination=outliers_fraction
                      )
clf.fit(X_train)
# 计算得分并设置阈值
scores_pred = clf.decision_function(X_train)
f = open('Score.txt','w')
```

图 5.4 python 处理数据代码

最终所得 Isolation Forest 示意图如下图 5.5 所示：

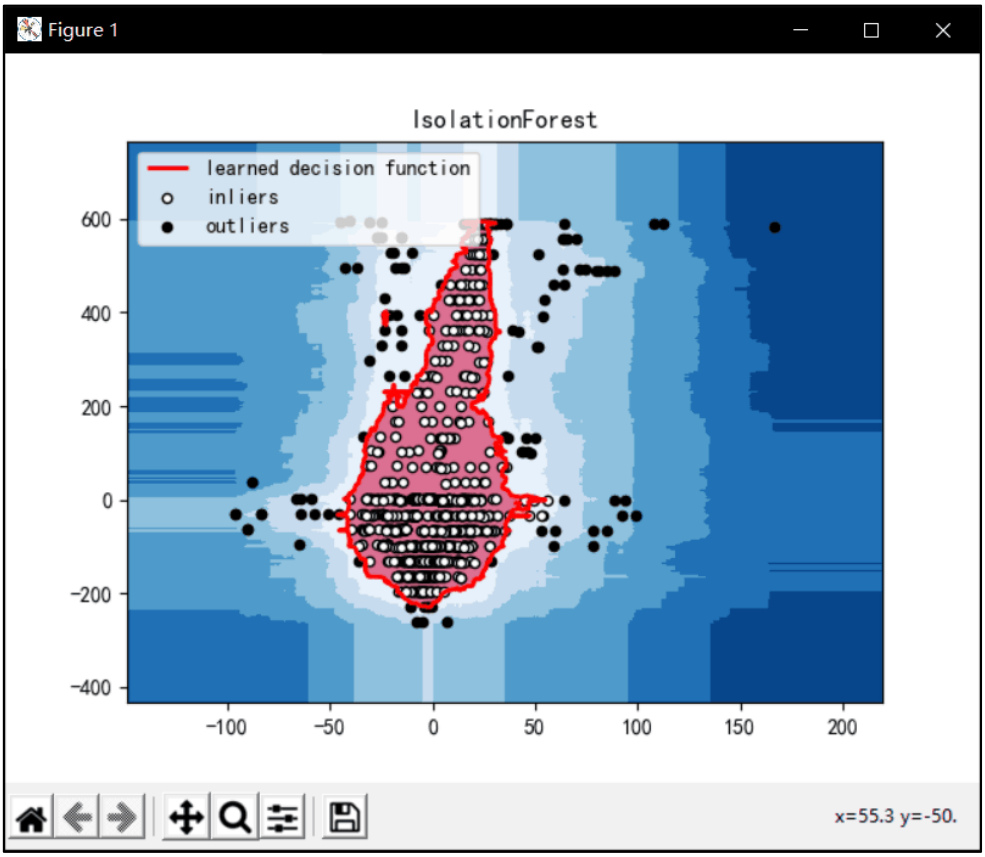

图 5.5 Isolation Forest 示意图

根据图 5.5 所示，红色区域为孤立森林决策树的学习决策函数边界，可以用来区分风险性异常数据和非风险型异常数据，白色的点表示非风险性异常数据,黑色的点表示离群点，也即风险性异常数据。综上，我们得以判定非风险性异常数据和风险性异常数据。

5.2 问题二的分析和求解

1.对问题二的分析

题设需要我们以百分制建立数学模型，给出风险性异常数据异常程度的量化评价方法，构建孤立森林中的随机二叉树，之后对数据进行归一化处理，最后得到的结果为[0,1]，越接近 1 表示异常程度越高，越接近 0 则表示异常程度越低，再利用代码求出最佳转化百分制的函数，筛选出得分最高的 5 个，一一对比后找到最高的 5 个时刻及这 5 个时刻对应的异常传感器编号填入表中。

2.对问题二的求解

为了更好的给出风险性异常数据异常程度的量化评价方法，我们对孤立森林中的孤立树进行分析处理，孤立树（Isolation Tree）是一种随机二叉树，每个节点要么有两个女儿节点，要么就是叶子节点。

孤立树构建好之后，就可以对数据进行导入处理，也就是把数据记录在孤立树上处理，观察数据记录在哪个叶子节点。孤立树能有效检测异常点的原因是异常点一般来说是稀疏的，可以用较小次划分把他们归结到单独的区域中，也即是异常点在孤立树中会很快被划分到叶子节点。

因此我们选择可以把叶子节点到根节点的路径$H(x)$长度来判断一条记录x是否是异常点（也就是根据$H(x)$判断x是否是异常点）。对于一个包含 100 条数据记录的数据集，其构造的树的高度最小值为$log(100)$，最大值为 99，需要的归一化公式如下：

$$S(x,n) = 2^{-\frac{H(x)}{C(n)}}$$

$$C(n) = 2H(n-1) - (2(n-1)/n)$$

$$H(k) = \ln(k) + \tau, \quad \tau = 0.5772156649$$

$S(x,n)$ 就是记录x在n个样本的训练数据构成的孤立树的异常指数，$S(x,n)$取值范围为[0,1]，异常情况的判断如下表 5.1 所示：

表 5.1 孤立树异常程度判断

$H(x) \to 1$	$H(x)$越接近 1 表示异常程度越高
$H(x) \to 0$	$H(x)$越接近 0 表示异常程度越低
$H(x) \to 0.5$	$H(x)$大部分接近 0.5 则说明整个数据集都没有明显的异常值

对数据进行归一化处理后的结果（部分）如下图 5.6 所示：

BU	BV	BW	BX
70	71	Type	Score
4.062556	3.840573	-1	-0.26226
4.076404	3.840573	-1	-0.15244
4.127178	3.840573	-1	-0.14207
4.127178	3.840573	-1	-0.08825
4.127178	3.840573	-1	-0.12513
4.436441	3.840573	-1	-0.11719
4.436441	3.840573	-1	-0.07164
4.159489	3.840573	-1	-0.07686
4.164105	3.840573	-1	-0.03741
4.13641	3.840573	-1	-0.03023
4.13641	3.840573	-1	-0.0305
4.131794	3.840573	-1	-0.04661
4.145642	3.840573	-1	-0.0774

图 5.6 对数据进行归一化处理结果（部分）

在得到数据归一化处理结果后，经过代码处理将其转化为百分制（分值越高

表示异常程度越高），代码如下图 5.7 所示：

```
# print(data['Score'].sort_values())
# data['Score']=(data['Score']-0.73)*(-100)
# data.to_csv('2—完整.csv',encoding='gbk')
# (data.sort_values('Score').iloc[-5:]).to_csv('5条.csv',encoding='gbk')
```

图 5.7 将归一化结果转化为百分制

由图 5.7 可知，Score 中数据转化为百分制（分值越高表示异常程度越高）的最佳公式为：

$$Y = Y_{SC} * （-0.73） * 100$$

转化过的结果如下图 5.8 所示：

	BU	BV	BW
)	71	Type	Score
5	3.840573	-1	99.22601
4	3.840573	-1	88.24358
3	3.840573	-1	87.20745
3	3.840573	-1	81.82526
3	3.840573	-1	85.51333
1	3.840573	-1	84.71936
1	3.840573	-1	80.16352
9	3.840573	-1	80.68571
5	3.840573	-1	76.74122

图 5.8 转化百分制

我们筛选出得分最高的 5 组数据，也即数据中异常分值最高的 5 个，与原数据进行对比，找到这五组数据所对应 5 个时刻及这 5 个时刻对应的异常传感器编号，对比代码（部分）如下图 5.9 所示：

```
ind = []
Name =[]
for i in data.values:
    for j in i:
        j = j.round(3)

        boolNum = (j==oringinData.values)
        for x in range(len(boolNum)):
            for k in range(len(boolNum[x])):
                if True == boolNum[x][k]:
                    Name.append(k)
                    ind.append(x)
```

图 5.9 转化后数据与原数据对比代码（部分）

将最终结果填入表 1，如下表 1 所示：

表 1　问题 2 的结果

	第一高分	第二高分	第三高分	第四高分	第五高分
异常程度得分	99.22601	89.6219	88.24358	87.24175	87.20745
异常时刻编号	12:38:00	12:37:45	22:32:30	22:30:30	6:13:45
异常传感器编号	63	72	90	53	31
异常传感器编号	54	68	61	51	52
异常传感器编号	44	43	14	48	38
异常传感器编号	100	8	95	97	55
异常传感器编号	22	49	16	42	35

5.3 问题三的分析和求解

1.对问题三的分析

本问需要建立模型用以提前发现未来生产过程中可能存在的风险隐患，我们选择利用 ARIMA 模型结合问题二的处理结果来建立风险性异常预警模型，按照模型处理结果得出 23:00:00-23:59:59 中四个时间段的的最高异常分值及对应的异常传感器编号，填入表中。

2.对问题三的求解

问题三需要建立风险性预警模型，我们进行模型对比，发现回归移动平均模型（ARIMA）较为适合进行模型建立，自回归移动平均模型（ARIMA）的目标是描述数据中彼此之间的关系。

首先我们利用代码对数据进行分析处理，对数据的处理代码（部分）如图 5.10 所示：

```
#         results = mod.fit()
#         print('x{}12 - AIC:{}'.format(param_seasonal, results.aic))
#         if results.aic < score_aic:
#             score_aic = results.aic
#             params = param_seasonal, results.aic
#     param_seasonal, results.aic = params
#     print('x{}12 - AIC:{}'.format(param_seasonal, results.aic))
# pdq = [0, 1, 1]
# get_ARIMA_params(train, pdq, m=10)

y_hat_avg = test.copy()
fit1 = sm.tsa.statespace.SARIMAX(train[1],
                                 order=(0, 1, 4),
                                 seasonal_order=(3, 2, 4, 10),
                                 enforce_stationarity=False,
                                 enforce_invertibility=False).fit()
y_hat_avg['SARIMA'] = fit1.predict(5279,5520)
plt.figure(figsize=(16, 8))
plt.plot(train[1], label='Train')
plt.plot(test[1], label='Test')
plt.plot(y_hat_avg['SARIMA'], label='SARIMA')
```

图 5.10 对数据进行分析处理（部分）

6.模型的改进及推广

本文在数据处理的过程中，由于时间及能力有限，在某些环节会有数据差异或者误差等等，可能会造成一定程度的结果失真，但从实际结果来看，也可较为完美的反映所得到的一般规律，可侧面反映出模型的实用价值。

7.参考文献

[1]姜启源、谢金星、叶俊，《数学模型》，北京高等教育出版社，2005 年:　.

[2]司守奎、孙兆亮,《数学建模算法与应用》(第二版)北京国防工业出版社,2016 年:

[3]盛骤、谢式千、潘承毅,《概率论与数理统计》，北京高等教育出版社，2015 年。