|  |  |
| --- | --- |
| 队伍编号 | MCB2403441 |
| 赛道 | B |

基于GM(1,1)模型变体与遗传算法的电商仓储优化研究

摘 要

随着电子商务行业的迅猛发展，电商平台面临日益复杂的库存管理和物流配送挑战。本文旨在通过预测库存量和销量，并优化多仓库系统中的分仓规划，以提高仓储效率和降低成本。

**针对问题一**，构建了**GM(1,1)灰色预测模型**，预测了350种品类在未来三个月的月平均库存量及其每日销量。通过比较三种不同的GM(1,1)模型变体（传统GM(1,1)、新信息GM(1,1)和代谢GM(1,1)），选择最优模型进行预测。category21中7-9月预测结果为1362152、1246384、1140454详情见表 1 。category21中7月1-3日预测结果为16929、16794、16659详情见表 3 。模型的检验结果，平均相对残差为0.056、平均级比偏差为0.024，表明预测模型具有较高的准确性和可靠性。

**针对问题二**，基于预测结果，采用**遗传算法（GA）**设计了一种合理的品类分仓方案，优化了仓容利用率、产能利用率和总仓租成本。通过遗传算法的迭代搜索，找到了一个接近最优的分仓方案，category1-3一品一仓的方案是：category1-warehouse80、category2-warehouse45、category3-warehouse64详情见表 4 ，该方案能够平衡各个仓库的库存量与产能，同时控制总仓租成本，实现了供应链规划的最优化。

**针对问题三**，进一步考虑了品类间的**关联度**，允许每个品类存放于最多三个仓库，但同一品类在不同仓库之间的库存量比例及出库量比例相同。通过遗传算法优化分仓方案，定义**适应度函数**来评估不同方案中的品类关联度，并确保每个品类最多只分配给三个仓库。category2一品多仓方案是warehouses54、warehouses28、warehouses137详情见表 5 ，结果表明，该方案不仅提高了仓容利用率和产能利用率，还最大限度地提升了品类间的协同效应。

本文提出的预测模型和分仓方案有效解决了电商平台的库存管理和分仓规划问题，提高了仓储系统的运营效率。研究结果为电商平台提供了有效的库存管理和分仓规划解决方案，并为类似问题提供了参考借鉴。

**关键词：**GM(1,1)灰色预测模型；遗传算法；库存管理；分仓规划；电子商务

目 录

[一、 问题重述 1](#_Toc3100)

[1.1 问题背景 1](#_Toc10831)

[1.2 问题提出 1](#_Toc12657)

[1.2.1 货量预测模型 1](#_Toc27657)

[1.2.2 品类分仓规划 1](#_Toc11449)

[二、 问题分析 2](#_Toc14321)

[2.1 问题一的分析 2](#_Toc4703)

[2.2 问题二的分析 2](#_Toc8961)

[2.3 问题三的分析 3](#_Toc5324)

[三、 模型假设 3](#_Toc2565)

[四、 符号说明 3](#_Toc27003)

[五、 模型的建立与求解 4](#_Toc28907)

[5.1 问题一模型的建立与求解 4](#_Toc32658)

[5.1.1 数据清洗 4](#_Toc24935)

[5.1.2 模型的建立 6](#_Toc23676)

[5.1.3 精度检验 7](#_Toc27572)

[5.1.4 模型的求解 8](#_Toc22037)

[5.2 问题二模型的建立与求解 12](#_Toc17032)

[5.2.1 模型的建立 12](#_Toc24417)

[5.2.2 模型的求解 14](#_Toc29548)

[5.3 问题三模型的建立与求解 17](#_Toc1010)

[5.3.1 模型的建立 17](#_Toc1821)

[5.3.2 模型的求解 18](#_Toc20312)

[六、 模型的分析与检验 23](#_Toc29085)

[6.1 模型分析 23](#_Toc2398)

[6.1.1 灵敏度分析 23](#_Toc14361)

[6.1.2 误差分析 23](#_Toc13535)

[6.2 模型检验 23](#_Toc8924)

[6.2.1 稳定性检验 23](#_Toc28267)

[七、 模型的评价、改进与推广 24](#_Toc17369)

[7.1 模型的优点 24](#_Toc28969)

[7.2 模型的缺点 24](#_Toc7837)

[7.3 模型的改进 24](#_Toc31624)

[7.4 模型的推广 24](#_Toc25759)

[八、 参考文献 25](#_Toc23235)

# 问题重述

## 问题背景

随着电子商务行业的迅猛发展，电商平台面临着日益复杂的库存管理和物流配送挑战。为了提高仓储效率、降低运营成本，合理的商品存储策略至关重要。本研究聚焦于电商平台的商品品类货量预测及其在多仓库系统中的分仓规划问题。

电商平台通过分布在全国各地的多个仓库来储存大量的商品，这些商品根据其属性（如品类、件型等）进行分类和标记，以便于库存管理。然而，随着商品种类和数量的不断增加，如何合理分配商品到各个仓库成为了一个亟待解决的问题。一方面，需要预测未来的库存量和销量，以提前做好仓储资源的规划；另一方面，需要设计一种有效的品类分仓方案，使得仓储系统的运作更加高效。

具体来说，库存量预测的目标在于估计特定时间内各品类商品所需的总存储量，而销量预测则是预估各品类商品在一定时间内需要出库的总量。合理的预测可以帮助企业提前调整仓储容量，避免因库存不足或过剩而导致的成本浪费。此外，合理的品类分仓方案不仅要考虑到单个仓库的容量和生产能力限制，还要确保整个仓储网络的高效运行。如果将过多的同类商品集中在少数几个仓库中，可能会超出仓库的承载能力，导致履约问题；相反，如果将同类商品分散到多个仓库，则会增加管理成本。

## 问题提出

### 货量预测模型

构建模型预测指定仓储网络中350种品类在未来三个月（7月至9月）的月平均库存量及其每日销量。

### 品类分仓规划

基于第一步的预测结果，设计一个合理的品类分仓方案。该方案需考虑单个仓库的容量和产能限制，并优化整个仓储网络的运营成本。具体而言，需要确定每个品类应存放于哪个仓库，并保证每个仓库的存储利用率和生产利用率处于合理范围。

# 问题分析

## 问题一的分析

针对问题1，首先从附件1和附件2中提取到各品类的历史库存量和各品类的历史销量并进行预处理，接着利用灰色预测模型，考虑到数据可能存在的波动性和趋势性使用三种不同的GM(1,1)变体模型，包括传统GM(1,1)、新信息GM(1,1)和代谢GM(1,1)模型，进行拟合和预测，并通过均方误差（SSE）选择最优模型，最终生成7月至9月的月均库存量和每日销量预测结果。问题1的解题流程如图1所示：

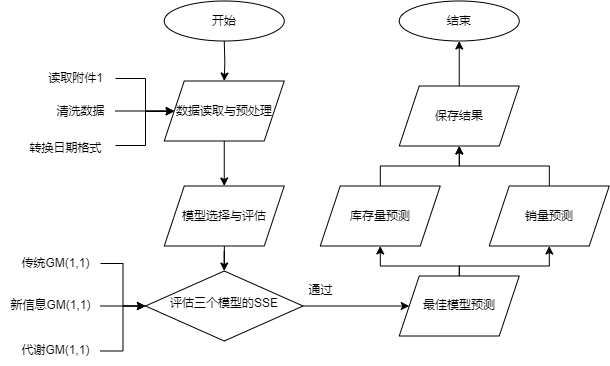


图 1 问题1的解题流程图

## 问题二的分析

针对问题2，需要基于预测的库存量和销量数据，为350个品类合理分配仓库，以优化仓容利用率、产能利用率、总仓租成本等指标。为此，采用遗传算法（GA）构建优化模型，通过定义适应度函数来评估不同分配方案的有效性，并确保每个品类仅分配在一个仓库中。通过遗传算法的迭代搜索，期望找到一个接近最优的分仓方案，该方案能够平衡各个仓库的库存量与产能，同时控制总仓租成本，以实现供应链规划的最优化。问题2的解题流程如图2所示：

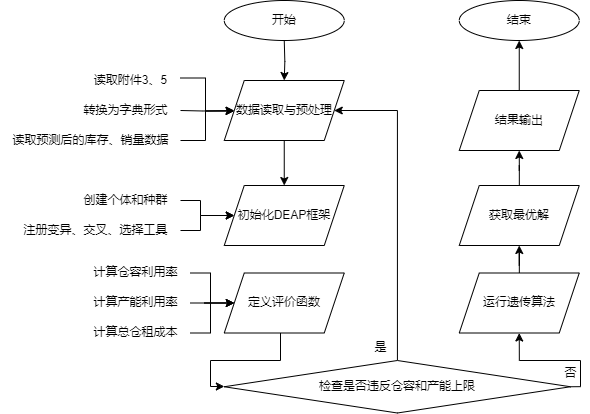


图 2 问题2的解题流程图

## 问题三的分析

针对问题3，需要在考虑品类间关联度的情况下，为350个品类制定新的分仓方案。不同于问题2的一品一仓假设，现在允许每个品类可以存放于最多三个仓库，但同一品类在不同仓库之间的库存量比例及出库量比例相同。的目标是在保证最大品类关联度的同时，还要兼顾仓容利用率、产能利用率和总仓租成本等其他业务指标。为此，首先读取并处理附件中的数据，创建品类到索引的映射表，并构造关联度矩阵来量化品类间的关联关系。然后，使用遗传算法（GA）来优化分仓方案，定义适应度函数来评估不同方案中的品类关联度，并确保每个品类最多只分配给三个仓库。通过遗传算法的迭代优化，期望找到一个既能最大化品类关联度，又能在仓容、产能和成本方面达到平衡的分仓方案。

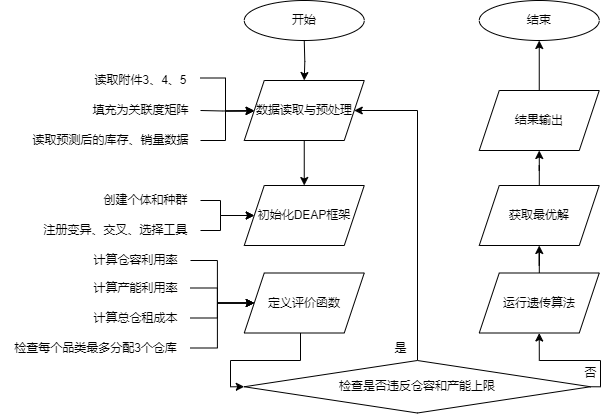


图 3 问题3的解题流程图

# 模型假设

* 不考虑季节性变化对库存量和销量的影响。
* 不考虑因极端天气导致的运输中断或其他物流问题。
* 假定所有仓库的生产设备始终处于良好状态，不会发生意外故障。
* 假定所有操作人员严格按照标准程序操作，不存在人为失误。
* 假定品类间的关联度在整个预测期内保持不变。
* 假定每个仓库的仓容利用率、产能利用率在不同时间段内大致相等。
* 假定仓租日成本在整个预测期内保持固定不变。

# 符号说明

| **分组** | **符号** | **说明** |
| --- | --- | --- |
| 通用符号 |  | 品类集合 |
|  | 仓库集合 |
|  | 时间点 |
|  | 时间区间 |
|  | 品类总数 |
|  | 仓库总数 |
| 品类相关符号 |  | 第个品类 |
|  | 第个品类在时间的预测库存量 |
|  | 第个品类在时间的预测销量 |
|  | 第个品类与第个品类之间的关联度 |
| 仓库相关符号 |  | 第个仓库 |
|  | 第个仓库的仓容上限 |
|  | 第个仓库的产能上限 |
|  | 第个仓库的仓租日成本 |
|  | 第个仓库在时间的实际库存量 |
|  | 第个仓库在时间的实际销量 |
| 分配相关符号 |  | 是否将第个品类分配到第个仓库 |
|  | 第个品类分配的仓库数量 |
|  | 分仓方案 |
| 目标函数相关符号 |  | 目标函数，通常为最小化或最大化的目标 |
|  | 最终优化后的分仓方案 |
| 约束条件相关符号 |  | 仓库的仓容利用率 |
|  | 仓库的产能利用率 |
|  | 仓库的总仓租成本 |

# 模型的建立与求解

## 问题一模型的建立与求解

### 数据清洗

（1）数据读取与初步检查

首先，需要读取数据并进行初步检查，通过读取附件中的文件，获得了库存数据和销量数据，并检查了数据列名称，确保数据列的名称正确无误。

附件1中，每个品类都有九个月的库存量数据，所以，无缺失值

附件2中，绝大部分品类的销量都在180天左右，部分品类的销售数据量较少。

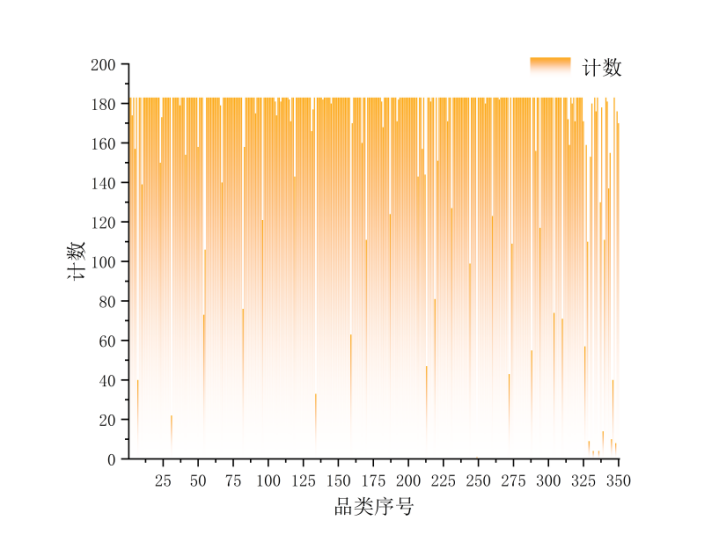
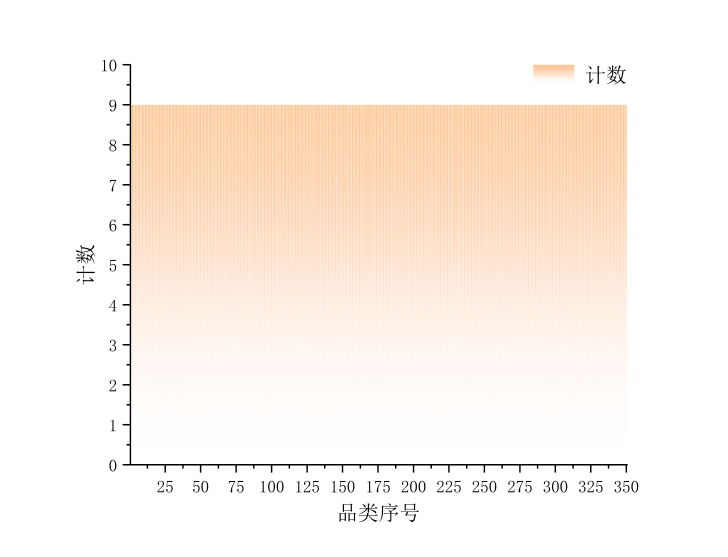


图 4 附件品类计数情况柱状图

由于数据是品类的库存量、销售数据，数据量较少也是一种正常现象，如周期品类（只能在特定的季节，月份或者节日进行销售）等。

（2）数据类型转换

确保数据列的类型正确无误，特别是日期和数值类型。对日期列进行了转换，确保它们是日期格式，以便后续的时间序列分析。

（3）缺失值处理

检查数据集中是否存在缺失值，并进行相应的处理。对于缺失的数据，若在研究中不占主导地位则采用前向填充的方法来填补缺失值，以保持数据的连续性和完整性。

（4）异常值检测与处理

通过绘制箱线图来检测异常值，并对其进行处理。绘制了库存量和销量的箱线图，以识别潜在的异常值。通过计算四分位数（Q1、Q3）和内距（IQR），确定了异常值的边界，并将超出这些边界的值视为异常值。对于这些异常值，选择删除，以避免它们对模型产生负面影响。

由于样品太多，为了不失一般性，选择了品类21、品类84、品类225、品类254。对上述的品类，绘制出如下箱线图：

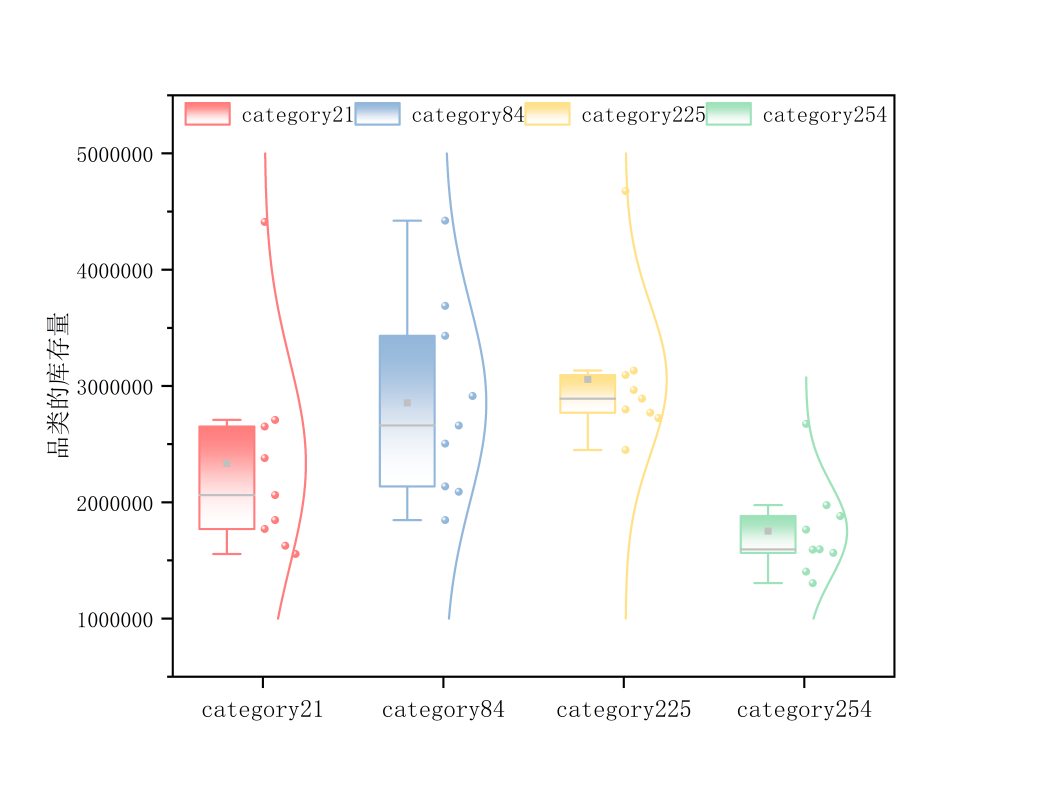


图 5 部分品类的库存量箱线图

从箱线图可以看出，品类21的库存量中位数位于较高位置，整体库存水平较高。箱体相对较窄，库存量的分布较为集中。箱线图的上下须较长，存在一些较高的异常值，库存量的变化较大。

品类84的库存量中位数较高，整体库存水平较高。箱体相对较宽，库存量的分布较为分散。箱线图的上下须较长，存在一些较高的异常值，库存量的变化较大。

品类225的库存量中位数较高，整体库存水平较高。箱体相对较宽，库存量的分布较为分散。箱线图的上下须较长，存在一些较高的异常值，库存量的变化较大。

品类254的库存量中位数较高，整体库存水平较高。箱体相对较宽，库存量的分布较为分散。箱线图的上下须较长，存在一些较高的异常值，库存量的变化较大。

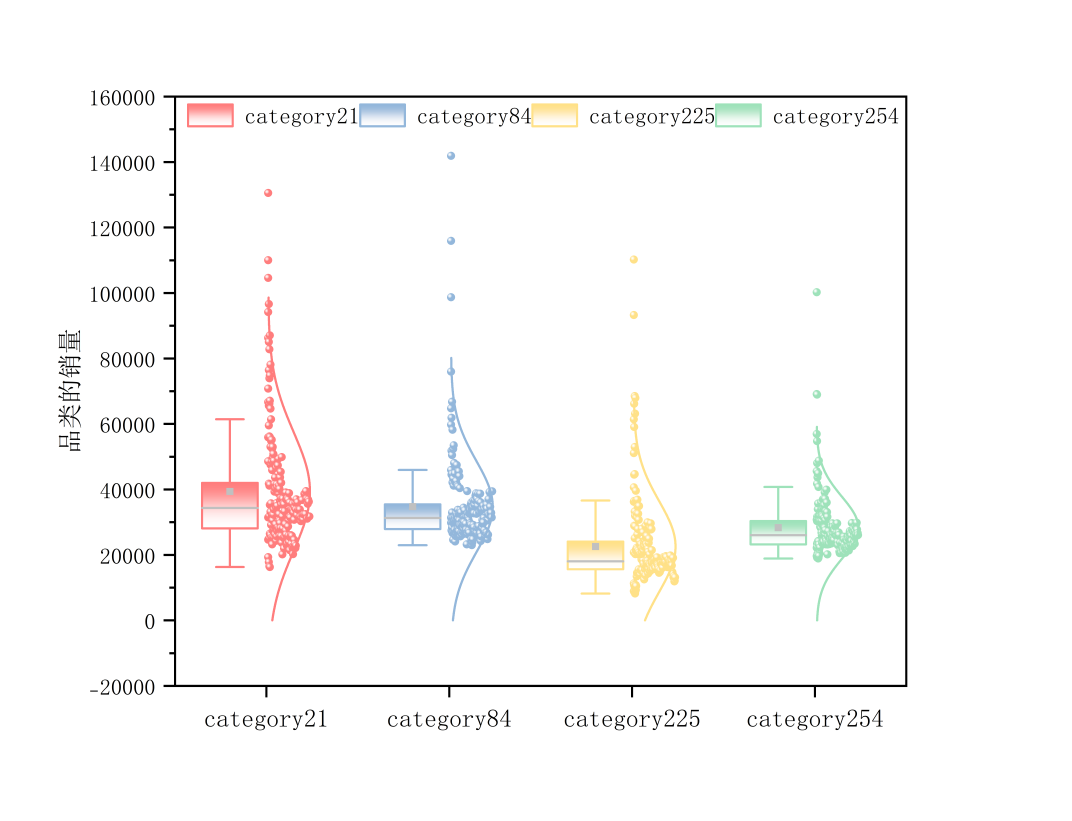


图 6 部分品类的销量箱线图

从箱线图可以看出，品类21的销量中位数位于较高位置，整体销量较高。箱体相对较窄，销量的分布较为集中。箱线图的上下须较短，没有显著的异常值，销量的变化相对稳定。

品类84的销量中位数较高，整体销量较高。箱体相对较宽，销量的分布较为分散。箱线图的上下须较长，存在一些较高的异常值，销量的变化较大。

品类225的销量中位数较高，整体销量较高。箱体相对较窄，销量的分布较为集中。箱线图的上下须较短，没有显著的异常值，销量的变化相对稳定。

品类254的销量中位数较高，整体销量较高。箱体相对较宽，销量的分布较为分散。箱线图的上下须较长，存在一些较高的异常值，销量的变化较大。

完成上述的数据预处理的步骤后，开始进行模型建立。

### 模型的建立

（1）初始GM(1,1)模型

首先，通过对历史库存量和销量数据进行预处理，选择了使用灰色预测模型（GM(1,1)）来预测未来三个月的库存量和销量。灰色预测的基本思想是通过生成有规律的数据序列，建立微分方程模型，通过累加生成（AGO）得到新的数据序列，然后利用最小二乘法估计模型参数。从而实现对系统行为的预测。

利用灰色预测模型的参数，求解微分方程，得到预测模型。然后，通过还原数据累积生成（AGO）得到预测结果。

利用Python软件，编写程序实现上述模型的求解过程。输入历史数据，输出预测的库存量和销量。

灰色微分方程：

|  |  |
| --- | --- |
|  | () |

GM(1,1) 模型：

|  |  |
| --- | --- |
|  | () |

若为参数列，系数向量

|  |  |
| --- | --- |
|  | () |

式中

|  |  |
| --- | --- |
|  | () |

微分方程的解为

|  |  |
| --- | --- |
|  | () |

GM(1,1)模型中的参数有：

|  |  |  |
| --- | --- | --- |
| **参数** | **名称** | **意义** |
|  | 发展系数 | 反映了时间序列数据的增长趋势 |
|  | 灰色作用量 | 衡量某一时间段内的累积影响 |

一般情况下，系统作用量应是外生的或前定的，而GM(1,1)是单序列建模，只用到系统的行为序列(或称输出序列、背景值)，而无外作用序列(或称输入序列、驱动量)。GM(1,1)中的灰色作用量是内涵外延化的具体体现，它的存在，是区别灰色建模与一般输入输出建模(黑建模)的分水岭，也是区分灰色系统观点与黑箱观点的重要标志。通过以上分析，可得下述结论：

（1）当时，GM(1,1)可用于中长期预测；

（2）当时，GM(1,1)可用于短期预测，中长期预测慎用；

（3）当时，用GM(1,1)作短期预测应十分谨慎；

（4）当时，应采用参差修正GM(1,1)模型；

（5）当时，不宜采用 GM(1,1)模型。

（2）变体GM(1,1)模型

1）新陈代谢GM(1,1)模型

可以通过不断更新数据（使用最新预测结果），能够更好地适应数据变化。在某些情况下，能够捕捉到数据变化的趋势，从而提高预测准确性。

模型的预测效果依赖于初始数据质量，前期数据较差会影响后续预测。在数据快速变化的情况下，短期预测可能不准确，因为模型依赖于之前的数据趋势。

2）新信息GM(1,1)模型

通过将新的预测值添加到数据中，模型能够更灵活地适应新信息。在处理时间序列时，可以利用新增数据，提高对未来的预测能力。

更新时可能会引入延迟，影响模型对新信息的反应速度。在信息频繁更新的情况下，模型可能会过拟合历史数据，导致预测不准确。

GM(1,1)模型适合简单、线性数据，而新陈代谢GM(1,1)和新信息GM(1,1)模型则在处理动态变化时表现更佳。选择哪种模型应根据具体数据特性、预测需求和可用信息进行综合考虑。

### 精度检验

灰色预测模型的精度检验一般有两种方法，分别是残差检验和级比偏差检验。

（1）残差检验

绝对残差： 

相对残差： 

平均相对残差： 

如果  则认为GM(1，1)对原数据的拟合达到一般要求。

如果  则认为GM(1，1)对原数据的拟合效果非常不错。

（2）级比偏差检验

首先计算原始数据级比 

再根据预测发展系数  计算级比偏差和平均级比偏差：



如果  则认为模型对原数据的拟合达到一般要求。

如果  则认为模型对原数据的拟合效果非常不错。

### 模型的求解

为了直观的展示上面所选品类的库存量和销量的变化趋势，绘制出时间序列图，如下所示：

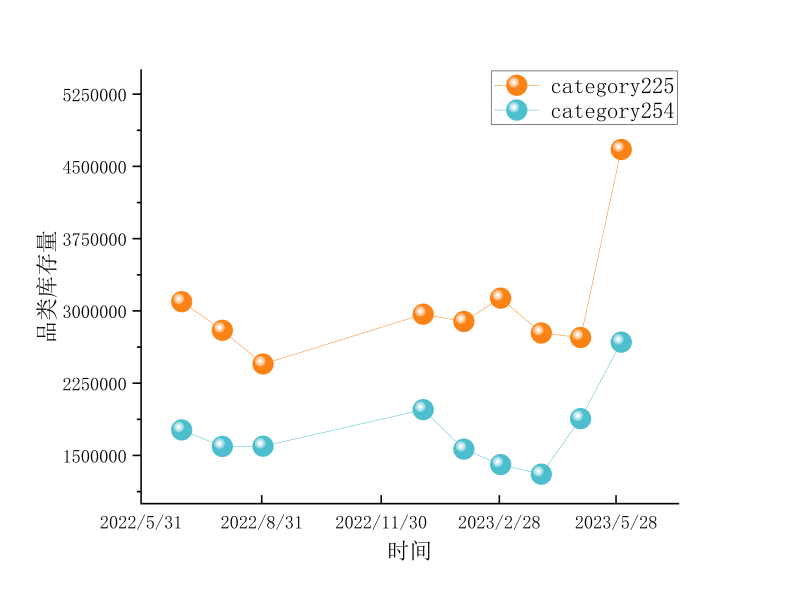
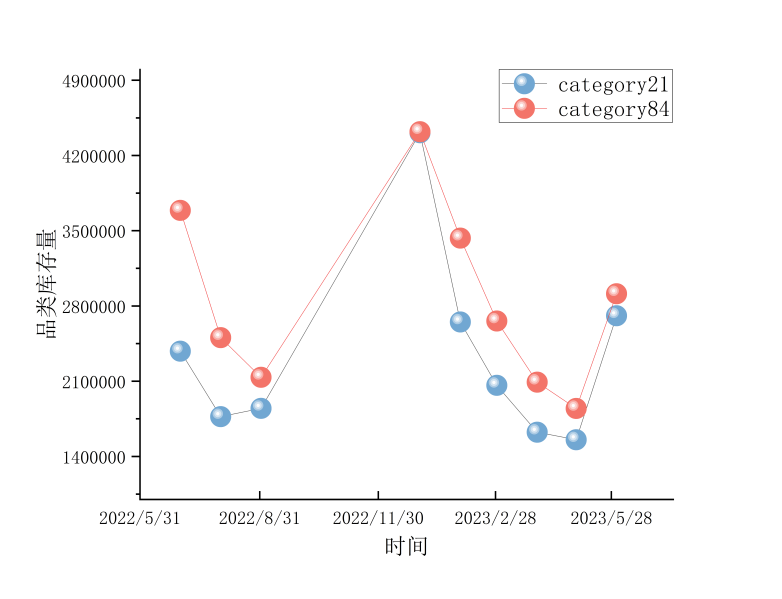


图 7 部分品类库存量的时间序列点线图

从上面的品类库存量的时间序列点线图可以看出，各个品类的库存量，比较平稳，波动幅度不大，说明市场稳定，有助于提高预测精度。

接着，绘制所选品类的销量时序图，如下所示：

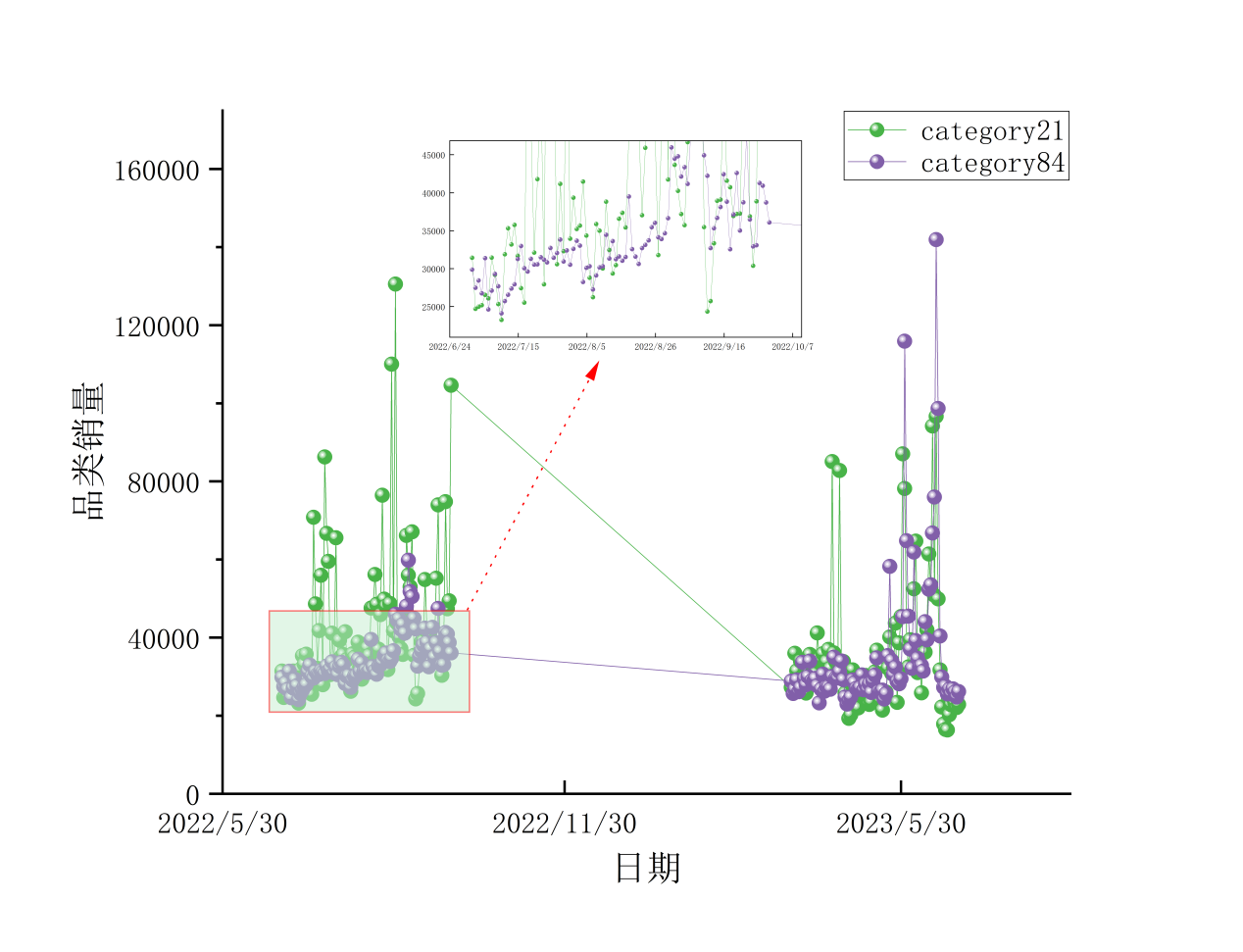


图 8 品类21和84销量时间序列点线图

品类21的销量在2022年夏季之前保持稳定，随后出现增长。2022年末到2023年初销量有所下滑，但之后又恢复增长。

品类84的销量呈现出较为稳定的模式，但偶尔会出现大幅波动。2022年夏季到2023年春季之间，销量经历了先升后降的过程。

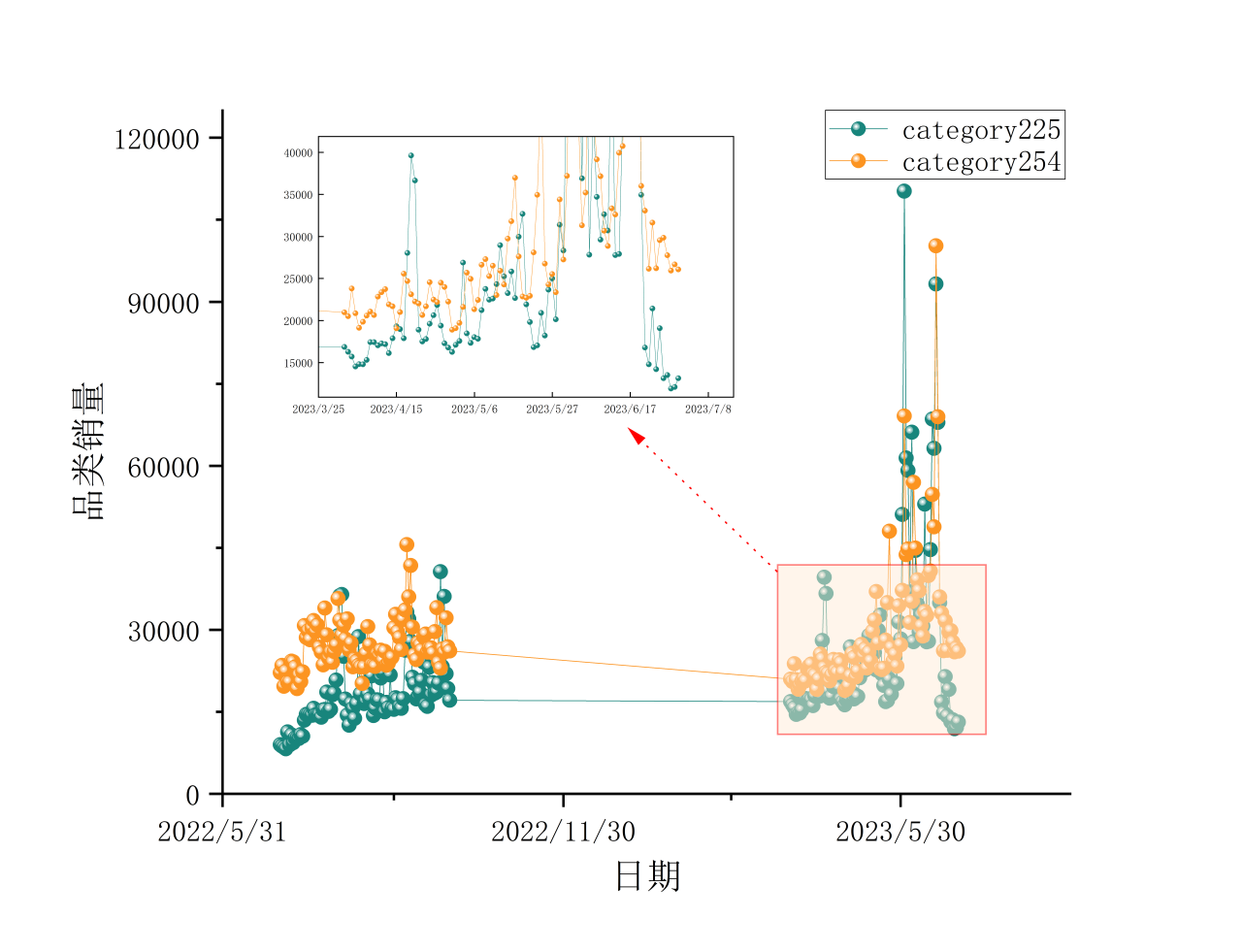
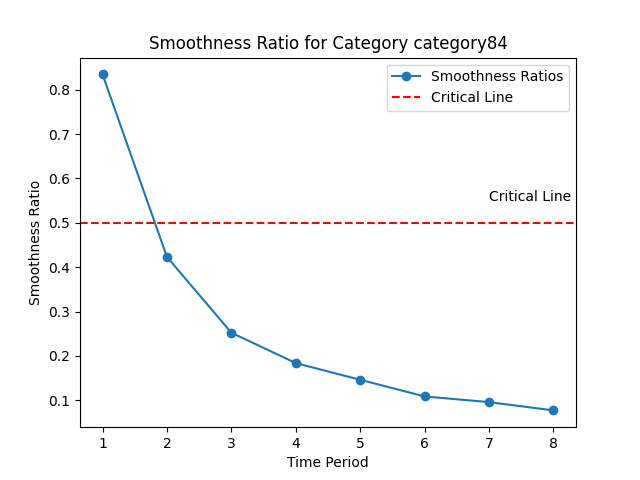
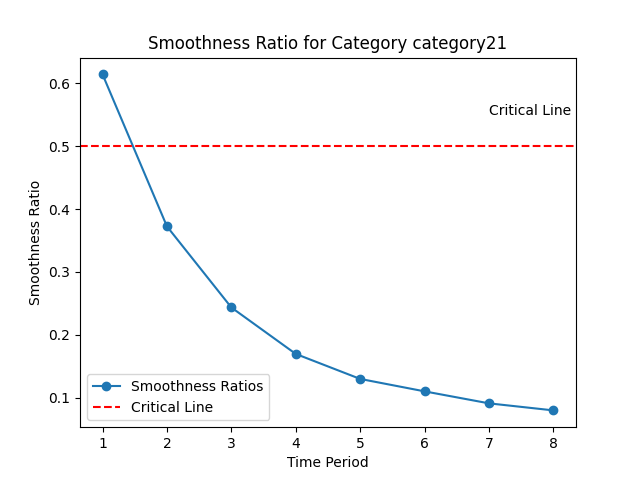


图 9 品类225和254销量时间序列点线图

品类225的销量在2022年秋季和冬季期间表现出较高的销售活动。进入2023年后，销量出现了明显的季节性下降。

品类254的销量在2022年夏季期间非常高，之后有所下降。2023年上半年销量经历了先增后减的趋势。



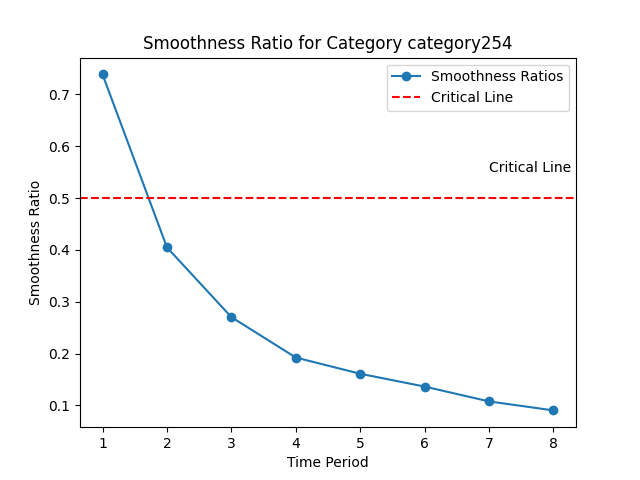
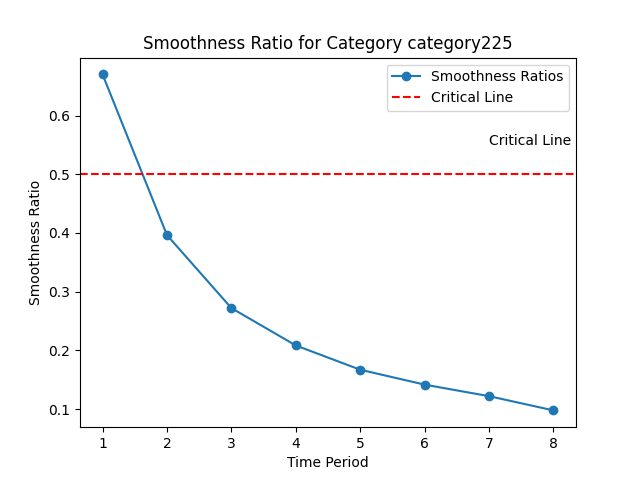


图 10 部分品类的平滑曲线

接下来，通过平滑处理原始数据，得到光滑度曲线，如图 10 所示。平滑处理有助于减少噪声干扰，突出数据的基本趋势。根据计算，光滑比小于0.5的数据占比达到87.5%，这表明数据符合准指数增长规律，适合采用指数平滑法进行预测。得到的预测结果，如下表 1 所示：

表 1 部分品类的库存量预测结果表

|  |  |  |  |
| --- | --- | --- | --- |
| **品类** | **7月库存量** | **8月库存量** | **9月库存量** |
| category21 | 1362152.319 | 1246383.748 | 1140454.283 |
| category84 | 1643421.839 | 1487220.171 | 1380657.105 |
| category225 | 2483365.827 | 2409095.714 | 2337046.767 |
| category254 | 1260919.054 | 1192248.008 | 1127316.864 |

通过对比，可以看出模型很好地捕捉到了数据的增长趋势，并对未来三个月进行了合理的预测。

|  |
| --- |
| **精确检验** |
| 平均相对残差为0.050577 |
| 平均级比偏差为0.024317 |

残差检验的结果表明，该模型对原始数据的拟合程度较好；而级比偏差检验的结果同样显示模型对原始数据的拟合效果良好。综合两项检验结果，可以认为该预测模型具有较高的精度和可靠性。

灰色预测的特性是普遍适用于短期数据，但在问题1的第二部分中，需要预测接下来三个月的日销量，所以，需要借助灰色预测模型中的发展系数。通过计算，得到品类的发展系数，如下表 2 所示：

表 2 模型发展系数表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **品类** | **21** | **21** | **84** | **84** | **225** | **225** | **254** | **254** |
| **所用模型** | **传统** | **传统**  **预测** | **传统** | **代谢**  **预测** | **传统** | **新信息**  **预测** | **传统** | **传统**  **预测** |
| 发展系数 | 0.0973 | 0.0973 | 0.1044 | 0.1044 | 0.0292 | 0.0292 | 0.0600 | 0.0600 |

可见，以上品类的发展系数均大于0.3，故灰色预测模型任然适用。

和上述的流程，步骤一样，通过编写Python代码，得到所选品类的销量预测结果，如下表 3 所示：

表 3 部分品类日销量预测结果表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **品类销量** | | | |
| **category21** | **category84** | **category225** | **category254** |
| 7月1日 | 16929.48755 | 20329.68383 | 7696.677586 | 17482.5028 |
| 7月2日 | 16793.89613 | 20223.78459 | 7620.722709 | 17399.67125 |
| 7月3日 | 16659.39068 | 20118.43699 | 7545.517394 | 17317.23215 |
| 7月4日 | 16525.96251 | 20013.63816 | 7471.054246 | 17235.18365 |
| 7月5日 | 16393.60299 | 19909.38523 | 7397.325939 | 17153.52389 |
| 7月6日 | 16262.30356 | 19805.67536 | 7324.325221 | 17072.25103 |
| 7月7日 | 16132.05573 | 19702.50573 | 7252.044914 | 16991.36324 |
| 7月8日 | 16002.85108 | 19599.87352 | 7180.477907 | 16910.85869 |
| 7月9日 | 15874.68126 | 19497.77593 | 7109.61716 | 16830.73557 |
| 7月10日 | 15747.53797 | 19396.21018 | 7039.455705 | 16750.99208 |
| 7月11日 | 15621.413 | 19295.17349 | 6969.986641 | 16671.6264 |
| 7月12日 | 15496.29819 | 19194.66312 | 6901.203133 | 16592.63676 |
| ... | ... | ... | ... | ... |
| 9月18日 | 8969.085803 | 13456.84733 | 3515.925305 | 12013.26546 |
| 9月19日 | 8897.250719 | 13386.74934 | 3481.228298 | 11956.34698 |
| 9月20日 | 8825.990977 | 13317.0165 | 3446.8737 | 11899.69818 |
| 9月21日 | 8755.301967 | 13247.64691 | 3412.858131 | 11843.31778 |
| 9月22日 | 8685.179118 | 13178.63867 | 3379.178245 | 11787.20452 |
| 9月23日 | 8615.617897 | 13109.9899 | 3345.83073 | 11731.35711 |
| 9月24日 | 8546.613804 | 13041.69872 | 3312.812307 | 11675.77431 |
| 9月25日 | 8478.162378 | 12973.76328 | 3280.119726 | 11620.45485 |
| 9月26日 | 8410.259192 | 12906.18173 | 3247.749773 | 11565.3975 |
| 9月27日 | 8342.899856 | 12838.95221 | 3215.699264 | 11510.60101 |
| 9月28日 | 8276.080013 | 12772.0729 | 3183.965046 | 11456.06414 |
| 9月29日 | 8209.795342 | 12705.54197 | 3152.543999 | 11401.78567 |
| 9月30日 | 8144.041558 | 12639.3576 | 3121.43303 | 11347.76437 |

## 问题二模型的建立与求解

### 模型的建立

#### 5.2.1.1 遗传算法在品类分仓规划中的应用

对于问题2，选择遗传算法来解决品类分仓规划问题。遗传算法是一种模拟自然选择和遗传机制的搜索算法，适用于解决复杂的优化问题。

定义染色体编码方式，每个染色体代表一个可能的分仓方案，染色体的每个基因代表一个品类分配给特定仓库的信息；定义适应度函数，以总仓租成本最小化为目标，同时考虑仓容利用率、产能利用率和品类关联度等指标；定义选择、交叉和变异操作，以模拟自然选择过程中的“优胜劣汰”和遗传变异。

（1）编码表示

在许多问题求解中,编码是遗传算法中首要解决的问题。对算法的性能有很重要的影响。二进制编码是遗传算法中最常用的一种编码方法,它采用最小字符编码原则,编解码操作简单易行,利于交叉、变异操作的实现,也可以采用模式定理对算法进行理论分析。但二进制编码用于多维、高精度数值问题优化时,不能很好地克服连续函数离散化时的映射误差；不能直接反映问题的固有结构,精度不高,并且个体长度大、占用内存多。针对二进制编码存在的不足，人们提出了多种改进方法,比较典型的有以下几种：

a)格雷码编码。为了克服二进制编码在连续函数离散化时存在的不足，提出了用格雷码进行编码的方法,它是二进制编码的变形假设有一个二进制编码为

其对应的格雷码为，则

|  |
| --- |
|  |

格雷码不仅具有二进制编码的一些优点,而且能够提高遗传算法的局部搜索能力。

b)十进制编码。该方法利用十进制编码控制参数,缓解了“组合爆炸”和遗传算法的早熟收敛问题

（2）适应度函数

在遗传算法中,适应度是描述个体性能的主要指标,根据适应度的大小对个体进行优胜劣汰。对于求解有约束的优化问题时,一般采用罚函数方法将目标函数和约束条件建立成一个无约束的优化目标函数：然后再将目标函数作适当处理，建立适合遗传算法的适应度函数。将目标函数转换成适应度函数一般应遵循两个原则：适应度必须非负；优化过程中目标函数的变化方向应与群体进化过程中适应度函数变化方向一致!。在使用遗传算法求解具体问题时,适应度函数的选择对算法的收敛性以及收敛速度的影响较大，针对不同的问题需根据经验或算法来确定相应的参数。

（3）遗传算子

在遗传算法中通过一系列算子来决定后代,算子对当前群体中选定的成员进行重组和变异。

1)选择算子选择操作通过适应度选择优质个体而抛弃劣质个体,体现了“适者生存”的原理。常见的选择操作主要有以下几种：

a)排序选择。对个体适应值取正值或负值以及个体适应度之间的数值差异程度无特殊要求,对群体中的所有个体按其适应度大小进行排序,根据排序来分配各个体被选中的概率。

b)最优个体保存。父代群体中的最优个体直接进入子代群体中。该方法可保证在遗传过程中所得到的个体不会被交叉和变异操作所破坏,它是遗传算法收敛性的一个重要保证条件：它也容易使得局部最优个体不易被淘汰,从而使算法的全局搜索能力变强。

2)交叉算子交叉是指对两个相互交叉的染色体按某种方式相互交换其部分基因,从而形成两个新的个体。它是产生新个体的主要方法,决定了遗传算法的全局搜索能力,在遗传算法中起关键作用。几种常用的适用于二进制编码或实数编码方式的交叉算子如下：

a)单点交叉。在个体编码串中随机设置一个交叉点后在该点相互交换两个配对个体的部分基因。

b)两点交叉。在相互配对的两个个体编码串中随机设置两个交叉点,并交换两个交叉点之间的部分基因。

c)均匀交叉。两个相互配对个体的每一位基因都以相同的概率进行交换，从而形成两个新个体。

d)算术交叉。由两个个体的线性组合而产生出新的个体。

3)变异算子变异是指将个体染色体编码串中的某些基因座上的基因值用该基因座的其他等位基因来替换，从而形成一个新的个体。它是产生新个体的辅助方法，决定了遗传算法的局部搜索能力。变异算子与交叉算子相互配合,可以共同完成对搜索空间的全局搜索和局部搜索，从而使得遗传算法以良好的搜索性能完成最优化问题的寻优过程。在遗传算法中使用变异算子主要有以下两个目的：改善遗传算法的局部搜索能力；维持群体的多样性,防止出现早熟现象。下面是几种常用的变异操作：

a)基本位变异。对个体编码串以变异概率P随机指定某一位或某几位基因进行变异操作。

b)均匀变异(一致变异)。分别用符合某一范围内均匀分布的随机数,以某一较小的概率来替换个体编码串中各个基因座上的原有基因值。均匀变异操作特别适合应用于遗传算法的初期运行阶段,它使得搜索点可以在整个搜索空间内自由地移动,从而可以增加群体的多样性,使算法能够处理更多的模式。

c)二元变异。需要两条染色体参与,通过二元变异操作后生成两条新个体中的各个基因分别取原染色体对应基因值的同或异或。它改变了传统的变异方式,有效地克服了早熟收敛,提高了遗传算法的优化速度

遗传算法运算过程示意图如下所示：

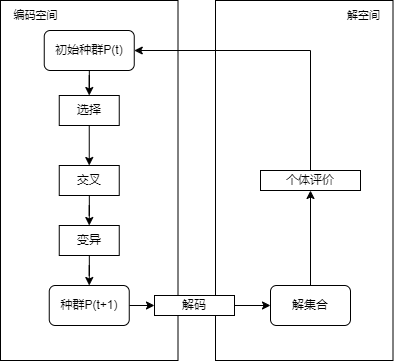


图 11 遗传算法运算过程示意图

#### 5.2.2.2 遗传算法关键技术及其在品类分仓中的实现

决策变量是每个品类分配到哪个仓库的映射。用一个长度为品类数的整数数组表示这个映射，其中数组的每个元素代表对应品类分配到的仓库编号。

目标函数是最小化总成本，包括仓租成本和因仓容或产能不足而产生的额外成本。因此，目标函数可以表示为：

|  |  |
| --- | --- |
|  | () |

其中的符号说明如下所示：

|  |  |
| --- | --- |
| **符号** | **含义** |
|  | 第个品类的预测库存量 |
|  | 第个品类的预测销量 |
|  | 第个仓库的仓容上限 |
|  | 第个仓库的产能上限 |
|  | 第个仓库的日仓租成本 |
|  | 第个品类分配到的仓库编号 |
|  | 违反约束条件的惩罚项 |

约束条件如下所示：

|  |
| --- |
| 每个仓库的总库存量不能超过其仓容上限 |
| 每个仓库的日销售总量不能超过其产能上限 |

### 模型的求解

（1）模型参数设置

|  |  |
| --- | --- |
| **参数** | **初始值** |
| 种群大小POP\_SIZE | 100 |
| 进化代数GENS | 50 |
| 交叉概率CXPB | 0.7 |
| 变异概率MUTPB | 0.2 |

（2）模型求解流程

***Step1*** 初始化参数

设置种群大小、进化代数、交叉概率和变异概率。

***Step2*** 读取并处理数据

读取仓库信息、品类信息、预测库存量和预测销量数据，并转换为字典格式，便于后续处理。

***Step3*** 初始化 DEAP Creator 类

创建适应度函数类 FitnessMin 和个体类 Individual。

***Step4*** 注册工具箱

注册个体生成、种群生成、评估函数、交叉、变异和选择算子。

***Step5*** 初始化种群

随机生成 POP\_SIZE 个个体。

***Step6*** 定义评估函数

评估函数 evaluate 用于计算每个个体的适应度，综合考虑仓容利用率、产能利用率和仓租成本，并检查是否违反了仓容和产能限制。

***Step7*** 迭代进化

使用遗传算法进行迭代进化，包括选择、交叉、变异和替换操作。

***Step8*** 输出结果

获取最优个体并输出结果，将最优分仓方案进行保存。

（3）模型核心代码

|  |
| --- |
| **代码1：遗传算法核心算法——评价函数** |
| def evaluate(individual)：  objective = 0  violations = 0  capacity\_usage = {w： 0 for w in warehouses['id']}  production\_usage = {w： 0 for w in warehouses['id']}   *# 计算目标函数* for cat\_idx, id in enumerate(individual)：  cat = category\_info['category'][cat\_idx]  inv\_pred = category\_inv\_dict.get(cat)  sales\_pred = category\_sales\_dict.get(cat)  cap\_limit = capacity\_dict.get(warehouses['id'][id])  daily\_max = production\_dict.get(warehouses['id'][id])  rent\_cost = rent\_dict.get(warehouses['id'][id])   if inv\_pred is not None and cap\_limit is not None：  objective += inv\_pred / cap\_limit  if sales\_pred is not None and daily\_max is not None：  objective += sales\_pred / daily\_max  if rent\_cost is not None：  objective += rent\_cost  *# 更新容量使用情况* if inv\_pred is not None：  capacity\_usage[warehouses['id'][id]] += inv\_pred  *# 更新生产使用情况* if sales\_pred is not None：  production\_usage[warehouses['id'][id]] += sales\_pred  *# 检查约束条件* for id, used\_cap in capacity\_usage.items()：  if used\_cap > capacity\_dict.get(id)：  violations += 1  for id, used\_prod in production\_usage.items()：  if used\_prod > production\_dict.get(id)：  violations += 1  return objective + violations,  *# 注册评估函数* toolbox.register("evaluate", evaluate) |

（4）模型结果分析

通过上述模型建立的内容，编写Python代码，运行得到如下一品一仓的方案表：

表 4 一品一仓方案表

| **category** | **warehouse** | **category** | **warehouse** | **category** | **warehouse** |
| --- | --- | --- | --- | --- | --- |
| category1 | warehouse80 | category91 | warehouse98 | category207 | warehouse6 |
| category2 | warehouse45 | category94 | warehouse123 | category211 | warehouse13 |
| category3 | warehouse64 | category96 | warehouse92 | category214 | warehouse23 |
| category5 | warehouse59 | category98 | warehouse115 | category215 | warehouse40 |
| category7 | warehouse61 | category99 | warehouse122 | category216 | warehouse21 |
| category8 | warehouse77 | category100 | warehouse127 | category220 | warehouse74 |
| category9 | warehouse106 | category101 | warehouse60 | category223 | warehouse137 |
| category10 | warehouse30 | category103 | warehouse130 | category227 | warehouse32 |
| ... | ... | ... | ... | ... | ... |
| category72 | warehouse65 | category184 | warehouse114 | category302 | warehouse97 |
| category74 | warehouse58 | category185 | warehouse87 | category304 | warehouse124 |
| category75 | warehouse15 | category191 | warehouse105 | category307 | warehouse5 |
| category78 | warehouse71 | category196 | warehouse138 | category309 | warehouse86 |
| category80 | warehouse28 | category198 | warehouse44 | category311 | warehouse43 |
| category81 | warehouse85 | category199 | warehouse27 | category312 | warehouse4 |
| category82 | warehouse57 | category200 | warehouse132 | category313 | warehouse111 |
| category83 | warehouse83 | category201 | warehouse56 | category316 | warehouse49 |
| category86 | warehouse89 | category204 | warehouse31 | category320 | warehouse99 |
| category88 | warehouse26 | category206 | warehouse78 | category321 | warehouse90 |

在上表的基础上，为了更清晰的分辨出各品类与仓库的联系，绘制品类与仓库的网络拓扑图，如下所示：

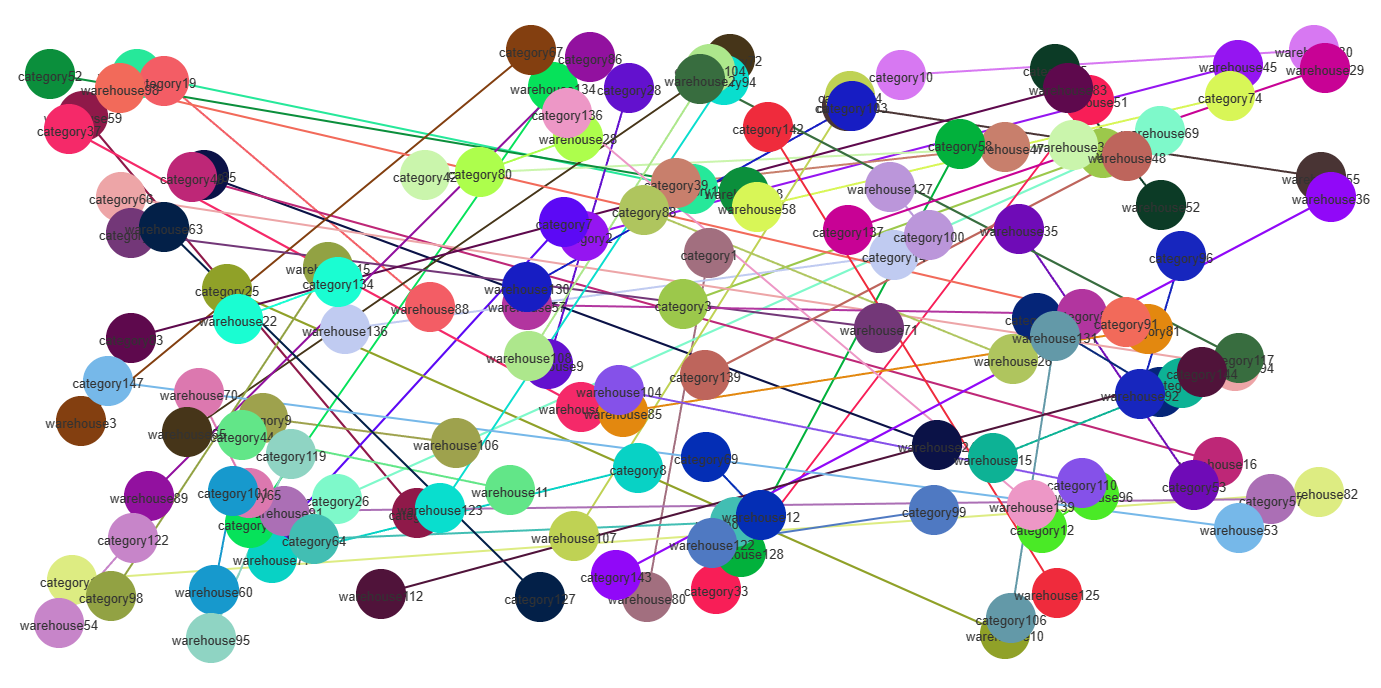


图 12 一品一仓网络拓扑图1

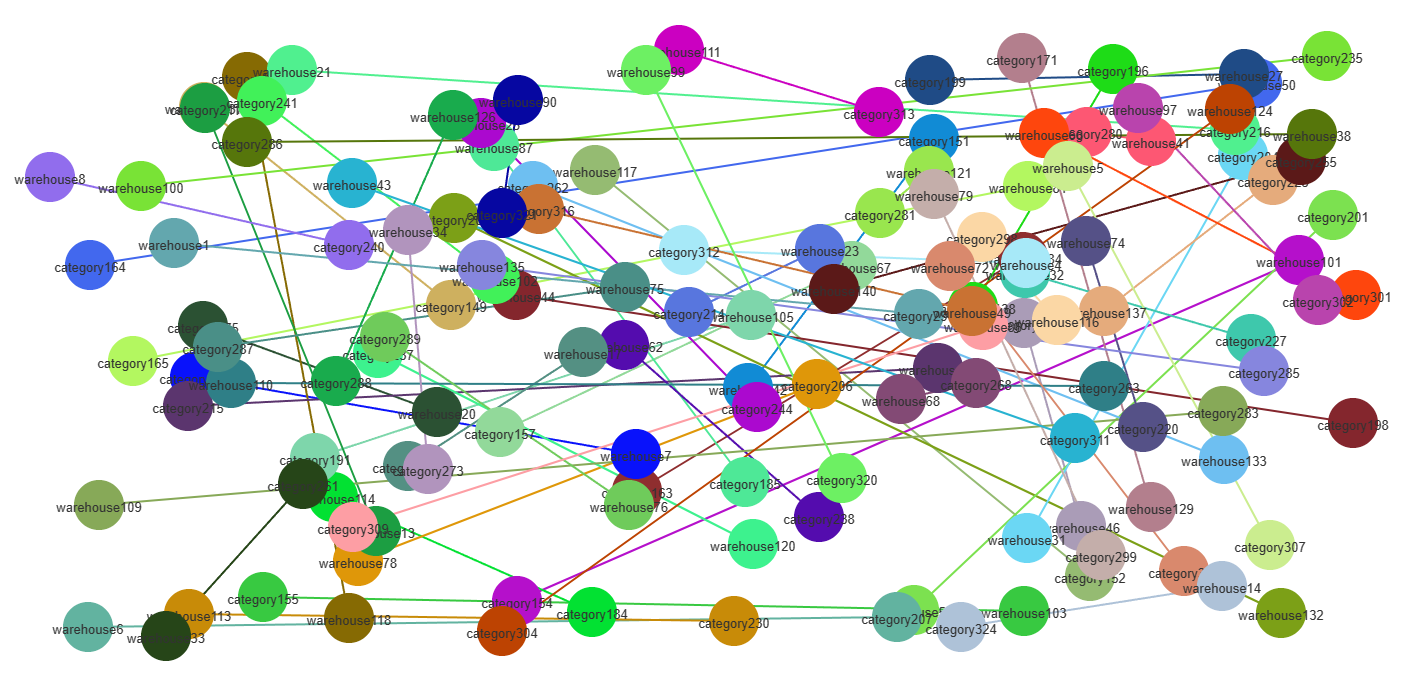


图 13 一品一仓网络拓扑图2

上面的网络拓扑图，能够更直观地理解品类与仓库之间的分配关系

## 问题三模型的建立与求解

### 模型的建立

在问题二使用的模型基础上，问题三要求放开“一品一仓”的假设，允许一个品类存放于多个仓库，但同一品类存放的仓库数量不能超过3个。需要建立一个多目标优化模型，以最大化品类关联度为主要目标，同时考虑仓容利用率、产能利用率、总仓租成本和品类分仓数等其他指标。

在数据处理方面，问题2处理的是单一仓库分配问题，即每个类别对应一个仓库；而问题3不仅处理了类别与仓库的映射关系，还引入了类别之间的关联度矩阵，并允许每个类别分配到多个仓库。编码方式上，问题2中的编码较为简单，每个类别对应一个仓库的索引；而在问题3中，每个类别对应三个仓库的索引，增加了编码的复杂性。适应度函数方面，问题2主要考虑成本和容量利用率等因素；问题3则在此基础上加入了类别间关联度的计算，旨在提高相关类别在分配仓库时的协同效应。约束条件方面，问题2只需考虑单一仓库的约束条件；而问题3不仅需要考虑单个仓库的约束条件，还需确保每个类别分配的仓库数量不超过限定值。综上所述，问题2关注的是单一仓库分配的优化，侧重于成本效益的最优化；而问题3则是一个更为复杂的多仓库分配问题，不仅要考虑成本效益，还需要最大化类别间的关联度，使得相关性强的商品能够尽可能地集中存放。

5.3.1.1 遗传算法关键技术及其在品类分仓中的实现2

决策变量是每个品类分配到哪些仓库的映射。用一个长度为品类数的整数数组表示这个映射，其中数组的每个元素是一个长度为3的列表，表示该品类分配到的3个仓库的编号。

目标函数是最大化品类关联度，同时考虑到其他业务指标。因此，目标函数可以表示为：

|  |  |
| --- | --- |
|  | () |

其中的符号表示如下：

|  |  |
| --- | --- |
| **符号** | **含义** |
|  | 是第个品类与第个品类之间的关联度 |
|  | 是第个品类分配到的第个仓库的编号 |
|  | 是指示函数，如果和相等则返回1，否则返回0 |

约束条件：

|  |
| --- |
| 每个品类存放的仓库数量不超过3个 |
| 每个仓库的库存量不得超过仓容上限 |
| 每个仓库的出库量不得超过产能上限 |
| 同一品类在不同仓库之间的分布比例相同 |
| 同高级品类和件型的品类尽量放在同一个仓库 |

### 模型的求解

问题3是在问题2基础上的补充，我任然使用遗传算法，解决一品多仓的问题。

（1）模型求解流程

***Step1*** 数据加载与预处理

读取数据文件，创建映射表，初始化关联度矩阵，汇总预测数据。

***Step2*** 遗传算法初始化

定义适应度函数、个体结构，注册工具箱函数。

***Step3*** 定义评价函数

计算适应度值，并检查约束条件。

***Step4*** 注册遗传操作

交叉、变异、选择。

***Step5*** 设置GA参数

定义种群大小、迭代次数及概率。

***Step6*** 初始化种群

生成初始种群。

***Step7*** 执行遗传算法

迭代优化，记录统计信息，保留最佳个体。

（2）模型核心代码

|  |
| --- |
| **代码2：遗传算法核心算法——评价函数** |
| *# 定义评价函数* def evaluate(individual)：  individual\_matrix = np.array(individual).reshape(num\_categories, 3) *# 每个品类分配3个仓库   # 计算目标函数* objective = 0  for i in range(num\_categories)：  for j in range(num\_categories)：  if i != j：  objective += assoc\_matrix[i, j] \* (1 if individual\_matrix[i, 0] == individual\_matrix[j, 0] else 0)  objective += assoc\_matrix[i, j] \* (1 if individual\_matrix[i, 1] == individual\_matrix[j, 1] else 0)  objective += assoc\_matrix[i, j] \* (1 if individual\_matrix[i, 2] == individual\_matrix[j, 2] else 0)   *# 检查约束条件* violations = 0  for i in range(num\_categories)：  unique\_warehouses = len(np.unique(individual\_matrix[i]))  if unique\_warehouses > 3：  violations += unique\_warehouses - 3   return objective - violations,   *# 注册评估函数* toolbox.register("evaluate", evaluate) |

（3）模型结果分析

在上述的模型建立的基础上，编写Python代码，实现遗传算法，得到如下一品多仓的方案表：

表 5 一品多仓方案表

|  |  |  |  |
| --- | --- | --- | --- |
| **category** | **warehouse** | **warehouse** | **warehouse** |
| category2 | warehouses54 | warehouses28 | warehouses137 |
| category5 | warehouses109 | warehouses53 | warehouses46 |
| category8 | warehouses109 | warehouses139 | warehouses74 |
| category11 | warehouses9 | warehouses24 | warehouses135 |
| category12 | warehouses66 | warehouses129 | warehouses139 |
| category14 | warehouses120 | warehouses63 | warehouses2 |
| category15 | warehouses102 | warehouses10 | warehouses53 |
| category17 | warehouses88 | warehouses72 | warehouses42 |
| category18 | warehouses132 | warehouses133 | warehouses15 |
| ... | ... | ... | ... |
| category325 | warehouses62 | warehouses66 | warehouses88 |
| category328 | warehouses129 | warehouses76 | warehouses10 |
| category330 | warehouses28 | warehouses108 | warehouses19 |
| category331 | warehouses137 | warehouses58 | warehouses118 |
| category333 | warehouses31 | warehouses69 | warehouses63 |
| category342 | warehouses31 | warehouses67 | warehouses66 |
| category343 | warehouses124 | warehouses60 | warehouses117 |
| category344 | warehouses107 | warehouses28 | warehouses137 |
| category347 | warehouses61 | warehouses28 | warehouses49 |

在上表的基础上，为了更清晰的分辨出各品类与仓库的联系，绘制品类与仓库的网络拓扑图，如下所示：

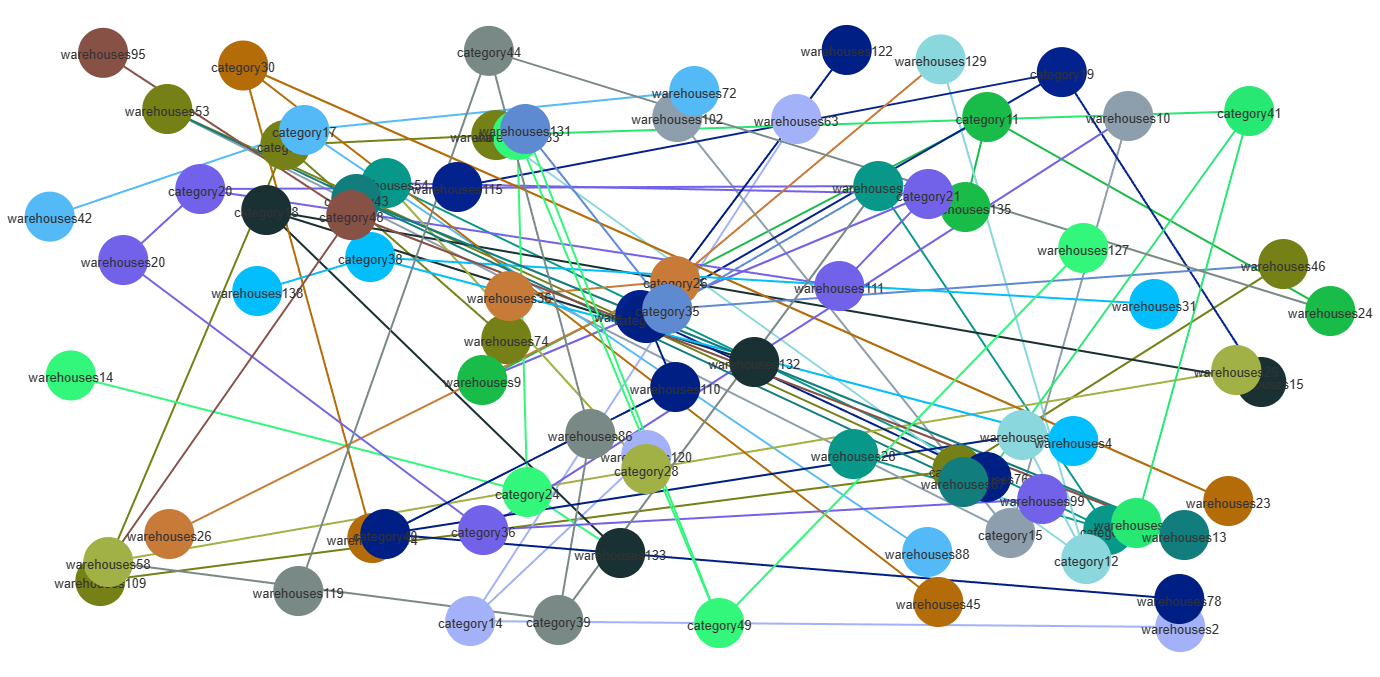


图 14 一品多仓网络拓扑图1

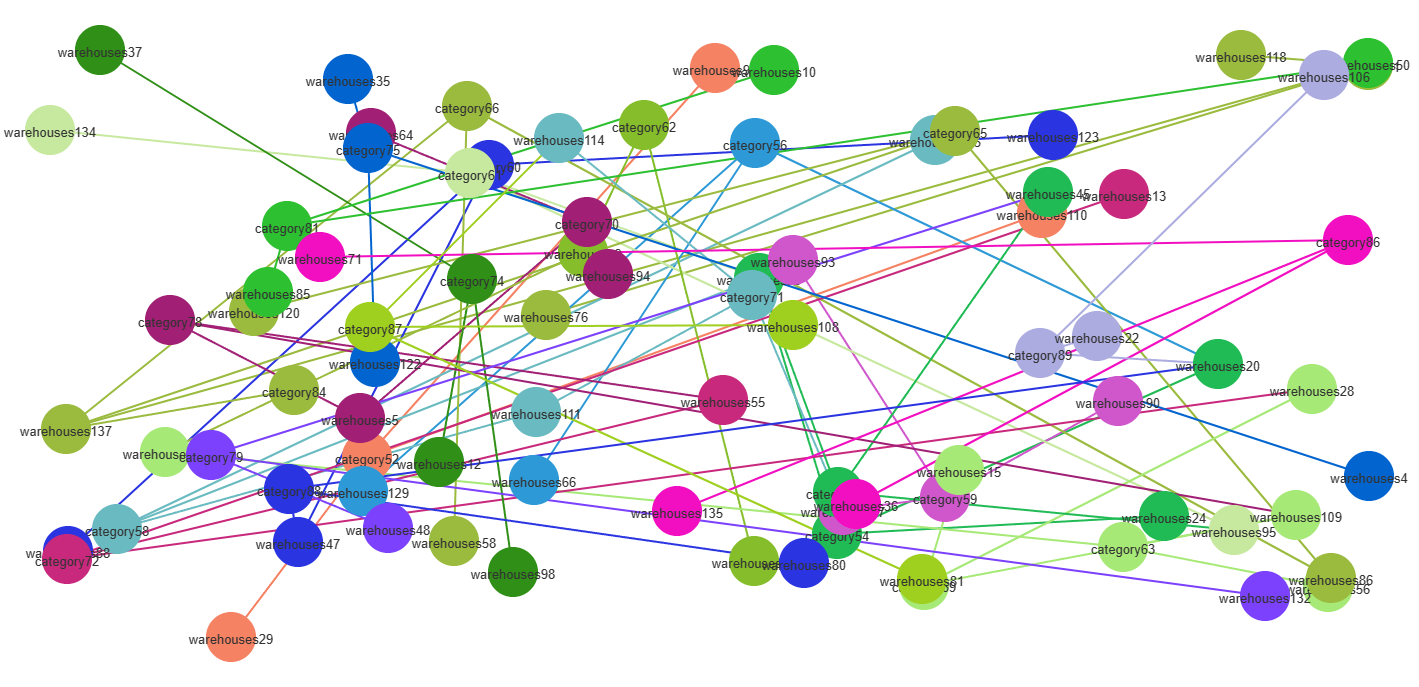


图 15 一品多仓网络拓扑图2

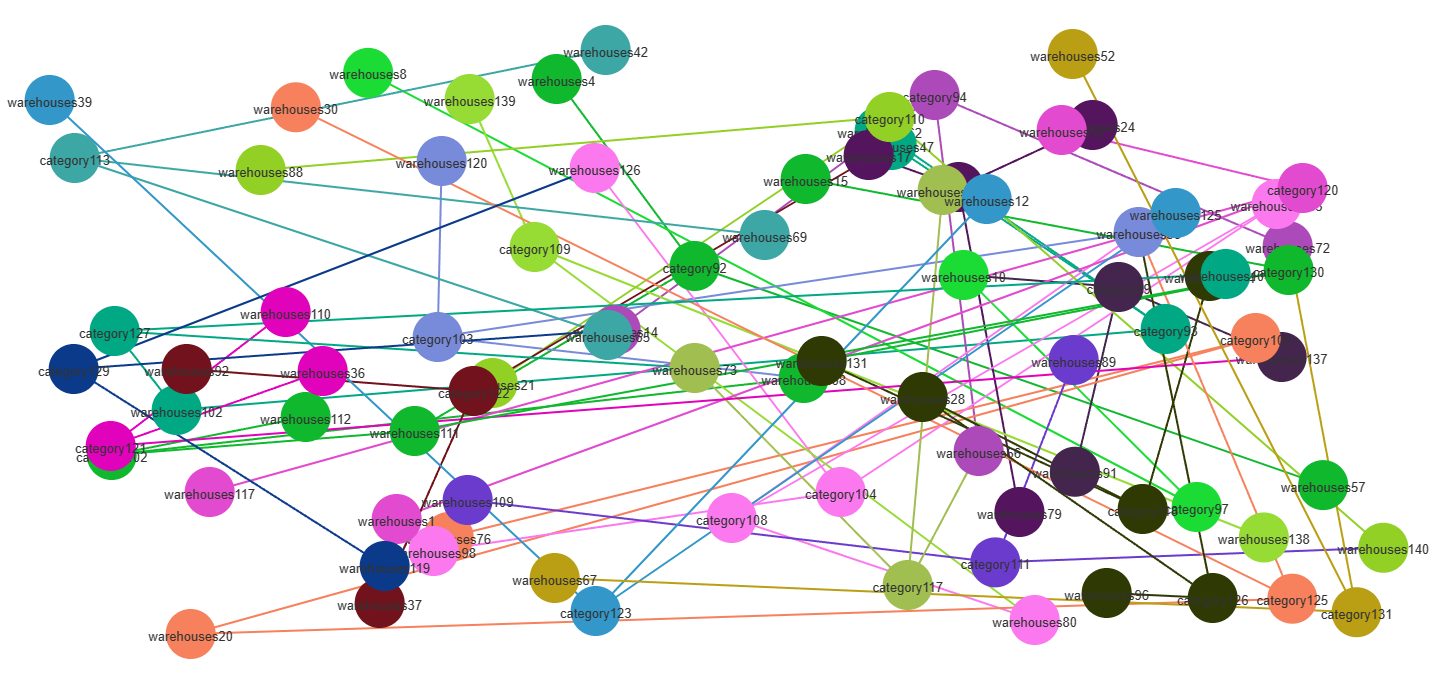


图 16 一品多仓网络拓扑图3

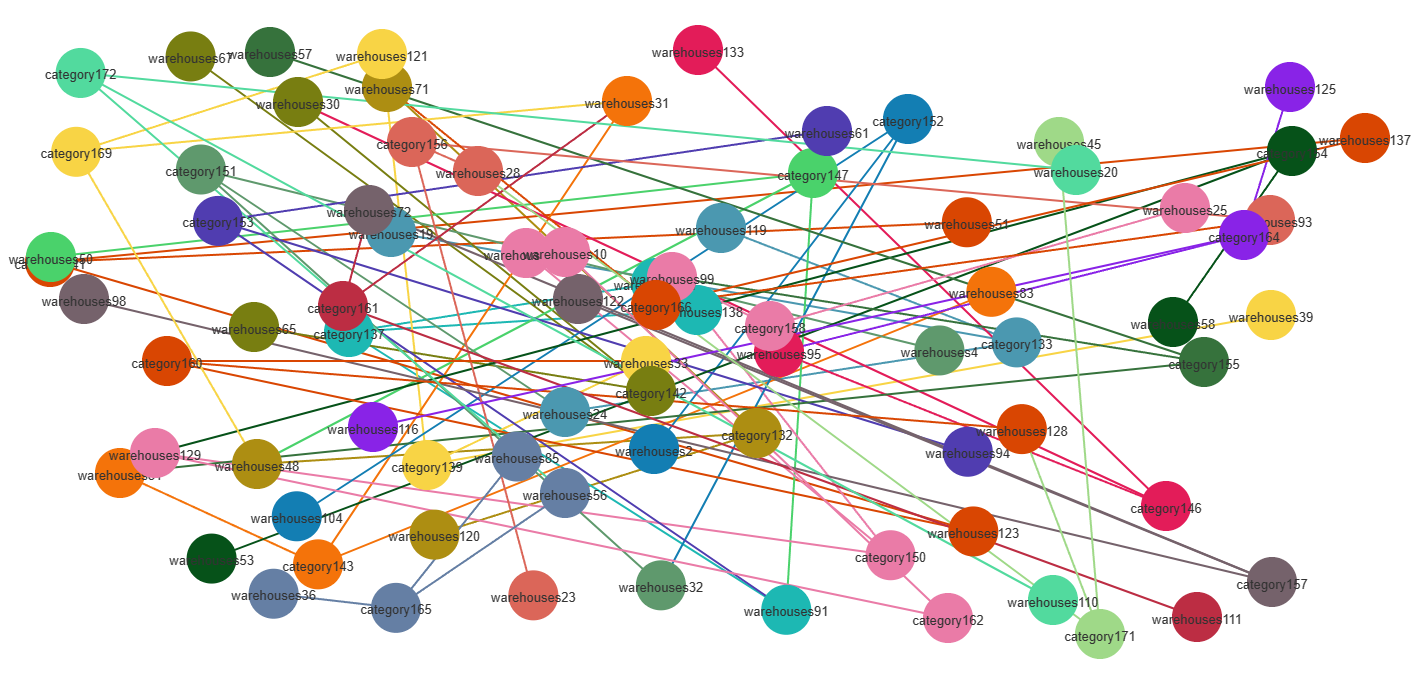


图 17 一品多仓网络拓扑图4

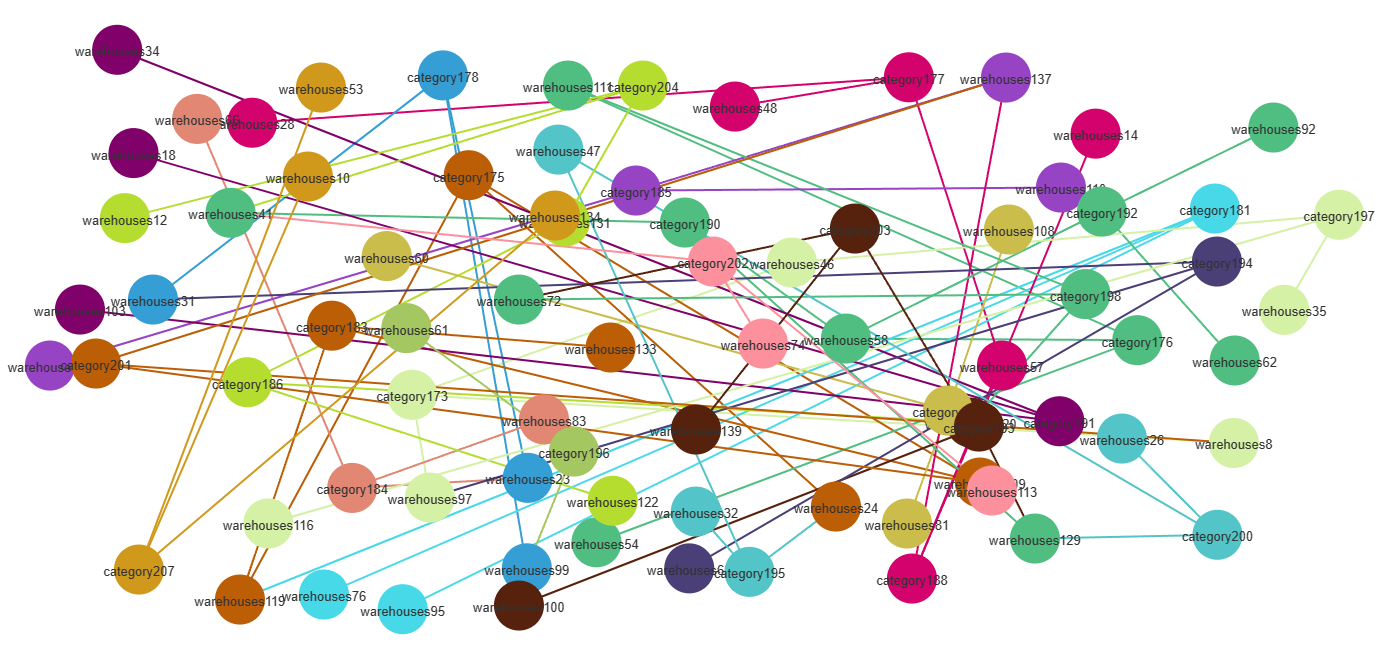


图 18 一品多仓网络拓扑图5

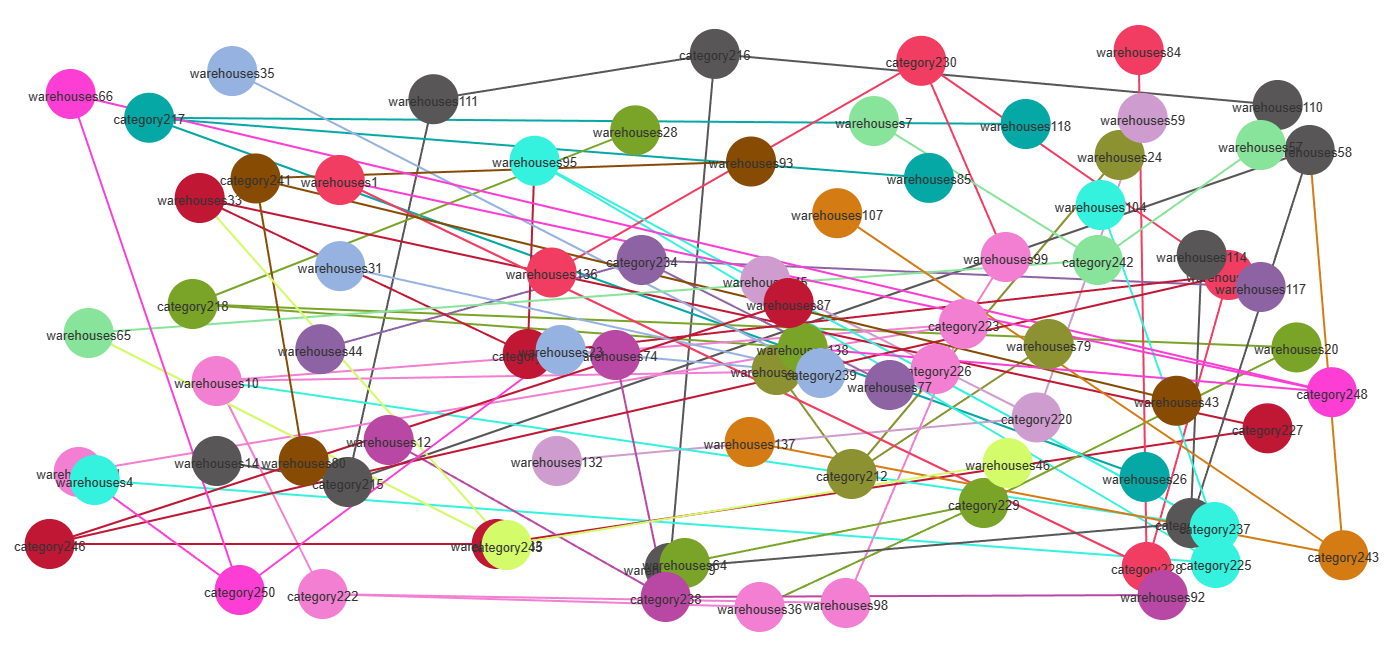


图 19 一品多仓网络拓扑图6

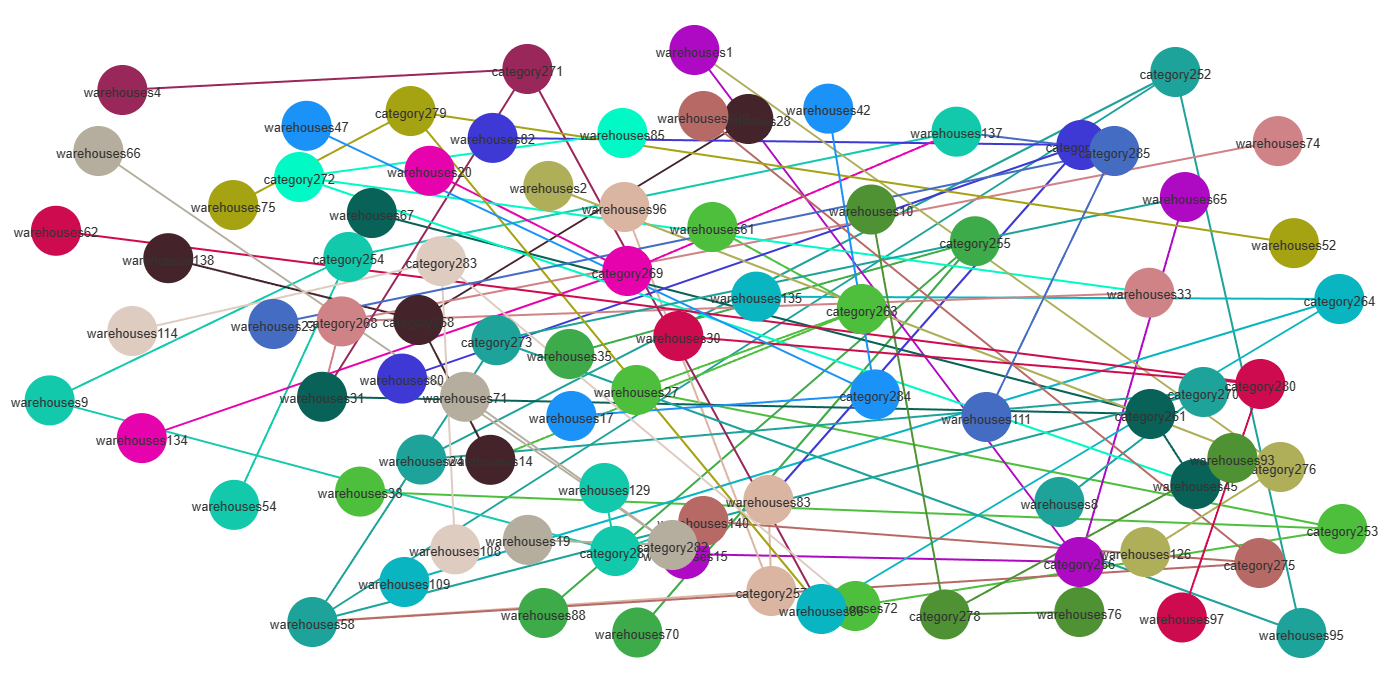


图 20 一品多仓网络拓扑图7

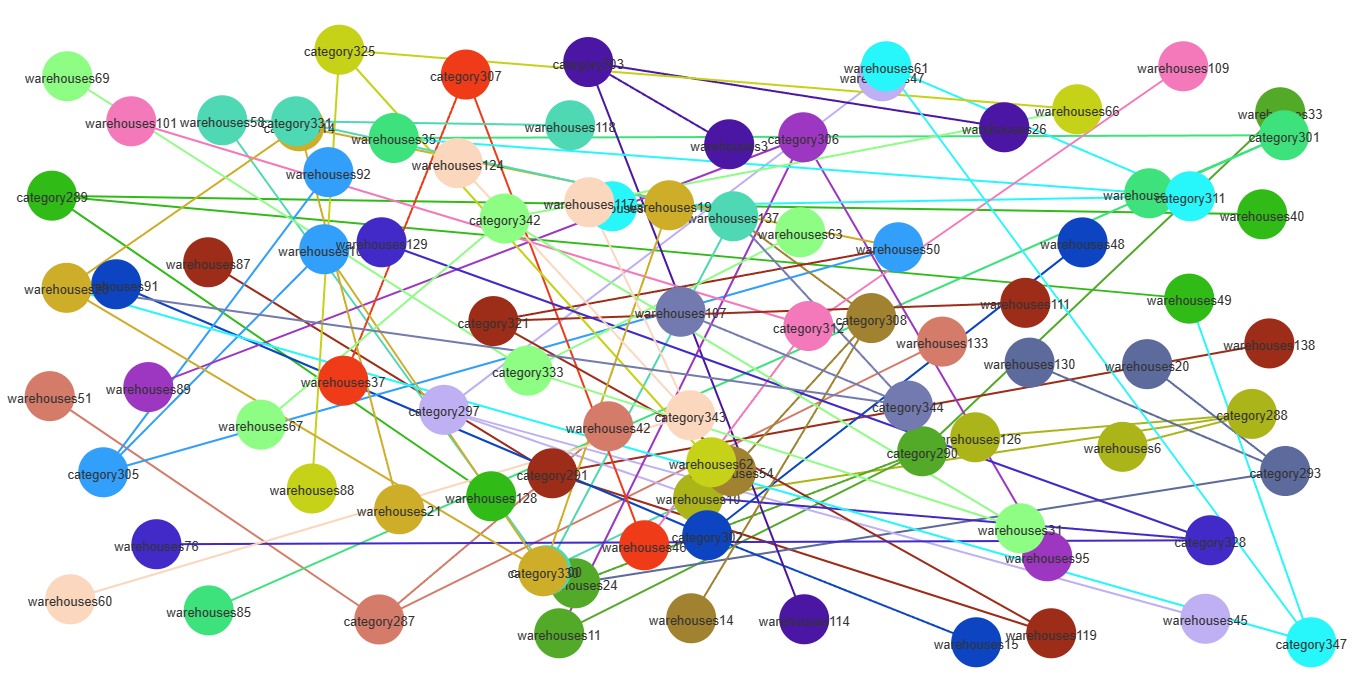


图 21 一品多仓网络拓扑图8

能够更直观地理解品类与仓库之间的分配关系，以及每个仓库的使用情况。这有助于优化仓库管理策略，提高物流效率。同时，通过图形化的方式，可以更容易地识别数据中的异常情况，为进一步的数据分析和决策支持提供了基础。

# 模型的分析与检验

## 模型分析

### 灵敏度分析

灵敏度分析的主要目的是研究模型对参数变化的敏感程度。通过调整模型中的关键参数，我们可以了解这些参数对模型结果的影响。

在灰色预测模型（GM(1,1)）中，关键参数包括累积生成序列 ，生成的新数据序列，以及发展系数和灰色作用量。在每次分析时，只改变一个参数，保持其他参数不变。改变参数后，重新运行预测模型，并观察预测结果的变化。记录不同参数值下的预测结果，分析这些变化对预测准确性的影响。

### 误差分析

误差分析是为了评估模型预测结果与实际数据之间的偏差。我们可以通过比较预测值与真实值来计算误差，并分析误差的来源。

使用拟合后的数据与预测值进行比较。使用标准的误差度量方法，如均方误差（MSE）、平均绝对误差（MAE）等，来计算预测值与真实值之间的误差。

识别导致误差的原因，比如数据质量、模型假设、外部因素（如节假日、促销活动等）等。根据误差分析的结果，调整模型参数或引入新的特征，以减少误差。

## 模型检验

### 稳定性检验

稳定性检验类似于灵敏度分析，主要是通过改变关键参数来观察预测结果的变化情况。

设定参数的变化范围，并在每次检验时只改变一个参数。在不同参数值的情况下重新运行模型，并记录预测结果。比较不同参数值下的预测结果，分析预测结果的稳定性。如果预测结果在参数变化时不发生显著变化，则说明模型具有较好的稳定性；反之，则需要进一步调整模型或参数。

# 模型的评价、改进与推广

## 模型的优点

预测准确性高：通过灰色预测模型（GM(1,1)），我们能够较为准确地预测未来销量趋势。尤其是对于具有准指数规律的数据，该模型表现出良好的拟合能力。

模型简单易用：GM(1,1)模型的构建过程简单明了，易于理解和实施。对于初学者来说，学习成本较低。

鲁棒性强：即使在数据量有限的情况下，灰色预测模型也能提供可靠的预测结果。相比于一些复杂的机器学习模型，灰色预测模型在数据量较少的情况下仍然有效。

灵活性高：通过引入新陈代谢GM(1,1)模型和新信息GM(1,1)模型，我们可以灵活地调整预测方法，以适应不同类型的数据和应用场景。

考虑了关联度：在问题3中，通过遗传算法优化分仓方案时，我们考虑了品类间的关联度，使得分仓方案更加合理，有利于提高整体运营效率。

## 模型的缺点

对数据质量依赖：虽然模型对数据量要求不高，但对于数据的质量有一定的依赖。如果输入数据存在较大噪声或异常值，可能会影响预测结果的准确性。

缺乏动态调整机制：模型在预测过程中假设了数据的平滑特性，但在实际应用中，市场环境和消费者行为可能会出现突发性变化，模型可能无法及时反映这些变化。

忽略季节性因素：目前的模型没有考虑季节性因素对销量的影响，这对于某些受季节影响明显的品类来说，可能会导致预测误差增大。

## 模型的改进

增强数据清洗：通过增强数据预处理阶段的数据清洗工作，去除异常值和噪声，提高输入数据的质量，从而提升预测的准确性。

引入动态调整机制：可以考虑引入在线学习机制，使模型能够在预测过程中不断调整自身参数，以适应市场的动态变化。

考虑季节性因素：在模型中加入季节性成分，通过分解时间序列数据来分离趋势和季节性成分，更准确地预测销量。

多模型融合：采用多模型融合的方式，将灰色预测模型与其他预测模型（如ARIMA、LSTM等）相结合，通过集成学习来提高预测的鲁棒性和准确性。

## 模型的推广

跨品类应用：将当前的模型应用于更多的品类上，不仅限于本次竞赛的数据集，可以拓展到其他行业的产品销售预测中。

实时预测系统：开发一个实时预测系统，利用实时数据流进行销量预测，并将预测结果反馈给库存管理系统，以实现实时库存调整。

供应链优化：将预测结果与供应链管理相结合，优化供应链各个环节的资源配置，提高整体供应链的效率和响应速度。

多仓库协同：将模型推广到多仓库协同管理中，通过优化不同仓库之间的协调，实现资源的最优配置，降低整体运营成本。

通过上述改进措施，我们不仅能提高现有模型的性能，还能进一步拓展其应用范围，使其在更广泛的场景下发挥更大的作用。这些改进和推广措施将有助于提高预测的准确性，增强模型的实用性，并为企业的决策提供更有力的支持。

# 参考文献

[1]马永杰,云文霞.遗传算法研究进展[J].计算机应用研究,2012,29(04)：1201-1206+1210.

[2]杨华龙,刘金霞,郑斌.灰色预测GM(1,1)模型的改进及应用[J].数学的实践与认识,2011,41(23)：39-46.

[3]葛继科,邱玉辉,吴春明,等.遗传算法研究综述[J].计算机应用研究,2008,(10)：2911-2916.

[4]谢乃明,刘思峰.离散GM(1,1)模型与灰色预测模型建模机理[J].系统工程理论与实践,2005,(01)：93-99.

[5]吉根林.遗传算法研究综述[J].计算机应用与软件,2004,(02)：69-73.

[6]温丽华.灰色系统理论及其应用[D].哈尔滨工程大学,2003.

[7]李晓磊.一种新型的智能优化方法-人工鱼群算法[D].浙江大学,2003.

[8]张大海,江世芳,史开泉.灰色预测公式的理论缺陷及改进[J].系统工程理论与实践,2002,(08)：140-142.

[9]席裕庚,柴天佑,恽为民.遗传算法综述[J].控制理论与应用,1996,(06)：697-708.

[10]陈涛捷.灰色预测模型的一种拓广[J].系统工程,1990,(04)：50-52.

附 录

|  |
| --- |
| 附录1 |
| 介绍：支撑材料的文件列表 |
|  |