基于规划模型的蔬菜商品定价与补货探究

摘　要

生鲜商超中，蔬菜的保鲜期较短，随时间的增加，蔬菜的价格与销量会因其品相的变差而受影响，因此，商超需要采取合理的补货和定价策略。本文运用 Pearson 相关系数，研究了不同蔬菜品类间的相互关系；使用 Spearman 相关系数分析了不同蔬菜单品间的相关性；建立规划模型有效解决了商超对于蔬菜品类以及蔬菜单品的补货决策与定价决策问题。

本文首先对附件数据进行了预处理，将附件 2 的扫码时间精确到小时，并筛选掉超出正常范围的销售量异常值。对于因退货产生的负销售量，我们进行了抵消或剔除。同时，根据附件 1，我们为附件 2 中的蔬菜单品标注了相应的品类。

针对问题一，首先，通过计算均值、方差、偏度和峰度统计量，我们探讨了蔬菜品类销售量的**基本分布特征**。并进一步分析长期趋势、周内动态和日内销售分布，得出了各蔬菜品类销售量的分布规律。接着，结合散点图和频数直方图，本文对蔬菜单品销售量的分布规律进行研究。然后，本文通过 **K-S 检验**发现各蔬菜品类销售量服从正态分布后，对各蔬菜品类销售量间的 **Pearson 相关系数**进行分析，观察到多数蔬菜品类销量间存在正相关，而食用菌与茄类蔬菜销量呈负相关。最后，本文通过系统聚类算法将蔬菜单品分为 6 类，从每类中选出一种代表性蔬菜，并利用 **Spearman 相关系数**研究了这些代表性蔬菜间的销售关系，发现部分蔬菜之间的销量呈正相关关系，而部分则呈反向变动。

针对问题二，首先，为探索蔬菜品类销售量与成本加成定价法间的关系，本文识别了成本加成定价模型中涉及的关键变量—价格，并通过**线性拟合**确定了其与销售量间的函数关系。然后，本文利用 **ARIMA 模型**预测了 2023 年 7 月 1 日至 7 月 7 日各品类蔬菜的平均批发价格。最后，本文建立了以蔬菜品类的原始单价P_{ij}和实际销售总量Q_{ij}为决策变量，以商超销售收益最大化为目标函数，以蔬菜品类的销售量-价格关联性、销售量界限及定价界限为约束条件的**单目标规划模型**，运用 **Metropolis-Hastings 算法**对模型求解后得到近似最优定价方案与补货方案（详细结果见正文 7.3），并得到商超的最大收益为 **22489 元**。

针对问题三，首先，本文采用 7 期滞后项的 **SARIMA(0,1,1)**模型预测各蔬菜单品的批发价格。接着，本文对问题二的模型进行更新，以商超销售收益最大化和销售量最大化为目标函数，将蔬菜单品原始价格P_{ik}、实际销售量Q_{ik}及单品补货状态d_{ik}作为决策变量，综合考虑可售蔬菜单品数量、最小陈列量、单品销售量和定价界限等约束条件，建立**双目标 0-1 规划模型**。最后，本文运用**ε约束法**和**遗传算法**对该模型进行求解，得到近似最优的定价方案与补货方案（详细结果见正文 8.6），并得到最大收益为 **1415.77 元**，最大日销量为 **602.86kg**。

针对问题四，为更好地解决上述问题，本文认为还需采集损耗率分布情况、未售出商品占比以及竞争对手销量及价格的数据。本文还对所采集数据的合理性做出了解释。

最后，本文进行了模型推广与优缺点评价。

关键词：相关分析　规划模型　时间序列　定价与补货

一、 问题重述

1.1 问题背景

生鲜商超中，由于蔬菜类商品水分含量高等原因，容易导致其保鲜期较短。与其他商品相比，蔬菜的易腐性使其变得更加珍贵，需要更为细致的管理。随着销售时间的增加，蔬菜的品相会逐渐变差，色泽、口感甚至营养价值都可能会受到影响，销量也随之下降。

因此，商超采取合理的补货策略尤为重要。商超常用“成本加价法”对蔬菜进行定价，并且会对运损及品相变差的商品进行打折销售。同时，商超需要从需求端出发，探究销量与时间的关系。在供应品种丰富的时期，商家需挑选重要的蔬菜品种进行销售。

总之，商超需综合考虑供应链及供需两端，才能更好地进行补货决策与定价决策。

1.2 问题要求

各问题层层递进且暗含联系，其中，关键的问题在于，商超根据以往的销售情况应该如何做出蔬菜类商品的补货决策以及定价决策。根据附件 1 中某商超各蔬菜品类的信息，附件 2、3 中该商超各蔬菜单品的历史销售数据以及附件 4 中各蔬菜商品的损耗率，我们现在进行分析与建立模型以解决以下问题：

问题一要求对蔬菜品类与单品分别进行研究。首先要求探究蔬菜各品类的分布规律，分析不同品类之间的相互关系。接下来需要分析蔬菜各单品的分布规律以及不同单品间的相互关系。

问题二要求商超基于蔬菜的品类进行补货规划。首先，需要探究各蔬菜种类的销售总量与“成本加价法”的关联。接着，依据过往的销售记录，为接下来的一周（2023 年 7 月 1-7 日）预测每种蔬菜的补货数量，并设定一个有效的定价方案，以确保商超获得最大的经济效益。

问题三是在问题二之后对商超补货计划与定价策略的进一步探究。要求在控制可售单品总数在 27-33 个，且各单品订购量的最小陈列量为 2.5 千克的前提下，根据 2023 年 6 月 24-30 日的可售品种，给出 7 月 1 日的单品补货量和定价策略。同时，要在尽量满足市场对各品类蔬菜的前提下，实现商超的收益最大化。

问题四要求采集相关数据，为商超更好地制定蔬菜地补货决策与定价决策提供帮助，并从帮助解决上述问题的角度来解释所收集数据的合理性。

二、 问题分析

在现代社会，随着技术的进步和消费者需求的多样化，商家面临的挑战也越来越大。特别是在食品零售业，如蔬菜类商品，由于其保鲜期短、易受环境和季节影响，如何准确定价和补货成为商家追求利润最大化的核心问题。此外，考虑到蔬菜的品类繁多、来源多样、销售模式不一，商家必须进行精细化管理，既要确保商品的新鲜度和品质，又要避免库存积压导致的损失。为此，商家需要借助先进的数学模型和算法，来帮助他们做出更科学、更合理的决策。

2.1 问题一的分析

该问目标是分析蔬菜各品类和单品销售量的规律及关系。问题指出蔬菜品类或单品间的潜在关联性。这推动我们分类蔬菜品类和单品，利用描述性统计揭示销售

模式和趋势，其中可能受季节、节假日、价格和供应链影响。要明确品类和单品销售量的分布，深入数据是关键，不仅要寻找销售高低峰，还要确保数据质量。关于品类与单品间关系，我们将对销售数据进行关联性分析，探索相似或对立的销售模式，并深化分析导致这些关联的因素，如市场趋势和消费者偏好。综合而言，这一分析旨在为蔬菜的定价和补货策略提供坚实的数据基础。

2.2 问题二的分析

成本加成定价模型体现了价格、成本及成本加成率间的关系。因为成本可从附件3中得到，是个给定量。价格和成本加成率间存在固定的等式关系，同时考虑价格和成本会产生多重共线性的问题。因此，为更准确地描述蔬菜品类销售量与成本加成定价法间的关系，本文仅对蔬菜品类的销售量与价格间的关系进行考查。最终通过线性拟合确定了蔬菜品类的销售量与价格间的函数关系。问题二的核心目标在于制定各蔬菜品类2023年7月1日至7月7日的补货策略和定价决策。本文以商超销售收益最大化为目标函数，将各蔬菜品类的实际销量和原始价格作为决策变量，综合考虑蔬菜品类的销量-价格关联性，销量界限以及定价界限约束，建立规划模型。最后，本文运用Metropolis-Hastings（M-H）算法对该模型进行求解，得到近似最优的补货决策和定价决策，以及近似最大的收益。

2.3 问题三的分析

核心在于制定合理的补货量与定价策略，确保商超最大化收益。首要挑战是满足补货约束，为此，应优先选取高销量或有特定市场需求的蔬菜进行补货。定价方面，需综合蔬菜进价、市场需求，定价策略应依库存和市场需求调整。而利用2023年6月24-30日的数据，应进行一定时间序列分析，以预测7月1日的市场趋势，从而制定相应的补货和定价策略。最终目标是在确保满足市场需求的同时，实现商超最大收益。这要求多目标优化，平衡收益与市场需求满足度，确保策略既不遗失客户，又能保证最佳利润。

2.4 问题四的分析

为制定蔬菜商品的补货与定价决策，商超需整合多维度数据。例如顾客满意度反映消费者对商品和价格的接受度，能指导品种选择和定价。天气预报影响蔬菜需求；例如，炎热可能促销冷食蔬菜，而雨天可能抑制购买意愿。节假日安排也会改变购买模式，如春节时对某些蔬菜的追求。并且如果考察竞争对手的定价策略有助于制定有竞争力的价格。综上，应多维度数据采集确保策略的精确性，满足消费者需求，最大化收益。

三、 模型假设

1.假设时间以小时为单位，不考虑分与秒；
2.假设营业时间为9时至22时；
3.假设商家2023年7月1日到7月7日正常经营；
4.假设补货量等于销售量。

四、 符号说明

符号	说明	单位
i	第i个蔬菜品类	/
j	2023 年 7 月j日	/
t	第t个时间段	/
Q_{ij}	第i个蔬菜品类在 2023 年 7 月j日销量	kg
P_{ij}	第i个蔬菜品类在 2023 年 7 月j日原始单价	元/kg
C_{ij}	第i个蔬菜品类在 2023 年 7 月j日的综合批发价格	元
α_i	第i个蔬菜品类的损耗率	/
w_{it}	第i个蔬菜品类在时间t的销售量在当日总销售量的占比	/
β_i	第i个品类蔬菜的平均折扣率	/
η_t	在时间t的蔬菜品类打折的概率	/
$Maxsale_i$	第i个品类三年中的最大日销售总量	kg
$Minsale_i$	第i个品类三年中的最大日销售总量	kg
$Minvalue_i$	第i个品类三年中的最大日销售价格	元
$Maxvalue_i$	第i个品类三年中的最大日销售价格	元
d_{ik}	第i个品类中第k种蔬菜单品的补货状态	/
μ_{ik}	品类i第k个单品总销量在对应品类中的占比	/

五、 数据预处理

5.1 数据处理

为了使数据处理更为高效，方便后续问题的解决，我们将附件 2 中的扫码时间细化至小时级别，忽略了具体的分钟和秒钟信息。

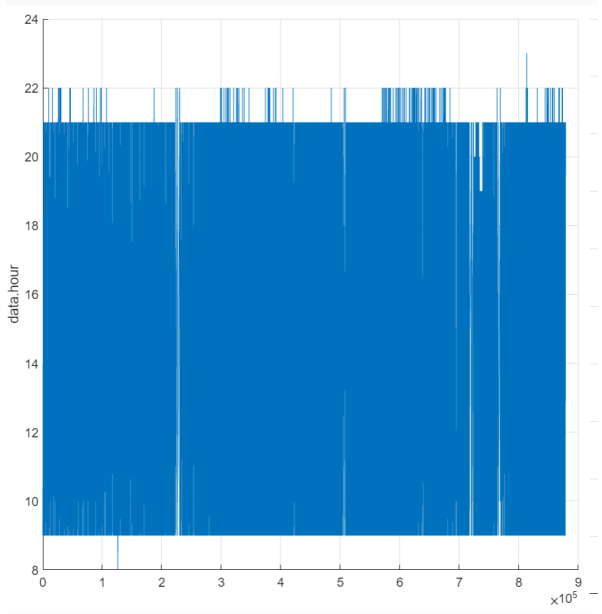


图 1 成交时间可视化

从上图中可以清晰地看到，蔬菜的销售活跃时段主要是上午 9 时至晚上 22 时。通过对附件 2 的详细分析，8 时和 23 时仅有一次交易记录。考虑到可能存在的数据记录误差、商店的营业时间异常或退货情况，我们将这些数据视为异常值并予以排除，避免此类特殊数据的干扰。

为了确保数据分析的准确性并为后续讨论和决策打下基础，我们假设商超蔬菜销售时段为 9 时至 22 时。这一决策不仅基于现有数据，还有助于我们更深入地探讨问题。

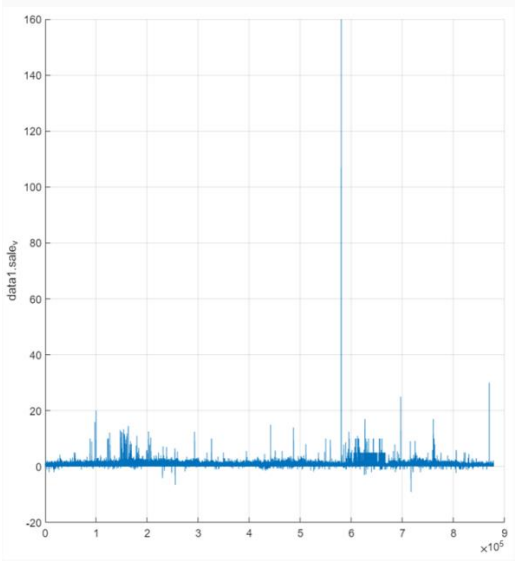


图 2 单笔销售量可视化

从上图中，我们注意到大部分蔬菜的单次销售量集中在 0 到 30kg 之间。但有一个显著的数据点：鲜粽叶在某次销售中达到了 160 公斤。进一步分析发现，这一异常销售量出现在端午节，与节日需求相符。尽管如此，与其他数据相比，这一数据点仍然显得突出。为了确保数据的连贯性，避免异常数据对后续建模的干扰，我们选择将其作为异常值排除。

在对上图进行观察后，我们发现某些数据为负值。这些负值是由于蔬菜退货造成的。为了深入了解这些退货数据，我们使用 MATLAB 对数据进行分析。分析结果显示，蔬菜退货有三种类型：全额退款（429 个）、非全额退款 A 类（12 个）和非全额退款 B 类（19 个）。非全额退款 A 类表示退货量与购买时一致，但退货价格不一致；非全额退款 B 类则表示退货价格与退货量和购买时均不一致。（由于篇幅有限，其余部分见附录 2）

表 1 非全额退款 A 类的数据（部分）

销售日期	销售时间	单品编码	销量	销售单价	是否打折销售
2020-07-07	17	102900005118831	-1	6.5	否
2020-07-08	10	102900005118831	-2	6.9	否
		・・・・・・			
2021-09-25	11	102900005116714	-1.269	7.4	否
2022-09-09	9	102900011033944	-0.207	9	否

表 2 非全额退款 B 类的数据（部分）

销售日期	销售时间	单品编码	销量	销售单价	是否打折
2020-09-24	13	102900005119098	-0.325	16	是
2020-11-12	11	102900005116899	-0.714	8	否
2020-12-20	12	106930274220092	-1	100	否
		・・・・・・			
2023-04-15	15	102900011010891	-0.492	10	否
2023-04-28	13	102900011010891	-0.402	10	否
2023-05-11	17	102900011016701	-0.335	7.2	否

对于全额退款数据，我们将其与对应正常数据进行抵消，累计删除 429 条退款数据和 429 条正常交易数据。对于非全额退款数据，由于其仅有 31 条，在海量数据中占比极低，我们直接将其删除。

在分析附件 2 时，我们注意到附件 2 仅给出了蔬菜的单品详细信息，但没有明确每个单品所属的具体品类。为了便于后续分析，我们先将附件 1 的品类重新编号，编号结果如下。

表 3 类别编号及名称

类别	1	2	3	4	5	6
分类名称	花叶类	花菜类	水生根茎类	茄类	辣椒类	食用菌

我们在附件 2 中新增一列，参考附件 1 增添单品类别，用于标识单品所述品类。数据处理部分 MATLAB 代码见附录 3。

5.2 数据侧写

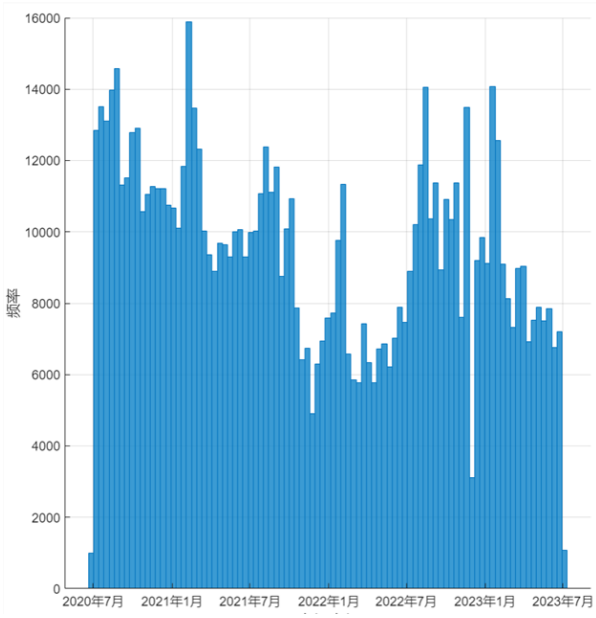


图 3 月度蔬菜购买频数统计图

上图表示过去 36 个月的销售次数统计图，蔬菜的购买高峰主要出现在 8 至 9 月和 1 至 2 月。这是因为 8、9 月份正值暑期，人们对蔬菜的需求量较大。同时，1、2 月是春节期间，由于中国传统习俗，人们对包括蔬菜在内的食品需求量大，消费频数高。

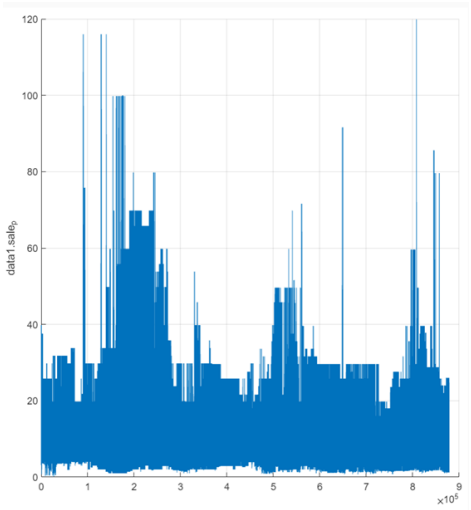


图 4 蔬菜商品销售价格

观察上述的蔬菜销价图，我们发现，蔬菜的价格主要分布在 0 到 70 元的范围内。这个价格区间可能反映了大部分常见蔬菜的标准售价，或者是大多数消费者对蔬菜价格的接受范围。然而，图中也显示了一些蔬菜的价格超过了 80 元，甚至有些达到或超过了 100 元。比如：四川红香椿于 2023 年 3 月 13 日的价格达到了 119 元，黑皮鸡纵菌的价格于 2020 年 9 月 12 日的价格达到了 116 元，蟹味菇（袋）的价格在 2020 年 12 月 19 日和 20 日达到了 100 元。这些显著的价格点可能是由于这些蔬菜的稀有性、供应链的特定变化、季节性需求或特定的市场活动所导致的。

六、 问题一的模型建立与求解

6.1 品类分布规律探究

6.1.1 统计量分析

期望（均值）表示数据的中心位置或平均水平，能够反映数据分布的“中心”。公式为：

$$\mathrm{E(x)} = \int_{-\infty}^{+\infty} xf(x)dx \tag{1}$$

方差描述了数据的离散程度或变异性，代表数据值与其期望值的偏差的平方的期望。公式为：

$$\mathrm{Var(x)} = \int_{-\infty}^{+\infty} [x - E(x)]^2 f(x)dx \tag{2}$$

偏度度量数据分布的不对称性。正偏度表示分布的右尾部比左尾部长，负偏度表示分布的左尾部比右尾部长。公式为：

$$\mathrm{S} = \frac{\mathrm{E}\{[\mathrm{X} - \mathrm{E(X)}]^3\}}{\sigma^3} \tag{3}$$

峰度描述了数据分布的尖锐程度。峰度大于 3 的分布比正态分布更尖锐，小于 3 的则比正态分布更平坦。公式为：

$$\mathrm{S} = \frac{\mathrm{E}\{[\mathrm{X} - \mathrm{E(X)}]^4\}}{\sigma^4} \tag{4}$$

通过 EXCEL，我们求得各个品类的均值、方差、偏度和峰度。结果如下表

表 4 不同品类均值、方差、偏度和峰度

	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
均值	0.5982	0.4830	0.6927	0.5000	0.4408	0.5132
方差	0.1379	0.0398	0.3030	0.0548	0.0952	0.1358
偏度	6.3558	6.0346	5.9096	5.0391	1.4243	3.2022
峰度	234.7308	236.8861	77.5194	166.7052	14.5772	152.6863

从均值上看，品类 1、品类 3 的中心位置会更靠右侧，而品类 2 与品类 5 的整体分布会更靠左侧；

从方差上看，品类 3 的方差明显高于其余品类，数值间的差异更大，分布更加分散，而品类 2 与 3 的分布较其余品类更加集中；

从偏度来看，六个品类均满足偏度大于 0，均为正偏斜，有较长的右尾部，均值大于中位数，其中品类 1 与品类 2 的正偏尤为明显；

从峰度来看，六个品类均远高于 3，具有典型的“尖峰”特征，品类 1 与品类 2 尤为如此。

综合以上几点，品类 1 与品类 2 呈现非常明显的“尖峰厚尾”特征，但品类 2 中心偏右，品类 1 中心偏左。品类 3、品类 4 与品类 6 分布特征较前两类没有非常明显。品类 5 峰度与偏度最低，较其它品类更加扁平和对称。

6.1.2 图像分析

为了深入了解不同蔬菜品类的销售量分布规律，我们将对 2020 年 7 月 1 日至 2023 年 6 月 30 日的时间段进行详细分析。具体来说，我们计划从三个维度出发：

1.长期趋势：通过观察这三年间各品类每天的销售量变化，我们可以了解哪些品类的销售量呈上升或下降趋势，以及是否存在某些特定时段的季节性变化。

2.周内动态：我们将进一步分析一周中各个日子（从周一到周日）的销售数据。这有助于我们揭示哪些品类在周末或工作日销售得较好，或是否有某些品类在特定的日子中受到消费者的更大欢迎。

3.日内销售分布：最后，我们将探讨一天中各个时段的销售量变化。了解品类在不同时间段的销售情况。

- **蔬菜销量的长期趋势**

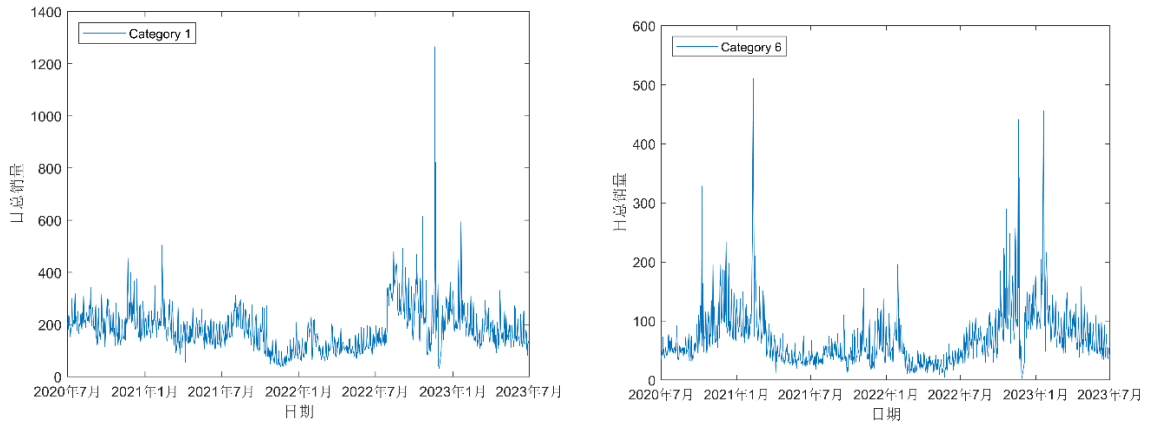


图 5 不同品类蔬菜日总销量趋势图（左图品类 1，右图品类 6，其余见附录 4）

从图中我们可以洞察以下几个蔬菜品类销售量的分布规律。

1.**花菜类**：整体上，该类蔬菜在 2020 年 7 月至 2021 年 7 月的销售较为稳定。尽管在 2021 年 12 月有所回落，但 2022 年 8 月的销售量经历了显著的突增，并在此后的时间段持续高位运行。

2.**花叶类**：销售模式显示了明显的峰谷变化，其中在 2021 年 10 月和 2022 年 3 月都有明显的销售谷值，但在 2022 年 1 月和 2023 年 8 月，销售量反弹至高峰。

3.**水生根茎类**：这类蔬菜的销售呈现了明确的季节性波动，大约每半年出现一次销售高峰和谷底。

4.**茄类**：从 2020 年 7 月开始至 2023 年 6 月底，茄类蔬菜的销售模式呈现了一个明确的一年周期，每年都经历了一次销售高峰和谷底。

5.**辣椒类**：2021 年 1 月至 2023 年中，辣椒类蔬菜的销售量表现出了两次明显的上升趋势，并在 2022 年 7 月之后显著增长，保持在较高的水平。

6.**食用菌类**：与茄类相似，食用菌的销售也呈现了明确的一年周期，每年都会出现一次销售高峰和谷底。同时，食用菌类销售量的变化方向与茄类相反。

各品类蔬菜的日销售量均显示出了季节性的变动趋势，但变动的时间点和幅度各不相同。例如，茄类和食用菌类都展示了年度周期性，而水生根茎类则表现为半年一次的波动。蔬菜日销量的季节波动性可能与种植季节、消费者的季节性需求或者节假日活动等因素有关。例如，花菜类和花叶类在某些特定月份出现销售峰值，这可能与它们的收获季节或者某些传统节日（如春节）的菜肴需求有关。

- **蔬菜销量的周内动态**

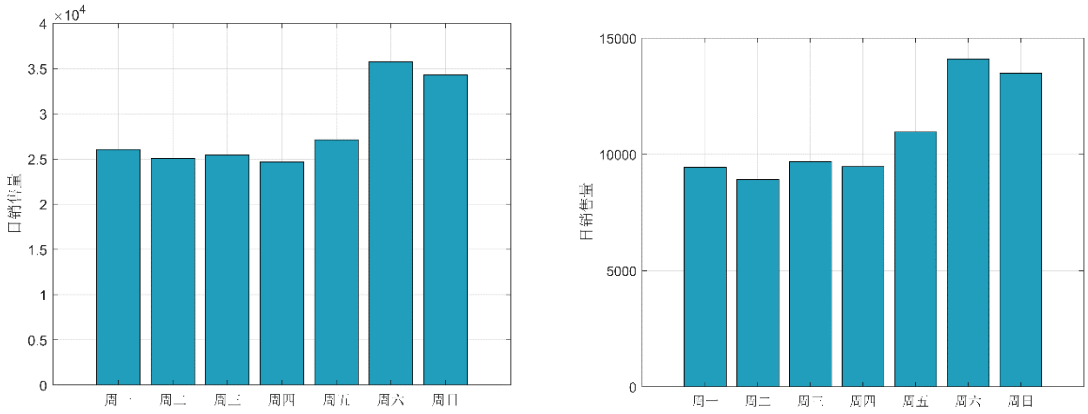


图 6 周内不同日期日销售总额（左图品类 1，右图品类 6，其余见附录 4）

根据上述图表展示，各品类蔬菜的销售量的分布规律呈现出以下特点：

1.**工作日销售趋势**：在一周的五个工作日中，周五成为了各品类蔬菜销量的小高峰。这可能是因为许多消费者为周末准备食材，或者因为某些市场促销活动常在周五举行，从而刺激了购买。

2.**周末销售特性**：相比工作日，各品类蔬菜在周末的销量显著增加。这或许是由于大多数家庭选择在周末进行大规模的采购，或是家庭聚餐、朋友聚会等活动在周末较为频繁，从而增加了蔬菜的消费需求。

➢ **蔬菜的销量的日内销售分布**

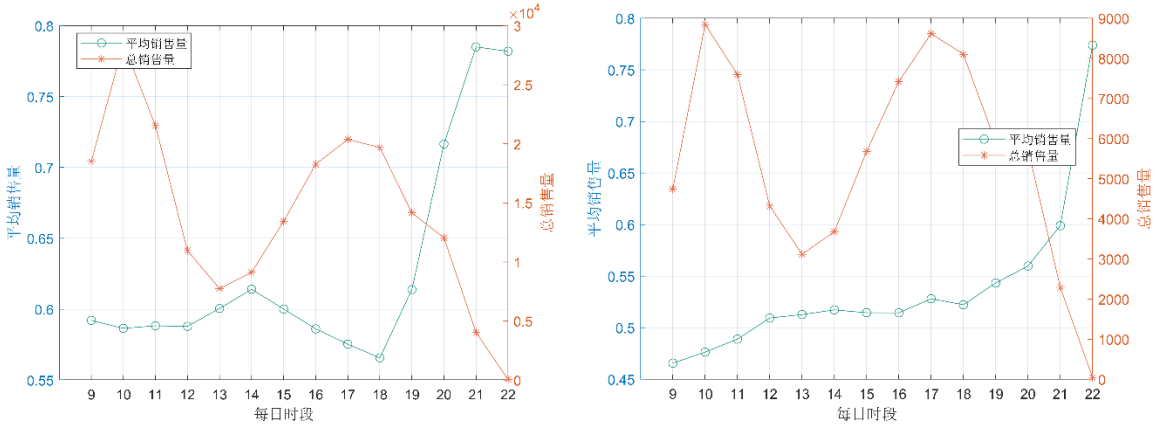


图 7 不同品类每小时销售趋势（左图品类 1，右图品类 6，其余见附录 4）

根据图表数据，我们可以进一步对蔬菜品类的每小时销售趋势进行深入解读和分析，对蔬菜品类日内销售量的分布规律进行探究：

日内销售高峰：无论是哪个品类，蔬菜的销售量都在上午的 9 时至 10 时之间快速攀升并达到日内的首个高峰。这可能是因为许多消费者早上进行采购，为一天的三餐做准备。特别是在工作日，许多人可能在上班前进行快速购物。

日中销售低谷：从 10 时之后，各品类蔬菜的销售量开始逐渐减少，至 13 时达到日内的最低点。这段时间是大多数人的午休时间，所以购买活动相对较少。

下午销售趋势：除了水生根茎类蔬菜在 16 时出现反弹外，大部分蔬菜品类在 17 时都有一个明显的销售高峰。这可以理解为许多上班族下班后的采购高峰，为晚餐和次日早餐做准备。

晚间销售情况：从 17 时的高峰开始，各品类蔬菜的销售量逐渐减少，并在 22 时达到一个相对低点。考虑到大多数商店在此时关闭或接近营业结束，这个低点是可以预期的。

平均销售量：虽然总销售量在特定时间点达到峰值，但各品类蔬菜的平均销售量在一天中总体呈现出上升的趋势，最终在 22 时达到峰值。这意味着，尽管总销售量在 22 时达到低谷，但平均购买量达到了峰值，这可能是消费者在商店即将关闭之前进行了大量的采购，同时交易笔数少的缘故。

6.2 单品分布规律探究

我们现在通过绘制散点图来深入分析蔬菜单品销售量的分布规律。下图中的“单品销售总量”指的是每种蔬菜单品在 2020 年 7 月 1 日至 2023 年 6 月 30 日这三年内的总销售数量。

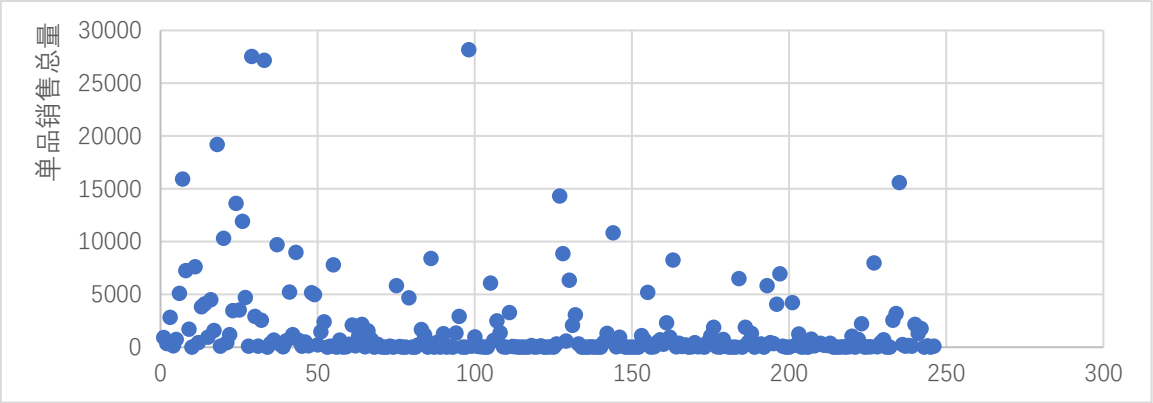


图 8 单品销售总量散点图

根据上述图表，蔬菜单品的销售总量呈现出明显的分布特征。大多数蔬菜单品在过去的三年内的销售总量集中在 0 到 5000 千克之间，我们认为这是一般的蔬菜销售水平。然而，也存在少数蔬菜单品，它们在过去的三年内销售量显著超过了这个范围，具体来说，芜湖青椒（1）的销售总量达到了 28164.33 千克，西兰花的销售总量达到了 27537.23 千克，净藕(1)的销售总量达到了 27149.44 千克等。这些蔬菜受到了特殊的市场需求、消费者喜好以及季节性因素的影响，因此具有高销售量。

为了更精确地研究蔬菜单品销售量的分布规律，使其更具一般性和代表性，我们决定将注意力集中在销售总量在 0 到 5000 千克范围内的蔬菜单品上进行分析。我们认为这个范围内的数据更能反映蔬菜销售的常态情况，因此我们绘制了下面的 "蔬菜单品销售量分布频数图"，以深入研究这些单品的销售量分布规律。

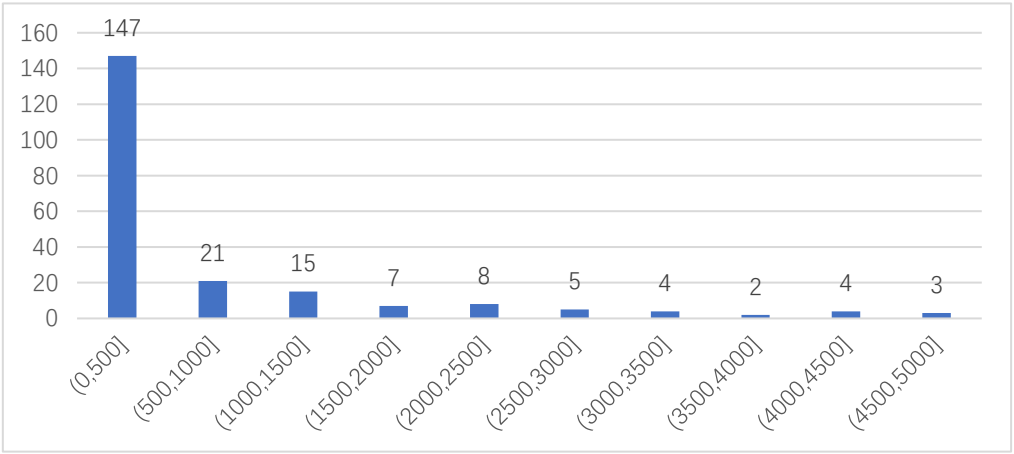


图 9 蔬菜单品销售量分布频数图

通过上述直方图的详细分析，我们可以更深入地了解蔬菜单品销售量的分布规律。观察发现，大多数蔬菜单品的销售量集中在较低的区间，主要分布在 0 到 500 千克的范围内。这表明了在蔬菜销售市场中，大部分单品的销售量相对较小。只有少部分单品因为季节性需求与自身的销售优势而超出这个范围。

6.3 品类相互关系探究

我们计算各蔬菜品类 2020 年 7 月 1 日至 2023 年 6 月 30 日各月的销售总量，并希望通过 Pearson 相关系数探究不同品类销售量之间的相互关系。

6.3.1 正态性检验

为了使用 Pearson 相关系数来研究各蔬菜品类月度销售总量之间的关系，我们首

先需要确保样本数据满足 Pearson 相关系数的假设之一，即数据服从正态分布。

表 5　K-S 检验结果

	花叶类	花菜类	水生根茎类	茄类	辣椒类	食用菌
K-S 值	0.1874	0.1645	0.1353	0.1782	0.2041	0.1979
P 值	0.7278	0.8504	0.9597	0.7794	0.6289	0.6660

为验证这一假设，我们利用 MATLAB 对样本数据进行 K-S 检验，发现 K-S 检验的 P 值小于 0.05，所以在 95%的置信水平下，我们拒绝原假设，即认为样本符合正态分布。

6.3.2 Pearson 相关系数

Peason 相关系数法是一种衡量两个变量间线性关系方向以及强弱的常用统计方法，它要求样本数据服从正态分布，且其值介于-1 到 1 之间。具体公式如下：

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}} \tag{5}$$

其中：r 代表两个变量间的 Pearson 相关系数，X_i 和 Y_i 分别代表两个变量的第 i 个观测值。$\bar{X}$和$\bar{Y}$分别代表两个变量的均值。Pearson 相关系数的绝对值越接近 1，则表示两个变量之间的相关程度越大，即两个变量越相似[1]。

我们用 MATLAB 软件计算得到的相关系数如下表。其中:“*”表示 0.05≤p<0.1，“**”表示 0.01≤p<0.05，“***”表示 p<0.01

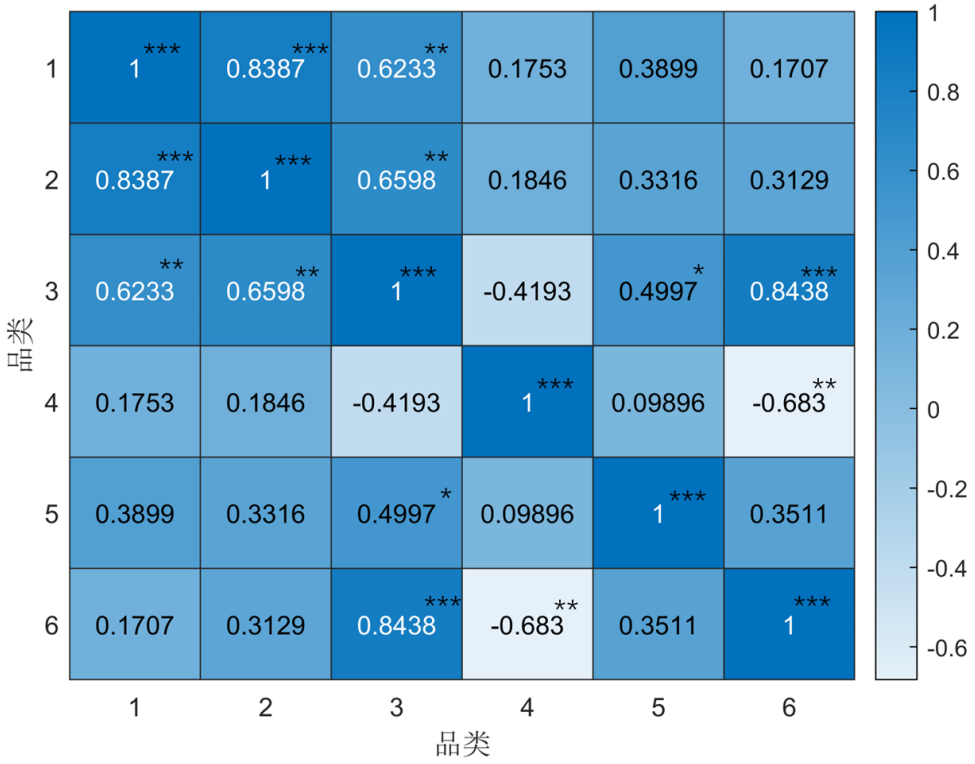


图 10　蔬菜品类销量皮尔逊相关系数及显著性

由图可知，根据 Pearson 相关系数的分析，我们在 95%的置信水平下发现了一些显著的关联。具体而言，花菜类蔬菜与花叶类蔬菜的销量、水生根茎类蔬菜与花叶类蔬菜销量、水生根茎类蔬菜与花菜类蔬菜销量、食用菌与水生根茎类蔬菜销量以及食用菌与茄类蔬菜销量之间的 Pearson 相关系数均显著。特别值得注意的是，花菜类与花叶类蔬菜的销量之间存在高度正相关，这表明它们的销售趋势非常相似，可能受到类似的季节性或市场需求因素的影响。

此外，根据对 Pearson 相关系数的分析，。我们观察到花菜类蔬菜与花叶类蔬菜销量、水生根茎类蔬菜与花叶类蔬菜销量、水生根茎类蔬菜与花菜类蔬菜销量、食用菌与水生根茎类蔬菜销量以及食用菌与茄类蔬菜销量之间存在正向关系，这意味着当一个品类的销量增加时，另一个品类的销量也有增加的趋势，这是因为他们互为互补品。特别地，食用菌与茄类蔬菜销量呈现负相关关系，这是因为它们是市场上的替代品，当一个品类销量增加时，另一个品类的销量会下降。

6.4 单品相互关系探究

附件 2 中有两百多种单品类型，若逐一进行相互关系探究，工作量大。因此，该部分先采取系统聚类，使月度销售量相近的蔬菜类商品能够归为同类。之后，我们再从不同类别随机抽取商品，对它们的相互关系进行探究。

6.4.1 单品归类

系统聚类算法是一种层次化聚类方法，它能够有效解决未知聚类数量的问题，其基本思想是将个体逐个合并形成一个子集，直到整个总体都在一个集合内为止[2]。右边是系统聚类算法的基本流程图。

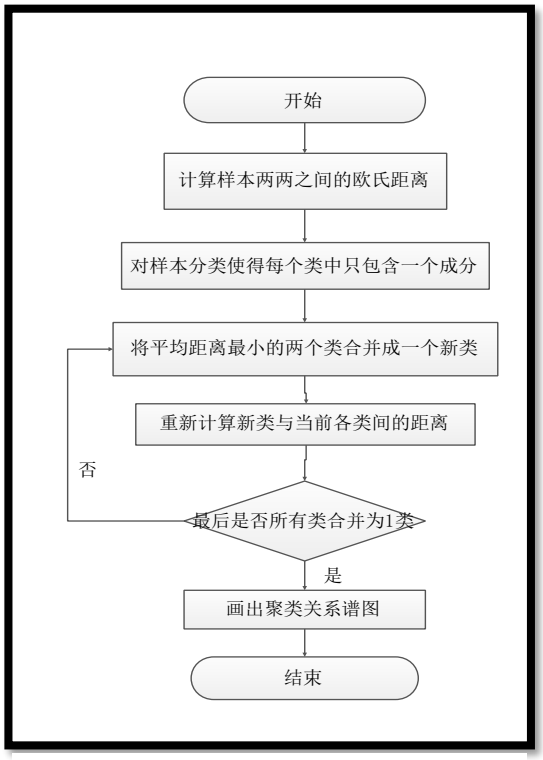


图 11 聚类流程图

我们将不同单品类型作为样本，将不同月度销售量作为指标，进而依据指标对样本进行聚类。在聚类之前，我们通过肘部法则分析聚类类别数。

肘部法则是一种常用的聚类数量确定方法，它的核心目标是寻找数据集的最佳聚类数量，以在保持聚类性能的平衡时取得最佳结果。为了确定系统聚类的最优聚类数量，我们引入肘部法则。具体来说，我们以聚类类别数 K 为横坐标，以聚合系数（所有类别总畸变程度）为纵坐标绘制聚合系数折线图。

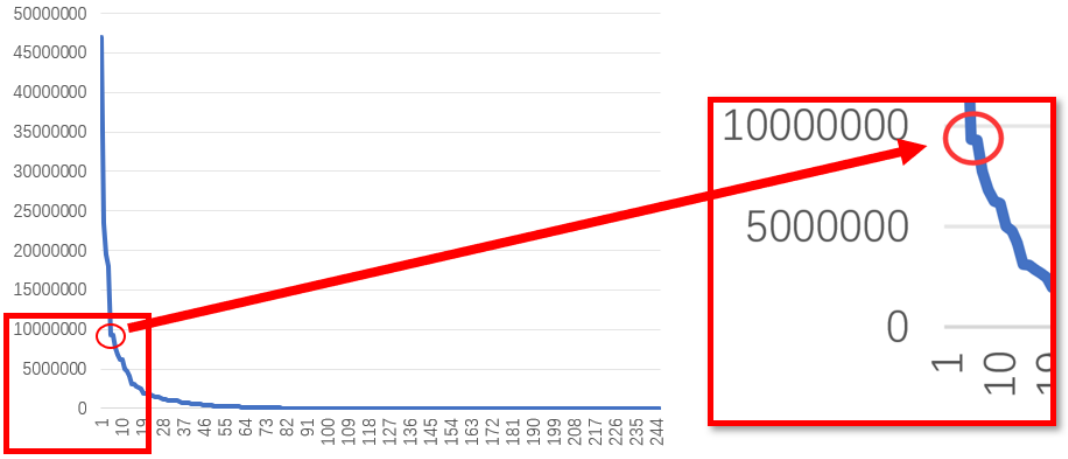


图 12 聚合系数折线图

通过观察聚合系数的变化趋势，我们发现在 K=6 时，聚合系数有明显的降低，形成了一个“肘部点”。这个“肘部点”通常被认为是最佳的系统聚类数量，因为增加聚类数量后，聚合系数的下降幅度不再显著，导致性能提升有限。因此，我们选择 K=6 作为最佳的聚类数量，将蔬菜单品划分为 6 个类别，以更好地理解它们之间的关系和特征。

通过绘制谱系图，我们得到了蔬菜单品分类的最终结果，并将结果展示在附录中。我们发现，同一类别的蔬菜单品在销售风格上具有相似性，表现为销售量的相似性。这意味着属于同一类别的蔬菜单品在销售方面有一定的共性。

6.4.2 相关性量化

为了深入研究不同蔬菜单品的销售量之间的相互关系，我们采取了以下方法：我们从每个类别中随机选择了一种代表性的蔬菜单品，并对它们的销售量进行了详细分析。所抽取的蔬菜单品如下：

表 6 各类别代表性蔬菜单品

类 1	类 2	类 3	类 4	类 5	类 6
牛首生菜	云南生菜	大白菜	紫茄子(2)	西兰花	净藕(1)

对于所抽出的蔬菜单品，我们首先制作了散点图，以直观展示它们之间的趋势和相关性。（仅展示部分结果，其余见附录）

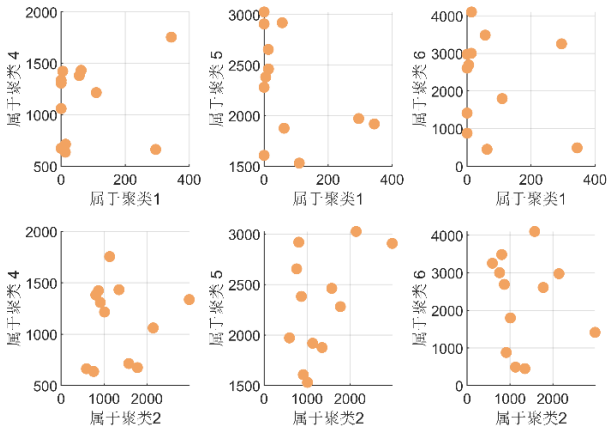


图 13 销量关系散点图

根据上图观察，我们可以明显看到蔬菜单品销售量之间并不呈现明显的线性关系。因此，为更准确地研究蔬菜单品销售量之间的相互关系，我们选择采用 Spearman 秩相关系数，而不使用 Pearson 相关系数。

Spearman 秩相关系数法是一种非参数检验方法，可用于衡量两个变量等级关系的强度和方向。公式为：

$$r = 1 - \frac{6\sum_{i=1}^{n} d_i}{n(n^2-1)} \tag{6}$$

其中，n 为数据个数，di 为两个变量的等级差。和 Pearson 相关系数比较，Spearman 相关系数没有过多限制，不需要关心数据是否线性，是否符合正态分布，只需要关心每个变量对应数值的具体位置。如果两个变量的对应值排列顺位是相同或相似，则具有显著的相关性[3]。

我们利用 MATLAB 软件得出的相关系数如下（***为在 0.01 水平上显著，**为在 0.05 水平上显著，*为在 0.1 水平上显著）：

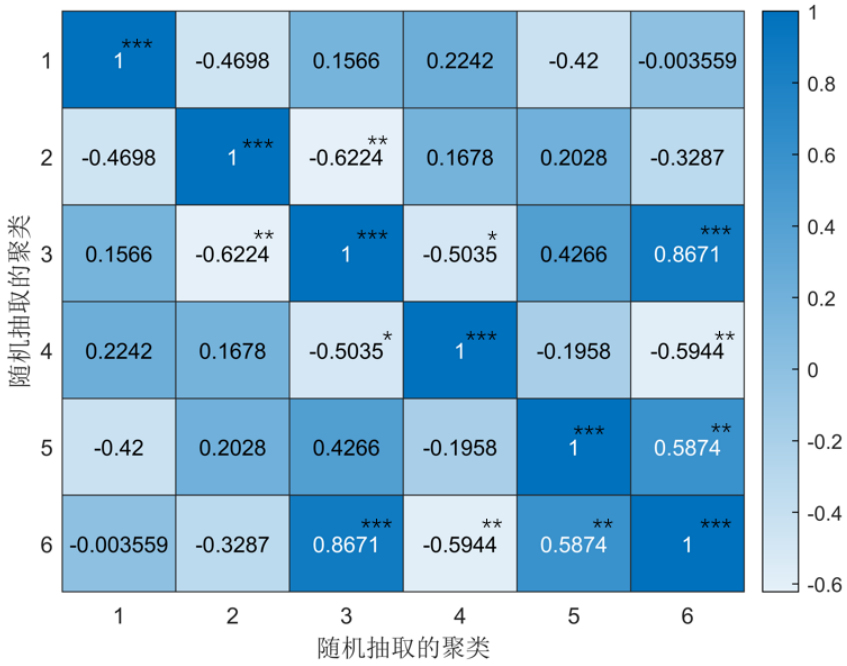


图 14 斯皮尔曼相关系数图及显著性

通过对相关系数的分析，在 90%的置信水平下，我们发现了多组蔬菜单品销售量之间 Spearman 秩相关系数显著，这表明这些蔬菜单品的销售量之间存在一定的关联性。其中，净藕(1)与大白菜的销量间呈现出极强的正相关性，这意味着它们的销售量在很大程度上会同时增加或减少。另一方面，大白菜与云南生菜、紫茄子(2)与大白菜以及净藕(1)与紫茄子(2)的销售量之间存在负向变动关系。这意味着在某些情况下，其中一个蔬菜单品的销售量增加，可能会导致另外一个蔬菜单品的销售量下降，或者反之亦然。此外，净藕(1)与大白菜的销量以及净藕(1)与西兰花的销量之间存在同向变动的关系，表明它们的销售趋势在某些时间段内是一致的。

七、 问题二的模型建立与求解

7.1 销量与定价关系分析

7.1.1 研究对象确定

成本加成定价法是一种价格制定策略，它着重确保产品或服务的价格足以覆盖生产和销售成本，并获得适当的盈利。在这一方法中，销售者被认为拥有决定价格的主要权力，而购买者主要能够影响的是价格的加成率。成本加成定价决策主要考虑成本基础、业务量水平以及加成率这三个主要因素[4]。以下是常用的一种成本定价模型:

$$P = C(1 + r) \tag{7}$$

其中，P 表示价格，C 表示成本，r 表示成本加成率。

基于上述模型，探究蔬菜品类销售总量(Q)与成本加成定价法之间的关系，本质上是探究 Q 与 C、P、r 的关系。其中，C 可以通过附件 3 得到，为给定的量，因此我们不对其进行研究。

值得注意的是，若同时考虑销售价格和成本加成率，则由于等式（7），将导致多重共线性问题，干扰分析的结果。

因此，为了更准确地研究蔬菜的销售总量与成本定价法间的关系，我们仅关注销售总量与销售价格之间的关系。

7.1.2 数量关系建立

考虑到成本加成定价模型是一个静态模型，同时题目也强调“蔬菜类商品的销售量与时间往往存在一定的关联关系”。为了剔除时间因素干扰，我们的研究数据的时间范围应当尽可能小。最终，我们选取了近三年 7 月 1 日至 7 月 7 日的数据作为研究样本。

在研究样本中，对于不同品类，我们分别从各个蔬菜品类里随机挑选了 10 组日销售总量与销售均价数据，以期得到一个更具代表性的样本。考虑到微观经济学中常使用线性模型对需求曲线简化处理，我们使用 MATLAB 工具箱来完成数据拟合工作。拟合结果如下：

表 7 不同类别日销量与均价关系

品类	关系
花叶类	$Q_1 = -27.54P_1 + 413.8$
花菜类	$Q_2 = -36.4P_2 + 468.8$
水生根茎类	$Q_3 = -6.276P_3 + 268.8$
茄类	$Q_4 = -9.47P_4 + 104.9$
辣椒类	$Q_5 = -17.12P_5 + 290.4$
食用菌类	$Q_6 = -0.9913P_6 + 47.68$

根据我们的分析结果，各蔬菜品类的销售总量与销售价格之间呈现出负向关系。具体来说，当销售价格上升时，销售总量往往出现下降的趋势；反之，当销售价格下降时，销售总量则有增长的趋势。这说明消费者在面对蔬菜价格上涨时会减少购买量，而在价格下跌时则更愿意购买。

7.2 规划模型建立

该小问的目标是实现商超收益最大化，为此，我们设计相应的目标函数和约束条件，并对其中的参数进行测算。在正式建立规划模型之前，我们需预测 2023 年 7 月 1-7 日的批发价格。

7.2.1 批发价格预测

附件 3 中提供的数据是单品数据，我们将单品按品类进行归类，取每日某品类各单品平均批发价格作为待预测数据。处理后，我们得到六个品类 3 年来的批发价时间序列数据。我们采用 ARIMA 模型分别对其进行预测。

ARIMA 模型，又名差分自回归移动平均模型，是一种用于时间序列分析和预测的统计方法。该模型通过差分法将非平稳时间序列平稳化，这一过程克服了仅适用于平稳时间序列的 AR 模型和 MA 模型的局限性[5]。

ARIMA 模型的使用步骤包括数据的差分、模型阶数的选择、模型的拟合及预测。这些步骤的有机结合使 ARIMA 模型成为了一个强大的工具，能够更准确地对未来的趋势和变化进行预测。记 ARIMA 模型为 ARIMA(p,d,q)，其中，p 是时间序列中自回归(AR)项的阶数，d 是使时间序列平稳需要进行的差分次数，q 是时间序列中移动平均(MA)项的阶数，其一般公式如下：

$$Y_t' = c + \Phi_1 Y_{t-1}' + \phi_2 Y_{t-2}' + \cdots + \Phi_p Y_{t-p}' + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots \theta_q e_{t-q} + e_t \quad (8)$$

其中，Y_t'表示经过 d 阶差分后的时间序列，即 d 次差分后的时间序列。c 是常数项。Φ_1，Φ_2，…，Φ_q是自回归（AR）系数，表示各阶滞后项的权重。$\theta_1, \theta_2, \ldots, \theta_q$是移动平均（MA）系数，表示各阶滞后的白噪声误差项的权重。e_t 表示白噪声误差项，是随机扰动项。

ADF 检验是一种用于检验时间序列数据是否具有非平稳性的统计方法。它的基本思想是通过对时间序列进行差分，然后利用单位根检验来判断序列是否平稳。

首先，根据附件 3，对于每一蔬菜品类，我们对其中的所有单品蔬菜的批发价格进行加权平均得到各蔬菜品类的批发价格，然后，我们对 6 个蔬菜品类的批发价格数据进行了 ADF 检验，确定这些时间序列数据是否具有平稳性。通过 ADF 检验，我们得到了各蔬菜品类批发价格的 P 值如下：

表 8 初始 ADF 检验结果

品类	花叶类	花菜类	水生根茎类	茄类	辣椒类	食用菌
P 值	0.0030	0.2068	0.0018	0.1087	0.0732	0.0059

如上表所示，花菜类、茄类与辣椒类的批发价格的 P 值大于 0.05，这意味着在 5%的显著性水平下，我们不能拒绝数据具有非平稳性的原假设。这提示我们需要对这三个蔬菜品类的批发价格数据进行一阶差分处理，将非平稳序列转化为平稳序列，使其更适用于时间序列分析和建模。

我们对进行一阶差分后的数据进行了 ADF 检验，以进一步验证平稳性。以下为对蔬菜品类批发价格一阶差分后的 ADF 检验 P 值：

表 9 一阶差分后 ADF 检验结果

品类	花菜类	茄类	辣椒类
P 值	0.2068	0.1087	0.0732

根据表格所示，各蔬菜品类批发价格的 P 值均小于 0.05，在 5%的显著水平下，

我们拒绝了原假设，这表明一阶差分后，各蔬菜品类批发价格数据已经成功实现了平稳化。

贝叶斯（BIC）准则是一种模型选择准则，通常用于确定在给定数据下，哪个统计模型更优。

我们采用 MATLAB 对各品类蔬菜批发价格数据进行了 ARIMA 模型参数搜索，具体是对 P（自回归阶数）和 Q（移动平均阶数）值进行了从 1 到 5 的遍历搜寻，并计算了相应的 BIC 值（贝叶斯信息准则）。旨在找到最佳的 ARIMA 模型，以更准确地描述各蔬菜品类的价格趋势。

最终，我们为各品类蔬菜选择了最优的 ARIMA 模型，这些模型的具体参数如下：

表 10 六个品类 ARIMA 模型的参数

类别	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
p	0	0	3	0	0	1
d	0	1	0	1	1	0
q	2	0	4	1	1	1

最后，我们利用所确定的 ARIMA 模型预测各品类蔬菜 2023 年 7 月 1 日至 2023 年 7 月 7 日的批发价格。结果如下：

表 11 7 月 1 日至 7 月 7 日各品类批发价格预测结果

类别	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
2023-7-1	3.6669	7.8135	11.8873	4.4427	5.9474	4.6856
2023-7-2	3.6876	7.8137	11.7604	4.4429	5.9462	4.6917
2023-7-3	3.6861	7.8138	11.7049	4.4431	5.9451	4.6900
2023-7-4	3.6847	7.8140	11.7488	4.4434	5.9439	4.6874
2023-7-5	3.6832	7.8142	11.6359	4.4436	5.9427	4.6846
2023-7-6	3.6818	7.8143	11.6397	4.4438	5.9416	4.6818
2023-7-7	3.6803	7.8145	11.6507	4.4440	5.9404	4.6789

7.2.2 目标函数的建立

商超的核心经营目标是追求收益的最大化。建立目标函数时应考虑三个关键要素：销售收入、销售成本、折扣损失。由于题目要求分析损耗率与折扣，该部分对这两个要点进行详细解释与量化。

我们定义i = 1，2,…，6表示第i个蔬菜品类，j = 1，2,…，7表示 2023 年 7 月j日，t = 1，2,…，13 表示每天的9 时，10 时，…，22 时。

- **销售收入与销售成本**

假设第i个蔬菜品类在 2023 年 7 月 j 日原始单价（打折前定价）为P_{ij}，实际销售总量为Q_{ij}，基于这两个关键变量，我们可以计算出 2023 年 7 月 j 日该蔬菜品类的销售收入为$P_{ij}Q_{ij}$。

根据上文 ARIMA 模型预测结果，我们得到 2023 年 7 月 1 日至 2023 年 7 月 7 日的批发价格信息。我们将第i个蔬菜品类在 2023 年 7 月j日的综合批发价格表示为C_{ij}。

根据附件 4，我们已知各蔬菜单品的损耗率。对于每一蔬菜品类，我们以销售价格为权数，对该类所有单品蔬菜的损耗率进行加权平均，从而得到该品类的损耗率。我们将第i个蔬菜品类的损耗率为α_i。

表 12 不同品类平均损耗率

品类	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
平均损耗率(%)	10.2803	14.1420	11.9747	7.1220	8.5153	8.1310

我们定义，损耗率表示补货量与销售量的差值占进货量的比重。我们假设对于第i个品类，批发总价为 A 元，补货量为 B 千克，则批发价格为A/B元。由于从补货到销售过程存在损耗，我们实际销售量为（$1-\alpha_i$）$\times B$，故实际单位成本为$A/$[（$1-\alpha_i$）$\times B$]。根据上述推导，我们知道真实单位成本，应该是：

$$c^* = \frac{c}{1-\alpha_i} \tag{9}$$

由此，我们得到 2023 年 7 月j日第i个蔬菜品类的销售收入为$\frac{c_{ij}}{1-\alpha_i}Q_{ij}$。

根据上述内容，将收入与成本相减，再进行求和，我们得到 2023 年 7 月 1-7 日的初始利润为：

$$z_1 = \sum_{i=1}^{6}\sum_{j=1}^{7}(P_{ij} - \frac{c_{ij}}{1-\alpha_i})Q_{ij} \tag{10}$$

- **折扣损失**

由于折扣会降低蔬菜商品定价，带来一部分损失，我们现对折扣损失进行量化。

为了更深入地了解每一种蔬菜品类的销售动态，我们首先计算第 i 个蔬菜品类在时间 t 的销售量相对于当日总销售量的占比，记作w_{it}。我们将不同品类过去三年 9 时、10 时、••• 22 时的销量分别除以该品类在这三年的总销量，得到如下结果：

表 13 w_{it}参数具体值

时间	9	10	11	12	13	14	15
品类 1	9.34%	14.73%	11.13%	5.72%	4.04%	4.60%	6.94%
品类 2	8.18%	13.74%	11.05%	5.59%	4.27%	4.87%	6.94%
品类 3	12.85%	18.74%	13.16%	6.08%	4.41%	4.91%	6.94%
品类 4	7.00%	11.92%	9.82%	5.05%	3.89%	4.75%	7.33%
品类 5	6.83%	12.28%	10.19%	5.62%	4.39%	5.42%	7.81%
品类 6	7.51%	13.32%	10.89%	5.75%	4.14%	4.86%	7.46%
时间	16	17	18	19	20	21	22
品类 1	9.62%	10.95%	10.72%	6.80%	4.24%	1.16%	0.01%
品类 2	9.17%	10.36%	9.71%	7.40%	6.65%	2.07%	0.02%
品类 3	8.54%	8.99%	7.51%	4.06%	2.85%	0.95%	0.01%
品类 4	9.46%	10.92%	10.56%	8.26%	8.02%	2.97%	0.06%
品类 5	9.80%	10.43%	10.44%	7.33%	6.80%	2.64%	0.04%
品类 6	9.72%	10.73%	10.18%	6.92%	6.26%	2.22%	0.03%

基于这一比重，我们可以进一步推算出第 i 个蔬菜品类在第 t 个时间的实际销售收入。具体来说，这个时间段内的销售收入为$\omega_{it}P_{ij}Q_{ij}$。

为了评估折扣力度对销售策略的影响，我们计算第 t 个时间的打折销售笔数与总

交易笔数的比值，并将这个比值记作 η_t，反映了在特定时间 t 的蔬菜品类打折的概率（计算结果见下表）。因此，这个时间段用于计算打折的销售收入为$\eta_t \omega_{it} P_{ij} Q_{ij}$。

表 14 不同时间打折概率

时间（时）	9	10	11	12	13	14	15
打折概率	3.06%	2.99%	2.72%	2.78%	2.55%	2.72%	2.61%
时间（时）	16	17	18	19	20	21	22
打折概率	2.74%	3.15%	4.77%	12.33%	21.83%	26.65%	38.52%

接着，为了更准确地了解各品类蔬菜的实际折扣程度，我们从每一种蔬菜品类中随机抽取 10 个打折后的销售价格，并计算这些价格与其对应日期的原价的比值。这样，我们可以估算出该品类蔬菜的平均折扣率，记作β_i（结果见下表）。因此，对于第 i 个品类，其在第 j 日第 t 个时间段因打折带来的损失为$\eta_t \beta_i \omega_{it} P_{ij} Q_{ij}$。

表 15 各品类蔬菜的平均折扣率

品类	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
折扣率	20%	20%	20%	20%	30%	40%

对该结果进行求和，得到打折带来的损失为：

$$z_2 = \sum_{i=1}^{6} \sum_{j=1}^{7} \sum_{t=1}^{14} \eta_t \beta_i \omega_{it} P_{ij} Q_{ij} \tag{11}$$

- **最终目标函数**

根据以上两部分，我们将销售收入减去销售成本减去打折损失，得到最终目标函数：

$$\max z = \sum_{i=1}^{6} \sum_{j=1}^{7} \left(P_{ij} - \frac{c_{ij}}{1-\alpha_i} \right) Q_{ij} - \sum_{i=1}^{6} \sum_{j=1}^{7} \sum_{t=1}^{14} \eta_t \beta_i \omega_{it} P_{ij} Q_{ij} \tag{12}$$

7.2.3 约束条件确定

- **约束一：蔬菜品类销售量-价格关联性约束**

根据我们的数据分析，我们已经通过线性拟合方法确定了每个蔬菜品类销售量与销售价格之间的函数关系。具体来说，对于任何给定的蔬菜品类价格P_{ij}，其预期销售量 Q_{ij} 可以由特定的函数f_i来预测。因此P_{ij}与Q_{ij}应满足等式约束：

$$Q_{ij} = f_i(P_{ij}), i = 1,2,\dots 6 \ \ j = 1,2,\dots 7 \tag{13}$$

- **约束二：蔬菜品类销售量界限约束**

为确保预测的蔬菜品类销售总量与过去实际销售数据相符，我们引入了基于历史数据的约束。具体地说，对于第 i 个蔬菜品类，我们已经统计出其近三年中的最大日销售总量和最小日销售总量，分别记为$Maxsale_i$和$Minsale_i$。

表 16 不同品类日销量上界与下界

品类	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
最大值	1265.473	186.155	296.792	118.931	604.231	511.136
最小值	1	0.632	0.926	0.252	5	3.012

这为我们提供了一个实际的参考范围，使我们避免对该品类的过高或过低估计。因此，为确保我们的预测与实际历史数据一致，我们设定以下的销售量可行域约束：

$$Minsale_i \le Q_i \le Maxsale_i, i = 1,2,\dots,6 \tag{14}$$

- **约束三：蔬菜品类定价界定约束**

为了确保我们设定的蔬菜品类价格与过去的销售数据保持一致，我们根据历史数据进行约束。具体来说，针对第 i 个蔬菜品类，我们已经根据其近三年的销售数据，确定了一个实际的价格范围，即最大销售价格$Maxvalue_i$和最小销售价格$Minvalue_i$。这样的参考范围有助于我们避免对某一蔬菜品类作出过高或过低的定价预测。

表 17 不同品类日销价上界与下界

品类	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
最大值	119.9	19.8	53.8	21.6	79.8	116
最小值	0.1	0.5	1	1.5	0.9	0.5

为此，我们设立了以下的定价区间约束来确保所定价格与历史数据相符：

$$Minvalue_i \le P_i \le Maxvalue_i, i = 1,2,\dots, 6 \tag{15}$$

7.2.4 规划模型的建立

最终，我们得到如下规划模型：

$$\max\ z = \sum_{i=1}^{6}\sum_{j=1}^{7}\left(P_{ij} - \frac{c_{ij}}{1-\alpha_i}\right)Q_{ij} - \sum_{i=1}^{6}\sum_{j=1}^{7}\sum_{t=1}^{14}\eta_t\,(1-\beta_i)\omega_{it}\,P_{ij}Q_{ij}$$

$$s.t.\begin{cases} Q_{ij} = f_i(P_{ij}), i = 1,2,\dots,6; j = 1,2,\dots,7 \\ Minsale_i \le Q_{ij} \le Maxsale_i, i = 1,2\dots,6; j = 1,2,\dots,7 \\ Minvalue_i \le P_{ij} \le Maxvalue_i, i = 1,2,\dots,6;\ j = 1,2,\dots,7 \end{cases}$$

7.3 规划问题求解

由于目标函数的 Hessian 矩阵非正定，故目标函数非凸，传统的算法难以解决此类问题。为此，我们采用 Metropolis-Hastings 算法解决此问题。

Metropolis-Hastings（M-H）算法是统计计算领域中一种卓越的蒙特卡罗马尔科夫链（MCMC）技术。这种算法主要目标是从某个特定且通常难以直接采样的复杂分布（往往是后验分布）中有效地进行样本抽取。为了实现这一目标，M-H 算法巧妙地构

建了一个马尔科夫链，并确保了这个链的平稳分布与我们期望的目标分布相符。以下概述了 Metropolis-Hastings 算法的核心步骤：

1.**初始化**：选择一个初始状态θ，并确定一个提议分布$q(\theta' \mid \theta)$，它给出了从当前状态θ转移到新状态θ'的概率。

2.**迭代**：对于每次迭代，执行以下步骤：

提议步骤：从提议分布$q(\theta' \mid \theta)$中抽样得到一个新的候选状态 θ'。

接受步骤：使用以下规则来决定是否接受这个新的候选状态：

首先，计算接受概率α：

$$\alpha(\theta, \theta') = min\,(1, \frac{p(\theta')\,q(\theta \mid \theta')}{p(\theta)\,q(\theta' \mid \theta)})$$

其中，$p(\theta)$是目标分布（通常是未归一化的后验分布）。

然后，从均匀分布$U(0,1)$中随机抽取一个值u。接着，如果$u < \alpha(\theta,\theta')$，则接受新的候选状态，即设置$\theta = \theta'$。最后，如果不接受新的候选状态，那么$\theta$保持不变。

3. **重复**：重复上述迭代过程直到达到预定的迭代次数。

我们利用 MATLAB 实现上述功能，得到如下结果：

表 18 2023 年 7 月 1-7 日补货总量

	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
2023-7-1	154.7150	65.7441	87.3709	21.2619	92.3717	26.3224
2023-7-2	148.2630	49.9931	87.6827	39.6709	103.7490	25.8309
2023-7-3	140.1741	73.0420	90.8833	46.8762	94.5461	25.4867
2023-7-4	181.5437	70.2386	92.4248	27.0068	80.4143	25.6267
2023-7-5	167.6649	50.3981	89.1336	27.7730	91.0206	21.5051
2023-7-6	130.1766	77.2107	101.8652	30.6452	90.5413	28.3547
2023-7-7	160.1794	70.1204	94.4301	43.1374	89.0182	24.8598

表 19 2023 年 7 月 1-7 日定价策略

	品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
2023-7-1	9.41	11.07	28.91	8.83	11.57	21.55
2023-7-2	9.64	11.51	28.86	6.89	10.90	22.04
2023-7-3	9.94	10.87	28.35	6.13	11.44	22.39
2023-7-4	8.43	10.95	28.10	8.23	12.27	22.25
2023-7-5	8.94	11.49	28.63	8.14	11.65	26.40
2023-7-6	10.30	10.76	26.60	7.84	11.67	19.49
2023-7-7	9.21	10.95	27.78	6.52	11.76	23.02

虽然该算法无法求出最优解，但我们可以认为所求得的结果为近似最优解。最终，商超根据所制定的 2023 年 7 月 1 日至 7 月 7 日的定价策略与补货策略，能够实现的最大收益为 22489 元。

八、 问题三的模型建立与求解

为了更好地研究单品销售方式与定价策略，我们对问题二的规划模型进行调整和改进，设计成双目标 0-1 规划模型，具体过程如下。

8.1 决策变量的引入

题目强调“根据 2023 年 6 月 24-30 日的可售品种制定策略”。因此，我们分析了 2023 年 6 月 24 日至 6 月 30 日的蔬菜单品销售数据，发现：在这一时段内，仅有 49 种蔬菜单品存在销售记录。为了更有系统地研究这些数据，我们将这 49 种蔬菜单品按其所属的蔬菜品类进行分类，详情如下：

品类1	品类2	品类3	品类4
102900011030059	102900005116714	102900005116899	102900005116257
102900011031100	102900011034026	102900051000944	102900011022764
102900005115786		102900005118824	102900005116509
102900005118831		102900011001691	102900051000463
102900005115762		102900011032732	102900011033982
102900011030110		102900011007969	
102900011008164		102900011018132	
102900005115946			
102900011023464			
102900005119975			
102900005115823			
102900005115779			
102900005115908			

图 15 出现的单品分类结果（部分，其余见附录）

我们设定 i 为蔬菜品类，并用 k 表示特定蔬菜品类下的蔬菜单品的索引。例如，$(i,k)=(1,3)$表示第 1 类第 3 个商品，即 102900005115786。在上表中，对于那些未填充的单元格，如（2,3），我们采用运筹学虚拟商品填补的思想，对空缺位置用虚拟蔬菜单品进行填补，这些虚拟蔬菜商品被设定为拥有高成本及损耗率。

我们引入二元变量 d_{ik} 来描述第 i 个品类中第 k 种蔬菜单品的补货状态。当 $d_{ik}=1$时，对该蔬菜单品进行补货；而 $d_{ik}=0$则表示不进行补货。进一步，设定第 i 个品类中第 k 种蔬菜单品的价格为P_{ik}，相应的销售量为Q_{ik}。

8.2 模型参数的确定

延续问题 2 的思路，我们已知第 t 个时间段蔬菜品类打折的概率η_t、各品类蔬菜的平均折扣率β_i以及第 i 个蔬菜品类在时间 t 的销售量相对于当日总销售量的占比w_{it}。

同时，我们根据附件 4 的数据得到第 i 个品类中第 k 个蔬菜单品的损耗率α_{ik}。特别地，对于上文提到的虚拟蔬菜单品，其损失率被设定为一个非常接近 1 的值，我们设定为 0.99。

基于近三年的蔬菜单品批发价格数据，我们对多个预测模型进行测试，最终采用 7 期滞后的 SARIMA(0,1,1)模型。我们利用该模型预测得到了第 i 个品类中第 k 个蔬菜单品在未来的批发价格，即C_{ik}。对于虚拟蔬菜单品，我们将其成本设定为很大的值，标记为 M。

8.3 目标函数的建立

现在对问题 2 中关于销售收益最大化的描述进行更新。由于在问题 3 的情境下，不是每种蔬菜单品都会进行销售，为此我们引入 0-1 决策变量d_{ik}以确定哪些蔬菜单品确实进行了销售。同时，我们对参数下标进行修改。最终，利润最大化目标函数得到修正：

$$\max\ z=\sum_{i=1}^{6}\sum_{k=1}^{17}\left(P_{ik}-\frac{C_{ik}}{1-\alpha_{ik}}\right)Q_{ik}d_{ik}-\sum_{i=1}^{6}\sum_{k=1}^{17}\sum_{t=1}^{14}\eta_t\beta_i\omega_{it}P_{ik}Q_{ik}d_{ik} \tag{16}$$

问题 3 中提及“尽量满足市场对各品类蔬菜商品需求”。对此，为了更好地满足消费者对各品类蔬菜商品的需求，我们希望最大化商超所有蔬菜单品的累计销售量。具体的数学表达为：

$$\max\ Q=\sum_{i=1}^{6}\sum_{k=1}^{17}Q_{ik}d_{ik} \tag{17}$$

根据上述步骤，我们得到了双目标 0-1 规划的目标函数。

8.4 约束条件的更新与设计

- **约束一：可售蔬菜单品数量界限约束**

鉴于商超的蔬菜销售空间所受的限制，我们必须确保可售蔬菜单品的总数维持在 27 至 33 个之间。据此，我们制定以下约束条件：

$$\begin{cases}27\le\sum_{i=1}^{6}\sum_{k=1}^{17}d_{ik}\le 33\\ d_{ik}=0\ or\ 1\end{cases} \tag{18}$$

- **约束二：最小陈列量约束**

考虑到每种蔬菜单品都需要达到最小陈列量的要求，即 2.5kg，我们根据预估的销售量来确定订购量。这意味着我们会根据预测的销售量来决定订购的数量，确保每种蔬菜单品都至少有 2.5kg 的存货。对于那些不在销售范围内的蔬菜单品，我们不考虑其订购和销售量。因此，这一约束可以如下描述：

$$(Q_{ik}-2.5)d_{ik}\ge 0 \tag{19}$$

- **约束三：蔬菜单品销售量-价格关联性约束**

我们已经通过线性拟合得出了各蔬菜品类的销售总量与销售价格间的函数关系。我们假设，对于品类 i，过去三年里第 k 个单品总销量占对应品类的比重为μ_{ik}，则对于各品类中的蔬菜单品，存在如下等式关系：

$$Q_{ik}=\mu_{ik}f_i(P_{ik}) \tag{20}$$

- **约束四：蔬菜单品销售量界限约束**

延用问题 2 的思路，我们统计出各蔬菜单品其近三年中的最大日销售量和最小日销售量，分别记为$Maxsale_{ik}$ 和$Minsale_{ik}$。则蔬菜单品的销售量应满足如下约束：

$$Minsale_{ik}\le Q_{ik}\le Maxsale_{ik}, i=1,2,\dots,6\ \ k=1,2,\dots,17 \tag{21}$$

- **约束五：蔬菜单品定价界定约束**

与问题 2 的分析方法相似，我们针对每种蔬菜单品统计了过去三年内的最高销售

价格$Maxvalue_{ik}$和最小销售价格$Minvalue_{ik}$。因此，为确保蔬菜单品的销售价格在一个合理的范围内，我们设置以下的价格约束条件：

$$Minvalue_{ik} \le P_{ik} \le Maxvalue_{ik}, i = 1,2,\dots，\ 6\ \ k = 1,2,\dots,17 \tag{22}$$

8.5 最终规划模型的确定

最终，我们得到如下双目标 0-1 规划模型：

$$\max\ z = \sum_{i=1}^{6}\sum_{k=1}^{17}\left(P_{ik} - \frac{C_{ik}}{1-\alpha_{ik}}\right)Q_{ik}d_{ik} - \sum_{i=1}^{6}\sum_{k=1}^{17}\sum_{t=1}^{14}\eta_t\beta_i\omega_{it}P_{ik}Q_{ik}d_{ik}$$

$$\max\ Q = \sum_{i=1}^{6}\sum_{k=1}^{17}Q_{ik}d_{ik}$$

$$s.t.\begin{cases} d_{ik} = 0\ or\ 1 \\ 27 \le \sum_{i=1}^{6}\sum_{k=1}^{17} d_{ik} \le 33 \\ Q_{ik} = \mu_{ik}f_i(P_{ik}) \\ (Q_{ik} - 2.5)d_{ik} \ge 0 \\ Minsale_{ik} \le P_{ik} \le Maxsale_{ik}, i = 1,2,\dots,6; k = 1,2,\dots,17 \\ Minsale_{ik} \le Q_{ik} \le Maxsale_{ik},, i = 1,2,\dots,6; k = 1,2,\dots,17 \end{cases}$$

8.6 双目标 0-1 规划模型求解

在西方经济学理论中，利润最大化是企业经营的第一目标，因此利润最大化是商超经营的首要目标。对于这一双目标规划问题，我们采用ε约束法，该方法又称参考目标法，需要设置决策者对次要目标容许接受阈值。

基于过去三年的数据，我们发现有 85.7%的日期日销量小于 600，比例足够高。因此，我们将日销量为 600 作为阈值，对另一个目标函数进行约束。

遗传算法（Genetic Algorithm, GA）是模拟自然选择和自然遗传机制的搜索算法。它是通过模拟生物进化过程中的自然选择、交叉、变异等机制，从而对解空间进行搜索的优化方法。遗传算法常用于求解一些复杂的、难以通过传统算法解决的问题。

该方法的优点是能够在大范围内快速搜索，对于某些复杂的非线性、非凸问题，可能找到比传统算法更好的解。但 GA 不能保证找到全局最优解，通常找到的是近似最优解。此外，遗传算法的参数（如交叉率、变异率等）的选择也会影响算法的效果，需要根据具体问题进行调整。

对ε约束法处理后的规划问题，我们采用遗传算法计算局部最优解，关键参数如下：

表 20 遗传算法关键参数信息

参数	PopulationSize	Generations	CrossoverFcn
信息	1000	1000	@crossovertwopoint
参数	MutationFcn	SelectionFcn	FunctionTolerance
信息	@mutationadaptfeasible)	@selectiontournament	1.00E-06

通过 MATLAB，我们得到相应的d_{ik}、Q_{ik}、P_{ik}。我们对结果进行整理，决策变量计算结果见下表。最终得到的 2023 年 7 月 1 日的最大收益为 1415.77 元

表 21 最终结果展示

单品名称	销售价格	补货量	单品名称	销售价格	补货量
云南生菜(份)	3.17	22.74	海鲜菇(包)	8.35	9.43
净藕(1)	9.05	14.76	娃娃菜	3.26	21.67
金针菇(盒)	8.26	9.47	菱角	8.86	15.29
小米椒(份)	2.89	26.24	螺丝椒(份)	3.56	16.17
芜湖青椒(1)	3.47	16.85	苋菜	2.96	25.33
双孢菇(盒)	8.60	9.32	高瓜(2)	9.50	13.50
竹叶菜	2.92	25.92	姜蒜小米椒组合装(小份)	3.26	18.42
高瓜(1)	9.86	12.48	虫草花(份)	8.50	9.36
小皱皮(份)	3.21	18.84	菠菜(份)	3.25	21.78
奶白菜	3.03	24.49	红莲藕带	8.19	17.18
野生粉藕	10.46	10.77	青线椒(份)	3.33	17.88
青红杭椒组合装(份)	3.45	17.01	上海青	3.30	21.18
木耳菜	3.07	23.99	云南生菜	3.64	16.91
鲜木耳(份)	8.29	9.46	菜心	3.30	21.15
小青菜(1)	3.01	24.75	外地茼蒿	3.55	18.10
红薯尖	2.98	25.08	云南油麦菜	3.45	19.27
木耳菜(份)	2.74	28.08			

九、 问题四的分析

9.1 数据 1：损耗率分布情况

商品损耗包括三个部分，分别是商品运损、品相变差以及其它部分。因此，商品损耗率满足如下公式：

$$\alpha = \alpha_{transport} + \alpha_{freshness} + \alpha_{other} \tag{23}$$

其中，α表示总计损耗率，$\alpha_{transport}$表示运损损耗率，$\alpha_{freshness}$表示因保鲜问题造成的损耗率，α_{other}表示其余损耗率。

对运损率拆分处理可以进一步考虑时间的影响，提高目标函数计算的准确率。以式（10）为例，我们先将α中的$\alpha_{freshness}$提出来，得到如下公式：

$$z_1 = \sum_{i=1}^{6}\sum_{j=1}^{7}(P_{ij} - \frac{c_{ij}}{1-\alpha_{transport}-\alpha_{other}})Q_{ij} \tag{24}$$

由于保鲜损耗率会随时间推进而上升，因此我们可以挖掘保鲜损耗率与时间 t 的关系，及其共同作用对利润带来的负面影响，修正结果见下：

$$z_1 = \sum_{i=1}^{6}\sum_{j=1}^{7}(P_{ij} - \frac{c_{ij}}{1-\alpha_{transport}-\alpha_{other}})Q_{ij} - g(\alpha_{freshness}, t) \tag{25}$$

上述修正可以帮助决策者，依据不同品类在不同时间段的保鲜情况，优化补货总量和定价策略，与实际情况更相符。

9.2 数据 2：未售出商品占比

题目指出“大部分品种当日未售出，则次日无法销售”。因此，部分品种因未及时出售而作废处理，即现实中补货量会高于销售量。若我们能够得到不同日期未售出商品占比，则可以根据该比率的变化趋势，设计补货策略。

例如，我们现在有θ_1，θ_2，……，θ_t，这些是不同日期的未售出商品占比。我们可采用时间序列预测等方法，预测未来可能的未售出商品占比，通过如下公式预测合理的补货量：

$$Q_{\text{补货}} = \frac{Q_{\text{销售}}}{1 - f(\theta_1,\ \theta_2,\ \cdots\cdots,\ \theta_t)} \tag{26}$$

9.3 数据 3：竞争对手销量及价格数据

在实证研究中，地域性因素在商业决策中常被视为一个关键变量。对于地理位置相近的商超来说，其销售数据和价格策略往往显示出显著的相关性。这种现象可以归因于消费者偏好、地域经济条件、物流成本等共同作用的结果。因此，收集并整合竞争对手在相同或相似地域的销量及价格数据，不仅可以丰富我们的数据集，从而提高统计分析的可靠性，还可以为前文所述的各类模型拟合、时间序列预测以及规划参数的设定提供更稳健的依据。

此外，除了传统的数据驱动策略之外，我们还可以借助现代决策理论，如博弈论，来加深对竞争对手定价策略的理解。通过对竞争对手的定价模式进行博弈分析，我们不仅可以预测其可能的策略反应，还可以优化自身的定价策略，以确保在市场竞争中获得更大的战略优势。简言之，结合博弈论的理论框架与实证数据，可以使我们的定价策略更加科学、合理且富有实践意义。

十、 模型的评价与推广

10.1 模型的优点

1. 模型量化了多个附件中的数据，便于后续处理分析，通过数据驱动使其更为准确和可靠；

2 本文模型涵盖了众多因素，包括销售量的关联关系、供应品种、季节性等，从多维度入手有效解决了模型问题；

3.本文问题一中的数据分析可视化易于后续模型解释处理，且建立的规划模型易于推广，模型建立层层递进，合理高效地解决实际需求。

10.2 模型的缺点

1.尽管模型预测批发价时考虑了蔬菜供应的季节性变动，但其他未考虑的动态因素，如突发事件、天气条件等，由于动态因素限制，可能会影响其效果；
2.本文对于一些边际效应没有进一步考虑，在某些情况下，小规模的补货和定价调整可能导致较大的收益变动，模型需要更细致的优化来处理这些情况。

10.3 模型的推广

蔬菜定价模型为我们提供了一个广泛适用的框架，不仅限于蔬菜，也可推广至其他保质期短的农产品，如水果和肉类。因此，利用蔬菜定价模型的基础结构，我们可以针对不同农产品的特点进行调整，形成更为合适的、更具实用价值的定价策略，从而为农业生产者和销售商带来更大的经济效益。

十一、 参考文献

[1] 刘子军.基于 Pearson 相关系数的低渗透砂岩油藏重复压裂井优选方法[J].油气地质与采收率,2022,29(02):140-144.
[2] 彭玮,龚俊梅.基于系统聚类法的返贫风险预警机制分析[J].江汉论坛,2021(12):23-31.
[3] 胡军,张超,陈平雁.非参数双变量相关分析方法 Spearman 和 Kendall 的 Monte Carlo 模拟比较[J].中国卫生统计,2008,25(06):590-591.
[4] 张辉锋,景恬.成本加成与消费者感知价值的结合:知识付费产品的定价模型[J].新闻与传播研究,2021,28(01):38-51+127.
[5] 翟静,曹俊.基于时间序列 ARIMA 与 BP 神经网络的组合预测模型[J].统计与决策,2016(04):29-32.

附录

附录 1

介绍：支撑材料的文件列表

表格：

支撑材料 1：附件 3 合并附件 1 后.xlsx

文件夹：

预处理代码
pearson 相关系数代码
spearman 相关系数代码
时间序列（问题 2）
M-H 算法
SARIMA（问题三）
遗传算法

附录 2

介绍：第一张是部分退款。第二章是部分退货

销售日期	扫码销售时间	单品编码	销量(千克)	销售单价(元/千克)	是否打折销售
2020-07-07	17	102900005118831	-1	6.5	否
2020-07-08	10	102900005118831	-2	6.9	否
2020-07-16	16	102900005118831	-1	5.7	否
2020-07-16	18	102900005118831	-1	5.5	否
2020-07-17	15	102900005118831	-2	6.1	否
2020-07-17	15	102900005118831	-1	5.2	否
2020-07-17	16	102900005118831	-1	5	否
2020-07-17	19	102900005118831	-1	5.5	否
2020-07-17	19	102900005118831	-1	6	否
2021-09-21	15	102900011016701	-0.246	9	否
2021-09-25	11	102900005116714	-1.269	7.4	否
2022-09-09	9	102900011033944	-0.207	9	否

销售日期	扫码销售时间（时）	单品编码	销量(千克)	销售单价(元/千克)	是否打折
2020-09-24	13	102900005119098	-0.325	16	是
2020-11-12	11	102900005116899	-0.714	8	否
2020-12-20	12	106930274220092	-1	100	否
2021-01-02	20	102900005117056	-0.92	16	否
2021-02-10	21	102900005115960	-4.128	3	否
2021-03-07	13	102900005116790	-6.505	6	否
2021-08-05	15	102900005116899	-0.778	12	否
2022-04-30	13	102900011032787	-1	2.9	否

2022-06-23	12	102900011030608	-1	5	否
2022-08-10	12	102900011033906	-2.804	3	否
2022-08-19	17	102900011034224	-2	2.9	否
2022-09-09	9	102900011032732	-0.172	17.8	否
2022-10-12	13	102900011031100	-2	3.5	否
2022-11-18	19	102900011033906	-9.082	2	否
2022-12-16	16	102900011034330	-2	5.9	否
2022-12-22	19	102900005122654	-0.502	12	否
2023-04-15	15	102900011010891	-0.492	10	否
2023-04-28	13	102900011010891	-0.402	10	否
2023-05-11	17	102900011016701	-0.335	7.2	否

附录 3

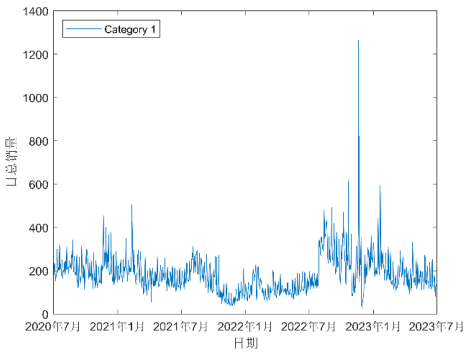
介绍：预处理 MATLAB 代码

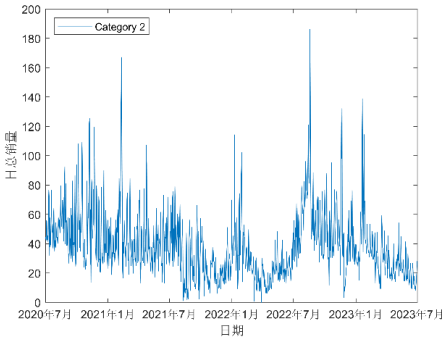
```
clc;clear;
%% 导入电子表格中的数据
% 用于从以下电子表格导入数据的脚本:
%
%     工作簿: C:\Users\handsomeju\Desktop\附件 2 量化处理后.xlsx
%     工作表: Sheet1
%
% 由 MATLAB 于 2023-09-08 09:07:03 自动生成

%% 设置导入选项并导入数据
opts = spreadsheetImportOptions("NumVariables", 7);

% 指定工作表和范围
opts.Sheet = "Sheet1";
opts.DataRange = "A2:G878504";

% 指定列名称和类型
opts.VariableNames = ["date", "hour", "num", "sale_v", "sale_p", "sale_r", "sale_d"];
opts.VariableTypes = ["datetime", "double", "double", "double", "double", "double",
"double"];

% 导入数据
data = readtable("C:\Users\handsomeju\Desktop\附件 2 量化处理后.xlsx", opts,
"UseExcel", false);


%% 清除临时变量
clear opts

%%
```

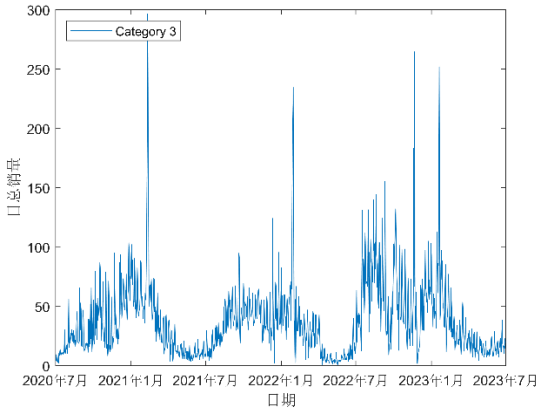
```
% %1.销售时间异常值初步处理
a=find(data.hour==8|data.hour==23)%找到异常交易时间
data(a,:) = [];   % 删除指定的行

%%
% 2.销量异常值初步处理
b=find(data.sale_v==160)
data(b,:)=[];

%%
%3.处理退款数据，销量全退情况下，全额退款标记为 1，部分退款记为 2.其余标记为 0
data.refund_flag = zeros(height(data), 1);
% 对于每一行数据，检查是否有匹配的行满足上述条件
for i = 1:height(data)
    if data.sale_r(i) == 1 % 只检查标记为退款的行
        % 查找具有相同日期、编号和非退款状态的行
        if iscell(data.date)
            matchedRows = data(strcmp(data.date, data.date{i}) & data.num == data.num(i) & data.sale_r == 0, :);
        else
            matchedRows = data(data.date == data.date(i) & data.num == data.num(i) & data.sale_r == 0, :);
        end

        if ~isempty(matchedRows) % 如果找到匹配的行
            % 检查是否有全额退款
            fullRefund = any(abs(matchedRows.sale_v) == abs(data.sale_v(i)) & matchedRows.sale_p == data.sale_p(i));
            % 检查是否有部分退款
            partialRefund = any(abs(matchedRows.sale_v) == abs(data.sale_v(i)) & matchedRows.sale_p ~= data.sale_p(i));

            if fullRefund
                data.refund_flag(i) = 1;
            elseif partialRefund
                data.refund_flag(i) = 2;
            end
        end
    end
end

% 显示带有 refund_flag 列的更新数据
disp(data);
```

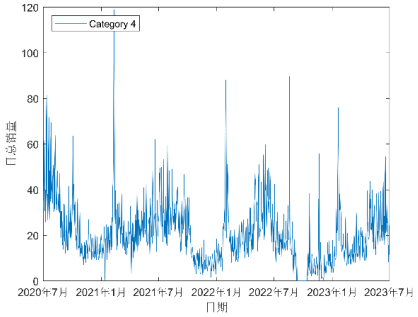
```
aa=data((data.refund_flag==1),:)%全额退款数据
ap=data((data.refund_flag==2),:)%销量全退的情况下，部分退款的数据
pp=data((data.refund_flag==0&data.sale_r==1),:)%仅退部分购买量以及部分退款的数据

%%
%4.1 删除完全退款
% 首先，获取所有标记为完全退款的记录的索引
full_refund_indices = find(data.refund_flag == 1);

% 初始化一个空数组来存储购买记录的索引
purchase_indices = [];

% 遍历每一个完全退款记录，找到其对应的购买记录
for i = 1:length(full_refund_indices)
    idx = full_refund_indices(i);
    matching_purchase = find(data.date == data.date(idx) & data.num == data.num(idx) & ...
                             data.sale_v == -data.sale_v(idx) & data.sale_p == data.sale_p(idx) & ...
                             data.sale_r == 0);
    purchase_indices = [purchase_indices; matching_purchase];
end

% 只取独特的 429 条购买记录的索引
unique_purchase_indices = unique(purchase_indices, 'stable');
unique_purchase_indices = unique_purchase_indices(1:429);

% 结合完全退款的索引和独特的购买记录索引
all_rows_to_delete = [full_refund_indices; unique_purchase_indices];

% 删除这些记录
data(all_rows_to_delete, :) = [];
%%
%4.2 删除其余退款
data(data.refund_flag==2|(data.refund_flag==0&data.sale_r==1),:)=[];
%此时 data1 为处理完退款后的数据
```

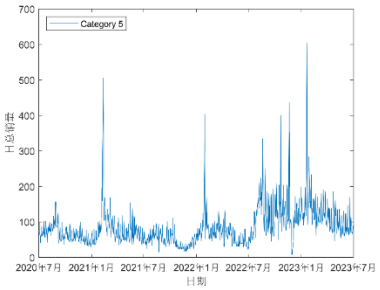
附录 4
介绍：品类间分布规律图

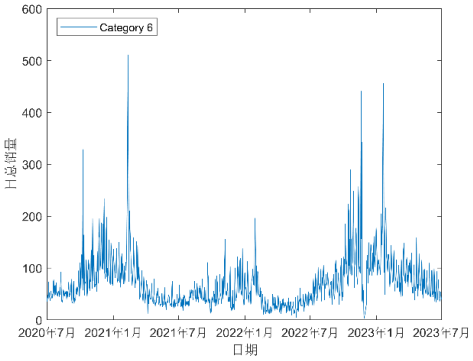

Category 1
日总销量
日期

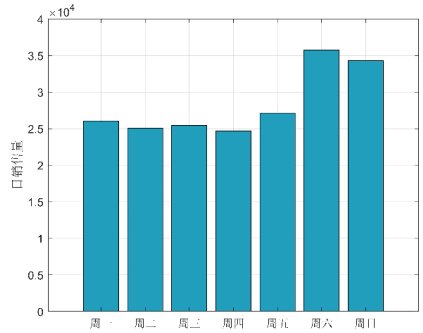

Category 2
日总销量
日期

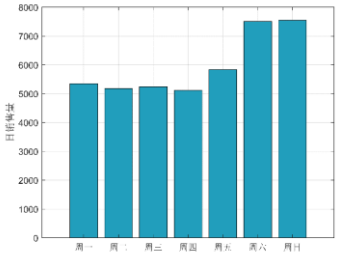

Category 3
日总销量
日期

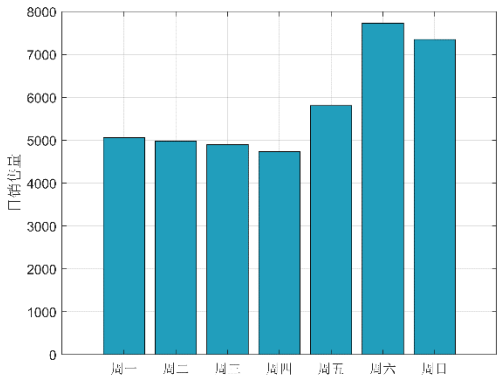

Category 4
日总销量
日期

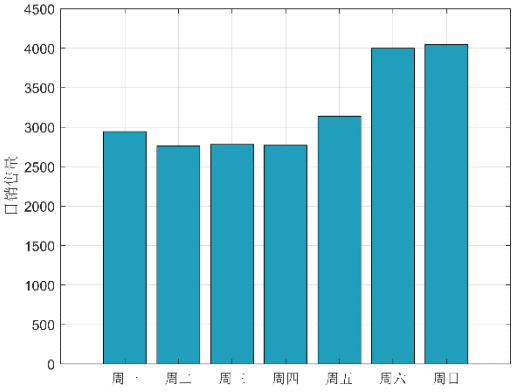

Category 5
日总销量
日期

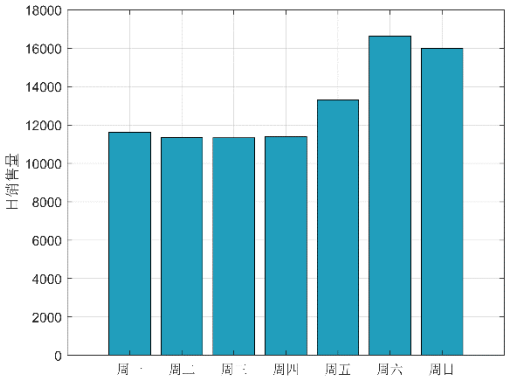

Category 6
日总销量
日期

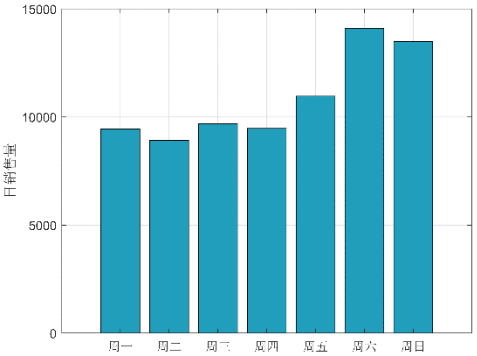

×10^4
日销售量
周一
周二
周三
周四
周五
周六
周日

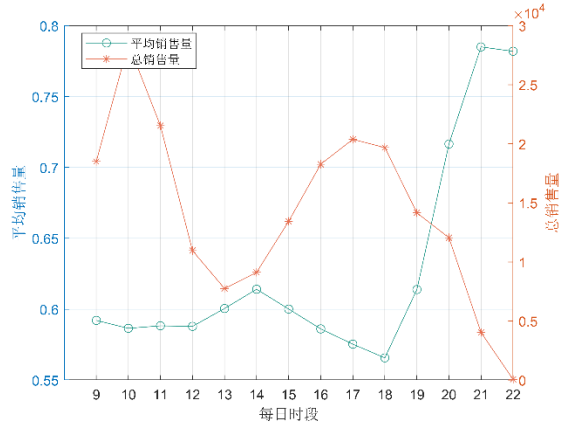

日销售量

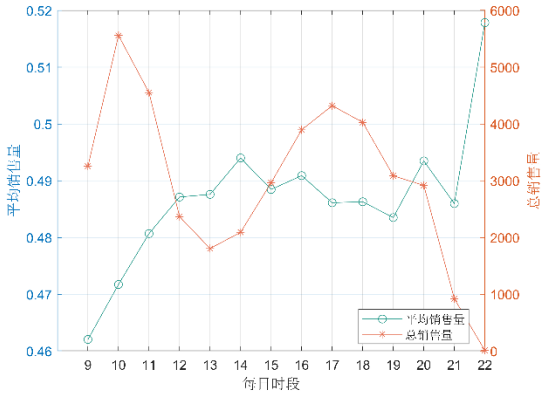

日销售量
周一
周二
周三
周四
周五
周六
周日

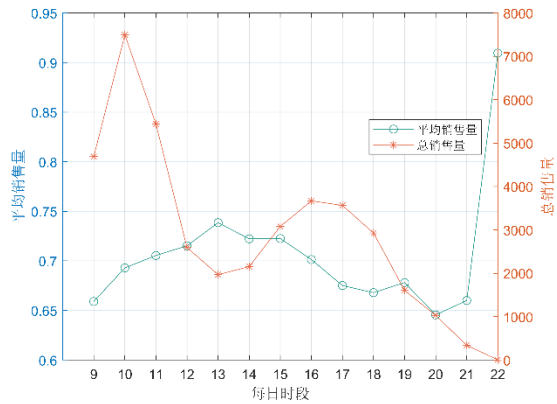

日销售量
周一
周二
周三
周四
周五
周六
周日

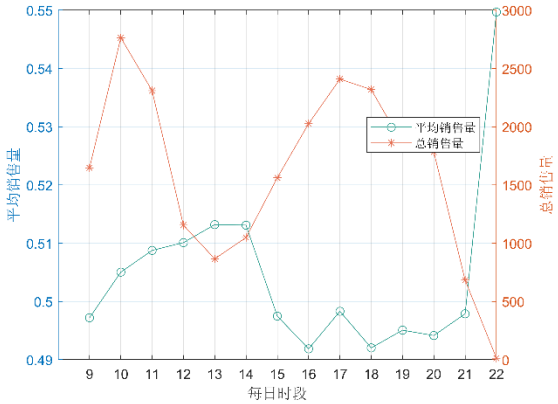

日销售量
周一
周二
周三
周四
周五
周六
周日

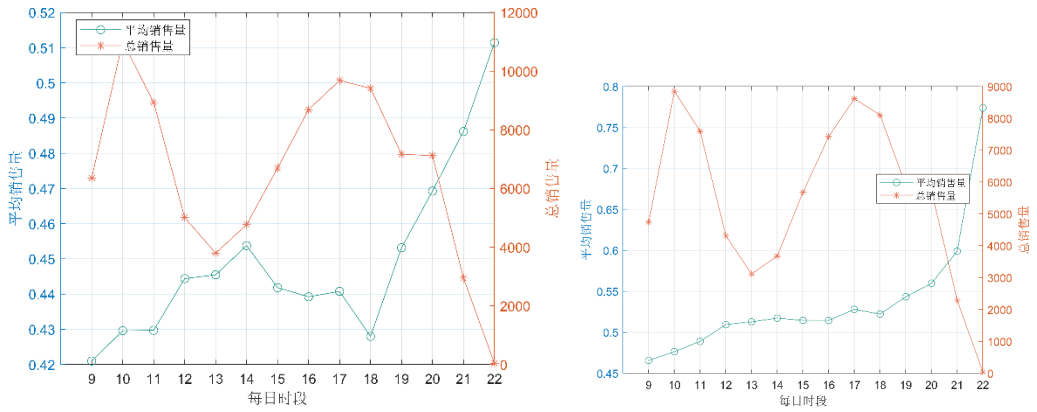

日销售量
周一
周二
周三
周四
周五
周六
周日


平均销售量
总销售量
×10^4
每日时段


平均销售量
总销售量
每日时段


平均销售量
总销售量
每日时段


平均销售量
总销售量
每日时段



附录 5

介绍：k-s 检验及 Pearson 相关系数代码

```
%%
clc;clear;
load data_q1.mat;

%%
%一.1.创建月销量数据集
categories = unique(data_q1.sort_num);

% 为数据集中的每一个日期分配一个月份标签
months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];
monthIndices = data_q1.month;

% 初始化一个表来保存月销售总和
salesSummaryByMonth = array2table(zeros(12, length(categories)), 'VariableNames', cellstr("Category" + string(categories)), 'RowNames', cellstr(months));

% 按品类和月份进行分组，计算销售量总和
for cat = 1:length(categories)
    for m = 1:12
        idx = (data_q1.sort_num == categories(cat)) & (monthIndices == m);
        currentMonthSales = sum(data_q1.sale_v(idx));
        salesSummaryByMonth{m, cat} = currentMonthSales;
    end
end
```

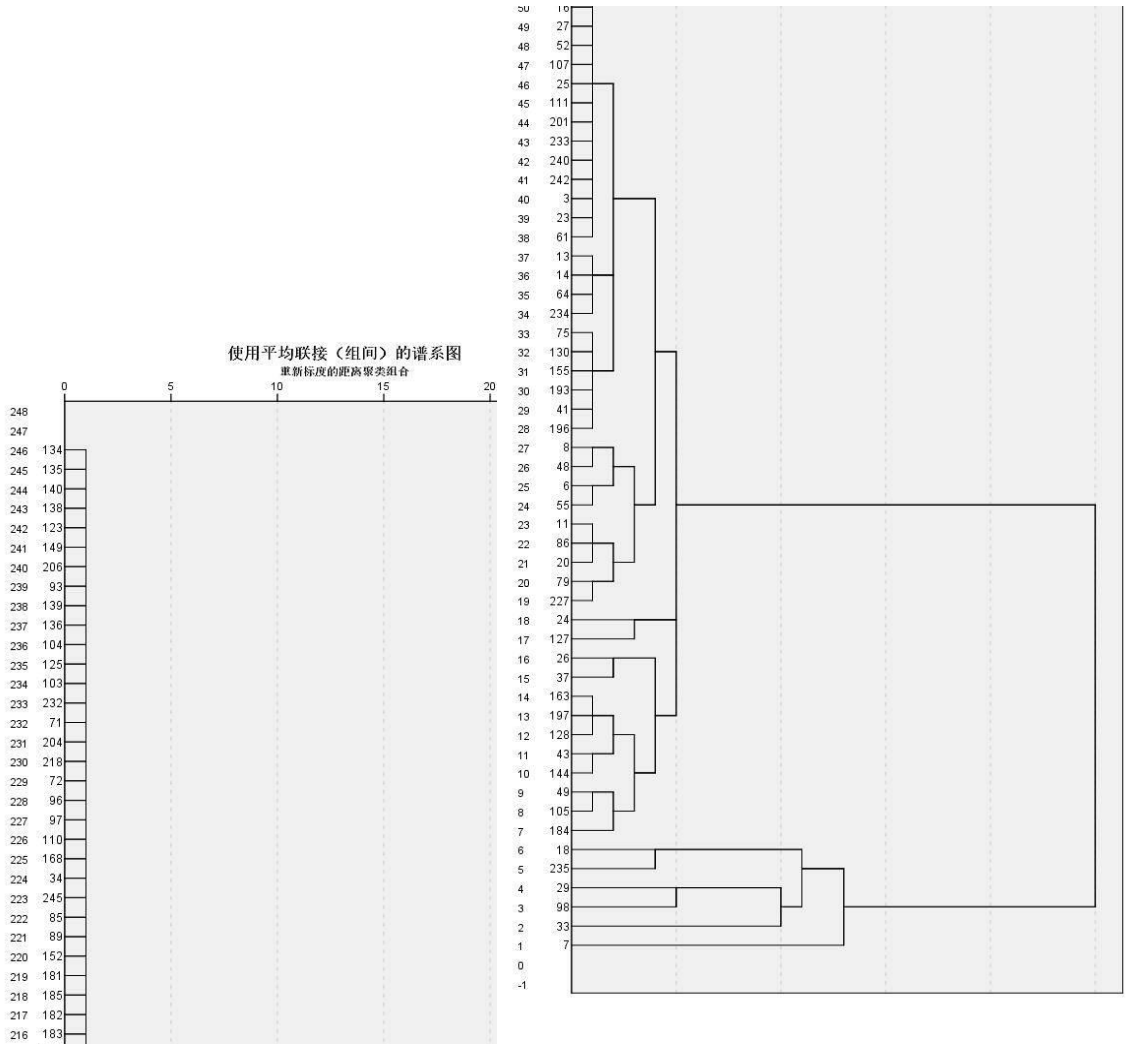
```
%%
%2.对月数据进行正态性检验
% pValues = zeros(1, size(salesSummaryByMonth, 2)); %  初始化 p 值数组
%
% for cat = 1:size(salesSummaryByMonth, 2)
%       %  提取当前品类的月销售数据
%       currentCategorySales = salesSummaryByMonth{:, cat};
%
%       %  使用 Kolmogorov-Smirnov 检验进行正态性检验
%                       [~,     pValues(cat)]     =     kstest((currentCategorySales     -
mean(currentCategorySales)) / std(currentCategorySales));
% end
%
% disp(pValues);
pValues = zeros(1, size(salesSummaryByMonth, 2)); %  初始化 p 值数组
ksValues = zeros(1, size(salesSummaryByMonth, 2)); %  初始化 KS 值数组

for cat = 1:size(salesSummaryByMonth, 2)
    %  提取当前品类的日销售数据并标准化
    currentCategorySales = salesSummaryByMonth{:, cat};
    standardizedSales  =  (currentCategorySales  -  mean(currentCategorySales))  /
std(currentCategorySales);

    %  使用 Kolmogorov-Smirnov 检验进行正态性检验
    [h, p, ksstat] = kstest(standardizedSales);

    pValues(cat) = p;
    ksValues(cat) = ksstat;
end

disp('p-values:');
disp(pValues);

disp('KS values:');
disp(ksValues);


%%
%3.进行线性性分析

[numDays, numCategories] = size(salesSummaryByMonth);

figure('Position', [100, 100, 1200, 1200]);   %  创建一个大的图形窗口
```

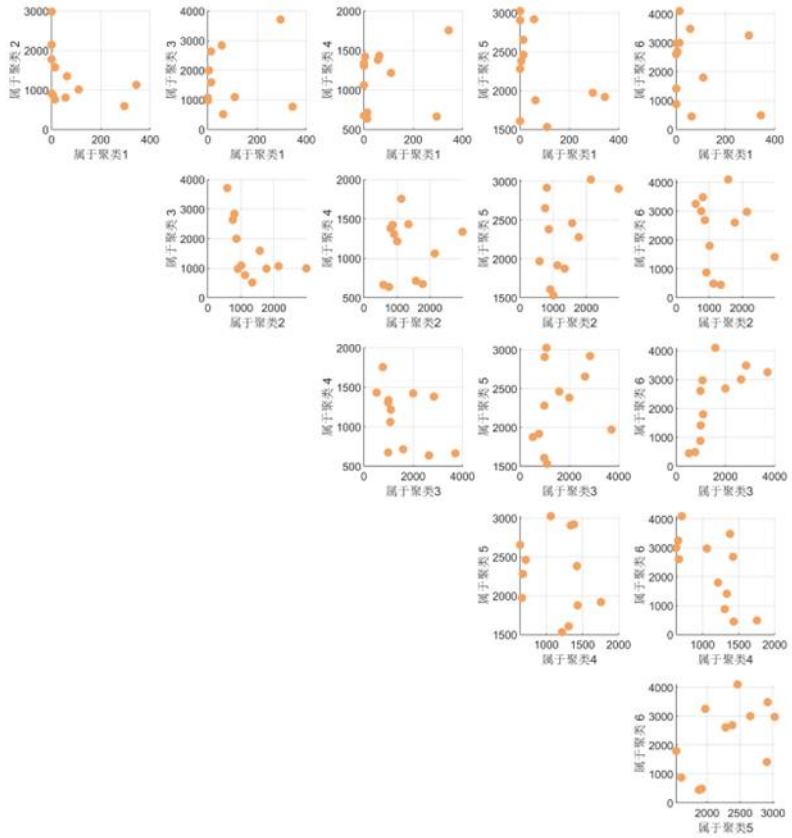
```
    % 遍历所有品类组合
    for i = 1:numCategories
        for j = i+1:numCategories
            subplot(numCategories-1, numCategories-1, (i-1)*(numCategories-1) + j-
1); % 在右上三角形中选择适当的位置
            scatter(salesSummaryByMonth{:, i}, salesSummaryByMonth{:, j}, 'filled',
'MarkerEdgeColor', [0.13, 0.62, 0.74], 'MarkerFaceColor', [0.13, 0.62, 0.74]);
            xlabel(['品类 ' num2str(i)]);
            ylabel(['品类 ' num2str(j)]);
            grid on;
        end
    end

    %%
    % 4.计算皮尔逊相关性系数及其 p 值
    [numDays, numCategories] = size(salesSummaryByMonth);
    pearsonCoefficients = zeros(numCategories, numCategories);
    pValues = zeros(numCategories, numCategories);

    for i = 1:numCategories
        for j = 1:numCategories
            [r, p] = corr(salesSummaryByMonth{:, i}, salesSummaryByMonth{:, j},
'Type', 'Pearson');
            pearsonCoefficients(i, j) = r;
            pValues(i, j) = p;
        end
    end

    % 生成带有星星标记的相关性系数矩阵
    coeffMatrixWithStars = strings(numCategories, numCategories);

    for i = 1:numCategories
        for j = 1:numCategories
            if pValues(i, j) < 0.01
                coeffMatrixWithStars(i, j) = sprintf('%.2f***', pearsonCoefficients(i,
j));
            elseif pValues(i, j) >= 0.01 && pValues(i, j) < 0.05
                coeffMatrixWithStars(i, j) = sprintf('%.2f**', pearsonCoefficients(i,
j));
            elseif pValues(i, j) >= 0.05 && pValues(i, j) < 0.1
                coeffMatrixWithStars(i, j) = sprintf('%.2f*', pearsonCoefficients(i, j));
            else
                coeffMatrixWithStars(i, j) = sprintf('%.2f', pearsonCoefficients(i, j));
            end
```

```
        end
    end

    disp('皮尔逊相关性系数矩阵（带显著性水平星星）：');
    disp(coeffMatrixWithStars);

    % 绘制热力图
    figure;
    colormap('hot');  % 单色系
    h = heatmap(pearsonCoefficients);
    h.ColorbarVisible = 'on';
    xlabel('品类');
    ylabel('品类');
```

附录 6

介绍：谱系图



附录 7
介绍：单品分布散点图


附录 8
介绍：第三问分类结果

品类 1	品类 2	品类 3	品类 4	品类 5	品类 6
10290001103 0059	10290000511 6714	10290000511 6899	10290000511 6257	10290001103 0097	10694971130 0259
10290001103 1100	10290001103 4026	10290005100 0944	10290001102 2764	10290001101 6701	10290001103 4330
10290000511 5786		10290000511 8824	10290000511 6509	10290001103 2022	10697153345 0003
10290000511 8831		10290001100 1691	10290005100 0463	10290001103 2251	10290000511 5250
10290000511 5762		10290001103 2732	10290001103 3982	10290001103 2848	10290001103 1926
10290001103 0110		10290001100 7969		10290001100 0328	10290001103 5740
10290001100 8164		10290001101 8132		10290001103 4439	10290001101 3274
10290000511 5946				10290001103 5078	10290001103 0929
10290001102 3464				10290001103 2343	
10290000511 9975				10290001103 2237	
10290000511 5823					
10290000511 5779					
10290000511 5908					
10290000511 8817					
10290001100 6948					
10290000511 5984					
10290001103 6686					

附录 9

介绍：斯皮尔曼相关系数代码

```
% 加载数据
clc;clear
load('data_q1.mat');
```

```
%%
%1.计算月销量和以便后续聚类
% 获取所有独特的 num 值
unique_nums = unique(data_q1.num);

% 初始化一个矩阵来存储结果，行数为商品数量，列数为 12（代表 12 个月）
result = zeros(length(unique_nums), 12);

% 使用 for 循环计算每个 num 的月销售量总和
for i = 1:length(unique_nums)
    current_num = unique_nums(i);

    for month = 1:12
        % 找到当前 num 和当前月的所有销售记录
        idx = (data_q1.num == current_num) & (data_q1.month == month);

        % 计算销售量总和
        monthly_sum = sum(data_q1.sale_v(idx));

        % 将结果存储在矩阵中
        result(i, month) = monthly_sum;
    end
end

% 现在 result 矩阵每行代表一个商品，每列代表一个月的销售量总和
%%
%2.增加来自 spss 的聚类结果
load mer_result.mat
% 其中 data(:, 1:12)是月销量数据，data(:, 13)是聚类结果
data=[result,mer_result]
%随机保留每一类的其中一个单品
%%
%3.随机保留每一类的其中样本（这里取第一个）
% 提取类别
categories = unique(data(:, end));

% 初始化一个空矩阵，用于存储每个类别的第一个数据
result1 = [];

% 对于每个类别，找到第一个数据并添加到结果矩阵中
for cat = categories'
    firstRow = find(data(:, end) == cat, 1, 'first');
    result1 = [result1; data(firstRow, :)];
end
```

```
rando=result1(:,1:12)'



%%
%4.观察是否有线性性
[numDays, numCategories] = size(rando);

figure('Position', [100, 100, 1200, 1200]);   % 创建一个大的图形窗口

% 遍历所有类间的随机样本
for i = 1:numCategories
    for j = i+1:numCategories
        subplot(numCategories-1, numCategories-1, (i-1)*(numCategories-1) + j-
1); % 在右上三角形中选择适当的位置
        scatter(rando(:, i), rando(:, j), 'filled', 'MarkerEdgeColor', [0.95, 0.64, 0.38],
'MarkerFaceColor', [0.95, 0.64, 0.38]);
        xlabel(['属于聚类' num2str(i)]);
        ylabel(['属于聚类 ' num2str(j)]);
        grid on;
    end
end

%%
%
% 5.计算斯皮尔曼相关性系数及其 p 值
[numDays, numCategories] = size(rando);
spearmanCoefficients = zeros(numCategories, numCategories);
pValues = zeros(numCategories, numCategories);

for i = 1:numCategories
    for j = 1:numCategories
        [rho, p] = corr(rando(:, i), rando(:, j), 'Type', 'Spearman');
        spearmanCoefficients(i, j) = rho;
        pValues(i, j) = p;
    end
end

% 生成带有星星标记的相关性系数矩阵
coeffMatrixWithStars = strings(numCategories, numCategories);

for i = 1:numCategories
    for j = 1:numCategories
```

```
                if pValues(i, j) < 0.01
                    coeffMatrixWithStars(i, j) = sprintf('%.2f***', spearmanCoefficients(i, j));
                elseif pValues(i, j) >= 0.01 && pValues(i, j) < 0.05
                    coeffMatrixWithStars(i, j) = sprintf('%.2f**', spearmanCoefficients(i, j));
                elseif pValues(i, j) >= 0.05 && pValues(i, j) < 0.1
                    coeffMatrixWithStars(i, j) = sprintf('%.2f*', spearmanCoefficients(i, j));
                else
                    coeffMatrixWithStars(i, j) = sprintf('%.2f', spearmanCoefficients(i, j));
                end
            end
        end

        disp('斯皮尔曼相关性系数矩阵（带显著性水平星星）：');
        disp(coeffMatrixWithStars);

        % 绘制热力图
        figure;
        colormap('hot');  % 单色系
        h = heatmap(spearmanCoefficients);
        h.ColorbarVisible = 'on';
        xlabel('随机抽取的聚类');
        ylabel('随机抽取的聚类');
```

附录 10

介绍：时间序列代码（问题 2）

```
        %% 导入电子表格中的数据
        % 用于从以下电子表格导入数据的脚本:
        %
        %    工作簿: C:\Users\handsomeju\Desktop\附件 3 合并附件 1 后.xlsx
        %    工作表: Sheet1
        %
        % 由 MATLAB 于 2023-09-09 13:07:20 自动生成

        %% 设置导入选项并导入数据
        opts = spreadsheetImportOptions("NumVariables", 4);

        % 指定工作表和范围
```

```
opts.Sheet = "Sheet1";
opts.DataRange = "A2:D55983";

% 指定列名称和类型
opts.VariableNames = ["date", "sale_num", "sale_v", "category"];
opts.VariableTypes = ["datetime", "double", "double", "double"];

% 导入数据
q2_pre = readtable("C:\Users\handsomeju\Desktop\附件 3 合并附件 1 后.xlsx", opts, "UseExcel", false);


%% 清除临时变量
clear opts

%%
data=q2_pre


% 使用分组统计来计算每日每个品类的批发价格平均值
averagePrices = groupsummary(data, {'date', 'category'}, 'mean', 'sale_v');

% 初始化结果矩阵
uniqueDates = unique(data.date);
uniqueCategories = unique(data.category);
resultMatrix = NaN(length(uniqueDates), length(uniqueCategories));

% 填充结果矩阵
for i = 1:length(uniqueDates)
    for j = 1:length(uniqueCategories)
        idx = ismember(averagePrices.date, uniqueDates(i)) & averagePrices.category == uniqueCategories(j);
        if any(idx)
            resultMatrix(i, j) = averagePrices.mean_sale_v(idx);
        end
    end
end

% 输出结果
disp(resultMatrix);

%%
% 初始化保存平稳性检验结果的矩阵
stationarityResults = zeros(6,2); % 第一列为原始数据的 p 值，第二列为差分后
```

```
的 p 值
    nonStationaryCategories = []; %  保存不平稳的品类

    %  对每个品类进行平稳性检验
    for i = 1:6
        %  对原始数据进行平稳性检验
        [~,pValue] = adftest(resultMatrix(:,i));
        stationarityResults(i,1) = pValue;

        %  如果原始数据不平稳，进行一阶差分并保存品类
        if pValue > 0.05
            nonStationaryCategories = [nonStationaryCategories, i];
            diffData = diff(resultMatrix(:,i));
            [~,pValueDiff] = adftest(diffData);
            stationarityResults(i,2) = pValueDiff;
        end
    end

    %  显示不平稳的品类
    disp('以下品类的原始数据不平稳:');
    disp(nonStationaryCategories);

    %  显示平稳性检验结果
    disp('原始数据的平稳性检验 p 值:');
    disp(stationarityResults(:,1));
    disp('差分后的平稳性检验 p 值:');
    disp(stationarityResults(:,2));

    %  保存平稳性检验结果
    save('stationarityResults.mat', 'stationarityResults');



    %%
    maxP = 5;
    maxQ = 5;
    bestModels = zeros(6,2);
    bicValues = inf(6,1);

    for i = 1:6
        for p = 0:maxP
            for q = 0:maxQ
                model = arima(p,1,q);
                try
```

```
                        [fit,~,logL] = estimate(model, resultMatrix(:,i), 'Display', 'off');
                        currentBIC = -2*logL + log(length(resultMatrix(:,i)))*(p + q +
1);
                        if currentBIC < bicValues(i)
                            bicValues(i) = currentBIC;
                            bestModels(i,:) = [p, q];
                        end
                    catch
                        continue;
                    end
                end
            end
        end
        disp(bicValues);
        %%
        % 确定哪些品类进行了差分
        diffCategories = find(stationarityResults(:,1) > 0.05);

        % 输出最佳 ARIMA 模型
        for i = 1:6
            d = 1;
            if ~ismember(i, diffCategories)
                d = 0;
            end
            disp(['品类', num2str(i), '的最佳 ARIMA 模型: ARIMA(',
num2str(bestModels(i,1)), ',', num2str(d), ',', num2str(bestModels(i,2)), ')']);
        end



        forecastedValues = zeros(7,6);%初始话预测结果
        for i = 1:6
            model = arima(bestModels(i,1),1,bestModels(i,2));
            fit = estimate(model, resultMatrix(:,i), 'Display', 'off');
            forecastedValues(:,i) = forecast(fit, 7, 'Y0', resultMatrix(:,i));
        end

        forecastedValues
```

附录 11

介绍：M-H 算法

```
        % C: 6x7, eta: 1x13, beta: 1x6, w: 6x13, P_up: 1x6, P_low: 1x6, Q_up: 1x6, Q_low:
1x6
        clc;clear
```

```
%%
load guihua1.mat
a=(eta)*(w')
aa=repmat(a',1,7)
aa=aa.*repmat(beta',1,7)
%%
% 定义目标函数
rng(6)
objective_function = @(P) sum(sum((P - C) .* (P .* coeff' + const'))) - sum(sum(P .*
(P .* coeff' + const') .* aa));
% (P_best.* coeff' + const')
% Metropolis-Hastings 算法参数
iterations = 10000;
acceptance_count = 0;
P_current = rand(6,7) * 5 + 5; % 任意初始值，介于[5, 10]之间
obj_value_current = objective_function(P_current);

P_best = P_current;
obj_value_best = obj_value_current;
%%

% 检查是否满足约束条件的函数
is_within_bounds = @(P) all(P >= P_low_mat & P <= P_up_mat, 'all');

% Metropolis-Hastings 算法循环
for i = 1:iterations
    % 提议新点
    P_proposed = P_current + randn(6,7) * 0.5; % 添加一些随机扰动

    % 检查提议点是否满足约束条件
    if ~is_within_bounds(P_proposed)
        continue; % 如果不满足约束条件，直接进入下一次迭代
    end

    obj_value_proposed = objective_function(P_proposed);

    % 计算接受概率
    alpha = min(1, exp(obj_value_proposed - obj_value_current));

    % 接受或拒绝新点
    if rand() < alpha
        P_current = P_proposed;
        obj_value_current = obj_value_proposed;
        acceptance_count = acceptance_count + 1;
```

```
            % 如果新点比最佳点更好，则更新最佳点
            if obj_value_current > obj_value_best
                P_best = P_current;
                obj_value_best = obj_value_current;
            end
        end
    end
    %%

    % 输出结果
    disp('Best P found:');
    disp(P_best);
    disp('Best Q found:');
    Q_best=(P_best.* coeff' + const')
    disp('maxprofit:')
    disp(obj_value_current)
```

附录 12

介绍：时间序列（问题三）

```
    clc
    clear
    % 所有的日期
    load data_q2.mat
    all_dates = unique(data_q2.date);

    % 初始化结果表格
    result = array2table(zeros(length(all_dates), length(data_q2_1.num) + 1));
    result.date = all_dates;

    % 对于每一个需要搜索的单品编码
    for i = 1:length(data_q2_1.num)
        code = data_q2_1.num(i);

        % 找到与该编码匹配的所有行
        idx = data_q2.num == code;

        % 提取匹配的数据
        matched_data = data_q2(idx, :);

        % 为每个日期填充批发价格
        for j = 1:height(matched_data)
            date_idx = result.date == matched_data.date(j);
```

```
            result{date_idx, i + 1} = matched_data.price(j);
        end

        % 设置列名为单品编码
        result.Properties.VariableNames(i + 1) = {sprintf('编号%d', code)};
    end

    % 显示结果
    disp(result);
    result(:,1)=[]

    %%

    data = result

    nProducts = size(data, 2) - 1; % 商品数量

    % 对每个商品进行预测
    predicted_values = zeros(1, nProducts);
    for i = 1:nProducts
        prices = data(:, i);

        % 剔除 0 值
        validIdx = prices ~= 0;
        validPrices = prices(validIdx);

        % 对数据进行对数变换
        logPrices = log(validPrices);

        % 构建 ARIMA 模型并预测
        Mdl = arima('Constant',0,'D',1,'Seasonality',7,'MALags',1,'SMALags',7);
        EstMdl = estimate(Mdl,logPrices);

        [logY,~] = forecast(EstMdl,1,'Y0',logPrices);

        % 将预测结果从对数形式转换回原始形式
        predicted_values(i) = exp(logY(end));
    end

    % 输出预测值
    disp('Predicted values for 2023-07-01: ');
    disp(predicted_values);
```

附录 14

介绍：遗传算法

```
clear;
load guihua3.mat
initialGuessP = rand(6, 17) * (max(max(P_up_mat)) - min(min(P_low_mat))) + min(min(P_low_mat));
initialGuessd = round(rand(6, 17));
initialGuess = [initialGuessP(:); initialGuessd(:)];
%%


% GA 参数设置
options = optimoptions('ga', ...
    'PopulationSize', 1000, ...
    'Generations', 1000, ...
    'CrossoverFcn', @crossovertwopoint, ...
    'MutationFcn', @mutationadaptfeasible, ...
    'SelectionFcn', @selectiontournament, ...
    'Display', 'diagnose', ...
    'PlotFcn', {@gaplotbestf, @gaplotstopping}, ...
    'FunctionTolerance', 1e-6, ...
    'ConstraintTolerance', 1e-6);

%
lb = [zeros(1, 102), P_low_mat(:)'];
ub = [ones(1, 102), P_up_mat(:)'];

%求解
[x, fval] = ga(@(x) fitnessFunction(x, C, aa, coeff, const), 204, [], [], [], [], lb, ub, @(x) constraintsFunction(x, P_low_mat, P_up_mat), options);
fval = -fval; % 由于 ga 优化最小值，故一开始目标函数取负，现取回正
Presult = reshape(x(103:end), [6, 17]);%定价
dresult = reshape(x(1:102), [6, 17]);%商品是否订货概率矩阵
Qresult=(repmat(coeff',1,17)) .* Presult + (repmat(const',1,17));%补货量
```

附录 15

介绍：遗传算法—子函数 1

```
function [c, ceq] = constraintsFunction(x, P_low_mat, P_up_mat)
    d = reshape(x(1:102), [6, 17]);
    P = reshape(x(103:end), [6, 17]);

    ceq = []; % We will have only inequality constraints
```

```
        c1 = 27 - sum(d(:));
        c2 = sum(d(:)) - 33;
        c3 = P(:) - P_up_mat(:);
        c4 = P_low_mat(:) - P(:);

        c = [c1; c2; c3; c4];
    end
```

附录 16

介绍：遗传算法—子函数 2

```
    function obj = fitnessFunction(x, C, aa, coeff, const)
        d = reshape(x(1:102), [6, 17]);
        P = reshape(x(103:end), [6, 17]);
        Q = (repmat(coeff',1,17)) .* P + (repmat(const',1,17));

        revenue = sum(sum((P - C) .* Q .* d));
        cost = sum(sum(P .* Q .* aa .* d));
        obj = -(revenue - cost);
    end
```