基于价值损耗的生鲜商超蔬菜定价模型

摘要

生鲜商超普遍面临蔬菜品类的保质期较短，且蔬菜的品质会随着销售时间的推移而降低。本文针对一家生鲜商超所提供的数据，构建了基于价值损耗的生鲜商超蔬菜定价模型。

针对问题一，我们分别将不同品类的蔬菜商品进行归类后进行分析，发现各蔬菜品类的销售数量在时间分布上具有周期性分布特点，并且在每年的 4 月至 10 月时间内总体的销售量较其余月份时间处于更高水平，其次蔬菜总销量的占比中花叶类的销售量总是处于高位，其余品类蔬菜的销售量则较为接近。通过运用**斯皮尔曼相关性系数**，得到蔬菜品类的销售量与蔬菜品类之间的关联系数为-0.886，表明两者之间相关性强烈。同理，对蔬菜单品的销售量数据进行分析，发现蔬菜单品存在一个明显的替代性特征，即上一年某一时间段销售量高的单品在下一年的同样时间段内则迅速变为普通蔬菜销量水平，利用斯皮尔曼相关性系数对蔬菜单品进行处理，并用热力图将不同蔬菜单品销售量相关性进行可视化处理。

针对问题二，我们团队采用对周期性时间序列数据预测结果良好的 **ARIMA 模型**对该生鲜商超各蔬菜品类的销售量进行预测，得到该生鲜商超各蔬菜品类未来一周的日补货总量。而针对各蔬菜品类的定价策略的制订问题上，我们团队通过查阅相关的文献，建立了基于**价值损耗**的**生鲜商超蔬菜定价模型**，并应用该定价模型预测得到对应各蔬菜品类的定价策略，预测所得结果已经在文章中展示。

针对问题三，我们团队在第二问对蔬菜品类的定价策略分析的基础上，继续对蔬菜单品的数据进行分析。通过对附件中给出的 2023 年 6 月 24-30 日的该生鲜商超的数据进行筛选后，得到销量满足基本要求且利润率较高的商品数据。在此基础上，继续筛选出排名前 33 个蔬菜单品的数据，同时每一蔬菜单品的单品补货量采用前一周对应该蔬菜单品的平均销售量。最后，利用问题二所构建的基于价值损耗的生鲜商超蔬菜定价模型得到对应蔬菜单品的定价策略，使得在尽量满足市场对各品类商品需求的前提下，使得商超收益最大。

针对问题四，我们从运输成本，连锁生鲜商超数据、政策等角度进行分析，得到商超还需要收集消费者价格敏感系数，观察消费者价格敏感系数等数据，并且给出采集这些数据的必要性。

关键词：斯皮尔曼相关性系数　ARIMA 模型　价值损耗　生鲜商超蔬菜定价模型

一、问题重述

1.1 问题背景

生鲜商超往往会面临一个问题 ，那就是对于一般蔬菜类商品而言，其保鲜期都比较短，并且其新鲜程度又会随着时间的推移而逐步变差，容易造成滞销的问题。事实上，生鲜商超中的蔬菜商品的销售与包括是否有折扣、蔬菜的销售区域大小以及顾客购买时间等因素都密切相关，在此基础上，可靠的市场需求分析对于蔬菜的销售则愈发重要。现已知某商超 2020 年至 2023 年的将近一年的蔬菜销售信息，需要解决以下问题。

1.2 问题要求

问题一：对蔬菜类商品不同品类或不同单品的数据进行处理，分析其各自的销售量的分布规律以及相互之间的关系。

问题二：以品类为单位对数据进行处理后，分析各蔬菜品类的销售总量与成本加成定价的关系，在此基础上，给出相对应的各蔬菜品类未来一周（即 2023 年 7 月 1-7 日）这一时间段内的日补货和定价策略，以此达到使得该商超收益的最大化。

问题三:在可售单品总数限制在 27-33 个且各蔬菜单品的订购量需满足最小陈列量 2.5 千克的条件下，结合附件中已知的 2023 年 6 月 24-30 日这一时间段内的可售蔬菜商品品种的数据，以及尽量满足市场对各品类蔬菜商品需求的情况下，使得商超当天的收益最大化。

问题四，考虑制订蔬菜商品的补货和定价决策的相关影响因素，并且分析具体某一相关因素对上述问题的解决的作用，进而向商超提出收集相应数据的需求。

二、问题分析

2.1 问题一分析

问题一事实上含有两个问题，即分析蔬菜类商品不同品类销售量的分布规律及相关关系与分析蔬菜类单品之间的分布规律及相互关系。对于蔬菜类商品不同品类的分布规律而言，可以考虑相应蔬菜品类的销售量随时间的变化以及在某一段时间内，各类蔬菜商品的品类的销售量占该段时间内的蔬菜销售数量的百分占比进行分析。

而对于相互关系的求解则可以通过相应的皮尔逊相关系数的方法对不同类别之间的相关性进行求解。同理，对于蔬菜不同单品销售量的分布规律以及相互关系的求解也可通过蔬菜单品数据随时间的变化可以得出。

2.2 问题二分析

问题二需要以商品品类为单位进行补货计划的制订，定价方式无疑是影响蔬菜品类销售量的相对重要的一个因素。该商超采用的“成本加成定价”的定价方法对蔬菜品类销售量的具体关系可以通过是否打折这一前后状态商品的销售量变化得出。另外，由于题目要求商超实现收益的最大化。因此，不能够单纯看蔬菜商品品类的销售量水平，还需要制订相应合理的定价策略。

2.3 问题三分析

问题三需要在可售单品总数限定在 27-33 个且各单品订购量满足最小陈列量 2.5 千克的限定条件下，参照 2023 年 6 月 24-30 日的可售品种，制订出 7 月 1

日的单品补货量和定价策略。因此，需要在筛选出 2023 年 6 月 24-30 日均有售的蔬菜单品的种类的基础上，根据当天的销量总和筛选出一部分蔬菜单品，然后根据这部分蔬菜单品的利润率进行排序，结合问题二所建立的模型分析可售单品在 27-33 个时候的 7 月 1 日单品补货量和定价策略。

2.4 问题四分析

问题四需要分析关于商超仍需采集的数据类型有哪些。可以从运输成本，连锁生鲜商超数据、政策等角度进行思考。

三、模型假设

假设一：假设某一蔬菜单品的可售时间与附件所给数据一致，并且保持一致；
假设二：该商超有多个收银台，可同时进行多种商品的销售；
假设三：假设 2023 年 6 月 24-30 日的可售品种在 7 月 1 日当天可售；

四、符号说明

符号	定义	单位
T	生鲜商超的订货周期	/
λ(t)	对应某一蔬菜品类的价值损耗率函数	/
p	该生鲜商超中对于某一类蔬菜品类的批发价格	/
H	该生鲜商超在单位时间内对于单位某品类蔬菜商品的库存持有成本率	%
Q(t)	该家生鲜商超在相应蔬菜单品订货周期 T 内在 t 时刻的某蔬菜品类商品的库存量。	/
S	该生鲜商超订购一批相应品类蔬菜的订货成本	/
x	在订货期 T 内该家生鲜商超对应于某一品类蔬菜商品的单位出售价格	/
G	生鲜商超单位时间的销售利润的总利润	/

(注：未列出符号及重复符号以出现处为准)

五、问题一模型

问题一需要对蔬菜类商品不同品类或不同单品的数据进行处理，分析其各自的销售量的分布规律以及相互之间的关系。而蔬菜商品的销售量数据由附件 2 提供，蔬菜商品所属的类别由附件 1 给出，因此可以通过将附件 1 和附件 2 进行数据的匹配与合并，从而便于进行蔬菜商品所属类别以及销售量的匹配信息。

而后，对附件 2 中原有的扫码销售时间进行处理，得到相对应的蔬菜品类商品的销售量对应时间的变化规律。同理，对于商品单品的处理也采用上述方法。而对于相关性的分析，由于商品的销售量属于连续型变量，故采用适用于求连续型变化变量相关性的皮尔逊相关系数分析法对不同蔬菜商品品类之间以及不同蔬菜单品之间的相关性进行求解。

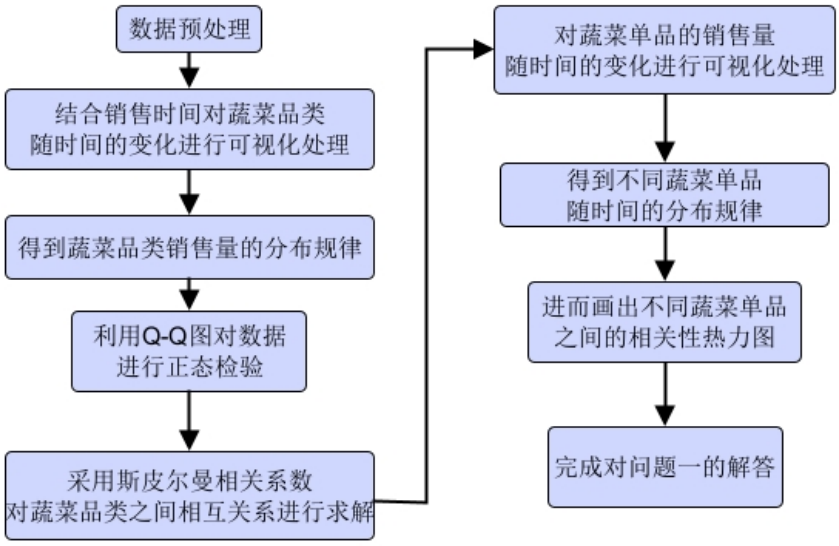

图 1 问题一思维导图

5.1 数据预处理

数据信息的合理性对于后续建立相应模型的准确性之间存在明显的影响。因此，为方便后续模型的建立，我们对附件的数据进行相应的预处理。

1. 缺失值

通过对题目所给附件的数据进行分析，我们发现题目提供的四个附件中均不存在缺失值，四个附件所提供的数据的完备性良好。

2. 异常值

通过对题目所提供的 4 个附件数据的最值，异常值进行筛选，发现存在一些不合理的数据。如单品编码为 102900011030417 的花叶大类的洪山菜尊莲藕拼装礼的批发价为 141 元/千克，而其销售单价则为 15 元/千克，虽然该商品属于打折销售并且打折的力度很大，但是其本身的销售数量并不明显——只有 3 条成交记录，这本身就并不合理，故对于此数据采取剔除处理。类似的数据剔除在附件中有相关说明，故不在此展示。

3. 数据编码

为便于后续模型的建立，对附件 2 中的销售类型以及是否打折销售进行数据编码。具体的编码规则如下：

表 1 编码规则表

销售类型	是否打折销售
退货——0，销售——1	否——0，是——1

4. 去量纲化处理

按照上述数据编码规则对数据进行编码后，发现上述数据是定类数据，而对应蔬菜单品的销量数据则为对应的定量数据。为将数据变量进行统一，可以将定类变量转化为相应的定性变量，但这存在一个问题，即转化后的数据量的量纲并不一致。为便于后续模型的建立，此处对蔬菜商品的总销量数据进行统一的去量纲归一化处理。具体的去量纲归一化处理方法采用极差值标准化的方法，即

$$y_i = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{1}$$

5.2 数据分析

根据附件 2 提供的该商超 2020 年 7 月 1 日至 2023 年 6 月 5 日各商品的销售流水明细数据，结合附件 1 所给的商品单品编码所对应的蔬菜的品类数据，可以得到该商超各蔬菜类商品不同品类随时间的销售量分布规律。相应的各蔬菜类商品品类随时间的分布规律图如图 2 所示：

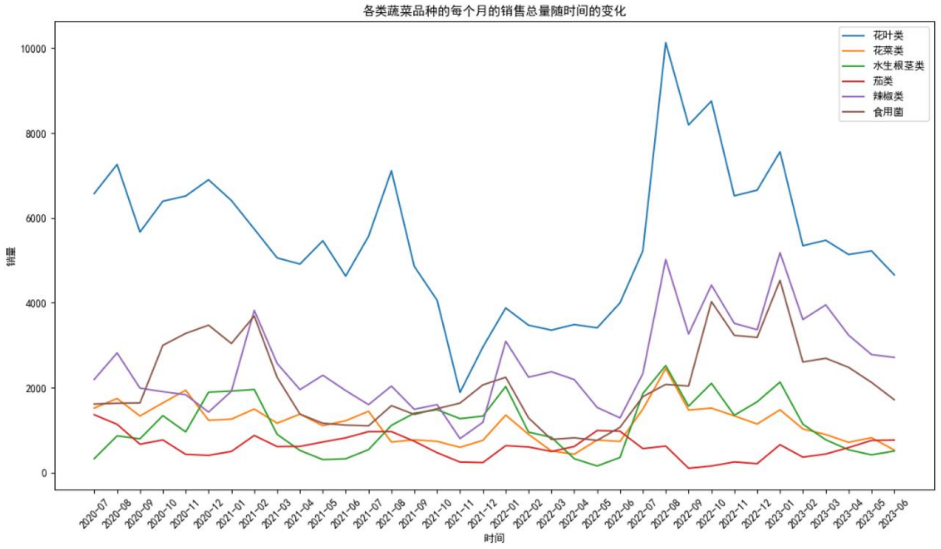

图 2 各蔬菜类商品品类随时间的分布规律图

从上图可以发现花叶类蔬菜商品的销售量一直维持在第一的水平，其次则是辣椒与食用菌的销售量水平相近，剩下的花菜类、水生根茎类以及茄类的销售量水平则处于较低的销售量水品。与此同时，由于是对 2020 年 7 月 1 日到 2023 年 6 月 30 日这一段时间进行分析，期间共有 3 年的时间，而蔬菜类产品的成熟时间往往以年为周期，故可以看到各蔬菜品类的销售量随时间的周期性变化。

各类蔬菜品类的销售量在每年的 4 月至 10 月的这段时间内所处的水平较该年其余时间都要高，而这条规律正好与题目中所提到的蔬菜的供应品种在 4 月至 10 月这部分较为丰富的规律所契合。另外，还有一点分布规律，那就是各蔬菜品类的销售量在每年的 7 月份的时间里往往会有一个较为明显的增长趋势，往往是在每年的 7 月份销售量达到顶峰后逐步进行回落。联系现实的情况，初步猜想初夏的来临有利于促进相应的蔬菜品类的上市，同时激发人们购买相应蔬菜的欲望。

对于各类蔬菜品类销售量的相互之间的关系的分析，可以运用饼状图将各蔬菜商品品类所占销售额的比例进行可视化处理，得到相应关系的可视化图表。下面是以月份为单位，分析各蔬菜类商品的销售量之间的相互关系图。由于数据量庞大的原因，此处只挑选了部分具有代表性特征的数据，其余数据图可在支撑材料处查询。

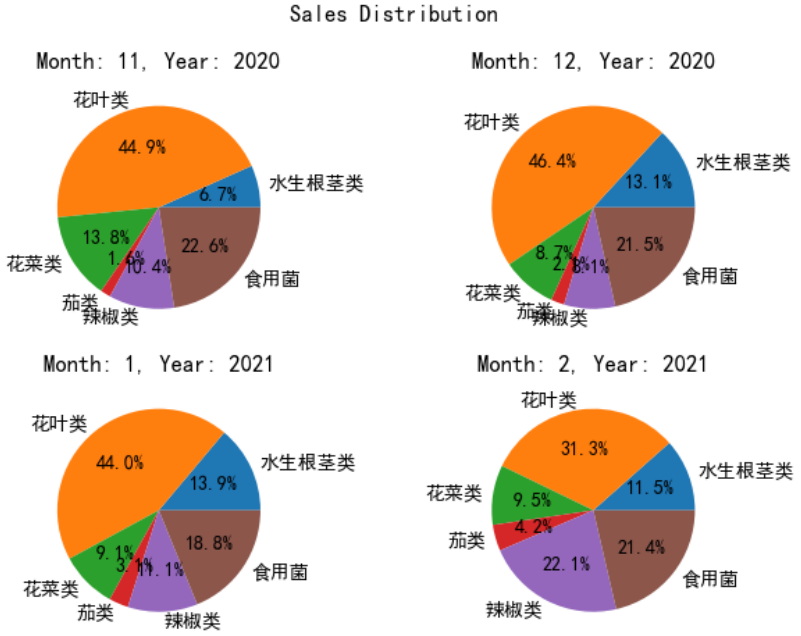

图 3 各蔬菜商品品类销售量各月份的占比

通过对上述饼状图的分析，可以发现各蔬菜商品品类的销售量与上述随时间变化的销售量一致，即花叶类销量总是占据总销量的第一位。另外，可以发现食用菌类的销售量所占份额较为稳定，几乎都是徘徊在当月蔬菜类商品总销量的20%左右，与其他蔬菜品类的销售量之间的关系从饼状图看来并不明显。反而是花叶类蔬菜的销售量与辣椒类商品品类的销售量之间的关系更为密切一些——当花叶类的销售量占总的蔬菜销售总量的比例增大时，辣椒类商品品类的销售总量占总订蔬菜销售总量的比例则在减少，经过皮尔逊相关系数的求解，发现两者的相关联系数为 0.624，属于明显较强的联系水平。为进一步深入分析各蔬菜品类的相关性情况，我们对各蔬菜品类的销量数据采用 Q-Q 图方法进行正态分布检验，以决定采用皮尔逊相关系数分析还是斯皮尔曼相关性分析。得到的结果如下：

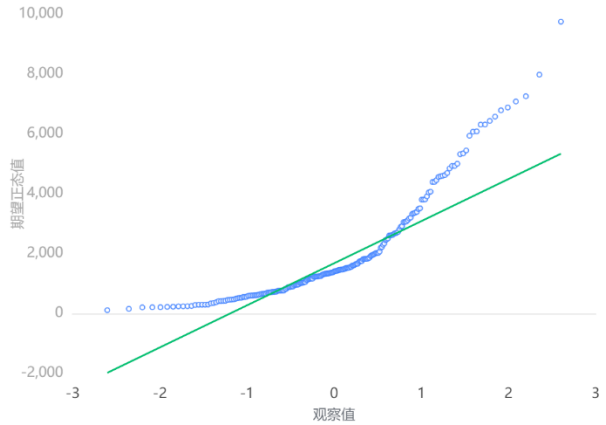

图 4 蔬菜产品正态分布检验图

从上图中可以看出，存在部分蔬菜产品数据与参考线的偏差较大，因此蔬菜产品销量的数据并不符合正态分布的规律。因此，我们采用斯皮尔曼相关系数分析蔬菜品类之间的相关性。相应的求解结果如下：

表 2 蔬菜品类销量与与所属蔬菜品类之间的关系

	销量总和	分类名称
销量总和	1(0.000***)	-0.886(0.019**)
分类名称	-0.886(0.019**)	1(0.000***)

注：***、**、*分别代表 1%、5%、10%的显著性水平

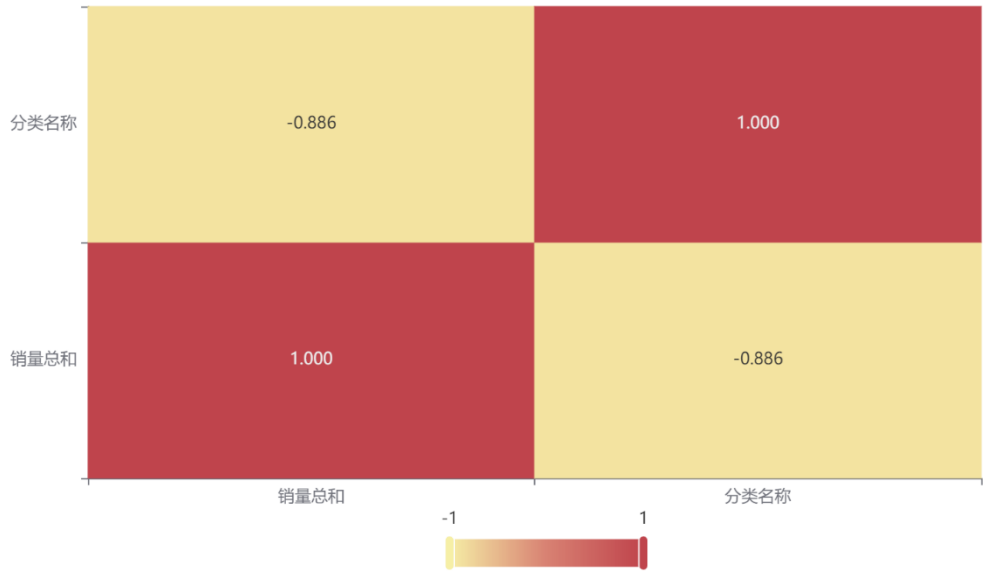

图 5 各蔬菜品类销售量与所属品类的相关性热力图

从热力图中可以发现，不同蔬菜品类的销售量之间销量的相关性是比较密切的，达到了在 0.05 下的显著性水平为-0.886。进一步细分蔬菜商品类型，得到各具体蔬菜品类销售量与其销售量之间的关系图如下：

图 6 各具体蔬菜品类销售量与具体所属品类的相关性热力图

其中，花菜类与辣椒类销量之间的相关性还是处于较高水平的。另外，花菜类的销量与花叶类的销量之间的相关性更是达到了在 1%的显著水平下 0.747 的强相关关系。

与此同理，对蔬菜单品的销售量按时间顺序从 2020 年 7 月 1 日开始至 2023 年 6 月 30 日这段时间进行分析，得到相应的各蔬菜单品随时间的分布规律如图 7 所示。

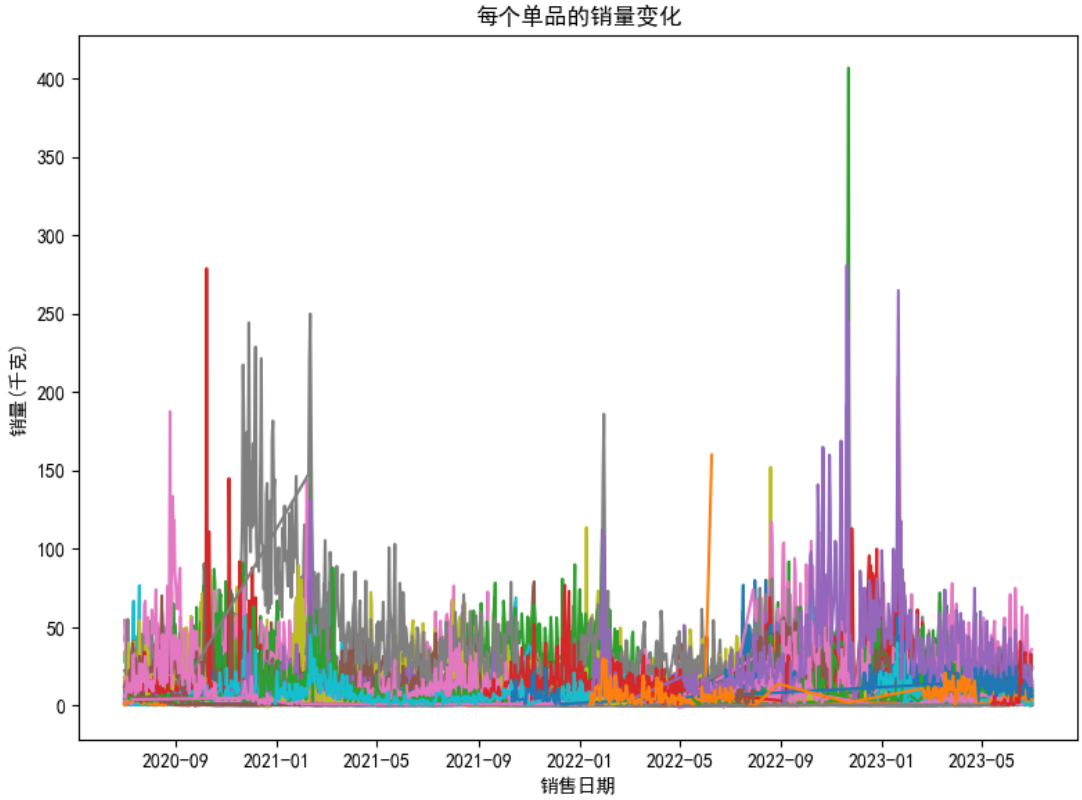

图 7 各蔬菜单品随时间的分布规律

（详细信息可于支撑材料进行查询）

由上图可以看到，大部分的蔬菜单品的销售量都集中在 0 到 50 千克的销量范围之间，少部分蔬菜单品的销售量则明显高于蔬菜的平均销量水平，达到了 50 千克以上。我们可以看到，在 2020 至 2023 年这三年的时间内，前一年某一季度的蔬菜爆品在下一年中并非仍然保留着爆品的地位。如 21 年 1 月销售量最高的蔬菜单品是金针菇（属食用菌类），而到了 2023 年 1 月，相应的爆品蔬菜单品则为娃娃菜（属花叶类）。

结合生活实际，猜测是冬季火锅引起这些蔬菜单品的销量的变动。这也说明对应蔬菜单品彼此之间存在相互替代的特性。另外，可以看出蔬菜单品的销售量随时间的分布存在明显的季节性特征，如娃娃菜在冬季这一段时间内的销量遥遥领先于其他蔬菜单品。

对于单品的销售量的相互关系的探讨，我们团队认为主要分析同一种蔬菜类品类内部单品之间的关系，而归属不同蔬菜品类单品之间的关系则可以用不同蔬菜品类之间的关系进行相应的替换，毕竟不同蔬菜商品品类之间的差异较为明显，不易进行简单的相关性分析。针对同一蔬菜品类的不同蔬菜单品的相关性可以从某一天的单品的销量相关性热力图进行分析，如图 8 所示：

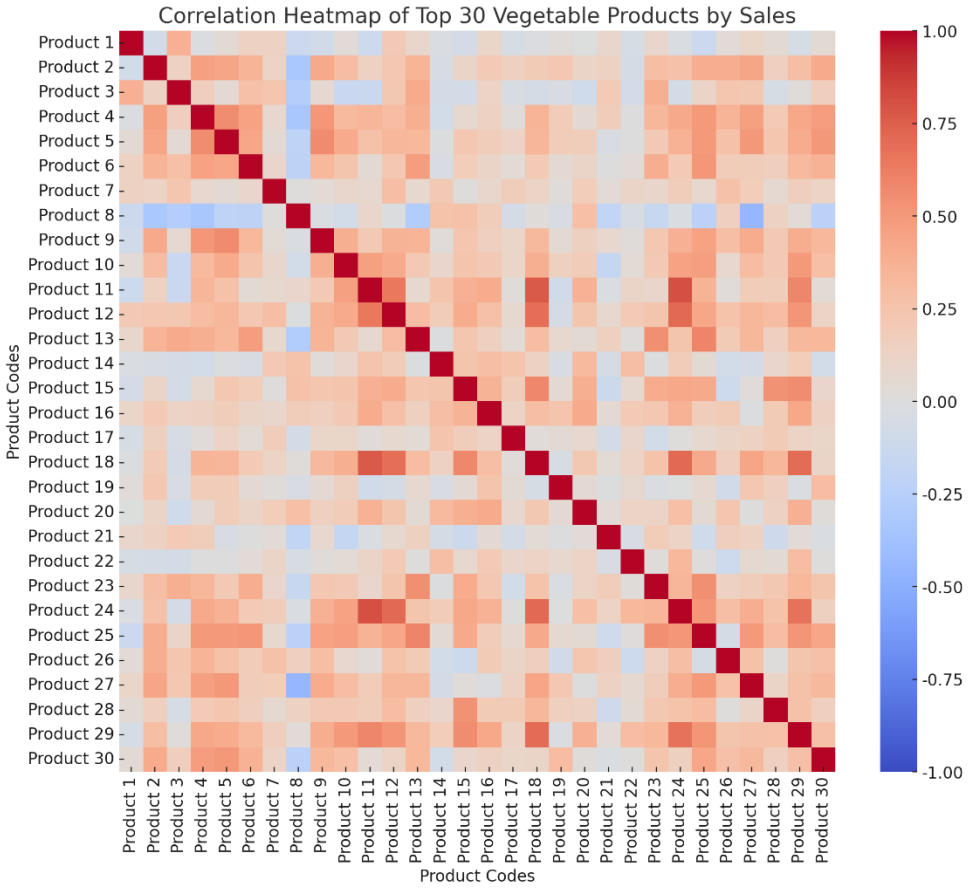

图 8 单天蔬菜单品销量相关性热力图

（上述图中详细信息可于支撑材料查询）

从单天蔬菜单品销量相关性热力图可以看出，单品的销售量占比之间存在较为明显的替代关系。如，同为食用菌类蔬菜品类的金针菇蔬菜单品与香菇单品的销售量在冬季的销售量之间呈现彼此相互替代的关系。同时，有一些蔬菜单品之间又存在互补的关系。如，姜、蒜、小米椒组合之间的销量占比在不同季度相近，尤其是在冬季，存在明显的互补关系。

六、问题二模型

问题二需要以商品品类为单位进行补货计划的制订，而由于蔬菜品类的销售总量受多种因素的影响，其中定价方式无疑是影响蔬菜品类销售量的相对重要的一个因素。通过建立 ARIMA 模型对相应蔬菜品类的销售量进行预测可以得到对应蔬菜商品的补货总量的预测。在此基础上，我们团队通过建立基于价值损耗的定价模型对对应蔬菜品类的价格进行计算，进而制订出每一类蔬菜商品品类的定价策略。

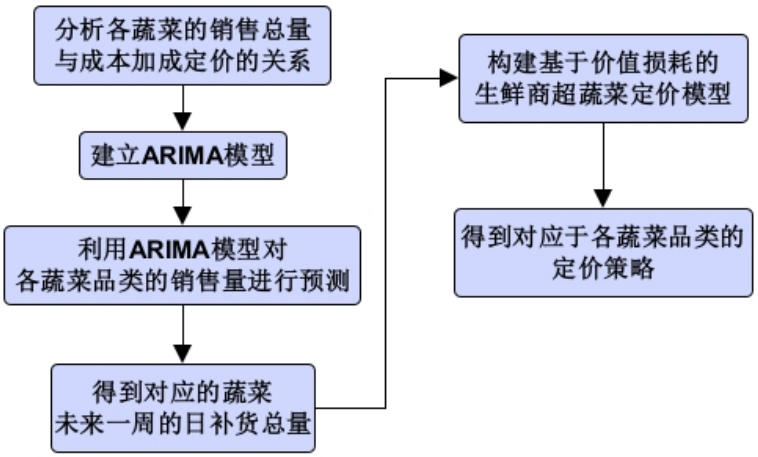

图 9 问题二流程图

6.1 问题分析

题目需要我们分析各蔬菜品类的销售总量与成本加成定价的关系，而这其中能够最能代表成本加成变价关系的就是打折与否，因为打折改变了蔬菜商品的成本定价。通过筛选 2020 年 7 月 1 日到 2023 年 6 月 30 日的数据，我们可以得到商品的打折情况。通过对比历史该商品数据没有打折时候的销量与打折后的销量，并且进行按蔬菜的品类进行规类即可得到各蔬菜品类的销售总量与成本加成定价的关系。另外，可以通过构建 arima 模型对各蔬菜品类的销售量进行预测，同时构建基于价值损耗的超市生鲜蔬菜定价模型进而分析得出各蔬菜品类未来一周(2023 年 7 月 1-7 日)的日补货总量和定价策略，使得商超收益最大。

6.2 分析各蔬菜品类的销售总量与成本加成定价的关系

通过筛选，发现商品打折与否的数据之间并无明显的区别，甚至出现打折的时候相应蔬菜单品的销量反而比打折的时候还要低。如 2020 年 7 月 1 日的娃娃菜打折的时候只卖出了 1kg，而相对应的 2020 年 7 月 3 日没有打折的娃娃菜的数量为 14kg。结合实际与理论知识，假设每一家生鲜商超都是理性的经济人，其只会在蔬菜单品的价值与价格已经不存在可比性的时候才会选择进行打折销售，而不会主动打折销售。为进一步说明蔬菜品类的销售总量与成本加成定价的关系，我们团队采用适用于非线性关系、受异常值影响较小的斯皮尔曼相关系数对蔬菜品类的销售量与成本加成定价的关系进行求解，得到的结果为在 1%的显著性水平下销售类型与销量的相关性为-0.04。

表 3 蔬菜商品品类的销量有所属品类的相关系数表

	销量	销售类型
销量	1(0.000***)	-0.04(0.000***)
销售类型	-0.04(0.000***)	1(0.000***)

注：***、**、*分别代表 1%、5%、10%的显著性水平

6.2 各蔬菜品类销售量的预测

6.2.1 模型分析

该问题需要对蔬菜品类销售量进行随时间序列的预测，预测出对应的2023年7月1日至7月7日的各蔬菜品类的销售量，便于得到模型基本的数据。

6.2.2 ARIMA模型建立

ARIMA一般用于时间序列数据的分析，通过拟合相关时间序列，进而预测基于该时间序列的未来数据值。事实上，ARIMA模型是由AR和MA这两步构成的。AR指的是利用解差分方程的方法解出通项公式，而MA是白噪声序列的移动平均，两者的组合共同构成ARIMA模型。

自回归与移动平均的结合：

$$y_t = \mu + \sum_{i=1}^{p} \gamma_i y_{t-1} + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-1} \tag{2}$$

其中，p与q分别为自回归模型与移动平均模型的阶数，需要进行人为定义，而γ_i与θ_i分别是两个模型的相关系数，需要进行模型的求解。

另外，如果满足γ_i满足方程：

$$\gamma_i = \varepsilon_i - \theta_i \varepsilon_{t-1} - \theta_2 \varepsilon_{t-1} - \cdots - \theta_q \varepsilon_{t-q} \tag{3}$$

则称时序γ_i为q阶滑动平均过程，简写为MA（q）。其中ε_{t-1}为白噪声，θ_1，θ_2，…，θ_q为滑动平均模型的参数。

自相关函数：

$$\mathrm{R}(\tau) = \int_{-\infty}^{\infty} x^2(t)dt = \int_{-\infty}^{\infty} x^2(t-\tau)dt \tag{4}$$

偏自相关函数：

$$\rho_{X_{t,}\ ,\ X_{t-k}|X_{t-1},\cdots,X_{t-k+1}} = \frac{E\left[\left(X_t - \hat{E}X_t\right)\left(X_{t-k} - \hat{E}X_{t-k}\right)\right]}{E\left[\left(X_{t-k} - \hat{E}X_{t-k}\right)\right]^2} \tag{5}$$

ARMA建模步骤：

（1）对输入的数据进行判断，判断其是否为平稳非纯随机序列，若平稳则直接进入步骤2；若不平稳则进行数据处理，处理后才能进入步骤2。

（2）通过自相关和偏自相关函数，并结合AIC或BIC准则对建立的模型进行模型识别和定阶。

（3）完成模型识别和定阶后，进入模型的参数估计阶段。

（4）完成参数估计后，对拟合的模型进行适应性检验。如果拟合模型通过检验，则开始进行预测阶段。若模型检验不通过，则重新进行模型识别和检验，即重复步骤2，重新选择模型。

（5）最后，利用拟合性高的拟合模型来预测序列的未来变化趋势。

ARIMA模型建模过程如图10所示：

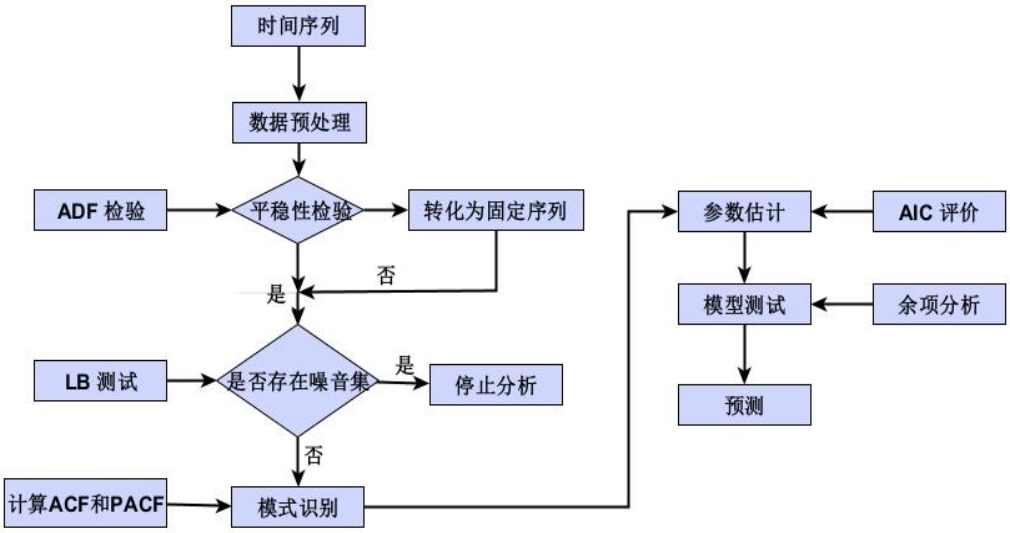

图 10 ARIMA 模型建模过程

6.2.3 模型求解

得到的 ARIMA 模型预测品类销量的可视化如下图所示：

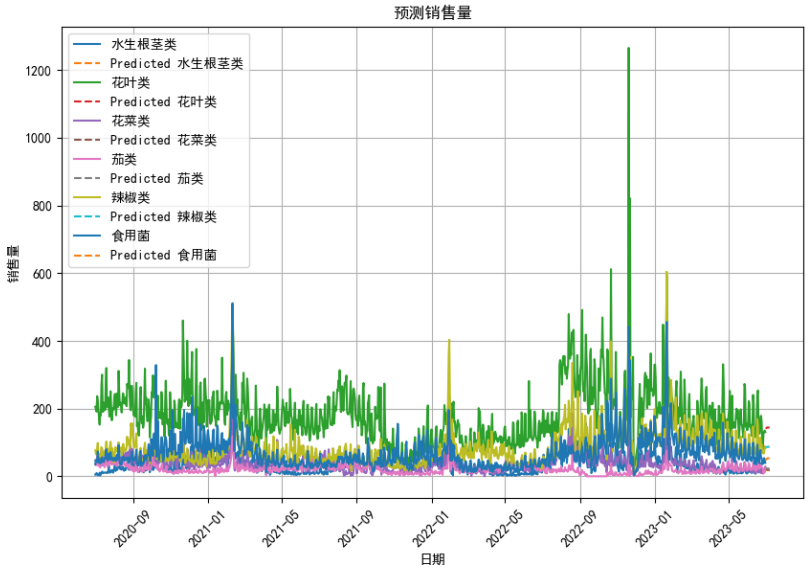

图 11 各蔬菜品类的日销售量的历史数据与预测数据可视化

相应的预测销售量为：

表 4 各蔬菜品类的总销售量预测表

预测日期	水生根茎类（kg）	花叶类（kg）	花菜类（kg）	茄类（kg）	辣椒类（kg）	食用菌（kg）
2023-07-01	18.45	140.28	23.10	23.33	84.87	47.95
2023-07-02	18.06	143.02	20.78	22.88	86.05	51.20
2023-07-03	17.91	143.79	19.71	22.71	86.58	52.47
2023-07-04	17.85	144.00	19.21	22.65	86.82	52.95
2023-07-05	17.82	144.06	18.98	22.62	86.93	53.15
2023-07-06	17.81	144.08	18.88	22.62	86.98	53.22
2023-07-07	17.81	144.08	18.83	22.61	87.00	53.25

6.3 基于价值损耗的超市生鲜蔬菜定价模型

本节模型主要根据题目中所给的商品品类的销售价格与对应该商品品类损耗率，其中损耗率选取能够代表平均损耗水平的平均值作为该品类商品的损耗率水平，从而建立蔬菜商品品类价值损耗的定价模型[5]。

正如经济模型需要在建立在一定范围内的前提一样，该定价模型也需要建立在一定前提的假设上。此次该模型 基于的假设主要有（Jeuland和 Shugan的寡头垄断模型假设假设,Lal 和 Staelin 的供应商假设）。另外，利用经典的库存理论中对于价格与需求、顾客对价格的 敏感程度，从而假设该 商超顾客对于某一品类蔬菜商品的 需求 $D(x)$是关于蔬菜商品品类价格的线性函数如下：

$$D(x) = a - bx, p < x < \frac{a}{b} \tag{6}$$

其中x是蔬菜品类的销售价格，a为该商超顾客对于该蔬菜品类商品的需求规模，b为该蔬菜品类商品的 价格敏感系数，因此得出该蔬菜品类商品的价格敏感系数的函数表达式为$b = \frac{a-D(x)}{x}$。

事实上，当一件物品在单位时间内以一定的 变质速率θ发生变质时，其库存系统可以描述为，在给定周期 T（题目给出的条件是 1 天）内，在时刻 t 某种蔬菜商品对应的 库存量Q(t)由于变质和出售而减少。因此，库存量的变化 可用下面的 一阶常微分方程表示。根据 Ghare 和 Schrader 提出的基于常数需求的指数变质库存模型：

$$\frac{dQ(t)}{dt} = -\lambda(t)Q(t) - D(x) \tag{7}$$

假设再相应每一订货周期 T结束的末尾 即t = T的时候，相应的该品类蔬菜的库存水平 Q(t)=0，与此同时根据一阶线性微分方程 求解（7）式，并代入 Q（t）=0 的这一结果 数据即得：

$$\mathrm{Q(t)} = \frac{D(x)}{\lambda(t)}\left(e^{\lambda(t)^{(T-t)}-1}\right), 0 \le t \le T \tag{8}$$

在某类蔬菜品类的订货 周期 T 内，当 t=0 时，代入（8）式可得该品类蔬菜商品的供应商将该蔬菜 商品品类转卖给该 生鲜商超时的库存水平为：

$$Q_o = \frac{D(x)}{\lambda(x)}\left(e^{\lambda(t)(T-1)} - 1\right) \tag{9}$$

假设该生鲜商超的各种成本构成如下：

（1）某批次某蔬菜品类商品订货 成本 ： S；

（2）某批次某蔬菜品类商品进货 成本 ： $\mathrm{p}Q_o$；

（3）某批次某蔬菜品类商品库存持有成本 ： $\mathrm{Hp}\int_0^T Q(t)dt$;

（4）某批次某蔬菜品类商品价值损耗成本 ： $\lambda(\mathrm{t})\mathrm{p}\int_0^T Q(t)dt$;

由上述该家生鲜商超的 成本分项可知，该生鲜商超销售某一蔬菜品类商品的总成本 TC 等于上述各式的和:

$$\mathrm{TC} = \mathrm{S} + \mathrm{p}Q_o + \mathrm{Hp}\int_0^T Q(t)dt + \lambda(t)p\int_0^T Q(t)\,dt$$

$$= S + p\frac{D(x)}{\lambda(t)}\left(e^{\lambda(t)T} - 1\right) + Hp\frac{D(x)}{[\lambda(t)]^2}\left(e^{\lambda(t)T} - 1 - \lambda(t)T\right)$$
$$+\lambda(t)p\frac{D(x)}{[\lambda(t)]^2}\left(e^{\lambda(t)T} - 1 - \lambda(t)T\right) \quad (10)$$

其中由于直接根据（10）式最值的精确 表达式进行求解的难度较高 ，我们团队应用指数函数的近似算法对该数据表达式进行求解 ，即当θt的数据取值达到足够小时 ，运用泰勒公式的展开式对该表达式进行转化化简 ：

当$\theta t < 1$时，$e^{\lambda(t)T}$可以用 $1+\lambda(t)T+\frac{1}{2}(\lambda(t)T)^2$近似代替；

将（9）式代入（10）式可得，则该生鲜商超在单位时间上销售某一蔬菜品类商品的平均总成本等于TC_1：

$$TC_1 = \frac{TC}{T} = \frac{S}{T} + p(a - bx) + \frac{1}{2}[H + 2\lambda(t)]p(a - bx)T \quad (11)$$

该生鲜商超单位时间的平均总 利润为 G：

$$G = x(a - bx) - TC$$
$$= x(a - bx) - \frac{S}{T} - p(a - bx) - \frac{1}{2}[H + 2\lambda(t)]p(a - bx)T \quad (12)$$

对(12)式中的λ求偏导可以得到下式：

$$\frac{\partial G}{\partial T} = \frac{S}{T^2} - \frac{1}{2}[H + 2\lambda(t)]p(a - bx) \quad (13)$$

再对(13)式求二阶 偏导

$$\frac{\partial^2 G}{\partial T^2} = -\frac{2S}{T^3} \leq 0 \quad (14)$$

(14)式表明该生鲜商超在单位时间内的平均总利润函数是关于 T 的凸函数，因此存在一个最值。与此同时，对于给定的零售商的销售价格 x，我们可以得到零售商的 最优订货周期如下所示：

$$T^* = \sqrt{\frac{2S}{[H + 2\lambda(t)]p(a - bx)}} \quad (15)$$

将(14)式代入式（15）中可以得到该生鲜商超某品类蔬菜的销售价格定价为 x 时，零售商的利润：

$$G = (a - bx)(x - p) - \sqrt{2S\left(H + 2\lambda(t)\right)p(a - bx)} \quad (16)$$

当某蔬菜品类商品的批发商和对应该蔬菜品类商品的生鲜商超独自确定自己的销售价格时，该蔬菜品类商品的批发商 的单位销售价格为 p,则对应该蔬菜品类商品的生鲜商的销售价格为：

$$x = \frac{1}{b}\left(a - \frac{2S}{pT^2[H + 2\lambda(t)]}\right) \quad (11)$$

销售额为$V = x * q$（12）

(11)和(12)式即为该家生鲜商超的定价模型。由(11)式能表明该生鲜商超某一蔬菜品类的销售价格与该品类蔬菜价值损耗率的关系——当价值损耗率λ减少

时，销售价格 x 越来越小。这一关系也是正常的，毕竟蔬菜是易变质的商品，价值损耗速率降低的本身就意味着能够存储的时间更长。当对应某一蔬菜品类价值损耗速率随着时间变得越来越小时，通过及时调整该类别的生鲜蔬菜的销售价格，使得销售价格降低从而促进零售商销售量的增加。（12）式销售价格乘以销售量为零售商最终的销售额。

6.4 模型求解

根据上述对各蔬菜品类的销售总量与成本加成定价关系的分析以及运用 ARIMA 模型对 2023 年 7 月 1-7日的销量进行预测后，代入所建立的基于价值损耗的超市生鲜蔬菜定价模型，得到各蔬菜品类的销售总量未来一周（2023 年 7 月 1-7 日）的日补货总量和定价策略如下所示：

表 5 各蔬菜品类未来一周(2023 年 7 月 1-7 日)的日补货总量和定价策略

时间	类型	日补货总量	定价策略
2023 年 7 月 1 日	水生根茎类	18.45	较前一天涨 34.37%
	花叶类	140.29	较前一天涨 3.77%
	花菜类	23.10	较前一天降 32.57%
	茄类	23.33	较前一天降 6.58%
	辣椒类	84.88	较前一天降 9.27%
	食用菌类	47.95	较前一天涨 18.40%
2023 年 7 月 2 日	水生根茎类	18.07	较前一天涨 132.44%
	花叶类	143.03	较前一天涨 39.30%
	花菜类	20.79	较前一天涨 93.74%
	茄类	22.88	较前一天涨 62.80%
	辣椒类	86.05	较前一天涨 81.57%
	食用菌类	51.20	较前一天涨 42.45%
2023 年 7 月 3 日	水生根茎类	17.91	较前一天涨 52.23%
	花叶类	143.79	较前一天降 1.85%
	花菜类	19.72	较前一天降 9.98%
	茄类	22.72	较前一天涨 18.60%
	辣椒类	86.58	较前一天降 8.47%
	食用菌类	52.47	较前一天降 20.60%
2023 年 7 月 4 日	水生根茎类	17.85	较前一天降 53.31%
	花叶类	144.01	较前一天降 24.43%
	花菜类	19.22	较前一天涨 1.79%
	茄类	22.65	较前一天降 15.81%
	辣椒类	86.82	较前一天降 12.77%
	食用菌类	52.96	较前一天降 10.98%
2023 年 7 月 5 日	水生根茎类	17.83	较前一天涨 11.63%
	花叶类	144.07	较前一天涨 6.44%
	花菜类	18.99	较前一天涨 10.10%
	茄类	22.63	较前一天降 1.37%
	辣椒类	86.93	较前一天降 4.41%

	食用菌类	53.15	较前一天涨 13.55%
2023 年 7 月 6 日	水生根茎类	17.82	较前一天涨 9.99%
	花叶类	144.09	较前一天涨 4.73%
	花菜类	18.88	较前一天降 18.35%
	茄类	22.62	较前一天降 41.82%
	辣椒类	86.98	较前一天降 7.53%
	食用菌	53.22	较前一天降 32.12%
2023 年 7 月 7 日	水生根茎类	17.81	较前一天涨 5.19%
	花叶类	144.09	较前一天降 9.25%
	花菜类	18.83	较前一天涨 31.71%
	茄类	22.62	较前一天涨 30.78%
	辣椒类	87.00	较前一天降 11.00%
	食用菌	53.25	较前一天降低 4.71%

七、问题三模型

问题三需要在可售单品总数限定在 27-33 个且各单品订购量满足最小陈列量 2.5 千克的限定条件下，参照 2023 年 6 月 24-30 日的可售品种，制订出 7 月 1 日的单品补货量和定价策略。根据附件所给数据，该生鲜商超一天售卖的蔬菜单品的种类是足够的，另需满足的条件是所选择的蔬菜单品的最小陈列量是 2.5kg 的基本要求，即进货量至少要 2.5kg 以上。另外，还需要使得该生鲜商超的利润最大化，即尽可能满足销量等于进货量的约束条件。因此，需要在筛选出 2023 年 6 月 24-30 日均有售的蔬菜单品的种类的基础上，根据当天的销量总和筛选出一部分蔬菜单品，然后根据这部分蔬菜单品的利润率进行排序，选取出可售单品在 27-33 个时候的 7 月 1 日单品补货量和定价策略。

7.1 数据的筛选

根据附件 2 所提供的单日流水明细数据，可以得到 2023 年 6 月 24 日到 6 月 25 日该商超的可售蔬菜单品的数据。对数据进行统计分析以及预测后，我们要从 250 个单品中选择出最优的 27-33 个单品。

整数线性规划是运筹学中的一种优化算法，用于解决在决策变量必须取整数时的问题。它的目标是在一组线性式子的约束下，让目标线性函数取得最优值。

在这个问题中，我们将每一种蔬菜单品看作为一个决策变量。目标函数就是所有选中的蔬菜单品在 7 月 1 日这天销售的总利润，并且有约束条件：决策变量数目介于 27 至 33 之间，并且单个选择蔬菜的期望销售量应该大于 2.5kg。

7.2 模型的构建

第一步，定义决策变量和相关参数。已知有 250 种蔬菜单品可供选择，我们定义决策变量$x[i]$表示第 i 种蔬菜单品，其值为 1 代表选择，值为 0 代表不选择。由于其值只能为 0 或 1，这就构成了整数决策变量问题。在此基础上为了便于表示和理解，我们又添加了一些变量和函数符号如下表 6 所示，其中后缀$[i]$均表示第 i 种蔬菜的情况。

表 6 参数符号及含义

符号	含义
$x[i]$	决策变量
$C[i]$	期望销售量
$S[i]$	期望售价
$F[i]$	期望批发价格
z	期望函数

第二步，构造目标函数，每个蔬菜的总利润 可以这样子计算：期望售价乘以期望销量，减去批发价格乘以期望销量。因此，7 月 1 日一整天的总利润就是所有选中蔬菜单品的当天利润之和，可以用数学形式表示如下：

$$z = \sum_{i=1}^{n} (x[i] * C[i] * (S[i] - F[i]))$$

第三步，添加约束条件，因为题目规定选取 27-33 个蔬菜单品，所以应当满足下面这个约束式子：

$$27 \leq \sum_{i=1}^{n} x[i] \leq 33$$

另一个约束是每一个选中的蔬菜单品，其预期销售量都要不低于 2.5kg，这里可以也简单用个不等式表示

$$x[i] * C[i] \geq 2.5$$

至此整数线性规划模型都构建完毕了，如同整数线性规划数学模型的一般形式，我们构建了如下模型：

$$\max z = \sum_{i=1}^{n} (x[i] * C[i] * (S[i] - F[i]))$$

$$\begin{cases} x[i] \in \{0,1\}, i = 1,2,\dots,n \\ 27 \leq \sum_{i=1}^{n} x[i] \leq 33 \\ x[i] * C[i] \geq 2.5, i = 1,2,\dots,n \end{cases}$$

7.2 问题的求解

在构建完整数线性规划模型之后，我们团队使用 python 的 pulp 库解决这个问题。pulp 作为 python 的线性规划库，可以将优化问题描述构建为数学模型，生成 MPS 或者 LP 文件，之后再调用 LP 求解器，如 CBC、GLPK、CPLEX 等来进行求解，可以用来建模和解决线性规划问题。

于是我们团队，利用 python 计算解决这个问题，导入 pulp 库，对问题进行初始化然后定义决策变量和目标函数，再添加约束条件，并最终统计了最终求得的数据结果。

通过对数据的求解，得到销售量满足基本要求且该蔬菜单品的利润率较大的

蔬菜单品数据。在此基础上，通过计算该蔬菜单品的销售额与进货成本的差值即利润，并根据蔬菜单品数量控制在 27-33 之间进行比较，最终得到相应的 7 月 1 日的单品补货量和定价策略如下：

表 7 7 月 1 日的单品补货量和定价策略

蔬菜单品	单品补货量	定价策略
上海青	3.86	较前一天涨 34.37%
云南油麦菜(份)	21.29	较前一天涨 3.77%
云南生菜(份)	32.29	较前一天降 32.57%
净藕(1)	6.03	较前一天降 6.58%
双孢菇(盒)	10.00	较前一天降 9.27%
姜蒜小米椒组合装(小份)	7.00	较前一天涨 18.40%
娃娃菜	10.43	较前一天涨 52.23%
小皱皮(份)	15.00	较前一天降 1.85%
小米椒(份)	21.43	较前一天降 9.98%
小青菜(1)	4.90	较前一天涨 18.60%
木耳菜	5.93	较前一天降 8.47%
洪湖藕带	4.03	较前一天降 20.60%
海鲜菇(包)	8.86	较前一天涨 5.19%
竹叶菜	13.30	较前一天降 9.25%
紫茄子(2)	10.90	较前一天涨 31.71%
红薯尖	4.51	较前一天涨 30.78%
芜湖青椒(1)	14.23	较前一天降 11.00%
苋菜	8.92	较前一天降低 4.71%
螺丝椒	6.84	较前一天涨 11.63%
螺丝椒(份)	11.29	较前一天涨 6.44%
西兰花	12.56	较前一天涨 10.10%
西峡花菇(1)	4.62	较前一天降 1.37%
金针菇(盒)	16.14	较前一天降 4.41%
长线茄	4.21	较前一天涨 13.55%
高瓜(1)	3.00	较前一天涨 34.37%
竹叶菜	10.42	较前一天涨 3.77%
枝江青梗散花	6.92	较前一天降低 4.71%
菠菜(份)	6.6	较前一天降 1.37%

八、问题四模型

问题四需要提出自己的意见关于商超仍需采集的数据类型有哪些，是指以商超制定更合适更优的蔬菜补货策略与蔬菜商品价格确定策略为目的所需要采集的数据。我们将考虑各个角度，涉及消费经济学、消费心理学、物流与采购管理学、市场营销学、金融学、信息网络技术等学科的知识，总结出以下的数据类型对于生鲜商超的补货策略与定价策略造成了直接或间接的影响，它们简单概括如下：客流量数据、消费者反馈与评价数据、蔬菜商超存储成本等。

8.1 消费者反馈和评价数据以及消费者价格敏感系数

为了确保做到生鲜商超补货量和零售价格的优化决策，商超必然需要通过一些途径收集收费这反馈和评价的数据，这是最直观最一手地获取消费者对于商超的看法，获取的这些数据经过商超整理归纳分析，可以得到全方面的改进和优化。我们侧重其中一点即就是关于消费者价格敏感系数的数据，当然途径不止通过消费者反馈获取，还可以统计商超消费记录中消费者消费变化与蔬菜商品价格变化两者之间的关系得到消费者对于价格的敏感程度。之所以考虑商超采集提取此数据的原因，就是因为它对于蔬菜零售价与订货量会有一定程度的影响。

首先，采集消费者价格敏感系数数据，首先可以通过 PSM 测试，全称为 Price Sensitivity Meter 既价格敏感度测试，由经济学家 Peter Van Westendorp 与 1976 年提出。商超可以向消费者提问四个问题从而总结出价格敏感度。

其次，为了科学精准地证明价格敏感度对于商超补货和定价存在相关性，我们决定构建考虑市场需求、批发价格、零售价格、商品补货量、消费者价格敏感系数、商超保鲜水平、消费者保险敏感系数、商朝风险规避系数、商超利润、期望总效用因素的模型研究消费者价格敏感系数关于商品补货量和零售价格之间的影响。

首先指定一些符号及其含义，如表 8 所示：

表 8 指定的符号以及函数表达式的含义

符号及函数表达式	含义
β	生鲜零售价格
γ	生鲜零售价格商超订购量
α	消费者对于商超保鲜努力敏感系数
$\mathcal{W}$	商品批发价格
e	商超保鲜水平
a	市场需求
b	消费者敏感系数
$f(\theta)$	概率密度函数
$F(\theta)$	累积分布函数
$D = d(\beta,e) + \theta = a - b*\beta + \alpha*e + \theta$	市场需求

生鲜零售价格商超订购量、消费者对于商超保鲜努力敏感系数、商品批发价格、商超保鲜水平、市场需求、消费者敏感系数。假设$f(\theta)$、$F(\theta)$分别为概率密度函数和累积分布函数，

市场需求。

（13）式为商超期望收益函数[7]，最后一项为商品保险成本与残缺商品亏损值。

$$E\pi = \beta * C(\beta,\gamma,\alpha) - \mathcal{W}*\gamma - k*\frac{e^2}{2} \tag{13}$$

其中所求的 C 是生鲜商超的期望销售量。

$$C(\beta,\gamma,\alpha) = \gamma - \int_0^{\gamma - d(\beta,e)} F(\theta)d\theta \tag{14}$$

所以我们可以得到剩余库存量相关式子（15）

$$I = \gamma - C(\beta,\gamma,\alpha) \tag{15}$$

至此，模型大体框架构建完毕，接着，结合生鲜商家损失规避模型结论，引入风险规避系数，再表示出商超期望效用（16）式

$$U(\pi)=\begin{cases}\pi，若\pi>0\\ \lambda*\pi，若\pi<0\end{cases} \tag{16}$$

进一步联立前式，应用多元积分学，求解出最终方程式，利用 MATLAB 求解方程式，我们可以得到所引入的各变量之间的相连关系，再带入商家已采集的数据，可以观察各数据间的变化以及各数据之间的影响。

经过模型运算，以消费者价格敏感系数为核心，通过已建模型函数对其进行灵敏度分析 ，我们采集抽样了部分具有代表性的附件中的数据并进行分析处理归纳总结，同时根据实际情况引入了新的数据，从而运算出了本次核心研讨的数据——消费者价格敏感系数，并对其简单的统计为折线图：

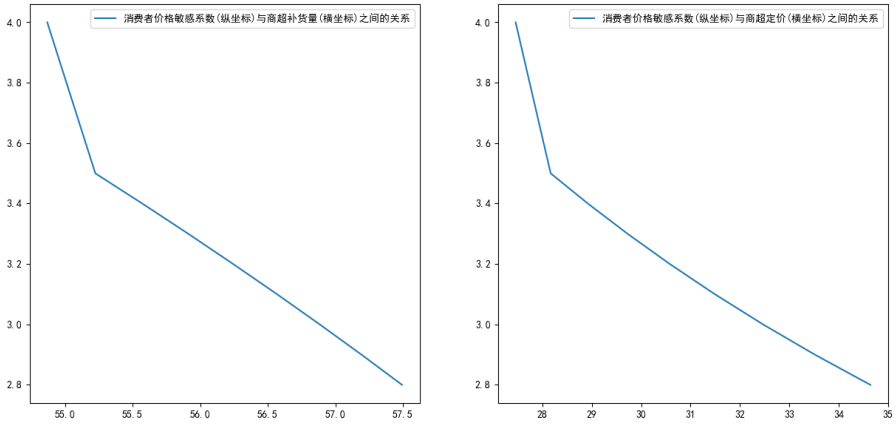

图 12 消费者价格敏感系数与商超补货量和商超定价之间的关系

根据折线图，关系很显然，消费者价格敏感系数对此两变量的影响显而易见[6]。随着消费者价格敏感系数的提高，商超商品价格随之降低，对于商超蔬菜产品，消费者对价格不敏感时，蔬菜价格较高；当消费者对价格敏感时，蔬菜价格较低。同理，随着消费者价格敏感系数的上升，商超补货量随之降低[8]。

总结来说，我们认为商超应采集收集消费者反馈数据与评价看，或通过种种途径，采集消费者价格敏感系数，观察消费者价格敏感系数，对其进行周期性预测并观察其动态变化，可以利于商超及时调整蔬菜商品定价策略与补货量策略。

8.2 数据角度二：蔬菜商超存储成本

在有关生鲜商品供应链方向的学习中，我们学习到了一种经济批量模型 EOQ，其主要目的就是通过分析以往的一些数据指定较为合适的补货策略。经济订货量通过平衡订货成本和仓库存储成本来确定最低总成本的最佳订货量。我们可以通过采集固定时间段周期性的仓库成本变化，从而推测运算出接下来的最佳订货量。

订货成本很好理解，就是总订货量与单词订货成本的乘机；商超存储成本即就是商超进货后仓库内所有商品成本，求其平均库存，就是某次进货总成本除以2，可以理解为进货时成本为总成本，销售完毕后总成本为零，在这一段时间上，平均进货成本就是进货总成本除以二。

EOQ 模型的结论为：持有成本 = 订货成本。相关参数及含义如下表 9 所示

表 9 参数及其含义表

符号	含义
A	全年需求
S	订货成本
i	库存持有成本率
C	蔬菜商品销售单价
Q	商品订货数量

第一步计算订货成本

$$w_1 = \frac{A}{Q} * S \tag{17}$$

紧接着计算平均持有成本

$$w_2 = \frac{Q}{2} * i * C \tag{18}$$

根据模型，令订货成本 = 平均持有成本，从而解出商品订货数量

$$EOQ = \sqrt{\frac{2 * A * S}{i * C}} \tag{19}$$

至此，就得到了EOQ的公式。

其实上述求Q的过程也可以通过图形表现出来，如下图 14 所示，横坐标为订货量，纵坐标有三个，蓝色是订货成本，红色是平均持有成本，绿色为红蓝之和即就是总成本。在图上也表现了出来，当某一订货量值使得订货成本和平均持有成本相等时，总成本最低

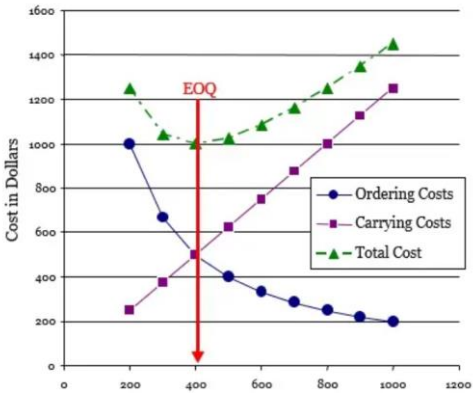

图 13 EOQ 中成本与订货量的关系

九、 模型的评价

9.1 模型的优点

问题一中，通过皮尔逊相关系数分析，对不同蔬菜商品品类之间以及不同蔬菜单品之间的相关性进行求解。

问题二中，通过 ARIMA 模型对各蔬菜品类的销售量进行了预测。

问题二中，构建的预测模型综合考虑了销售量、损耗率、利润率等各个方面的因素。

9.2 模型的缺点

数据预处理中删除了少部分数据，忽略了此数据带来的影响，导致分析不够全面。

参考文献

[1] 潘晓飞,解志恒,王淑云.考虑损失规避的生鲜品商超保鲜努力和定价的优化决策[J].公路交通科技,2022,39(06):177-185+190.

[2] 韩丹丹,马硕.浅析新零售业态下生鲜商超的创新型运营模式——以“盒马鲜生”为例[J].中国集体经济,2022(01):109-110+164.

[3] 沈正舜,唐步龙.基于实体损耗和价值损耗的生鲜农产品供应链协调机制[J].江苏农业科学,2014,42(02):388-390.DOI:10.15889/j.issn.1002-1302.2014.02.283.

[4] 陈军. 考虑流通损耗控制的生鲜农产品供应链订货策略及供需协调研究[D].重庆大学,2009.

[5] 但斌,陈军.基于价值损耗的生鲜农产品供应链协调[J].中国管理科学,2008(05):42-49.DOI:10.16381/j.cnki.issn1003-207x.2008.05.025.

[6] 毛莉莎. 供应链视角下蔬菜批发市场定价策略及产销模式研究[D].中南林业科技大学,2023.DOI:10.27662/d.cnki.gznlc.2022.000680.

[7] 卢亚杰. 我国超市优质生鲜蔬菜动态定价问题研究[D].北京交通大学,2010.

[8] 佘梓航,徐嘉桦,姚志玉等.基于皮尔逊相关系数的网购大数据分析——以天猫佰润居旗舰店交易记录为例[J].韩山师范学院学报,2020,41(03):16-22.

附录 A 文件列表

表 10 支撑材料文件列表

文件名	文件描述
第一问.py	问题一的代码
第二问.py	问题二的代码
第三问.py	问题三的代码
Util.py	工具函数
daily_sales.py	商品品类每一天的销售量
split_data.xlsx	分割数据集
Subplots_1.png	每一类商品的饼状占比图
top_40_categories	2023 年 6 月 24-30 日销量前 40 的商品

附录 B

第一问.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'SimHei'
matplotlib.rcParams['font.sans-serif'] = ['SimHei']

# 数据清洗

data1 = pd.read_excel('附件 1.xlsx')
print(data1.head())
print(data1.tail())
print(data1.dtypes)
print(data1.shape)
print(data1.info())
print(data1.isnull().any())
print(data1[data1.duplicated(subset=["单品编码"],keep='first')])
print(data1[data1.duplicated(subset=["单品名称"],keep='first')])


data2 = pd.read_excel('附件 2.xlsx')
print(data2.head())
print(data2.tail())
print(data2.dtypes)
print(data2.shape)
print(data2.info())
print(data2.isnull().any())
print(data2[data2.duplicated(subset=["扫码销售时间"],keep='first')])

data3 = pd.read_excel('附件 3.xlsx')
```

```
print(data3.head())
print(data3.tail())
print(data3.dtypes)
print(data3.shape)
print(data3.info())
print(data3.isnull().any())


data4 = pd.read_excel('附件 4.xlsx')
print(data4.head())
print(data4.tail())
print(data4.dtypes)
print(data4.shape)
print(data4.info())
print(data4.isnull().any())

# 第一问
import pandas as pd
import matplotlib.pyplot as plt

# 读取原始数据表
df = pd.read_excel('附件 2.xlsx')

# 将销售日期和扫码销售时间列转换为日期时间类型
df['销售日期'] = pd.to_datetime(df['销售日期'])
df['扫码销售时间'] = pd.to_datetime(df['扫码销售时间'])

# 根据单品编码和销售日期进行分组，并计算每天每个单品的销售量总和
result = df.groupby(['单品编码', '销售日期']).agg({'销量(千克)':
'sum'}).reset_index()

# 创建子图，并指定图表标题和坐标轴标签
fig, ax = plt.subplots(figsize=(8, 6))  # 调整子图的大小

# 绘制每个单品的销量折线图
for item in result['单品编码'].unique():
    item_data = result[result['单品编码'] == item]
    ax.plot(item_data['销售日期'], item_data['销量(千克)'], label=item)

# 设置图表标题和坐标轴标签
ax.set_title('每个单品的销量变化')
ax.set_xlabel('销售日期')
ax.set_ylabel('销量(千克)')
```

```
# 创建图例，并将其放置在子图外部
fig.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# 调整子图布局，以便容纳图例
fig.tight_layout()

# 显示图表
plt.show()

# 按销售量降序排序，并获取销售量前 30 种的单品编码
top_30_items = result.groupby('单品编码')['销量(千
克)'].sum().sort_values(ascending=False).head(30).index

# 创建子图，并指定图表标题和坐标轴标签
fig, ax = plt.subplots(figsize=(8, 6))

# 绘制销售量前 30 种单品的折线图
for item in top_30_items:
    item_data = result[result['单品编码'] == item]
    ax.plot(item_data['销售日期'], item_data['销量(千克)'], label=item)

# 设置图表标题和坐标轴标签
ax.set_title('销售量前 30 种单品的销量变化')
ax.set_xlabel('销售日期')
ax.set_ylabel('销量(千克)')

# 创建图例，并将其放置在子图外部
fig.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# 调整子图布局，以便容纳图例
fig.tight_layout()

# 显示图表
plt.show()

import pandas as pd
import matplotlib.pyplot as plt

# 读取原始数据表
df = pd.read_excel('附件 2.xlsx')

# 将销售日期列转换为日期时间类型
df['销售日期'] = pd.to_datetime(df['销售日期'])
```

```
# 根据单品编码和销售日期进行分组，并计算每天每个单品的销售量总和
result = df.groupby(['单品编码', '销售日期']).agg({'销量(千克)':
'sum'}).reset_index()

# 按月份进行分组，并计算当月每个单品的销售量和当月总销售量
monthly_data = result.groupby([result['销售日期'].dt.year.rename('年份'),
                               result['销售日期'].dt.month.rename('月份'),
                               '单品编码'])['销量(千
克)'].sum().reset_index()
monthly_total = monthly_data.groupby(['年份', '月份'])['销量(千
克)'].sum().reset_index()

# 获取唯一的月份列表
months = monthly_data[['年份', '月份']].drop_duplicates().values

# 分割月份列表为每个画布的子图数量
num_subplots = 4
num_plots = len(months)
num_figures = num_plots // num_subplots + (1 if num_plots %
num_subplots != 0 else 0)
subplot_indices = [months[i * num_subplots:(i + 1) * num_subplots] for
i in range(num_figures)]

# 创建存储前 30 单品数据的 DataFrame
top30_data = pd.DataFrame()

# 遍历每个画布
for i, subplot_index in enumerate(subplot_indices):
    fig, axs = plt.subplots(2, 2, figsize=(12, 8))
    fig.subplots_adjust(hspace=0.5)

    # 遍历当前画布中的子图索引
    for j, (year, month) in enumerate(subplot_index):
        month_data = monthly_data[(monthly_data['年份'] == year) &
(monthly_data['月份'] == month)]
        month_total = monthly_total[(monthly_total['年份'] == year) &
(monthly_total['月份'] == month)]['销量(千克)'].values[0]

        month_data = month_data[month_data['销量(千克)'] > 0]  # 去除销量
为 0 的单品

        if len(month_data) == 0:
            continue  # 如果没有需要绘制的单品，则跳过该月份
```

```
        sizes = month_data['销量(千克)'] / month_total

        row = j // 2
        col = j % 2

        axs[row, col].pie(sizes, labels=None, autopct='%.1f%%',
startangle=90, pctdistance=0.85)
        axs[row, col].set_aspect('equal')

        # 设置图表标题
        axs[row, col].set_title(f'{year}年{month}月每个单品销量占比')

        # 将当前月销量前30的单品信息添加到top30_data
        top30_month_data = month_data.nlargest(30, '销量(千克)')
        # top30_data = top30_data.append(top30_month_data)
        top30_data = pd.concat([top30_data, top30_month_data])
    # 调整子图之间的间距
    plt.tight_layout()

    # 保存当前画布为图片，或者根据您的需求进行其他处理
    fig.savefig(f'subplots_{i+1}.png')

    # 关闭当前画布
    plt.close(fig)

# 将top30_data保存为新的Excel文件
top30_data.to_excel('top30_sales.xlsx', index=False)
```

第二问.py

```
import pandas as pd

# 读取原始数据表格
data = pd.read_excel('C2.xlsx')

# 将销售日期列的数据类型转换为 datetime
data['销售日期'] = pd.to_datetime(data['销售日期'])

# 提取所需列
columns = ['销售日期', '大类', '销量']
data = data[columns]

# 按日期和大类分组并求和
grouped_data = data.groupby(['销售日期', '大类'])['销量
'].sum().reset_index()
```

```
# 将数据透视为每一天为行，每一大类为列的形式
pivot_data = pd.pivot_table(grouped_data, values='销量', index='销售日期
', columns='大类', fill_value=0)

# 将结果保存到 xlsx 文件中
pivot_data.to_excel('daily_sales.xlsx')

print("已保存每一天每一大类商品的销售量总和到 daily_sales.xlsx 文件中。")

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
import matplotlib
matplotlib.rcParams['font.family'] = 'SimHei'
matplotlib.rcParams['font.sans-serif'] = 'SimHei'
# 读取原始数据表格
data = pd.read_excel('daily_sales.xlsx')

# 提取所需列（销售日期、水生根茎类、花叶类、花菜类、茄类、辣椒类、食用菌列）
columns = ['销售日期', '水生根茎类', '花叶类', '花菜类', '茄类', '辣椒类',
'食用菌']
data = data[columns]

# 将销售日期列的数据类型转换为 datetime
data['销售日期'] = pd.to_datetime(data['销售日期'])

# 设置日期列为索引
data.set_index('销售日期', inplace=True)

# 使用 ARIMA 模型进行预测
predictions = {}
for col in data.columns:
    model = ARIMA(data[col], order=(1, 1, 1))
    model_fit = model.fit()
    forecast = model_fit.forecast(steps=7)
    predictions[col] = forecast

# 创建预测结果的数据框
pred_data = pd.DataFrame(predictions)

# 创建日期范围（2023 年 7 月 1 日到 2023 年 7 月 7 日）
date_range = pd.date_range(start='2023-07-01', end='2023-07-07')
```

```
# 设置日期范围作为索引
pred_data.set_index(date_range, inplace=True)

# 保存预测结果到 xlsx 文件
pred_data.to_excel('predicted_sales.xlsx')

# 可视化预测过程
plt.figure(figsize=(10, 6))

for col in pred_data.columns:
    plt.plot(data.index, data[col], label=col)
    plt.plot(pred_data.index, pred_data[col], label='Predicted '+col,
linestyle='dashed')

plt.title('预测销售量')
plt.xlabel('日期')
plt.ylabel('销售量')
plt.legend()
plt.xticks(rotation=45)
plt.grid(True)

plt.savefig('sales_forecast.png')
plt.show()

print("已保存预测结果到 predicted_sales.xlsx 文件，并生成了销售预测的可视化
图像 sales_forecast.png。")

import pandas as pd

# 读取原始数据表格
data = pd.read_excel('C2.xlsx')

# 按单品类和是否打折进行数据拆分
discounted_data = data[data['是否打折销售'] == '是']
non_discounted_data = data[data['是否打折销售'] == '否']

# 保存拆分后的数据到新的 Excel 表格
with pd.ExcelWriter('split_data.xlsx') as writer:
    discounted_data.to_excel(writer, sheet_name='打折销售', index=False)
    non_discounted_data.to_excel(writer, sheet_name='非打折销售',
index=False)

print("已将分好的数据保存到 split_data.xlsx 文件中。")
```

```
import pandas as pd

# 读取原始数据表格
data = pd.read_excel('C2.xlsx')

# 按日期、是否打折和单品类进行分组，并计算销售量总和
grouped_data = data.groupby(['销售日期', '是否打折销售', '单品类'])['销量
'].sum().reset_index()

# 拆分数据为打折和非打折两个 DataFrame
discounted_data = grouped_data[grouped_data['是否打折销售'] == '是']
non_discounted_data = grouped_data[grouped_data['是否打折销售'] == '否']

# 按照日期和单品类计算打折销售量平均值
discounted_avg_sales = discounted_data.groupby(['销售日期', '单品类'])['
销量'].mean().reset_index()

# 按照日期和单品类计算非打折销售量平均值
non_discounted_avg_sales = non_discounted_data.groupby(['销售日期', '单品
类'])['销量'].mean().reset_index()

# 保存统计结果到新的 Excel 表格
with pd.ExcelWriter('average_sales.xlsx') as writer:
    discounted_avg_sales.to_excel(writer, sheet_name='打折销售',
index=False)
    non_discounted_avg_sales.to_excel(writer, sheet_name='非打折销售',
index=False)

print("已将每一天打折商品和非打折商品的销售量平均值按单品类保存到
average_sales.xlsx 文件中。")
```

第三问.py

```
import pandas as pd

# 读取 Excel 文件
df = pd.read_excel('6 月 24 到 30 的数据.xlsx')

# 将日期列转换为日期类型
df['销售日期'] = pd.to_datetime(df['销售日期'])

# 筛选出指定时间段内的数据
start_date = pd.to_datetime('2023-06-24')
end_date = pd.to_datetime('2023-06-30')
```

```
filtered_df = df[(df['销售日期'] >= start_date) & (df['销售日期'] <=
end_date)]

# 统计每个单品类在每天的出现次数
daily_counts = filtered_df.groupby(['单品类', '销售日期
']).size().reset_index(name='出现次数')

# 筛选出在每天都出现的单品类
common_categories = daily_counts.groupby('单品类').filter(lambda x: x['
出现次数'].count() == (end_date - start_date +
pd.Timedelta(days=1)).days)
common_categories = common_categories['单品类'].unique()

# 筛选出在每天都出现的单品类的数据
result_df = filtered_df[filtered_df['单品类'].isin(common_categories)]

# 保存结果到新的 Excel 文件
result_df.to_excel('重复出现的单品.xlsx', index=False)

# 计算每个单品类每天的销量总和
daily_sales_totals = result_df.groupby(['单品类', '销售日期'])['销量
'].sum().reset_index(name='销量总和')

# 筛选出每天销量总和前 40 的商品单类
top_40_categories = daily_sales_totals.groupby('销售日期').apply(lambda
x: x.nlargest(40, '销量总和')).reset_index(drop=True)

# 保存结果到新的 Excel 文件
top_40_categories.to_excel('top_40_categories.xlsx', index=False)

filtered_df = df[df['销量总和'] > 2.5]
filtered_df.to_excel('top40>=2.5.xlsx', index=False)


import pandas as pd

# 读取 Excel 文件
df = pd.read_excel('top_40_categories.xlsx')

# 计算每个单品的销量总和的平均值
average_sales = df.loc[(df['销售日期'] >= '2023-06-24') & (df['销售日期']
<= '2023-06-30')].groupby('单品类')['销量总和'].mean()

# 将结果保存为新的 Excel 文件
```

```
output_filepath = 'path_to_save_output.xlsx'
average_sales.to_excel(output_filepath)

print("结果已保存到文件:", output_filepath)
```