

基于销量及需求分析的蔬菜自动定价与补货模型

摘要

生鲜商超中的蔬菜品相随销售时间变差，商家每天需在缺乏具体信息时补货决策。蔬菜的定价一般采用"成本加成定价"方法，并对运输过程破损和品相变差的商品进行打折销售。在需求方面，蔬菜类商品的销售量与时间有关；在供给方面，供应品种在 4 月至 10 月较为丰富。

针对问题一，分别按日、按月份以及按单品、按品类来进行销量统计，绘制色阶图、聚合气泡图、双坐标轴图、热力图等可视化图表，进行描述性统计以及 **Pearson 相关性检验**。分析结果显示，花叶类蔬菜是单品种数最多、最大日销量次数最高、平均销量最高的品类；此外，不同品类之间正向相关的单品和品类显著于负向相关的单品和品类。

针对问题二，在已有每日单品售价数据的情况下，以单品月销量占所属品类月销量的比例为权重，计算各品类的月加权售价。通过**数据可视化及 Kendall 一致性检验**，得到各蔬菜品类的销售总量与成本加成定价的关系。然后基于单品日批发价数据，以销量为权重，计算各品类的日加权批发价。建立 **ARIMA 时间序列模型**，拟合过去三年的品类销量数据和加权批发价，并得到未来一周的预测数据。根据预测销量区间，筛选区间内历史最优利润率作为定价基准；结合各品类**加权损耗率**，得到日补货量。

针对问题三，本文采用**粒子群多目标优化模型**解决，首先设定有关单品数和最少进货量的约束条件以及满足品类需求和收益最大的目标，然后分别设立对应公式。设置**事件指示器 δ** ，通过设立约束使得单品总数在 27-33 个之间；设置单品订购量 x ，满足最小陈列量 2.5 千克的要求；设置**虚拟变量**，设定目标为尽量满足市场对品类的需求；最后定义变量总利润 W 、批发价 C 、利润率 ω ，设定目标为总利润最大。用上述条件建立多目标规划模型。由于模型较为复杂，运行时间较长，加入针对此题目作出优化的粒子群算法加快模型运行速度，得出预测结果。

针对问题四，经过查阅文献和本次建模分析，本团队认为应该从商家供给、消费需求、商品特性和市场竞争环境四个角度，进一步获取有价值的信息。

关键词：ARIMA 时间序列 **Kendall 一致性检验** **粒子群多目标优化模型**

一、 问题重述

1.1 问题背景

生鲜商超中的蔬菜品相随销售时间变差，商家每天需在缺乏具体信息时补货决策。蔬菜的定价一般采用"成本加成定价"方法，并对运损和品相变差的商品进行打折销售。准确的市场需求分析对补货和定价决策至关重要。在需求方面，蔬菜类商品的销售量与时间有关；在供给方面，供应品种在 4 月至 10 月较为丰富。商超销售空间有限，需合理安排销售组合。

1.2 问题回顾

根据附件中的数据，探索数据规律及相互关系，并基于结果进一步分析给出相关的商超营销建议：

问题 1：对蔬菜各品类及单品的销售数据进行分析，探索它们的分布规律和相互关系。

问题 2：通过分析蔬菜品类的销售总量和成本加成定价的关系，制定未来一周每天的补货总量和定价策略，以实现商超收益最大化。

问题 3：基于 2023 年 6 月 24-30 日的可售品种数据，制定 7 月 1 日的单品补货量和定价策略，以最大化商超收益，并确保可售单品总数在 27-33 个范围内，同时满足最小陈列量的要求。

问题 4：分析和确定商超需要采集的相关数据，以支持蔬菜商品的补货和定价决策。说明这些数据对解决问题的帮助，给出具体的意见和理由。

二、 问题分析

2.1 问题一分析

首先根据题目要求，分别通过按日、按月份以及按单品、按品类，进行销量统计。通过数据可视化、描述性统计以及相关检验，分析不同单品、不同品类之间的销量分布规律。主要流程如下：

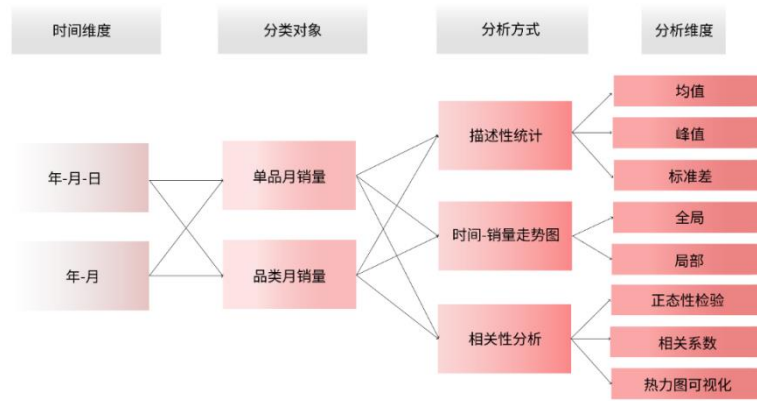


图 1：问题一分析流程

2.2 问题二分析

在分析各蔬菜品类的销售总量与成本加成定价的关系时，首先根据每日的单品售价数据，以销量为权重，计算出各品类的按月的加权售价数据作为成本加成定价方法的代表数据。首先绘制各品类折线图，初步观察比较数据趋势，难以得出有效结论；再将各品类每月销售量及加权售价的两列时间序列数据一一进行 Kendall 关联性分析，计算 Kendall's W 系数汇总成表，得到各蔬菜品类的销售总量与成本加成定价均存在显著性相关的结论。

搭建 ARIMA 时间序列模型，预测各品类未来七天的销量，得到销量区间，查找区间内的历史最优利润率，作为定价策略参考。同时根据预测出的七日销量，结合各品类加权损耗率，得到补货量。

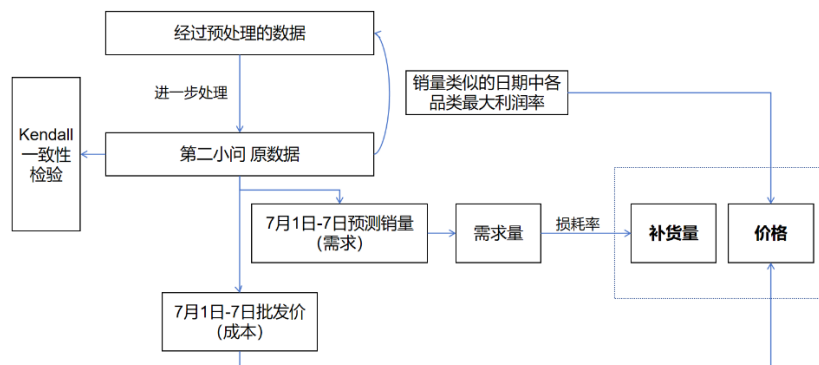


图 2：问题二分析流程

2.3 问题三分析

考虑蔬菜类商品的销售空间有限，在进一步制定单品的补货计划时，根据题目给出的约束条件，构建两目标两约束的多目标线性规划模型。同时筛选出 2023

年 6 月 24-30 日的可售单品，计算其近一周的周平均批发价和周加权利润率，作为模型的基础数据。最后通过 Matlab 及 R 语言的代码实现，得到 7 月 1 日的补货单品，以及其补货量和定价策略。

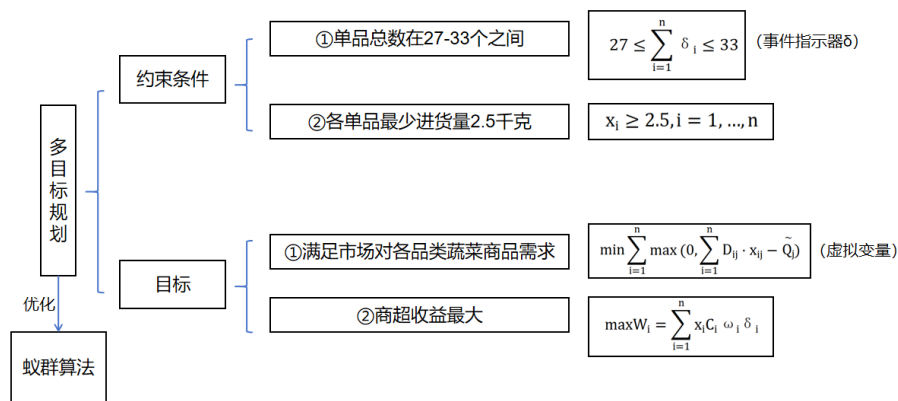


图 3：问题三分析流程

2.4 问题四分析

全面的数据采集和分析将为商超提供更全面、准确的信息基础，对于商家的持续性盈利有重大意义。经过查阅文献和本次建模分析过程，本团队认为应该从商家供给、消费需求、商品特性和市场竞争环境四个角度，进一步获取有价值、综合性的信息。

三、 基本假设及符号说明

- 1、销售数据稳定性假设：假设销售数据在短期内是相对稳定的，不受突发事件(如自然灾害) 的影响。
- 2、价格影响假设：假设价格对销售量有一定的影响，即价格上涨可能导致销售下降，价格下降可能导致销售增加。
- 3、市场竞争假设：假设市场上的竞争对手行为相对稳定，他们的价格策略和促销活动不会剧烈波动。
- 4、库存清空假设：所有的商品当日正好卖出，不占用次日的仓储和销售，即当日的库存恰好满足当日的销售和退货需求。
- 5、退货规则假设：退货行为发生在购买该商品的当日，即消费者不能退还之前日期的商品。

符号说明		
符号	含义	单位

c_i	商品单品 i 的成本	元
y_i	商品单品 i 的定价	元
p_i	商品单品 i 的批发价	元
x_i	第 i 件单品的销售量	件
Q_{it}	第 i 个品类第 t 日的销售量	件
S_{it}	第 i 个品类第 t 日的补货量	件
w_i	商品单品 i 的利润率	\
w_a	商品品类 a 的利润率	\
L_i	品类 i 的运损率	\
δ_i	第 i 件单品是否有进货	\
D_{ij}	第 i 件单品是否属于第 j 个品类	\

在文中出现、而未在本表格中提及的符号，将会在使用时进行详细说明。

四、 数据预处理

首先，根据附件 1 的单品所属品类，对附件 2 有销售记录的商品进行品类标注。通过观察 4 个初始附件，发现单品的销售单价、批发价和损耗率都为非负数，初步认为符合客观事实，无需进行异常值处理。

观察附件 2 数据，发现存在现象——对于在同一天销售的一件商品，打折后的销售单价不低于打折前的销售单价，可认为是错误数据，进行剔除。同时，发现有 6 样商品单品没有销售记录，因此在后续的销售分布规律和补货政策中，均不对这六样单品进行考虑。去除此 6 类单品后，涉及销售和订购的单品和品类关系分布如下：

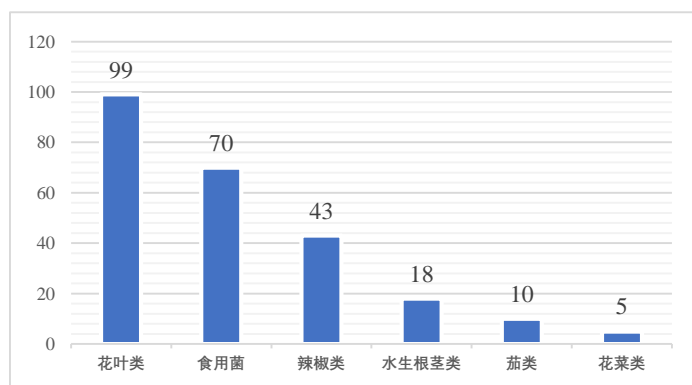


图 4：蔬菜品类的数量分布

筛除错误数据后，观察到同一天内，单品在同一天内有多次购买记录。为便于后续不同问题的分析，用两种标准统计销量：

- 1) 根据“销售类型”和“是否打折销售”两个维度，对于同一日内的同一商品，按“销售-打折”“销售-不打折”“退款-打折”“退款-不打折”四个类别进行销量汇

总。按照该步骤处理完毕，从原 788503 条数据中得到 56894 条有效数据。

表 1：双指标（“销售类型”及“是否打折销售”）分类销量汇总（按日）

销售日期	单品编码	销售类型	是否打折销售	分类名称	销量（千克）	单品名称
2020-7-1	102900005116530	销售	是	食用菌	0.195	西峡香菇(1)
2020-7-1	102900005116530	销售	否	食用菌	7.498	西峡香菇(1)
2020-7-1	102900005116547	销售	否	食用菌	13.863	金针菇(1)
2020-7-1	102900005118817	销售	否	花叶类	5.472	菠菜
2020-7-1	102900005118824	销售	否	水生根茎类	4.119	高瓜(1)
.....

2) 根据“销量（千克）”进行统计，无论商品是否打折出售，“销售类型”为“销售”的，销量则为正，“退款”的销量则为负，进行销量汇总。按照该步骤处理完毕，从原 788503 条数据中得到 46659 条有效数据。

表 2：单指标（“销售类型”）分类销量汇总（按日）

销售日期	单品编码	分类名称	销量（千克）	单品名称
2020-7-1	102900005118824	水生根茎类	4.119	高瓜(1)
2020-7-1	102900051000944	水生根茎类	0.731	洪湖藕带
2020-7-1	102900005115762	花叶类	6.841	苋菜
2020-7-1	102900005115786	花叶类	11.352	竹叶菜
.....

五、 模型建立与求解

5.1 问题一的分析与求解

5.1.1 可视化图表

在问题一中，由于只需分析品类和单品销售量的关系，故暂不考虑折扣和损耗率。为了全面得出商品品类及单品销售量的分布规律，对附件 2 进行以下分类统计销量：

1、以年-月-日为时间维度，统计各单品的日销量。对于单品的分布规律分析，

由于过小的样本量会影响模型精度，故需要在数据预处理的基础上，进一步筛除销售天数过少的单品。对于当日无销售记录的单品，将销量记录为 0。

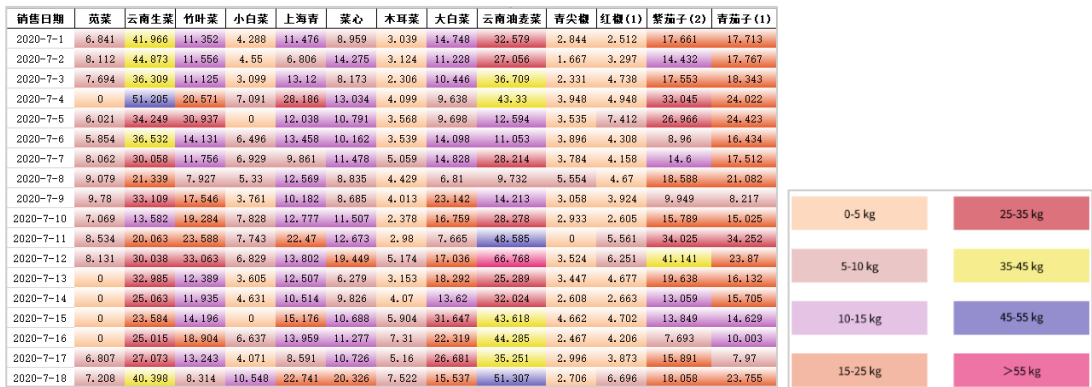


图 5：蔬菜单品日销量（部分）

绘制聚合气泡图和二维饼状图，发现有 6 种单品，占据最大日销量的天数在 3 位数以上，从多至少分别为：

表 3：占据最大日销量天数最多的商品信息

品类名称	单品名称	天数
辣椒类	芜湖青椒（1）	199
水生根茎类	净藕（1）	170
花叶类	大白菜	127
花叶类	云南生菜（份）	122
食用菌	金针菇（盒）	119

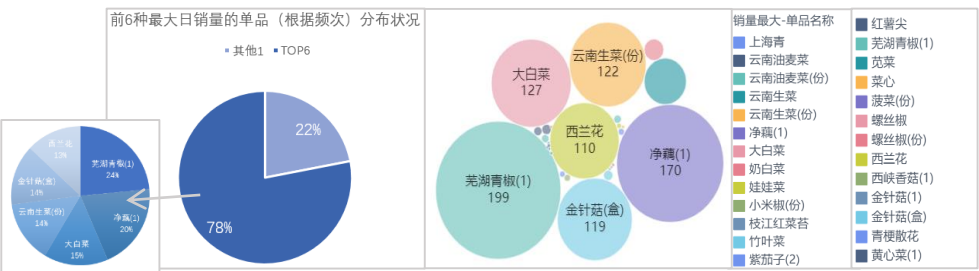


图 6：蔬菜单品前 6 种最大日销量的单品（根据频次）的分布情况

2、以年-月-日为时间维度，统计各品类的日销量。

表 4：蔬菜品类日销量

销售日期	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
------	-------	-----	-----	----	-----	-----

2020-7-1	4.85	205.402	46.64	35.374	76.715	35.365
2020-7-2	4.6	198.362	43.943	32.199	66.064	48.51
2020-7-3	9.572	190.779	42.076	35.896	64.253	42.442
2020-7-4	5.439	236.587	55.662	57.067	81.282	47.262
.....

绘制折线图和柱状图，发现花叶类商品作为最高日销量的品类的次数显著最多（1051 次），而茄类商品作为最低日销量的品类的次数出现最多次（639 次）；同时，花叶菜类商品整体上的销量最高

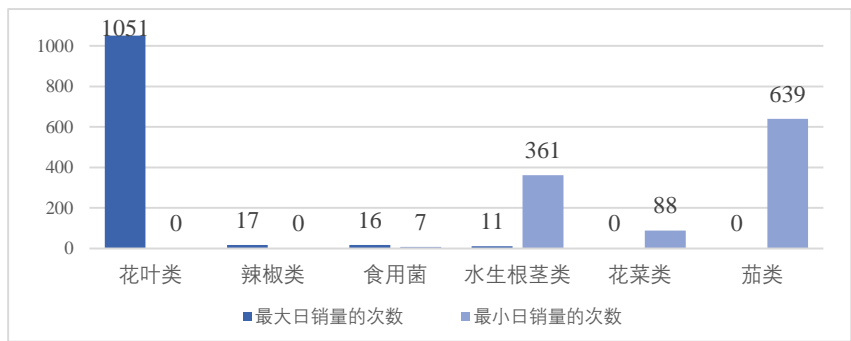


图 7：最高、最低日销量品类的频次统计

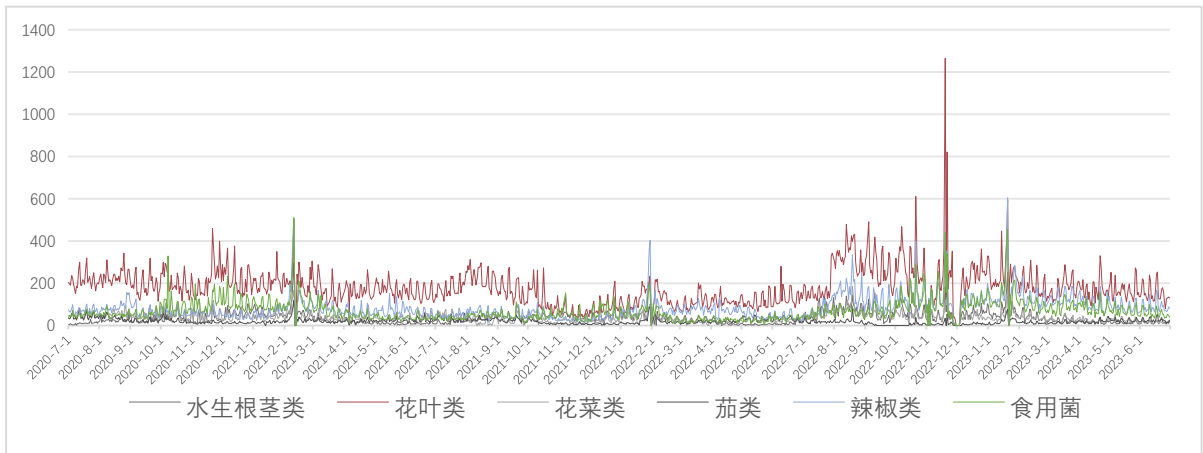


图 8：各品类日销售量折线图

3、以年-月为时间维度（纵向指标），以单品名称和单品编号为分类依据（横向指标）。对于单品的分布规律分析，由于过小的样本量会影响模型精度，故需要在数据预处理的基础上，进一步筛除销售天数过少的单品。将数据预处理后的有效数据转化为 36×47 矩阵，即 36 个月、47 类有效单品。对于当日无销售记录的单品，将销量记录为 0。

表 5：商品单品月销售量

销售日期	上海青	东门口小白菜	云南油麦菜	云南生菜	净藕(1)	圆茄子(2)	大白菜	奶白菜	
2020-7	452.999	55.09	935.567	958.044	214.199	0	505.871	0
2020-8	702.708	50.767	745.185	905.914	720.562	15.558	558.961	19.555
2020-9	529.028	47.205	319.056	856.958	757.64	176.29	600.052	225.305
2020-10	256.74	52.2	337.648	689.728	1326.863	175.213	1457.17	222.643
2020-11	152.065	24.107	330.112	375.571	943.696	55.449	2622.412	201.579
.....

根据柱状图及折线图，初步认为商品单品销售量在每个月份的分布无周期性。

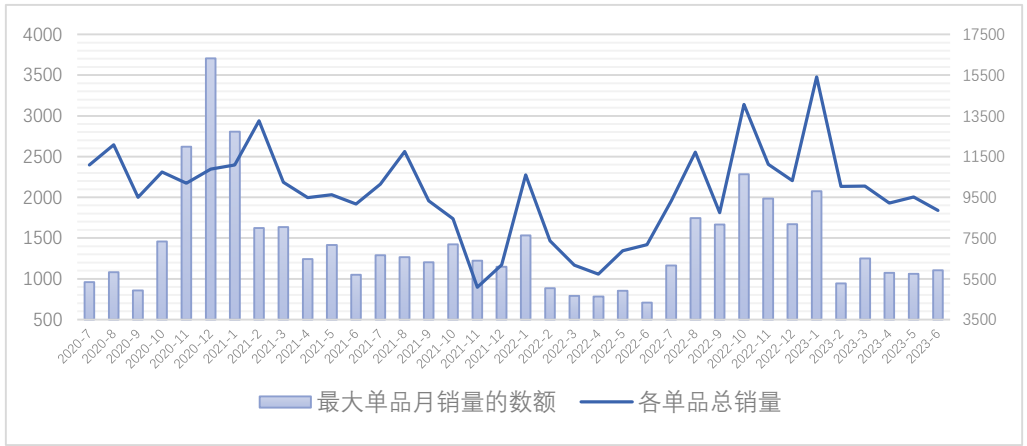


图 9：各单品最大月销售量（柱）及每月单品总销量（折）

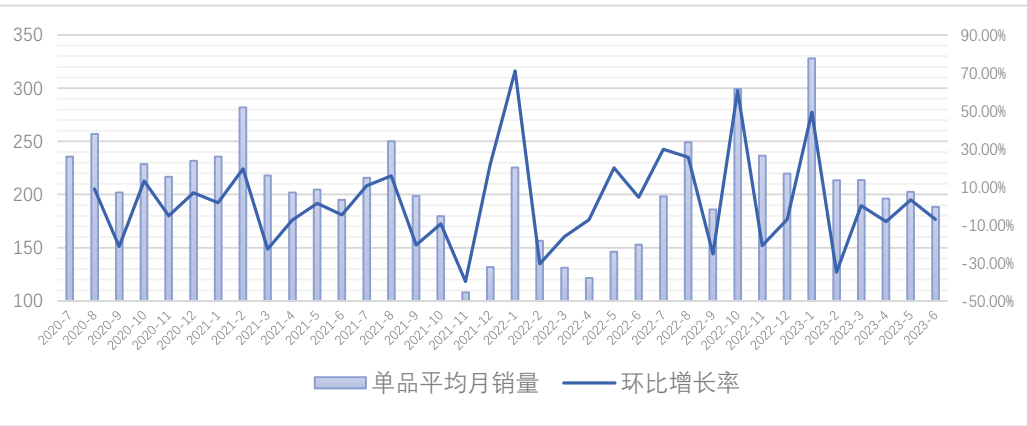


图 10：每月单品平均销售量（柱）及每月平均销量的环比增长率（折）

4、以年-月为时间维度，以品类名称和品类编号为分类依据，将数据预处理

后的有效数据分类统计至 36×6 矩阵，即 36 个月、6 种分类。

表 6：商品品类月销售量

销售日期	花叶类	花菜类	水生根茎类	茄类	辣椒类	食用菌
2020-7	6576.321	1522.555	332.056	1365.551	2198.049	1617.836
2020-8	7259.23	1748.658	867.18	1139.393	2822.344	1636.524
2020-9	5668.902	1336.066	798.931	671.692	1992.413	1642.741
2020-10	6395.312	1644.963	1344.118	771.967	1908.944	3000.532
.....

5.1.2 单品销售量分布规律

对单品销售量（按月）进行描述性统计，初步得到分布概况。在过往的 36 个月中，单月平均销量 7458.849 千克，销量最大的单品为芜湖青椒(1)(28163.89kg)，最小的单品为西峡香菇（份）（379kg）。该结果与 5.1.1 中的初步分析结论相同。

表 7：单品月销量-描述性统计

单品月销量-描述性统计					
Variable	Obs	Mean	Std. dev.	Min	Max
销量(千克)	47	7458.846	6928.364	379	28163.89

对单品月销量进行正态性分布检验，绘制每件商品的正态 Q-Q 图（Quantile-Quantile Plot）和正态 P-P 图（Probability-Probability Plot），发现样本点较好地呈一条围绕第一象限对角线的直线，可以认为单品月销量基本服从正态分布。

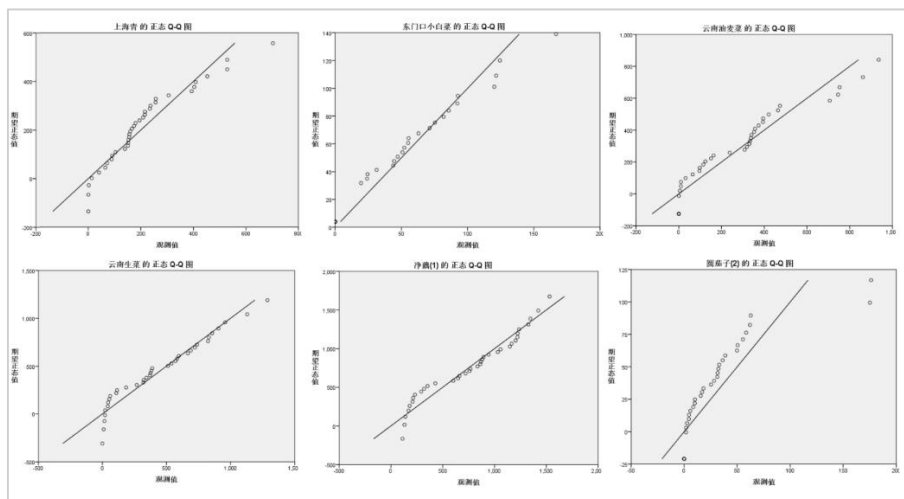


图 11：对商品单品进行 Q-Q 正态检验（详见支撑材料）

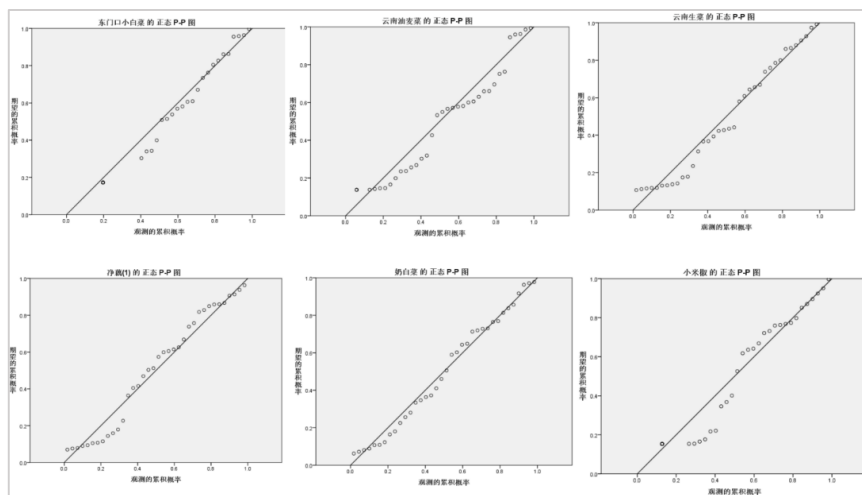


图 12：对商品单品进行 P-P 正态检验（部分）

绘制各商品单品销量的热力关系图，进行相关性分析，对较为显著的正相关和负相关商品进行总结归纳。

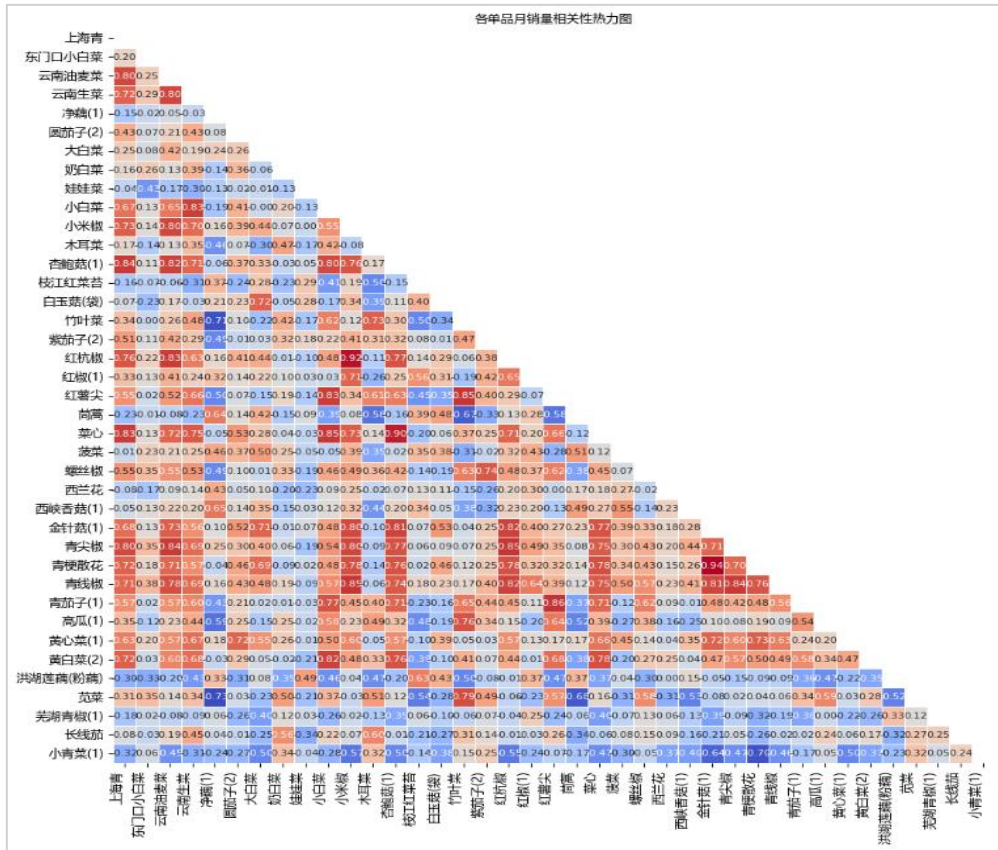


图 13: 各单品月销量相关性热力图

表 8: 根据热力图，归纳相关程度较显著的单品及其系数

正相关		负相关	
相关程度最大	青梗散花-金针菇(1) 相关系数: 0.94	相关程度最大	茼蒿-净藕(1) 相关系数: 0.73
相关性较显著	红杭椒-小米椒 0.92 菜心-杏鲍菇(1) 0.90 青茄子-红薯尖 0.86 菜心-小白菜 0.85 轻线椒-小米椒 0.85 青椒尖-红杭椒 0.85 红薯尖-竹叶菜 0.85	相关性较显著	竹叶菜-净藕(1) 0.71 小青菜(1)-青梗散花 0.70 茼蒿-茼蒿 0.68 茼蒿-竹叶菜 0.67 小青菜(1)-金针菇 0.64 高瓜(1)-净藕(1) 0.59 茼蒿-木耳菜 0.58

5.1.3 品类销售量分布规律

对品类销售量（按月）进行描述性统计，初步得到分布概况。在过往的 36 个月中，单月平均销量 58427.62 千克，销量最大的单品为花叶类(152543.51kg)，最小的单品为茄类（20662.866kg）。

表 9：单品月销量-描述性统计

单品月销量-描述性统计					
Variable	Obs	Mean	Std. dev.	Min	Max
销量千克	6	58427.62	48797.49	20662.87	152543.5

对品类月销量进行正态性分布检验，绘制每件商品的正态 Q-Q 图（Quantile-Quantile Plot）和正态 P-P 图（Probability-Probability Plot），发现样本点较好地呈一条围绕第一象限对角线的直线，可以认为品各类的月销量基本服从正态分布。

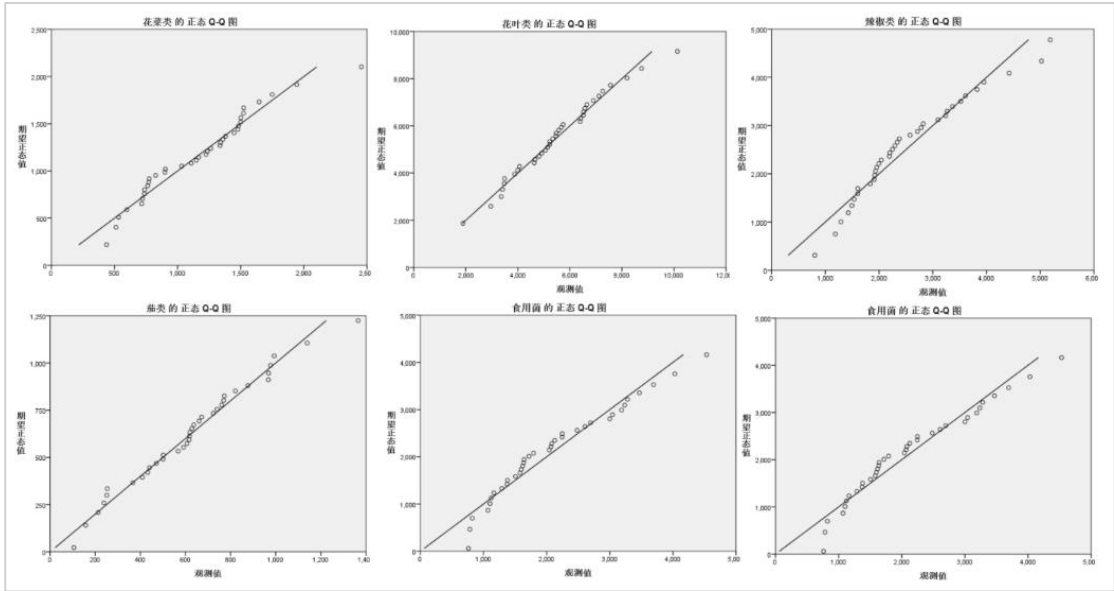


图 14：对商品品类进行 Q-Q 正态检验

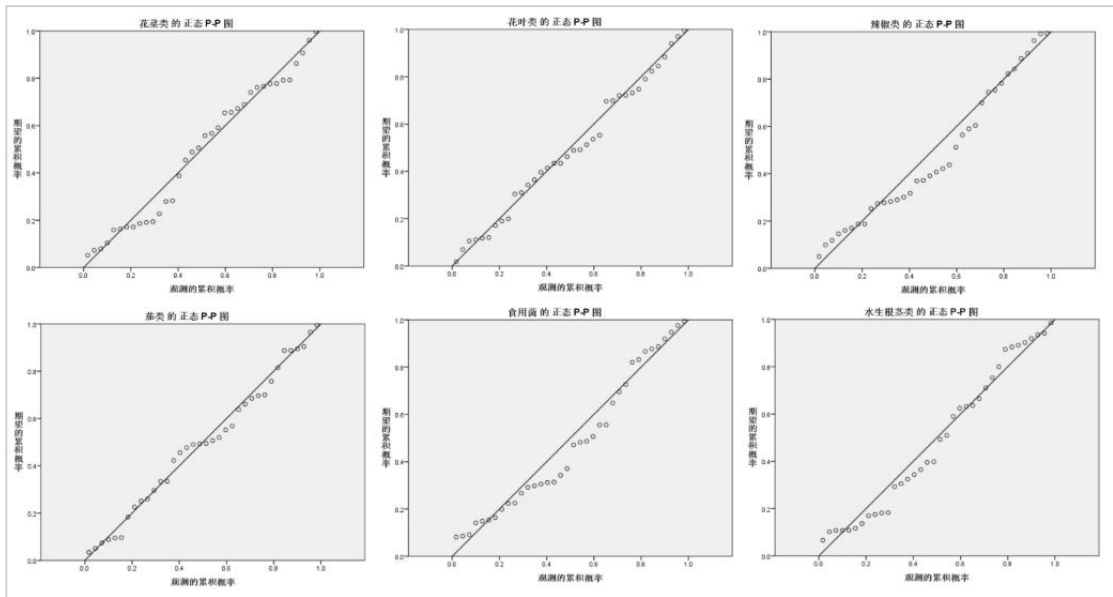


图 15: 对商品品类进行 P-P 正态检验

使用热力图，进行相关性分析。

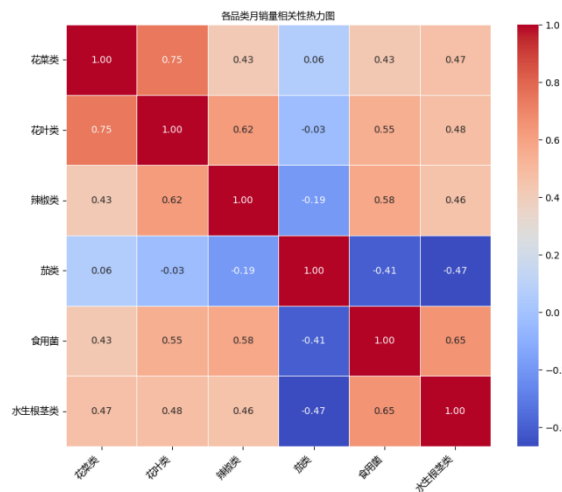


图 16: 各品类月销量相关性热力图

由热力图得知，6 种分类中，从相关关系的数量、相关系数的最大值看，正相关的情况都显著于负相关的情况。

正相关程度方面，花叶类与花菜类的正相关程度最高（相关系数为 0.75），水生根茎类与食用菌类（相关系数 0.65）、辣椒类与花叶类（相关系数 0.62）的正相关程度较高，其余的正相关数值在 0.43 到 0.58 之间，依然具有较明显的相关性；

负相关程度方面，水生根茎类与茄类商品的负相关程度最高，相关系数为-0.47，其次为食用菌与茄类商品（相关系数-0.41），其余负相关关系的组合系数的绝对值都低于 0.2，因此负相关的显著程度不及正相关商品的显著程度。

5.2 问题二的分析与求解

5.2.1 第一小问求解

5.2.1.1 Kendall 一致性检验

Kendall秩相关系数是一种非参数方法，可以度量有序变量或排名数据之间的关联性，而不考虑实际值。这使得Kendall秩相关系数成为评估大多数现实情况下的数据一致性的一种简单而有效的工具。因为在实际应用中，往往无法从数据中获得准确的数据值，而仅能得到数据的顺序或排名。

5.2.1.2 销售总量与成本加成定价的关系

要求分析各蔬菜品类的销售总量与成本加成定价的关系。利用预处理之后的数据，绘制统计图，直观上看出品类销售量与售价的关系，以水生根茎类为例：（其余图表详见附录）



图17：水生根茎类商品销售量与售价的关系

直观来看，销售量与价格并不呈现明显的线性或者有规律的非线性分析，我们使用一致性检验对二者进行进一步分析，分析结果如下：

表10：各品类商品的销售量与销售价格的一致性检验

品类名称	数据类别	秩平均值	中位数	Kendall's W 系数	X ²	P
水生根茎类	销售量	1.837	30.039	0.457	500.617	0.000***
	销售单价	1.163	8.355			
花叶类	销售量	1.995	171.919	0.991	1085	0.000***
	销售单价	1.005	5.479			
花菜类	销售量	1.926	33.858	0.735	804.756	0.000***
	销售单价	1.074	9.286			
茄类	销售量	1.821	18.262	0.431	472.015	0.000***
	销售单价	1.179	8.009			
辣椒类	销售量	1.995	72.537	0.991	1085	0.000***
	销售单价	1.005	7.556			
食用菌	销售量	1.983	57.258	0.94	1029.723	0.000***
	销售单价	1.017	10.139			

注：***、**、*分别代表 1%、5%、10%的显著性水平

5.2.2第二小问求解思路

第二小问要求给出2023年7月1-7日日补货总量和定价策略，使得商超收益率最大。第二问的模型中，品类 i ($1 \leq i \leq 6$) 第 t 天 ($1 \leq t \leq 7$) 的预期需求量和预期成本分别用 Q_{it} 和 C_{it} 表示。

5.2.2.1 ARIMA时间序列分析

ARIMA模型是一种常用的时间序列分析和预测模型，用于对数据的趋势、自回归和移动平均成分进行建模。建立ARIMA模型后，可以使用已有的历史数据进行模型的拟合和训练，然后对未来的数据进行预测。根据各品类销量随时间的趋势图，本题所使用的时间序列数据具有趋势和季节性成分，并且需要较高的解释性，适合使用ARIMA模型进行预测。

本题采用ARIMA时间序列模型，根据各品类蔬菜日补货量和批发价格数据向后预测7个单位，预测出未来七天各品类的需求量以及进货价格（成本）。使用spsspro预测，输出的部分图表如下（以辣椒类进货价格为例）：

(1) 模型参数表

表 11：模型参数表

ARIMA 模型 (0,1,2) 检验表		
项	符号	值
样本数量	Df Residuals	1091
	N	1095
	Q6(P 值)	0.018(0.895)
Q 统计量	Q12(P 值)	4.769(0.574)
	Q18(P 值)	17.61(0.128)
	Q24(P 值)	22.093(0.228)
	Q30(P 值)	39.596(0.024**)
信息准则	AIC	3294.574
	BIC	3314.564
拟合优度	R ²	0.928

注：***、**、*分别代表 1%、5%、10%的显著性水平

通过Q统计量初步判断这个模型在大部分滞后期数上残差序列上表现较好，没有明显的未捕捉到的自相关性。拟合优度达到0.928，说明模型拟合程度较好，解释能力较强。

(2) 时间序列图



图 18：时间序列图

分析图片可以看出，模型的拟合程度较好，拟合值和真实值基本相等。

(3) 时间序列预测表

表 12：时间序列预测表

预测值	
阶数（时间）	预测结果
1	3.41772903206333
2	3.351725930755436
3	3.3479429639088907
4	3.3441599970623455
5	3.3403770302158
6	3.336594063369255
7	3.33281109652271

根据表格可以得出时间序列模型最近7期数据预测情况。
分析预测每个品类销量和批发价格7月1-7日数据，汇总结果如下：

表 13：2023 年 7 月 1-7 日各品类销量预测表

预测销量(千 克)	水生根茎 类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	19.95936	131.46404	21.99442	23.93601	81.06961	42.84362
7 月 2 日	20.82457	128.34283	18.78876	22.00092	77.86205	42.77119
7 月 3 日	21.13631	136.58493	17.83956	21.98304	76.19992	42.16139
7 月 4 日	21.44137	143.62866	17.67549	21.96516	74.85626	44.67527
7 月 5 日	21.73990	143.34558	17.66854	21.94728	77.46515	45.74211
7 月 6 日	22.03203	139.04024	17.66213	21.92940	78.99609	44.46082
7 月 7 日	22.31790	136.37431	17.64139	21.91152	79.60576	43.82176

表 14： 2023 年 7 月 1-7 日各品类价格预测表

预测批发 价 (元/千克)	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	10.90289	3.28349	7.50799	4.70818	3.41773	1.52423
7 月 2 日	10.92002	3.28236	7.50287	4.73136	3.35173	1.50563

7月3日	10.92472	3.28123	7.47685	4.75362	3.34794	1.49703
7月4日	10.92942	3.28010	7.45885	4.77500	3.34416	1.49102
7月5日	10.93412	3.27897	7.44017	4.79554	3.34038	1.48567
7月6日	10.93882	3.27784	7.42329	4.81526	3.33659	1.48049
7月7日	10.94351	3.27671	7.40737	4.83420	3.33281	1.47536

5.2.2.2 成本加成定价法:

成本加成定价是一种定价策略,用于确定产品或服务的售价。这种定价方法的主要思想是将成本和所需利润相结合,从而确定最终的销售价格。具体而言,可以将每个产品或服务的成本乘以一个固定的利润率来计算利润部分,然后将成本和利润相加得到最终售价。假设利润率为 w ,成本为 c ,最终的定价 y 可以表示为:

$$y = (1 + w) * c \quad (1)$$

其中,利润率 w 和成本 c 是根据实际情况确定的参数。通过这种定价策略,企业可以确保覆盖成本并实现预期利润,同时考虑市场需求和竞争环境来决定最终的售价。

5.2.2.3 利润率的计算

为了得到定价策略,我们要求出合适的利润率。为使得商超收益最大,我们将预测出的销量带回到以往的数据中,选择本品类销量相似的历史数据中利润率最大的价格作为此品类预测日期的定价。

根据成本加成定价法,单品的利润率为:

$$\text{单品利润率} = \frac{\text{单品售价} - \text{单品批发价}}{\text{单品批发价}} \quad (2)$$

根据单品销量占品类总销量的比例,可以用加权平均法计算出品类的利润率,最终得到历史时点及销量情况下品类的利润率。假设某品类有 n 个单品,公式为:

$$\text{品类利润率} = \sum_{i=1}^n \frac{\text{单品销量}}{\text{品类销量}} \times \text{单品利润率}$$

(3)

5.2.2.4 求解未来七日的价格

假设品类*i*第*t*日的利润率为 w_{it} ，根据成本加成定价法，品类*i*第*t*日最终定价：

$$Y_{it} = (1 + w_{it}) \times C_{it} \quad (4)$$

使用Python求解得到定价策略及对应的历史最高利润率如下表：

表 15: 2023 年 7 月 1-7 日各品类定价策略预测表

成本加成						
定价法	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
(元/千克)						
7 月 1 日	19.75121	6.78968	18.11053	14.99354	7.18800	3.34216
7 月 2 日	19.78225	6.78734	18.09817	15.06736	7.04919	3.30139
7 月 3 日	19.79076	6.78501	18.03541	15.13826	7.04123	3.28253
7 月 4 日	19.79928	6.78267	17.99199	15.20634	7.03328	3.26934
7 月 5 日	19.80779	6.78033	17.94693	15.27173	7.02532	3.25761
7 月 6 日	19.81630	6.77800	17.90621	15.33453	7.01736	3.24626
7 月 7 日	19.82481	6.77566	17.86782	15.39484	7.00941	3.23501
历史最高加 权利润率	81.16%	106.78%	141.22%	218.46%	110.32%	119.27%

5.2.2.5 求解未来七日的补货总量

由于各品类蔬菜均有一定损耗，在固定利润率较高的价格情况下，当供给量满足需求量时，商超收益最大。如果补货量与需求量相同，由于损耗的影响会使得销售量小于预测的需求量，减少商超的收益。因此需要考虑运损率，增加补货量。

因此我们将附件3中各单品的运损率加权为各个品类的运损率，加权方法同5.2.2.3。设品类*i*的运损率为 L_i ，则第*i*个品类第*t*日的补货量 S_{it} 的计算公式如下：

$$S_{it} = \frac{Q_{it}}{1 - L_i}$$

通过整理各品类未来七日补货量和最终定价，可以得到未来七天的补货总量和定价策略。结果如下表：

表 16： 2023 年 7 月 1-7 日各品类日补货总量汇总表

日补货量(千 克)	水生根茎 类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	21.04530	143.97551	29.94068	25.05077	84.53557	44.69395
7 月 2 日	21.95758	140.55726	25.57686	23.02556	81.19088	44.61839
7 月 3 日	22.28628	149.58376	24.28473	23.00684	79.45769	43.98226
7 月 4 日	22.60794	157.29784	24.06138	22.98813	78.05658	46.60470
7 月 5 日	22.92271	156.98782	24.05192	22.96942	80.77701	47.71762
7 月 6 日	23.23074	152.27274	24.04319	22.95071	82.37340	46.38099
7 月 7 日	23.53216	149.35309	24.01496	22.93199	83.00913	45.71433
加权损耗率	5.16%	8.69%	26.54%	4.45%	4.10%	4.14%

表 17： 2023 年 7 月 1-7 日各品类定价策略汇总表

成本加成 定价法 (元/千克)	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	19.75121	6.78968	18.11053	14.99354	7.18800	3.34216
7 月 2 日	19.78225	6.78734	18.09817	15.06736	7.04919	3.30139
7 月 3 日	19.79076	6.78501	18.03541	15.13826	7.04123	3.28253
7 月 4 日	19.79928	6.78267	17.99199	15.20634	7.03328	3.26934
7 月 5 日	19.80779	6.78033	17.94693	15.27173	7.02532	3.25761
7 月 6 日	19.81630	6.77800	17.90621	15.33453	7.01736	3.24626
7 月 7 日	19.82481	6.77566	17.86782	15.39484	7.00941	3.23501
加权损耗率	5.16%	8.69%	26.54%	4.45%	4.10%	4.14%

5.3 问题三的求解

5.3.1 多目标规划模型

多目标规划（MOP）属于优化类模型，其中存在多个相互竞争的目标函数需要最大化或最小化。通常情况下，这些目标函数之间存在冲突和牵制关系，改善一个目标函数可能会对其他目标函数造成不利影响。

本题中商超需要考虑两个目标：单品补货量和收益最大化。使用多目标规划可以将这两个目标同时考虑进去，通过权衡不同的解来找到最优解。在给出单品补货量和定价策略时，可以通过调整权重来平衡两个目标之间的优先级，使得商超在尽量满足市场需求的前提下获得最大的收益。

5.3.2 模型建立

对于第三问，采用多目标优化模型，依题意，主要的约束条件为：

1)选择的单品总数在 27-33 个之间；

2)各单品最少进货量 2.5 千克。

目标为：

1)尽量满足市场对各品类蔬菜商品需求；

2)商超收益最大。对于约束条件和目标，分别建立相关等式。

5.3.2.1 单品总数控制

设置事件指示器 δ_i ,表示第 i 件单品的进货情况

$$\delta_i = \begin{cases} 1, & \text{此单品进货} \\ 0, & \text{此单品不进货} \end{cases} \quad (6)$$

单品数量为 n ，则约束条件①可以用以下公式表示：

$$27 \leq \sum_{i=1}^n \delta_i \leq 33 \quad (7)$$

5.3.2.2 各单品最少进货量限制

设 x_i 为销售量，约束条件②可用以下公式表示：

$$x_i \geq 2.5, i = 1, \dots, n \quad (8)$$

5.3.2.3 满足市场对各品类蔬菜商品需求

设定虚拟变量 D_{ij} ，表示第 i 个单品是否属于第 j 个品类（ $j=1,\dots,6$ ）。共有六个虚拟变量 $D_{i1}, D_{i2}, D_{i3}, D_{i4}, D_{i5}, D_{i6}$ ，满足：

$$D_{ij} = \begin{cases} 1, & \text{第 } i \text{ 个商品属于第 } j \text{ 个品类} \\ 0, & \text{第 } i \text{ 个商品不属于第 } j \text{ 个品类} \end{cases} \quad (9)$$

对于品类 j ，根据6月24-30日数据预测7月1日的需求量设为 \tilde{Q}_j ，依据题目，希望同一品类的单品销量和能够满足预测需求量：

$$\tilde{Q}_j \leq \sum_{i=1}^n D_{ij} x_i \quad (10)$$

对应的目标函数如下：

$$\min \sum_{i=1}^n \max(0, \sum_{j=1}^6 D_{ij} \cdot x_{ij} - \tilde{Q}_j) \quad (11)$$

5.3.2.4 商超收益最大

设根据6月24-30日数据预测7月1日的第 i 件单品批发价为 C_i ，第 i 件单品利润率为 ω_i ，则总收益表达式为：

$$W_i = \sum_{i=1}^n x_i C_i \omega_i \delta_i \quad (12)$$

在总收益最大的同时，需要对 x_i 进行进一步约束，要求 x_i 要在预测需求量附

近波动：

$$\tilde{Q}_j - 5 \leq x_i \leq \tilde{Q}_j + 5 \quad (13)$$

5.3.2.5 多目标规划模型建立

根据上文涉及的约束条件及目标设定，最终确定多目标规划模型如下：

$$\begin{aligned} \min & \sum_{i=1}^n \max(0, \sum_{i=1}^n D_{ij} \cdot x_{ij} - \tilde{Q}_j) \\ \max & W_i = \sum_{i=1}^n x_i C_i \omega_i \delta_i \\ \text{s. t.} & \begin{cases} 27 \leq \sum_{i=1}^n \delta_i \leq 33 \\ x_i \geq 2.5, i = 1, \dots, n \\ \tilde{Q}_j - 5 \leq x_i \leq \tilde{Q}_j + 5 \end{cases} \end{aligned}$$

5.3.3 模型优化

将单品数 $n=49$ 带入模型，本题中目标一的优先因子设定为 5，目标二的有限因子设定为 10。用 python 相关代码求解时，为了加快模型运算速度，对代码进行优化，加入粒子群算法。

5.3.3.1 粒子群算法

粒子群算法（PSO）粒子群优化算法（Particle Swarm Optimization, PSO）是一种启发式优化算法，模拟了鸟群或鱼群等群体行为的优化过程。该算法通过迭代更新一组称为粒子（particle）的解决方案，以寻找最优解。

粒子的位置和距离公式。

$$z_{ij} = z_{ij} + d_{1ij} \times rand() \times (pbest_{ij} - x_{ij}) + d_{2ij} \times rand() \times (pbest_{ij} - x_{ij}) \quad (14)$$

$$x_{ij} = x_{ij} + z_{ij} \quad (15)$$

针对本题目进行改进，其中， $\text{rand}()$ 为 0 到 1 之间随机数。

$$z_{ij} = \omega \times z_{ij} + d_{1ij} \times \text{rand}() \times (pbest_{ij} - x_{ij}) + d_{2ij} \times \text{rand}() \times (pbest_{ij} - x_{ij}) \quad (16)$$

ω 为惯性因子，其值非负。数值越大，则寻找最优解的能力越强，局部寻找最优解的能力越弱，反之结论正好相反。我们这里采用线性递减权值策略。

$$\omega^0 = (\omega_{ini} - \omega_{end})(G_k - g) / G_k + \omega_{end} \quad (17)$$

G_k 为最大迭代次数， ω_{ini} 为初始惯性权值， ω_{end} 迭代到最大进化数时的惯性权值。

根据蚁群算法能够较快速得到最终的单品补货量和定价策略。

5.3.4 模型结果

我们选取了预处理数据中 6 月 24-30 日单品的销售数据，共 49 个单品，最终得到的 7 月 1 日的单品补货量和定价策略表格如下：

表 18：7 月 1 日的单品补货量和定价策略表

分类	单品名称	销量	定价	利润
花叶类	红薯尖	0.520874474	3.692789134	1.923479598
花叶类	云南生菜	1.834026437	6.921442648	12.6941088
花叶类	菜心	1.453610684	5.237813444	7.613741581
花叶类	上海青	2.937377258	5.126698948	15.0590489
茄类	紫茄子(2)	2.099	3.954480606	8.300454791
食用菌	白玉菇(袋)	2.102301169	5.672467611	11.92523529
花菜类	西兰花	4.27	8.205066582	35.0356343
水生根茎类	净藕(1)	1.065	11.32193291	12.05785855
花叶类	云南生菜	0.343325621	6.921442648	2.376308594
食用菌	白玉菇(袋)	0.237280042	5.672467611	1.345963351
辣椒类	芜湖青椒(1)	6.055914992	3.514065571	21.28088237

食用菌	西峡花菇(1)	0.373834706	17.41655287	6.510911916
花叶类	木耳菜(份)	2.096398811	3.428536012	7.18757882
食用菌	双孢菇(盒)	1.186400208	3.566680577	4.231510579
辣椒类	青红杭椒组合装(份)	1.12484026	4.257431817	4.788930713
辣椒类	红椒(2)	0.62136176	15.84117433	9.843099956
食用菌	蟹味菇与白玉菇双拼(盒)	0.118640021	3.937879322	0.467190085
水生根茎类	红莲藕带	2.317	11.09284241	25.70211587
花叶类	小青菜(1)	0.857427114	3.317769342	2.844745391
花叶类	云南生菜(份)	2.096398811	3.622940151	7.595127426
花叶类	云南油麦菜(份)	1.451353023	2.922154103	4.241077192
花叶类	菠菜(份)	2.418921705	4.232869578	10.2389801
花叶类	菠菜	0.161261447	11.5579018	1.863843969
辣椒类	小米椒(份)	4.274392988	2.284919315	9.766643098
花叶类	外地茼蒿	2.096398811	10.54002561	22.09609716
食用菌	金针菇(盒)	2.372800416	1.477601297	3.506052972
花叶类	云南油麦菜	1.25703298	6.092676316	7.658695063
茄类	圆茄子(2)	2.268	6.100064905	13.8349472
花叶类	奶白菜	1.043522824	2.833829468	2.957165728

5.4 问题四的分析与求解

全面的数据采集和分析将为商超提供更全面、准确的信息基础，以支持合理有效的蔬菜商品补货和定价决策，提高运营效率和利润水平，对于商家的持续性盈利和满足消费者需求都有重大意义。除了往年曾销售过的商品交易记录、单品损耗率和批发价以外，获取以下信息，将更加有利于进行高收益的蔬菜商品补货和定价决策：

1、供给角度：

（1）供应链数据：了解蔬菜供应链的生产、加工、运输等环节数据，可以帮助商超预测供应稳定性，优化补货计划，并确保有足够的库存满足市场需求。

（2）成本数据：分析蔬菜生产、采购和运输等环节的成本数据，可以帮助商超评估不同蔬菜品类的毛利润水平，制定合理的定价策略，确保盈利最大化。

（3）生鲜蔬菜的易腐烂程度：建立在随机需求下单产品以及相互之间有替代关系的多种产品多个周期的定价模型，有利于解决随机环境下易腐烂的生鲜商品需求受库存以及价格影响的价格决策问题。

（4）季节性因素数据：了解蔬菜季节性供应的规律和特点，可以帮助商超合理安排补货计划，避免因季节性变化造成的库存积压或供应不足。

2、需求角度：

（1）客户需求数据：通过调查消费者的消费偏好动向，对商品的需求价格弹性进行分析，了解消费者对不同蔬菜品类的偏好和购买习惯，可以帮助商超精确定位目标客户群体，调整商品组合，提供个性化的补货和定价策略，从而增加销售额和顾客满意度。

（2）顾客反馈数据：收集和分析顾客对蔬菜品质、新品需求、购物体验等方面的反馈数据，可以帮助商超改进商品选择、品质管理和服务质量，提高顾客满意度和忠诚度。

3、经营角度：

（1）市场竞争数据：了解竞争对手的蔬菜品类销售情况和定价策略，可以帮助商超进行市场定位和差异化竞争，制定相应的补货和定价策略，增加市场份额和竞争优势。

（2）促销活动数据：分析过去的促销活动数据，包括折扣、套餐等优惠形式以及其对销售量的影响，可以帮助商超优化促销策略，吸引更多顾客并提高销售额。

（3）智能 POS 数据：利用智能 POS 系统收集的数据，如销售额、销售数量、交易频次等，可以帮助商超实时监测蔬菜销售情况，及时调整补货和定价策略，提高效率和准确性。

六、 模型评价

1、数据预处理：利用 spss、stata 等软件进行了数据预处理，将折后价大于售价的异常值进行剔除处理，并对缺失值进行检验，保证了数据的准确性和可用性，提高后续分析和建模的可靠性。

2、数据分析全面：论文对蔬菜各品类及单品销售量的分布规律和相互关系进行了充分的数据分析。通过观察和统计销售数据，深入了解不同品类和单品的销售情况，并揭示了可能的关联关系。为后续问题的解决提供了良好的基础。

3、模型选取恰当：合理选用时间序列模型、多目标规划模型、蚁群算法等多个模型，模型选取较为贴合主题，可以提高分析结果的解释能力。

4、对于问题二和问题三，综合选用模型，并做恰当的补充变换，与题目要求具有高度的契合性，展现出灵活性和准确性。

模型的不足：

1、模型需要特定的数据，数据处理过程较为复杂，需要大量的时间和精力。

2、第二小问的时间序列模型选用了 ARIMA 模型，由于时间问题没有再用其它模型（如 LSTM 模型）进行模型对比并选出最优模型。

3、模型内部分权重设置需要进一步的验证。

七、 参考文献

[1] 范凤岩. 超市生鲜商品定价模型研究 [D]. 福建师范大学,2020.DOI:10.27019/d.cnki.gfjsu.2019.000850.

[2] arima: 王燕. 应用时间序列分析[M]. 北京: 中国人民大学出版社 2005.

[3] 吉闫军. 基于外部种群自适应的多目标粒子群优化算法及其应用[D]. 陕西师范大学,2020.


[4] 李娟生,李江红,刘小宁等.Kendall's W 分析方法在医学数据处理中的应用及在 SPSS 中的实现方法[J].现代预防医学,2008(01):33+42.

八、 附录

附录一 支撑材料：

表:

 表1和表2

 问题1 统计销量-第1步

 问题1 统计销量-第2步

 问题二_按品类_日加权利润率

 问题二_按品类_日加权售价_日销量


 问题二_按品类_筛选销售区间_寻找历史最优利润率


 问题二_按品类_销量_日加权利润率


 问题三_多目标优化模型导入数据


 问题三_上周有效单品_周加权利润率


代码:

 P-P图正态检验 商品单品

 Q-Q图正态检验 商品单品

 问题1 描述性统计

 问题2 数据处理(1).ipynb

 问题2 数据处理(2).ipynb



 多目标规划模型

图:

 图17-1: 水生根茎类商品销量与售价的关系




 图17-2: 花叶类商品销量与售价的关系

 图17-3: 花菜类商品销量与售价的关系

 图17-4: 茄类商品销量与售价的关系

 图17-5: 辣椒类商品销量与售价的关系

 图17-6: 食用菌商品销量与售价的关系

附录二 数据预处理 (Python)

#预处理第一步 把附件1和附件2的品类编码、品类名称进行匹配

```
import openpyxl  
wb1 = openpyxl.load_workbook("附件1.xlsx")  
wb2 = openpyxl.load_workbook("附件2.xlsx")  
sheet1 = wb1.active  
sheet2 = wb2.active
```

```

mapping = {}
for row in sheet1.iter_rows(values_only=True):
    item_code = row[0]
    category_code = row[1]
    category_name = row[2]
    mapping[item_code] = (category_code, category_name)
for row in sheet2.iter_rows(min_row=2):
    item_code = row[0].value
    if item_code in mapping:
        category_code, category_name = mapping[item_code]
        sheet2.cell(row=row[0].row, column=3, value=category_code)
        sheet2.cell(row=row[0].row, column=4, value=category_name)
wb2.save("附件 2+分类编码及名称.xlsx")

```

#预处理第 2 步, 按照销售日期、单品编码、销售单价、销售类型、是否打折销售进行分类, 并统计销量

```

import pandas as pd
df = pd.read_excel("附件 2+分类编码及名称.xlsx")
grouped = df.groupby([df.columns[0], df.columns[2], df.columns[4],
df.columns[5], df.columns[6], df.columns[7], df.columns[8]])
summed = grouped[df.columns[3]].sum().reset_index()
summed.columns = ['销售日期', '单品编码', '销售单价', '销售类型', '是否打折销售', '分类编码', '分类名称', '销量 (千克)']
new_file = '附件 2+分类编码及名称_按销售类型和是否打折.xlsx'
summed.to_excel(new_file, index=False)

```

#预处理第 3 步, 查找并剔除折后价大于等于折前售价的数据条

```

import pandas as pd
df = pd.read_excel('附件 2+分类编码 名称_按销售类型和是否打折.xlsx')
for i, row in df.iterrows():
    date = row['销售日期']
    code = row['单品编码']
    sales_type = row['销售类型']
    category_code = row['分类编码']
    category_name = row['分类名称']

    filtered_rows = df[(df['销售日期'] == date) & (df['单品编码'] == code)
& (df['销售类型'] == sales_type) & (df['分类编码'] == category_code) &
(df['分类名称'] == category_name)]

    if len(filtered_rows) > 1:
        discount_price = filtered_rows[filtered_rows['是否打折销售'] == '是']['销售单价']
        no_discount_price = filtered_rows[filtered_rows['是否打折销售']]

```

```

== '否'] ['销售单价']
    if len(discount_price) > 0 and len(no_discount_price) > 0 and
discount_price.iloc[0] >= no_discount_price.iloc[0]:
        print(filtered_rows.drop_duplicates())
#查找折后价大于等于折前售价的数据条, 由于数据量较小, 根据打印结果, 可通过人为删
除, 储存为附件 2-分类编码 名称_按销售类型和是否打折_去除不合理折扣价.xlsx, 出于简
化代码的目的, 更名为'preprocess-1.xlsx'

```

附录三 问题一代码 (Python)

#第一步, 进一步预处理, 在统计单品销量之前, 先去除销售天数太少的商品
#首先, 使用 excel 统计出每个单品的销售天数, 将天数不超过 366 的标记为 0, 多于 366 天的标记为 1 然后, 用 vlookup 写进表单, 利用 python 代码删除掉标记为 0 的单品

```

import pandas as pd
source_file = 'preprocess1.xlsx'
df = pd.read_excel(source_file, sheet_name='区分折扣')
new_df = pd.DataFrame(columns=df.columns)
for _, row in df.iterrows():
    if row['临时标记'] == 0:
        print(row.tolist())
    else:
        new_df = new_df._append(row)
df = df[df['临时标记'] != 0]
df.to_excel(source_file, index=False, sheet_name='区分折扣',
engine='openpyxl')
new_file = '附件 2-去除不合理价格-去除不合理销售天数.xlsx'
new_df.to_excel(new_file, index=False, sheet_name='去除不合理价格-去除不
合理销售天数', engine='openpyxl')

```

#第二步, 销售类型为“退款”的销量数值为负数, 合并同样单品的销售和退款记录, 然后继续用 python, 统计以年-月-日和商品单品为分类的单品日销量

```

df = pd.read_excel('附件 2-去除不合理价格-去除不合理销售天数.xlsx')
grouped_df = df.groupby(['销售日期', '单品编码', '单品名称'])['销量 (千克)']
.sum().reset_index()
#检查无误并保存后, 再将表格转化为以日期为纵向指标、单品为横向指标的矩阵
grouped_df.to_excel('question1.xlsx', index=False)
source_file = '附件 2-去除不合理价格-去除不合理销售天数.xlsx'
df = pd.read_excel(source_file, sheet_name='合并退款')
item_names = df['单品名称'].unique()
new_df = pd.DataFrame(columns=['销售日期'] + list(item_names))
for _, row in df.iterrows():
    sales_date = row['销售日期']
    item_name = row['单品名称']
    sales_quantity = row['销量 (千克)']
    # 检查新数据框中是否已经有该日期的行

```

```

    if sales_date in new_df['销售日期'].values:
        # 更新对应单品的销售数量
        new_df.loc[new_df['销售日期'] == sales_date, item_name] =
sales_quantity
    else:
        # 创建新的行, 并填充对应单品的销售数量
        new_row = {'销售日期': sales_date, item_name: sales_quantity}
        new_df = new_df._append(new_row, ignore_index=True)
# 保存新 Excel 文件
new_file = 'question1 销量-按日-按单品.xlsx'
new_df.to_excel(new_file, index=False)

#第三步, 统计以年-月和商品单品为分类的单品日销量
#先手动把“合并退款-按日”里的销售日期换成 yyyy 年-mm 月形式, 储存在'question1-
month.xlsx'
# 读取源 Excel 文件
source_file = 'question1-month.xlsx'
df = pd.read_excel(source_file)
item_names = df['单品名称'].unique()
new_df = pd.DataFrame(columns=['销售日期'] + list(item_names))
for _, row in df.iterrows():
    sales_date = row['销售日期']
    item_name = row['单品名称']
    sales_quantity = row['销量 (千克) ']

# 检查新数据框中是否已经有该日期的行
if sales_date in new_df['销售日期'].values:
    # 更新对应单品的销售数量
    new_df.loc[new_df['销售日期'] == sales_date, item_name] =
sales_quantity
else:
    # 创建新的行, 并填充对应单品的销售数量
    new_row = {'销售日期': sales_date, item_name: sales_quantity}
    new_df = new_df._append(new_row, ignore_index=True)
new_file = 'question1 销量-按月-按单品.xlsx'
new_df.to_excel(new_file, index=False)

#第四步, 以日为单位, 按品类, 统计销量
import pandas as pd
df = pd.read_excel('preprocess-1.xlsx.xlsx', sheet_name='区分折扣')
grouped_df = df.groupby(['销售日期', '分类编码', '分类名称', '销售类型
'])['销量 (千克) '].sum().reset_index()
grouped_df.to_excel('question1 销量-按日-按品类.xlsx', sheet_name='按日统
计', index=False)

```



```

#第五步, 将以月为单位, 按品类统计销量, 先使用 excel 格式, 将 output_question1-
day.xlsx 中的销售日期手动改为“yyyy 年-mm 月”形式
df = pd.read_excel('question1 销量-按日-按品类.xlsx ', sheet_name='按日统
计-y 年 m 月')
grouped_df = df.groupby(['销售日期', '分类编码', '分类名称', '销售类型
'])['销量 (千克)'].sum().reset_index()
with pd.ExcelWriter('output_question1-month.xlsx') as writer:
    grouped_df.to_excel(writer, sheet_name='按月统计', index=False)
#在按月统计的基础上, 把销售和退货的销量加总
df = pd.read_excel('问题 1 销量统计 按月-按品类.xlsx', sheet_name='按月-按
品类-区分销售退货')
grouped_df = df.groupby(['销售日期', '分类编码', '分类名称'])['销量 (千克)
'].sum().reset_index()
with pd.ExcelWriter('output_question1-month(2).xlsx',
engine='xlsxwriter') as writer:
    grouped_df.to_excel(writer, sheet_name='按月统计', index=False)
#output_question1-month(2).xlsx 已保存至'问题 1 销量统计 按月-按品
类.xlsx', sheet_name='按月-按品类-不区分销售退货'
#把按月-按品类-不区分销售退货 sheet 转化成矩阵
import pandas as pd
from openpyxl import Workbook
df = pd.read_excel('问题 1 销量统计 按月-按品类.xlsx', sheet_name='按月-按品
类-不区分销售退货')
dates = df['销售日期'].unique()
categories = df['分类名称'].unique()
wb = Workbook()
ws = wb.active
ws['A1'] = '销售日期'
for i, date in enumerate(dates):
    ws.cell(row=i+2, column=1, value=date)
ws['B1'] = '分类名称'
for i, category in enumerate(categories):
    ws.cell(row=1, column=i+2, value=category)
for _, row in df.iterrows():
    date = row['销售日期']
    category = row['分类名称']
    sales = row['销量 (千克)']
    date_index = dates.tolist().index(date) + 2
    category_index = categories.tolist().index(category) + 2
    ws.cell(row=date_index, column=category_index, value=sales)
wb.save('output_question1-month(3).xlsx')
#output_question1-month(3)已保存至'问题 1 销量统计 按月-按品
类.xlsx', sheet_name='按月-按品类-不区分销售退货-矩阵形式'

```

#第六步, 绘制各品类月销量相关性热力图

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
df = pd.read_excel('data.xlsx', header=0)
axis_labels = df.columns.tolist()[1:7]
data_subset = df.iloc[:, 1:7]
correlation_matrix = data_subset.corr()
# 设置支持汉字的字体
font = FontProperties(fname=r"C:\Windows\Fonts\msyh.ttc", size=10)
# 绘制热力图并设置汉字标签
plt.figure(figsize=(10, 8))
ax = sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f", linewidths=0.5)
ax.set_xticklabels(axis_labels, rotation=45,
horizontalalignment='right', fontproperties=font)
ax.set_yticklabels(axis_labels, rotation=0,
horizontalalignment='right', fontproperties=font)

plt.title('各品类月销量相关性热力图', fontproperties=font)

plt.show()
```

附录四 问题二代码

根据同一日期、品类和单品进行分组, 然后对销售量进行求和

```
import pandas as pd

# 读取Excel 文件
df = pd.read_excel('python 操作数据.xlsx')

# 使用groupby 函数按日期、品类和单品进行分组, 然后对销售量进行求和
result = df.groupby(['销售日期', '分类名称', '单品编码'])['销量'
].sum().reset_index()

# 将结果保存到新的Excel 文件
result.to_excel('summarized_sales_data.xlsx', index=False)

# 根据同一日期、品类进行分组, 然后对销售量进行求和

import pandas as pd
```

```

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

# 使用groupby 函数按日期、品类进行分组，然后对销售量进行求和
result = df.groupby(['销售日期', '分类名称'])['销量'].sum().reset_index()

# 将结果保存到新的Excel 文件
result.to_excel('summarized_sales_data2.xlsx', index=False)

```

附录五 问题三代码 加蚁群算法（R）

```

library(lpSolve)
library(readxl)

# 读取数据
data <- read_excel("data.xlsx")

# 添加虚拟变量 Dij
categories <- unique(data$分类名称)
for (j in categories) {
  data[paste0("D_", j)] <- ifelse(data$分类名称 == j, 1, 0)
}

# 提取必要的变量
D <- data[, grep("^D_", colnames(data))]
Q <- data$品类销量
C <- data$周加权利润率
w <- data$上一周平均批发价
p <- ifelse(data$品类销量 > 0, 1, 0)

# 定义目标函数 1:  $\min \sum_{i=1,n} \max(0, \sum_{i=1,n} D_{ij} * x_{ij} - Q_j)$ 
profit <- function(x) {
  result <- numeric(length(Q))
  for (j in 1:length(Q)) {
    result[j] <- max(0, sum(D[, j] * x) - Q[j])
  }
  return(sum(result))
}

# 定义目标函数 2:  $\max \sum_{i=1,n} x_i * C_i * w_i * p_i$ 
weight_profit <- function(x) {
  return(sum(x * C * w * p))
}

# 定义线性规划模型

```

```

n <- ncol(D)
model <- lp(direction = "min", objective.in = rep(0, n), const.mat = D, const.dir = ">=",
const.rhs = rep(0, length(Q)),
            all.int = TRUE)

# 添加约束
for (i in 1:n) {
  const_mat <- as.matrix(D[i, , drop = FALSE])
  model <- lp("max", model, const.mat = const_mat, const.dir = rep(">=", 3), const.rhs =
c(Q[i] - 5))
  model <- lp("min", model, const.mat = const_mat, const.dir = rep("<=", 2), const.rhs =
c(Q[i] + 5))
}

# 设置目标函数
model$objfn <- rep(0, n)

# 求解线性规划问题
solution <- solve(model)

# 输出结果
cat("Optimal Solution:\n")
print(get.variables(model))

cat("Objective Value 1 (min): ", get.objective(model), "\n")

x <- get.variables(model)
cat("Objective Value 2 (max): ", weight_profit(x), "\n")

```