基于 ARIMA 模型与优化模型的蔬菜类商品的自动定价与补货决策

摘要

在生鲜商超中，由于蔬菜类商品保鲜期短，经营者往往需要采取合理的定价、补货和库存管理策略，以确保收益最大化。本文通过对蔬菜各品类及单品的大数据分析，建立了蔬菜商品各指标的 ARIMA 时间序列预测模型，使用最优化模型解决了蔬菜商品的自动定价与补货决策问题。

针对问题一：本文首先对数十万条数据进行预处理，通过对时序图的分析、对**主成分分析**提取因子得到的周期性的分析，得出结论：蔬菜销售有显著的 30 天周期性规律，因此对数据进行按月压缩处理。对于所有单品，通过 **K 均值聚类**，得到代表全体单品的 18 个聚类中心，分析它们的均值、方差、峰度、偏度、极差，结合堆积面积图等数据可视化，得出单品的分布规律结论；由于其不通过正态分布检验，我们选择斯皮尔曼**相关系数**，结合热力图等数据可视化，较好地得出单品间的相互关系。对于所有品类，分析它们的均值、方差、峰度、偏度、极差，结合柱状图等数据可视化，得出品类的分布规律结论；由于其通过了正态分布检验，我们选择皮尔逊相关系数，结合热力图等数据可视化，详细地得出品类间的相互关系。

针对问题二：本文首先剔除了退货数据，通过累加与加权平均得到不同品类的销售总量和定价，使用 Spss 对二者进行回归性分析，得到不同品类的销售总量和定价的**线性回归方程**。为了确定各品类未来一周的日补货量，我们分析出了：计划补货量 = 销售量 / （1 - 损耗率）。其中损耗率取均值，并且将销售量使用一阶差分将其转化为平稳非白噪声序列，对其进行 **ARIMA 时间序列预测**，得到各品类未来七天的销售量。进而计算出了各品类未来一周的日补货总量。为了确定各品类未来一周的定价策略，我们选择 ARIMA 时间序列预测来得到未来七天的批发价，建立了利润与定价的**优化模型**。最后使用 ARIMA 时间序列预测来得到未来七天定价的置信区间，结合利润曲线，得到最优定价策略。

针对问题三：本文首先根据题目要求，建立了目标函数，考虑了所有约束条件。我们把该问题看作六个**完全背包问题**，通过 Python 程序进行动态规划求解，成功得出 27 种选中单品、对应日补货总量、对应最大收益。我们根据成本加成定价公式，用利润率、最大收益、批发价计算得到 27 种选中单品的定价策略。

针对问题四：本文对可能会影响蔬菜商品补货和定价决策测的因素进行了举例和分析，分别从供给侧和需求侧对影响因素进行分析，最终提出建议：应该更加关注当地的气候和季节数据、更详细的损耗率数据、了解当地居民的饮食习惯，和更详细的市场调研数据。

关键字：主成分分析、相关性分析、k-Means 聚类、线性回归、ARIMA 时间序列模型、背包问题

一、 问题重述

1.1 问题背景

三月桃花芳菲，四月杏花梅雨。南方气候潮湿，蔬菜水果更是容易受潮发霉。在生鲜商超中更是如此，蔬菜的保质期短，基本隔日就无法再次出售。商家通常会根据每种蔬菜的供需情况进行补货，以防蔬菜滞销而导致亏本。这就要求商家具有可靠的市场需求分析能力，对供给端和需求端都需要制定一套合理的购买和销售方案，使得利润最大化。

1.2 问题提出

为了协助商超能够以更合理的方案进货和定价，本文将用数学建模解决以下问题：

问题一：

蔬菜分为不同的品类，每个品类下分为不同的单品。不同品类之间、不同单品之间的销售量可能存在一定联系。我们将分析蔬菜各品类以及单品的销售量的分布规律和他们之间的相互关系。

问题二：

假设商超以蔬菜的品类为单位做补货计划，我们将分析各种蔬菜的销售总量与成本加成定价的关系，并且预测各蔬菜品类在未来一周(2023 年 7 月 1-7 日)的日补货总量和定价策略，以实现商超的最大收益。

问题三：

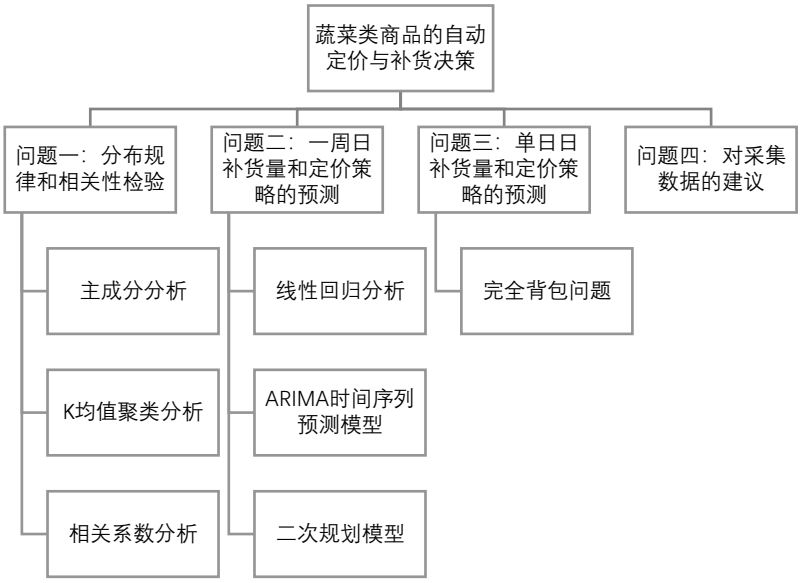
由于蔬菜类产品的销售空间有限，商超将计划进一步制定单品的补货计划。要求可售单品总数控制在 27-33 个，并且各单品的采购量满足最小陈列量 2.5kg 的要求。根据 2023 年 6 月 24-30 日的可售品种，我们需要给出 7 月 1 日的单品补货量和定价策略，在满足市场对各品类蔬菜的供需关系下，尽量使得商超的收益最大。

问题四：

为了帮助商超更好地制定蔬菜商品的补货和定价计划，试分析商超还需要采集哪方面的数据来更好地解决上述的问题。

二、 问题分析

文章框架图：



2.1 问题一的分析

问题一要求我们分别分析蔬菜各品类和单品销售量的分布规律、各品类销售量之间的相互关系以及单品销售量之间的相互关系。我们通过对数据预处理，将数十万条数据处理完毕后，使用 K 均值聚类模型，将所有单品替换成具有代表性的聚类中心，并通过统计学指标分析所有聚类中心的分布规律；随后对各品类也进行了规律性分析，得出最终的销售量分布规律。相关性方面我们对聚类中心进行了正态分布检验和斯皮尔曼相关分析，对各品类进行了正态分布检验以及皮尔逊相关分析，最终得出了相关性结论。

2.2 问题二的分析

问题二以品类为单位进行补货，要求我们分析六种蔬菜品类的销售总量与成本加成定价之间的关系，并且要求我们预测出未来一周的日补货量和利润最高的定价策略。我们删除了退货数据、对定价进行了加权平均处理后得到了销售总量与成本加成定价之间的关系，并对其进行了线性拟合得到了它们的线性关系。为了求得未来一周的补货计划和定价策略，我们将销售量使用一阶差分将其转化为平稳非白噪声序列，对其进行 ARIMA 时间序列预测得到未来七日的销售量预测，并同理可得批发价的预测结果、预测定价的置信区间，进而建立利润与定价的最优化模型。

2.3 问题三的分析

问题三要求我们根据 6 月 24-30 日的可售单品品种，给出 7 月 1 日的单品补货量和定价策略，并且使得商超收益最大。题目给出了约束条件：可售单品总数需要控制在 27-33 个，并且订货量需要满足最小陈列量 2.5 千克。在此约束条件之下，我们将问题转换为六个完全背包问题进行求解，并且运用 Python 软件进行动态规划求解。根据成本加成定价公式，最终获得了最合理的单品补货量和定价策略。

2.4 问题四的分析

问题四要求我们分析商超为了提高利润，还应该采集哪些相关数据，并且给出建议。我们分别从供给侧和需求侧分析可能会影响商超销售量的原因，并且提出了应该如何制定更加合理的定价决策的建议，有助于商家更好地盈利。

三、 模型假设

（一）假设在处理完异常值过后附件所给的数据真实有效；

（二）假设附件提供的损耗率能真实准确地反映近期商品的损耗；

（三）假设商品的利润仅考虑批发价、定价，不考虑税率；

（四）假设所有的蔬菜如当日未售出，隔日就无法再售，不能作为积货放到第二天售卖，每天没卖出的货算作利润损失；

四、 符号说明

符号	说明
ρ_s	斯皮尔曼相关系数
ρ_p	皮尔逊相关系数
f_{ij}	品类 i 的第 j 个单品的权重

符号	说明
$\bar{x}_i$	品类 i 的加权平均定价
AIC	最小化信息量准则
T_i	品类 i 的总利润
Q_i	品类 i 的销售量
k_i	品类 i 的销售量与定价关系的常数
b_i	品类 i 的销售量与定价关系的系数

五、 模型的建立与求解

5.1 问题一模型的建立和求解

5.1.1 数据预处理

问题一要求我们根据附件数据分析蔬菜各品类及单品销售量的分布规律及相互关系。在此之前，我们需要对数据进行清洗和筛选，剔除重复值或异常值，防止其对最终分析结果产生误差。

对于附件二，我们使用 Python 对 878503 条销售流水明细进行了逐一查重，确保了其中没有重复的销售记录；同时通过 Python 的正则表达式：

```
Pattern = r'^([0-1][0-9]|2[0-3]):([0-5][0-9]):([0-5][0-9](\.\d{1,3})?)$'
```

来检查时间数据是否符合合理区间（即小时部分是 00 到 23 之间的任意数字、分钟秒钟毫秒部分是 00 到 59 之间的任意数字、有可选的 1 到 3 位小数的毫秒部分），最终我们确认销售时间没有异常值；最后，我们发现存在 461 条销量为负数的记录，同时发现 461 条销售类型为退货的记录，而它们正好一一对应，符合常理，不影响对本问题的分析。至此，我们完成了对附件二数据的检查处理，并未出现重复值与异常值。

附件二的原始数据为 878503 条，如此庞大的数据量不便于我们进行分析。考虑到我们的目标是分析蔬菜销售量的分布规律及相互关系，根据生活常识与题目背景，我们假设蔬菜销售量与月份（季节）之间的关系通常会比与具体日期和时间段（早、中、晚）之间的关系更为显著。为此，我们使用 python 画出了六种不同蔬菜品类的时序图：

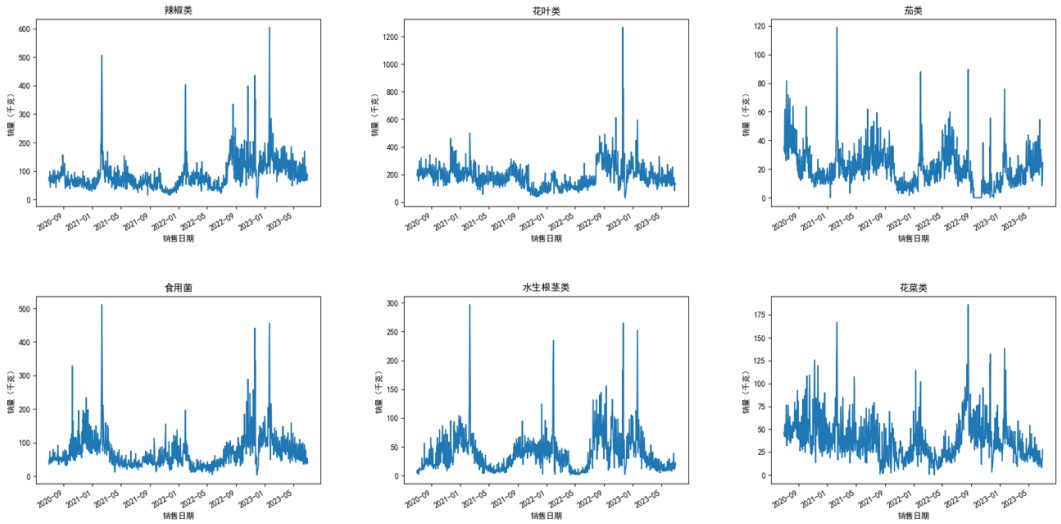


图 1 六种蔬菜品类销售量的时序图

由图，我们猜测蔬菜产品的销售可能具有周期性规律，因此我们对蔬菜单品进行主成分分析。

根据文献[1]，主成分分析（PCA）是一种常用的降维技术，用于将高维数据转换为低维空间。其主要思想是通过线性变换将原始数据映射到新的坐标系，使得新的特征尽可能多地保留原始数据的信息。通过主成分分析法，我们可以将所有 251 种蔬菜单品提取成数量较少但具有代表性的主要因子，以便后续对其进行周期性的分析。

我们用 Spss 软件对蔬菜单品进行主成分分析，提取出 18 个主要因子，累积百分比达到 84.315%，因此我们只需要对这 18 个主要因子检验其周期性。我们先通过傅里叶变换[3]估计各主要因子可能的周期，再结合预设周期 15 天、30 天及 60 天计算自相关系数，最后得到显著分数最大值所对应周期即主要因子的周期。

表 1 对 18 个主要因子的主成分周期性分析（部分）

fft_periods	[9 18 37]		fft_periods	[7 12 9]	
lag	9 fft acf	-0.03259	lag	7 fft acf	-0.04179
lag	18 fft acf	-0.03587	lag	12 fft acf	-0.05409
lag	37 fft acf	0.000779	lag	9 fft acf	-0.00974
lag	15 expected acf	-0.02848	lag	15 expected acf	-0.02738
lag	30 expected acf	0.007503	lag	30 expected acf	0.003219
lag	60 expected acf	0.000779	lag	60 expected acf	-0.00035

综合各主要因子的周期，认为以 30 天为周期具有显著性，因此把时间压缩为以月作单位是合理的。

随后，我们将同种蔬菜单品的销售量按月份进行了聚合。处理后，每个蔬菜单品有 36 列数据，代表从 20 年 7 月 1 日至 23 年 6 月 30 日该种蔬菜单品在该月的销售总量，如表 1 所示：

表 2 蔬菜单品每月的销售总量（部分）

单品编码	单品名称	2020/7	2020/8	2020/9	…	2023/5	2023/6
102900005115762	苋菜	143.737	236.18	139.108	…	270.735	236.008
102900005115779	云南生菜	958.044	905.914	856.958	…	22.194	45.721
102900005115786	竹叶菜	483.748	381.047	262.864	…	404.436	433.414
102900005115793	小白菜	185.146	198.076	149.285	…	0.000	0.000
102900005115816	南瓜尖	4.875	0.000	0.000	…	0.000	0.000
…					…		…
102900005115823	上海青	452.999	702.708	529.028	…	215.385	166.023

5.1.2 品类或单品分布规律分析

（一）对单品的分布规律分析

根据附件一，蔬菜单品共有 251 种，其中附件二涉及 246 种。对如此大的数据量进行分析，需要处理大量的计算和统计操作；单品分类过多，全部统计可能会引入冗余信息和噪声，使分析结果更加复杂；尝试可视化 246 种单品的分布规律也会十分困难。

因此我们选择聚类分析，从 246 种单品中聚类出具有代表性的单品，进行统计分析。这样不仅可以减少数据量、降低处理难度，还可以更容易地可视化数据、推导分

布规律。

由文献[2]可知，K 均值聚类（K-means clustering）是一种常用的无监督学习算法，用于将一组数据点划分为不同的簇（clusters）。该算法的目标是将数据点划分到 k 个不同的簇中，使得每个数据点与所属簇的中心点（质心）之间的距离最小化。

具体步骤为：

Step1. 选取 K 个点作为初始聚集的簇心；

Step2. 分别计算每个样本点到 K 个簇核心的距离（一般取欧氏距离或余弦距离），找到离该点最近的簇核心，将它归属到对应的簇；

Step3. 所有点都归属到簇之后，重新计算每个聚类的中心对象（均值）；

Step4. 反复迭代 2-3 步骤，直到每个聚类不再发生变化为止

肘部法（Elbow Method）是一种用于确定聚类分析中最佳簇数的常用方法。在聚类分析中，肘部法通过计算不同簇数下的聚类模型的误差平方和，帮助确定最合适的簇数。我们使用肘部法来寻找合适的 K 值：

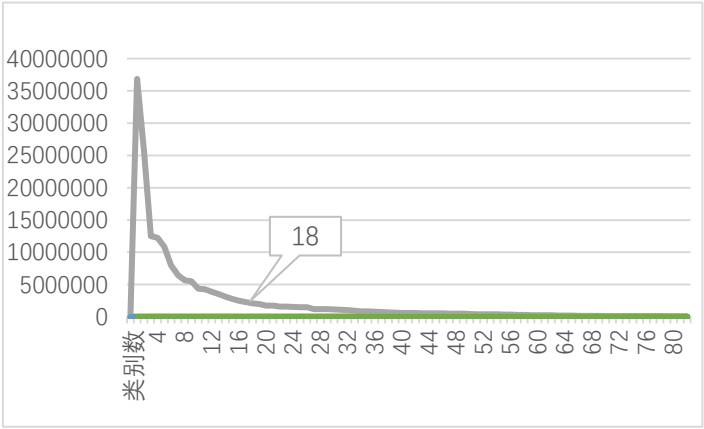


图 2 历年单品销量系统聚类肘部图

由肘部图可知，聚类系数曲线下降速度从类别数增加到 18 后开始放缓，因此确定聚类数目为 18。随后用 Spss 进行 K-means 聚类分析，最终得到 18 个最终聚类中心，对应的每月销售数据如下：

表 3 18 个聚类中心对应的每月销售数据（部分）

类别	2020/7	2020/8	2020/9	…	2023/4	2023/5	2023/6
1	14.30	30.73	63.85		39.80	15.57	12.62
2	638.60	672.67	424.17		43.61	53.85	44.54
3	0.00	0.00	13.67		693.80	691.31	509.70
…							…
16	1081.91	1825.90	1251.85		0.00	0.00	0.00
17	831.58	900.22	802.84		441.25	466.39	430.48
18	505.87	558.96	600.05	…	0.00	0.00	0.00

我们还构造了 18 类代表单品的销售量堆积面积图，使得数据的分布更加直观：

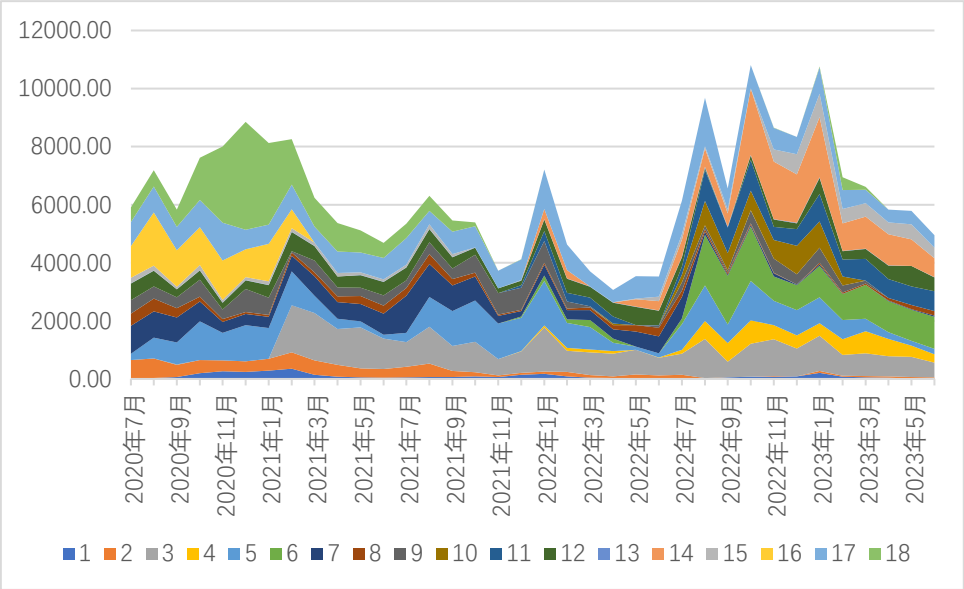


图 3 18 类单品的在不同月份下的销售量堆积面积图

接着，我们计算了 18 个聚类中心的均值、方差、峰度、偏度、极差，如下：

表 4 18 个聚类中心的统计指标（部分）

类别	均值	方差	峰度	偏度	极差
1	93.94	7395.74	1.61	1.51	347.40
2	215.30	40971.26	-0.73	0.75	672.48
3	782.34	239727.22	-0.67	-0.21	1632.92
…					…
16	269.53	284672.21	1.41	1.69	1825.90
17	764.92	72338.55	2.79	1.43	1234.00
18	532.98	784262.21	5.04	2.27	3706.82

通过对具有代表性的 18 个聚类中心的分析，得出单品销售量的分布规律如下：

均值在 93.9 到 782.3 之间变化，说明不同单品的销售量平均水平存在较大差异；方差在 38.5 到 784262.2 之间变化，说明不同单品的销售量离散程度差异较大；极差在 347.4 到 3,706.8 之间变化，说明不同单品之间的销售量波动范围差异较大；大多数单品的峰度值接近 0 或略小于 0，说明销售量分布相对平坦或稍微平缓；大部分单品的偏度值接近 0 或稍微大于 0，说明销售量分布近似对称。

（二）对品类的分布规律分析

根据附件一可知，蔬菜品类分为了六大类：花菜类、花叶类、辣椒类、茄类、食用菌、水生根茎类。我们分别计算了品类的均值、方差、峰度、偏度、极差，如下：

表 5 六种蔬菜品类的统计指标

品类名称	均值	方差	峰度	偏度	极差
花菜类	1160.18	198339.46	0.51	0.55	2018.15
花叶类	5514.47	2963371.46	0.46	0.39	8240.31

品类名称	均值	方差	峰度	偏度	极差
辣椒类	2544.13	1117095.04	0.26	0.82	4379.66
茄类	623.11	80789.33	0.16	0.32	1260.23
食用菌	2113.52	940215.40	-0.26	0.68	3770.10
水生根茎类	1127.26	416839.29	-0.97	0.33	2366.18

由此，可得品类销售量的分布规律如下：

花菜类的销售量分布相对平坦，接近正态分布，峰度略高，说明销售量在数据平均值附近相对集中；花叶类的销售量分布较广，离散程度较大，但整体呈现类似正态分布的形态，峰度略高，说明销售量分布相对陡峭；辣椒类的销售量分布接近正态分布，但具有稍微的正偏斜（右尾重），方差较大，说明销售量波动较大；茄类的销售量分布相对平坦，接近正态分布，方差相对较小，说明销售量波动不大，峰度也较低，显示出分布的平缓特征；食用菌的销售量分布接近正态分布，但具有明显的正偏斜（右尾重），方差较大，销售量波动较大；峰度较低，表明分布相对平缓；水生根茎类的销售量分布相对平坦，偏度接近于 0，说明数据分布相对对称，而峰度较低，表明分布相对平缓。

我们统计了六种蔬菜品类在不同月份下的销售量的堆积条形图，如下：

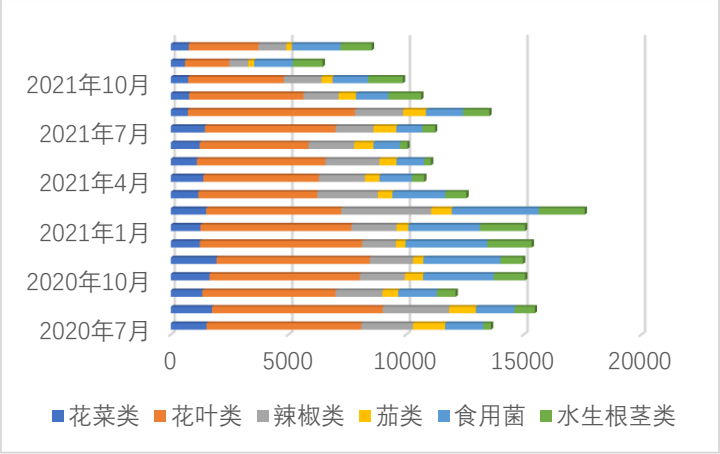


图 4 六种蔬菜品类在不同月份下的销售量的分布(2020-2021)

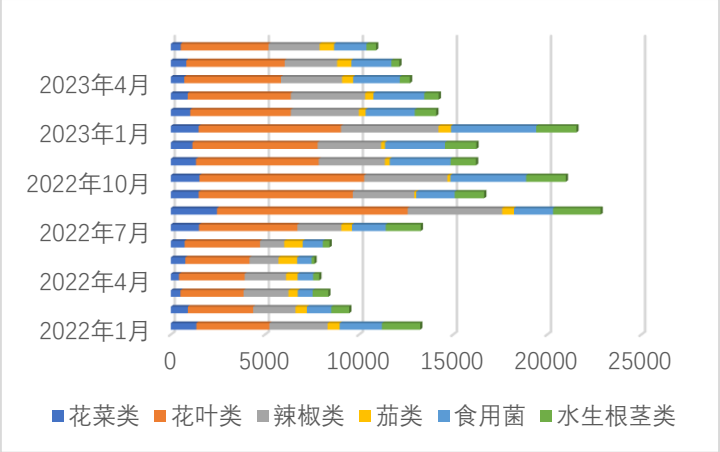


图 5 六种蔬菜品类在不同月份下的销售量的分布(2022-2023)

由此，可得蔬菜不同品类销售总量的分布规律如下：

从 2020 年 7 月到 2023 年 6 月的三年时间内，蔬菜销量呈现出波动的总体增长趋势。销售总量从初始的 13613kg 逐步增加至最高点 22842kg，并在接下来的几个月中下降到 14106kg 左右。通常在每年的夏季和秋季，销售总量明显增加，并保持在较高水平。总体而言，销售总量在时间段内有波动，但整体呈逐步增长的趋势。

5.1.3 品类及单品销售量相关性分析

（一）对单品的相关性分析

与对单品的分布规律分析相同，我们选择 18 个聚类中心作为 246 个单品的代表，进行相关性分析。为了选择适合的相关性分析方法，我们使用 Spss 软件对 18 个聚类单品的销售量进行了正态分布检验：

表 6 18 个聚类单品的正态性检验

	柯尔莫戈洛夫-斯米诺夫 [a]			夏皮洛-威尔克		
	统计	自由度	显著性	统计	自由度	显著性
1	.277	36	.000	.812	36	.000
2	.205	36	.001	.862	36	.000
3	.136	36	.089	.930	36	.025
4	.295	36	.000	.717	36	.000
5	.127	36	.156	.934	36	.033
6	.296	36	.000	.709	36	.000
7	.127	36	.149	.933	36	.031
8	.159	36	.022	.897	36	.003
9	.109	36	.200*	.941	36	.054
10	.438	36	.000	.554	36	.000
11	.284	36	.000	.771	36	.000
12	.090	36	.200*	.988	36	.953
13	.228	36	.000	.665	36	.000
14	.299	36	.000	.707	36	.000
15	.266	36	.000	.797	36	.000
16	.471	36	.000	.558	36	.000
17	.160	36	.020	.878	36	.001
18	.274	36	.000	.659	36	.000

可以发现只有第 3 个、第 5 个、第 7 个、第 9 个、第 12 个聚类单品的 Shapiro-Wilk 显著性水平大于 0.05，因此我们得出结论：蔬菜单品的销售量不符合正态分布的要求。基于此结论，我们选择用斯皮尔曼相关系数进行分析。

斯皮尔曼相关系数（Spearman correlation coefficient）是一种用于衡量两个变量之间的非线性关系的统计量，它基于两个变量的秩次而不是原始数值，不要求变量服从正态分布。斯皮尔曼相关系数的计算公式如下：

$$\rho_s = \frac{\frac{1}{n}\sum_{i=1}^{n}\left(R(x_i)-\bar{R(x)}\right)\cdot\left(R(y_i)-\bar{R(y)}\right)}{\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}\left(R(x_i)-\bar{R(x)}\right)^2\right)\cdot\left(\frac{1}{n}\sum_{i=1}^{n}\left(R(y_i)-\bar{R(y)}\right)^2\right)}} \tag{1}$$

我们使用 Spss 软件，对 18 个聚类单品的销售量数据进行斯皮尔曼相关分析，得到了相关系数矩阵，绘制了热力图：

表 7 18 个聚类的相关系数矩阵（部分）

相关系数	单品 1	单品 2	单品 3	单品 4	…	单品 18
单品 1	1					
单品 2	.223	1				
单品 3	.144	-.127	1			
单品 4	-.278	-.874	.105	1		
…					…	
单品 18	.526	.751	-.105	-.749	…	1

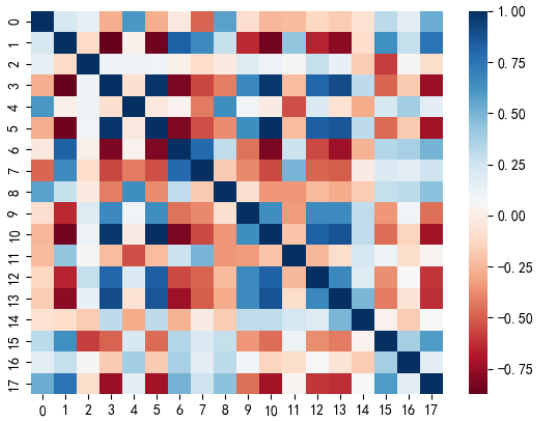


图 6 18 个聚类的斯皮尔曼相关系数热力图

分析相关系数矩阵与热力图，在 0.05 的显著性水平下，可得结论：

单品 1、3、9、17 与其他单品之间的相关性较弱，大多数相关系数接近于 0；单品 2 与单品 4 之间的相关系数为-0.874，存在较强的负相关性；单品 2 与单品 7 之间的相关系数为 0.829，存在较强的正相关性；单品 4 与单品 11 之间的相关系数为 0.968，存在非常强的正相关性；单品 6 与单品 11 之间的相关系数为 0.996，存在非常强的正相关性。

以此为代表，我们得出了各蔬菜单品间销售量的相关关系：大多数单品之间并无显著的相关性，少部分单品之间存在相关性，这与不同单品的上市季节、消费者的饮食习惯有关。

（二）对品类的相关性分析

为了选择适合的相关性分析方法，我们使用 Spss 软件对六种蔬菜品类的销售量进行了正态分布检验：

表 8 对六种蔬菜品类的正态性检验

	柯尔莫戈洛夫-斯米诺夫[a]			夏皮洛-威尔克		
	统计	自由度	显著性	统计	自由度	显著性
花菜类	.112	36	.200*	.954	36	.140
花叶类	.087	36	.200*	.983	36	.854
辣椒类	.146	36	.052	.943	36	.063
茄类	.078	36	.200*	.979	36	.724
食用菌	.131	36	.124	.944	36	.068
水生根茎类	.122	36	.191	.948	36	.090

可以发现 Shapiro-Wilk 检验的 p 值均大于 0.05 水平，因此我们认为六种蔬菜品类的销售量符合正态分布的要求，可用皮尔逊相关分析。

皮尔逊相关系数（Pearson correlation coefficient）是一种衡量两个连续变量之间线性相关程度的统计指标。它可以用来度量两个变量之间的线性关系强度和方向，取值范围为-1 到 1 之间。皮尔逊相关系数的计算公式如下：

$$\rho_p = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \tag{2}$$

我们使用 Spss 软件，对六种蔬菜品类的销售量数据进行皮尔逊相关分析，得到了相关系数矩阵，绘制了热力图：

表 9 六种品类间的相关系数矩阵

相关系数	花菜类	花叶类	辣椒类	茄类	食用菌	水生根茎类
花菜类	1					
花叶类	. 747	1				
辣椒类	. 425	. 624	1			
茄类	. 058	−. 035	−. 191	1		
食用菌	. 429	. 546	. 576	−. 409	1	
水生根茎类	. 472	. 484	. 459	−. 466	. 653	1

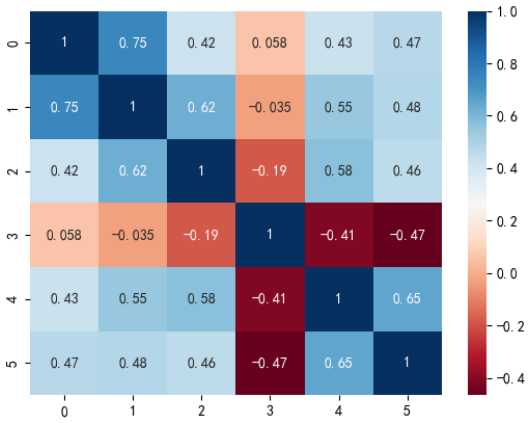


图 7 六种品类的皮尔逊相关系数热力图

分析相关系数矩阵与热力图，在 0.05 的显著性水平下，可得结论：

一、茄类与食用菌、水生根茎类的销售呈显著的负相关，且与其他品类相关性不强，这可能是因为：茄子具有独特的口感和味道，其质地柔软、具有一定的苦味，这些特点使得茄子在饮食中具有独立的地位，并较少与其他食材进行搭配，消费者偏向于单独购买茄子进行烹制。

二、除此之外，其他蔬菜品类之间的相关性较明显，这可能是因为：不同蔬菜品类在烹饪中经常被搭配使用，以此增强菜品口感并获得各种营养；蔬菜品类的季节性供应，使得消费者趋向于购买应季的品类。

5.2 问题二模型的建立和求解

5.2.1 探究品类的销售量与定价的关系

在对问题二的分析之前，我们需要对数据进行预处理。分析附件二，我们发现：存在 461 条退货记录，其对应的销售量为负数。为了更清晰地得到销售量与定价的规律，我们假设理想情况：没有消费者退货。因此，我们剔除了退货数据。

为了分析各蔬菜品类的销售总量与成本加成定价的关系，我们需要得到六种蔬菜品类每日的销售总量，以及每日每个品类的代表定价。以日为单位，对每个品类所有单品的销售量求和，得到不同品类每日的总销售量：

表 10 六种品类每日的销售总量（部分）

销售日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2020/7/1	4.85	35.365	35.374	76.715	205.402	46.64
2020/7/2	4.6	48.51	32.199	66.064	198.362	43.943
2020/7/3	9.572	42.442	35.896	64.253	190.779	42.076
…						…
2023/6/28	14.946	53.742	15.651	68.534	130.182	16.069
2023/6/29	22.945	48.314	11.511	89.113	135.09	24.367
2023/6/30	19.419	39.572	24.53	82.286	130.464	28.087

由于蔬菜的定价采用的是“成本加成定价”，运损和品相变差的商品通常会被降价或打折出售，因此同一天内同一单品可能会有不同的定价。为了得到每个品类的代表定价，需要先计算每一个单品对应总销量的权重：

$$\text{对应权重} = \frac{\text{单品销量}}{\text{品类总销量}} \tag{3}$$

算出单品的对应权重为f，再使用加权平均法计算品类的平均定价：

$$\bar{x} = \frac{x_1 f_1 + x_2 f_2 + \cdots + x_n f_n}{n} \tag{4}$$

其中$\bar{x}$为加权平均定价，x_i为该品类第 i 个单品的单价，算出不同品类每日的加权平均定价如下表：

表 11 六种品类每日的加权平均定价

销售日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2020/7/1	14.49	10.43	5.00	9.91	7.45	12.84
2020/7/2	11.55	12.49	5.34	10.14	7.00	12.43
2020/7/3	10.00	12.87	5.47	10.44	7.43	11.77
…						…
2023/6/28	18.06	4.71	7.77	6.26	5.21	12.33
2023/6/29	18.05	4.78	7.50	6.32	5.00	12.26
2023/6/30	17.80	6.61	8.10	6.19	5.24	11.92

由于我们对定价进行了加权平均处理，因此我们将该平均定价作为成本加成定价。在得到每种品类的成本加成定价与销售总量之后，我们用 Spss 对销售总量与成本加成定价进行回归性分析并作出散点图，以水生根茎为例：

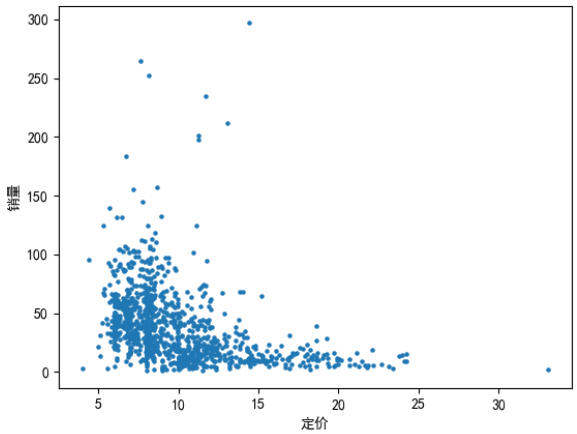


图 8 水生根茎类平均定价和销量总量的散点图

根据六个不同蔬菜品类的散点图描绘，我们发现所有蔬菜品类都大致按照上图的分布规律，在定价低时分布密集，在定价高时分布稀疏，并且在定价低时对应的销量普遍较高，符合价格升高需求降低的市场规律。

接着我们以水生根茎类为代表进行了线性回归拟合，得出显著性水平在 0.001 以下，拟合效果好。因此我们得到了销售总量与成本加成定价之间的线性回归方程：$y = -0.042x + 11.415$ 其中 x 表示成本加成定价，y 表示销售总量。根据相同的方法，我们可以得到所有六种蔬菜品类的线性回归方程：

表 12 六种蔬菜品类的线性回归方程

蔬菜品类名称	回归线性方程
水生根茎类	$y = -0.042x + 11.415$
食用菌类	$y = -0.013x + 9.611$
茄类	$y = -0.002x + 8.574$
辣椒类	$y = -0.012x + 9.178$
花叶类	$y = -0.004x + 6.384$
花菜类	$y = -0.031x + 10.667$

由表，我们可以发现所有六个线性回归方程都是减函数，但是其减小的速度都很慢，因此我们可以得出结论：各蔬菜品类的销售总量与成本加成定价呈负相关的关系，但下降并不明显。由于蔬菜属于经济学上生活必需品的范畴，因此销量不会随定价变化发生大幅度的下降，这也符合经济学的原理。

5.2.2 探究未来一周的日补货计划

由题可知，蔬菜商品的保鲜期短，若当日未售出，隔日就无法再售，会影响商超利润。当日未售出商品数与商超利润呈负相关。此外，蔬菜商品在运输到商超以及存储的过程中会有损耗，提前计划的补货量不等于实际的补货量。因此，在做补货计划时应考虑损耗问题。附件 4 中的损耗率反映了近期损耗情况。为了使商超收益最大，我们需要尽量让所有商品都在当日售出，即让销售量等于实际补货量，得到模型：

销售量 = 实际补货量 = 计划补货量 *（1 - 损耗率）

考虑商超以品类为单位做补货计划，为了得到各品类未来一周的计划补货量，我们需要得到每个品类的平均损耗率。根据附件 4，对每个品类所有单品的损耗率求均值，得到了每个品类的平均损耗率：

表 13 每个品类的平均损耗率

品类	平均损耗率（%）
花菜类	14.14
花叶类	10.28
辣椒类	8.52
茄类	7.12
食用菌	8.13
水生根茎类	11.97

我们还需要得到各个品类未来一周的销售量。结合题干、生活常识与对问题一的分析，可知：蔬菜类商品的销售量可能有季节效应。因此我们考虑使用时间序列预测模型来预测各蔬菜品类未来一周的销售量。时间序列预测模型可以捕捉到时间序列数据中的趋势和季节性变化，提高预测的准确性。

在问题一中，我们绘制了六种品类销售量的时间序列折线图，均呈现周期性的波动，因此我们认为：六种品类销售量的时间序列为非平稳序列。

非平稳序列是不稳定的数据，无法捕捉到时序规律，因此我们需要进行差分处理，尝试将其转化为平稳序列。使用 python 对时间序列进行一阶差分后，我们对其进行单位根 ADF 检验与白噪声检验：

表 14 单位根 ADF 检验与白噪声检结果

单位根 ADF 检验结果			白噪声检验结果		
品类	ADF	P-value	品类	stat	P-value
水生根茎类	-10.3	0.000	水生根茎类	106.549	0.000
食用菌	-11.335	0.000	食用菌	92.585	0.000
茄类	-10.635	0.000	茄类	72.037	0.000
辣椒类	-14.941	0.000	辣椒类	50.105	0.000
花叶类	-11.77	0.000	花叶类	197.361	0.000
花菜类	-11.653	0.000	花菜类	44.403	0.000

六种品类的单位根检验与白噪声检验的 p 值均小于 0.05，即通过了平稳性检验和白噪声检验。因此我们认为：原序列经过一阶差分后，得到了平稳非白噪声序列，符合 ARIMA 时间序列预测模型的条件。

随后，我们绘制了一阶差分时间序列的时序图、直方图、自相关图与偏自相关图并作出分析：

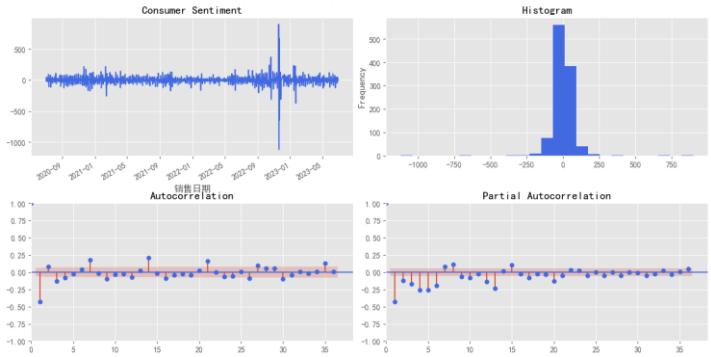


图 9 花叶类一阶差分的时序图、直方图、自相关图和偏自相关图

花叶类：自相关图 1 阶截尾，偏自相关图 6 阶截尾

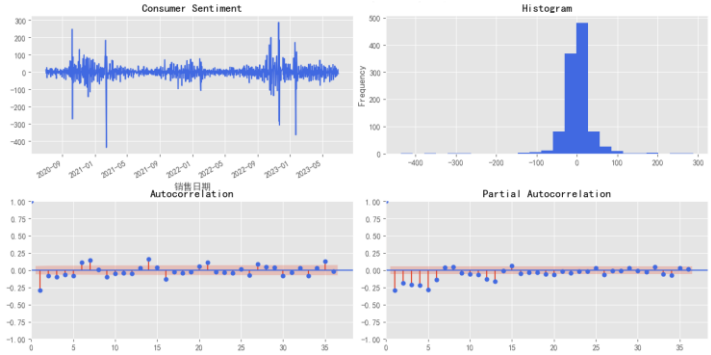


图 10 食用菌类一阶差分的时序图、直方图、自相关图和偏自相关图

食用菌类：自相关图 1 阶截尾，偏自相关图 6 阶截尾

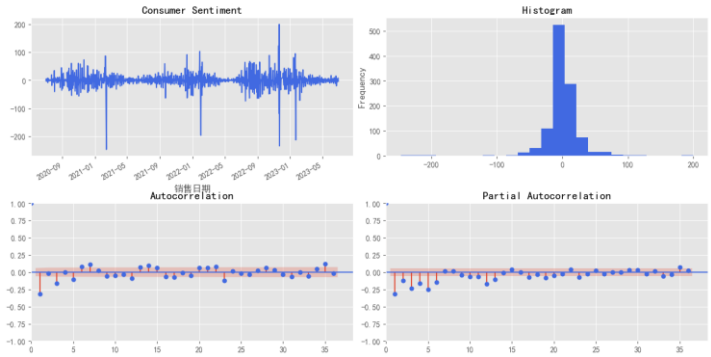


图 11 水生根茎类一阶差分的时序图、直方图、自相关图和偏自相关图

水生根茎类：自相关图拖尾或者 1 阶截尾，偏自相关图 6 阶截尾

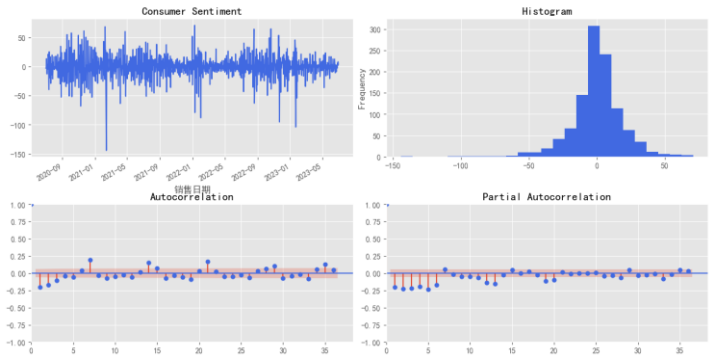


图 12 花菜类一阶差分的时序图、直方图、自相关图和偏自相关图

花菜类：自相关图拖尾或者 3 阶截尾，偏自相关图 6 阶截尾

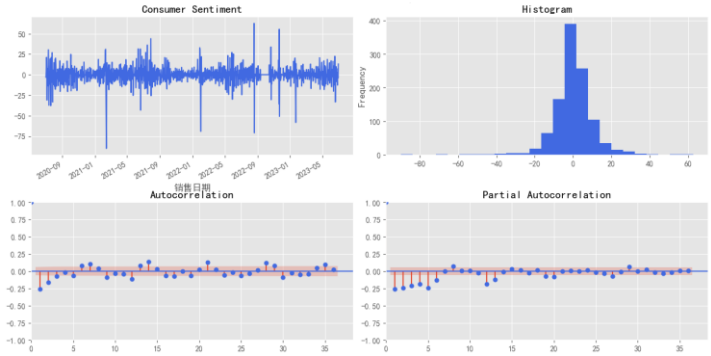


图 13 茄类一阶差分的时序图、直方图、自相关图和偏自相关图

茄类：自相关图拖尾或者 1 阶截尾，偏自相关图 6 阶截尾

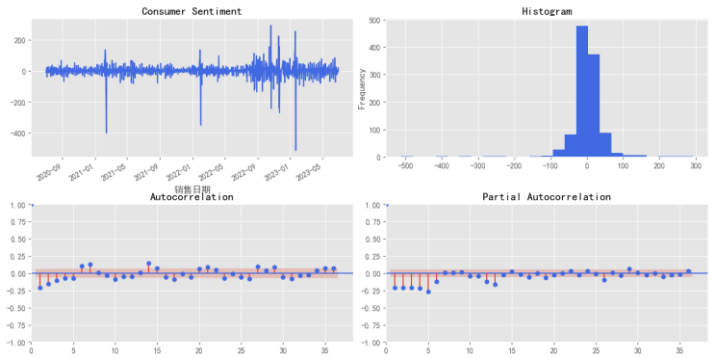


图 14 辣椒类一阶差分的时序图、直方图、自相关图和偏自相关图

辣椒类：自相关图拖尾或者 3 阶截尾，偏自相关图 6 阶截尾

为了进一步确定拟合模型，我们使用 AIC 准则（最小化信息量准则）来选择相对最优模型。AIC 的计算公式如下：

$$AIC = -2\ln(L) + 2k \tag{5}$$

其中，ln(L)是模型的最大似然函数值，k 是模型的参数个数。AIC 的计算结果越小，表示模型具有较好的拟合效果和较低的复杂度。

AIC 准则可以用于确定 ARIMA 模型的相对最优阶数，对模型进行拟合，模型可以预测未来 7 天的数据。对于 ARIMA(p, d, q)模型，其中 p 表示自回归阶数，q 表示滑动平均阶数，d 表示差分阶数。对于每个拟合的模型，计算 AIC 的值。AIC 的计算公式为：

$$AIC = -2\ln(L) + 2(p + q + 1) \tag{6}$$

我们使用 Python，比较每个模型的 AIC 值，选择具有最小 AIC 值的模型作为相对最优的 ARIMA 模型。最后，我们预测出了六种品类未来七天的销售量，如下：

表 15 六种品类蔬菜未来七天的销售量预测

日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2023/7/1	24.39	40.31	26.95	75.35	138.38	22.95
2023/7/2	18.01	44.15	25.68	74.13	130.90	20.64
2023/7/3	21.65	44.71	23.01	77.77	112.76	19.59
2023/7/4	15.15	43.16	21.86	79.32	109.56	19.12
2023/7/5	14.70	42.58	22.10	78.26	119.63	18.91
2023/7/6	13.66	43.12	22.78	77.11	130.19	18.82
2023/7/7	13.45	43.49	23.16	77.22	136.79	18.78

至此，我们得到了六种品类的平均损耗率与未来一周的销售量，根据公式：
计划补货量 = 销售量 / （1 - 损耗率）
我们可以计算六种蔬菜品类未来一周的日补货总量，如下：

表 16 六种品类蔬菜未来七天的日补货总量预测

日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2023/7/1	27.70	43.87	29.02	82.36	154.23	26.73
2023/7/2	20.46	48.06	27.65	81.04	145.90	24.03
2023/7/3	24.59	48.66	24.78	85.02	125.68	22.82
2023/7/4	17.20	46.98	23.53	86.71	122.12	22.27
2023/7/5	16.69	46.35	23.79	85.54	133.34	22.03
2023/7/6	15.52	46.94	24.52	84.29	145.11	21.92
2023/7/7	15.28	47.34	24.93	84.42	152.46	21.87

5. 2. 3 探究未来一周的定价策略

为了得到利润最大的定价策略，我们需要最优化利润函数。利润等于销量乘以定价与批发价的差值。在探究销售总量与成本加成定价的关系中，我们得到了它们的线性回归方程。因此，对于每个品类，建立以下模型：

$$\begin{cases} T_i = Q_i(R_i - W_i) \\ Q_i = k_i R_i + b_i \end{cases} \tag{7}$$

即：

$$T = (k_i R_i + b_i)(R_i - W_i) = k_i R_i^2 + (-W_i k_i + b_i)R_i - W_i b_i \tag{8}$$

其中T_i为品类 i 的利润，Q_i为品类 i 的销售量，R_i为品类 i 的定价，W_i为品类 i 的批发价，k_i为品类 i 的销售量与定价关系的常数，b_i为品类 i 的销售量与定价关系的系

数。由公式可知，利润函数是以定价为自变量的二次函数。为了最优化利润，得到适合的定价，根据式子，我们需要得到未来一周不同品类的批发价。在对附件 3 中的单品批发价格进行了分类、求均值、按日压缩处理过后，我们得到了不同品类每日的批发价：

表 17 六种品类蔬菜每日的批发价（部分）

日期	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
2020/7/1	17.54	5.33	7.63	4.10	7.20	7.89
2020/7/2	17.65	5.25	7.21	2.59	7.01	7.84
2020/7/3	5.61	5.09	6.89	4.02	6.85	7.82
…						…
2023/6/28	11.03	3.84	8.28	4.61	5.86	4.75
2023/6/29	12.10	3.31	8.31	4.48	6.44	4.84
2023/6/30	12.04	3.50	7.81	4.40	5.73	4.62

与对销售量的预测方法类似，为了得到未来一周的批发价，我们选择时间序列分析对批发价进行预测，且认为：六种品类批发价的时间序列为非平稳序列。使用 Python 对时间序列进行一阶差分后，我们对其进行了单位根 ADF 检验与白噪声检验，以水生根茎类为例：

表 18 批发价时间序列的 ADF 检验与白噪声检验结果

	单位根 ADF 检验结果		白噪声检验结果	
	ADF	P-value	stat	P-value
水生根茎类	-6.755	0.000	29.808	0.000

我们对六种品类均进行了检验，结果发现，六种品类的单位根检验与白噪声检验的 p 值均小于 0.05，即通过了平稳性检验和白噪声检验。因此我们认为：原序列经过一阶差分后，得到了平稳非白噪声序列，符合 ARIMA 时间序列预测模型的条件。随后，我们绘制了一阶差分时间序列的自相关图与偏自相关图，同样以水生根茎类为例：

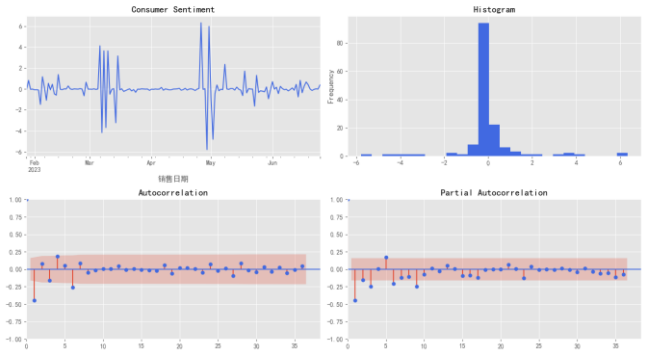


图 15 水生根茎类一阶差分的时序图、直方图、自相关图和偏自相关图

我们使用 Python，比较每个模型的 AIC 值，选择具有最小 AIC 值的模型作为相对最优的 ARIMA 模型，预测出了六种品类未来七天的批发价：

表 19 六种品类蔬菜未来七天的批发价预测

日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2023/7/1	12.01	4.74	4.42	6.00	3.58	7.74
2023/7/2	11.30	4.75	4.41	6.00	3.47	7.94
2023/7/3	11.28	4.76	4.40	6.11	3.50	7.77
2023/7/4	11.47	4.76	4.41	5.97	3.52	7.79
2023/7/5	11.30	4.76	4.41	6.15	3.50	7.86
2023/7/6	11.26	4.76	4.41	6.01	3.51	7.79
2023/7/7	11.26	4.76	4.41	6.16	3.51	7.81

至此，我们得到了利润函数所需的全部参数，绘制出了函数图，以 2023/7/1 的水生根茎类为例：

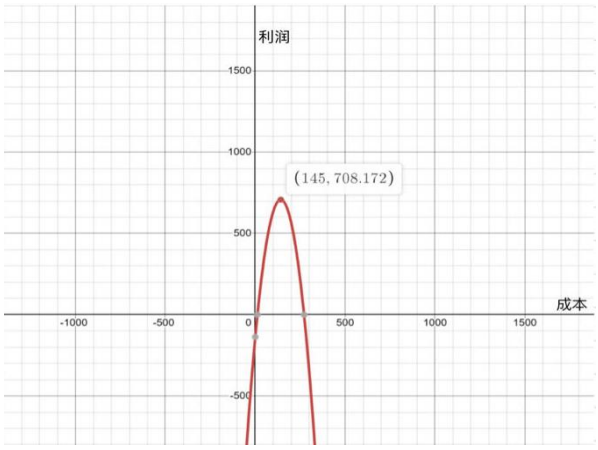


图 16 2023/7/1 的水生根茎类成本与利润的二次函数图像

由于缺乏约束条件，且销售总量与定价的回归拟合不是特别好，因此直接使用二次函数的顶点作为利润最大值、定价最优解是不合理的，如图：2023/7/1 的水生根茎类的利润函数曲线的最大利润为 708.172，对应的定价为 145，显然不符合常理。因此，我们对定价使用时间序列分析，预测未来一周的定价，在预测的置信区间内结合利润函数寻找定价最优解。

由于批发价与定价的相似性，参考批发价的时间序列分析，我们认为：六种品类定价的时间序列为非平稳序列。使用 python 对时间序列进行一阶差分后，我们对其进行单位根 ADF 检验与白噪声检验，同样发现 p 值均小于 0.05。接着使用 Python，选择具有最小 AIC 值的模型作为相对最优的 ARIMA 模型，根据置信区间，预测出了六种品类未来七天的定价区间，以水生根茎类为例：

表 20 水生根茎未来七天的定价区间

日期	最低价	最高价
2023/7/1	14.151	22.007
2023/7/2	13.963	22.430
2023/7/3	13.866	22.334
2023/7/4	13.735	22.448
2023/7/5	13.590	22.682

日期	最低价	最高价
2023/7/6	13.514	22.749
2023/7/7	13.430	22.803

得到可信的预测定价区间后，通过分析利润与定价的二次函数最大值所对应的定价与预测定价区间的关系，我们可以得出定价的最优解。例如：若二次函数最大值所对应的定价大于预测定价区间的最右端，说明在定价区间内利润曲线单调递增，因此区间最右端即为定价的最优解；　　若函数最大值所对应的定价在定价区间内，说明函数的最大值即为区间最大利润，所对应的定价为最优解。

最后，我们得到了使商超收益最大的各蔬菜品类未来一周的最优定价，如下表所示：

表 21 六种品类蔬菜未来七天的最优定价

日期	水生根茎类	食用菌	茄类	辣椒类	花叶类	花菜类
2023/7/1	22.01	6.85	9.21	7.46	5.87	13.27
2023/7/2	22.43	6.54	9.41	7.53	5.97	13.61
2023/7/3	22.33	6.61	9.58	7.59	6.13	13.84
2023/7/4	22.45	6.77	9.73	7.73	6.03	14.03
2023/7/5	22.68	6.89	9.87	7.77	6.06	14.20
2023/7/6	22.75	6.95	10.00	7.78	6.19	14.36
2023/7/7	22.80	6.99	10.12	7.86	6.15	14.51

5.3 问题三模型的建立和求解

5.3.1 数据预处理

问题三要求我们给出 7 月 1 日的单品补货量和定价策略，需要满足五个条件：一、可售单品总数控制在 27-33 个；二、各单品订购量满足最小陈列量为 2.5 千克；三、只能选择 2023 年 6 月 24-30 日的可售品种；四、尽量满足市场对各品类的需求；五、尽量使得商超收益最大。

我们首先对数据进行预处理：筛选出 2023 年 6 月 24-30 日有销售记录的单品，计算出每个的平均利润与平均销量，作为该单品的代表：

表 22 可选单品的七日内平均利润与平均销量（部分）

单品名称	分类名称	平均利润	平均销量
西兰花	花菜类	5.801696	12.55771
枝江青梗散花	花菜类	6.631604	3.600429
菠菜	花叶类	7.512092	2.273333
…			…
净藕(1)	水生根茎类	7.257724	2.158
菱角	水生根茎类	4.073478	2.037333
野生粉藕	水生根茎类	15.76577	0.813

根据问题二的模型，我们可以得到 7 月 1 日当天的各品类的预测总销售量：

表 23 7 月 1 日当天各品类的预测销量

	花菜类	花叶类	辣椒类	茄类	食用菌	水生根茎类
预测销量	26.72837	154.2337	82.36465	29.01505	43.87358	27.70364

5.3.2 背包问题

对于该最优化问题，我们可以考虑将其视为一个完全背包问题，使用动态规划求解。完全背包问题是在给定一个固定的背包容量 m 和一组不同物品 n 的情况下，要求从这组物品中选择一些放入背包中，以使得在满足背包容量限制的前提下，所选物品的总价值最大化。状态转移方程如下：

$$dp[i][j] = \max_{k=0}^{+\infty}(dp[i][j], dp[i-1][j-kw_i] + kv_i) \tag{9}$$

其中，$dp[i][j]$ 为在只能放前 i 个物品的情况下，容量不超过 j 的背包所能达到的最大总价值；w_i 为第 i 个物品的体积；v_i 为第 i 个物品的价值。

在本问中，为了满足条件：尽量满足市场对各品类的需求，我们使用六个完全背包，其中每个背包都代表一个品类。背包的容量 m 为该品类的总销量，即当天该单品的所有销量不应超过该总销量。背包物品的体积 w 为单品销量，价值 v 为单品利润。

我们使用 Python 代码，对该完全背包问题进行求解，同时在代码中加入回溯部分，记录每个被纳入背包的单品，得到：

表 24 选中单品的补货量及最大收益

编号	单品名称	单品补货量	最大收益
水生根茎类	净藕(1)	10.81814574	78.51512
	高瓜(1)	8.120835607	38.14813
	红莲藕带	4.016115466	10.66712
	野生粉藕	4.748545126	74.86447
	高瓜(2)*	6.925910485	18.2126
	菱角*	6.925910485	28.21255
花菜类	西兰花	18.3121487	106.2415
	枝江青梗散花	8.416224233	55.81306
辣椒类	螺丝椒	39.24105568	148.6321
	小皱皮(份)	27.38965652	8.714945
	青线椒(份)	6.136480423	12.99988
	七彩椒(2)	4.330309685	24.05132
	红椒(2)	5.267145696	33.0372
	青红杭椒组合装(份)*	16.4729296	48.25924
食用菌	西峡花菇(1)	31.85002	327.3674
	白玉菇(袋)	5.918334	9.007085
	木耳菜(份)	6.105222	7.457325
茄类	紫茄子(2)	19.85017	7.318648
	青茄子(1)	7.689306	5.392772
	紫茄子(1)	1.475565	6.435824
	长线茄*	9.671682	42.39039
花叶类	苋菜	83.34822	-27.8024
	云南生菜	27.30749	112.0819
	云南油麦菜	15.54486	44.52571
	菠菜	22.10477	166.0531
	木耳菜(份)	5.928386	7.241325
	上海青*	30.84675	112.3071

成功选出 27 种单品，满足单品数范围在 27-33 的约束条件；且补货量均大于 2.5 千克，满足最小陈列量的约束条件。满足全部条件，计算出的各单品的补货量有效。

商家为了营销不亏本，需要在制定价格时将成本损耗纳入考虑。为了满足市场对新鲜蔬菜单品的需求，商家在补货时需要考虑货品在运输途中造成的损伤或者新鲜度下降等原因带来的损耗，因此实际补货量一般会比预期销量多，它们的关系如下：

损耗率=（实际补货量-预期销量）/实际补货量
即：
实际补货量=预期销量/(1-损耗率)
为了抵消这部分损失，考虑成本加成定价，应该满足
定价=批发价格*（1+利润率）
由此，我们可得 7 月 1 日选出单品的定价，即最优的定价策略：

表 25 选中单品的最优定价

编号	单品名称	定价	编号	单品名称	定价
水生根茎类	净藕(1)	19.26713	食用菌	西峡花菇(1)	15.01908
	高瓜(1)	16.70697		白玉菇(袋)	6.262574
	红莲藕带	14.66548		木耳菜(份)	5.962146
	野生粉藕	27.77517	茄类	紫茄子(2)	4.786091
	高瓜(2)	14.63904		青茄子(1)	5.118731
	菱角	16.08288		紫茄子(1)	8.778997
花菜类	西兰花	13.53766		长线茄*	8.800336
	枝江青梗散花	14.36757	花叶类	苋菜	3.244323
辣椒类	螺丝椒	9.78304		云南生菜	7.682331
	小皱皮(份)	6.313556		云南油麦菜	6.442228
	青线椒(份)	8.113831		菠菜	11.08998
	七彩椒(2)	11.54955		木耳菜(份)	4.799359
	红椒(2)	12.26769		上海青	7.218699
	青红杭椒组合装(份)*	8.924981			

5.4 问题四意见的提出和分析

为了能够得到更加完整和全面的市场调研数据，以便更好地制定蔬菜商品的补货和定价的决策，我们认为商超还应该更加关注以下方面：

（一）当地的气候和季节数据。可以帮助商超了解当季的作物，并加大对应季蔬菜的进货量，可以减少非应季蔬菜出现的品质问题，有助于增大销量。

（二）不同蔬菜更详细的损耗率数据。本题中蔬菜损耗率为常量，了解更详细的损耗率数据可以帮助商超优化进货策略，减少损失，从而提高利润。

（三）当地居民的饮食习惯数据。能够帮助商超了解附近居民对某些蔬菜品类的喜好，并加大其进货量，可以有效提升销售量。

（四）市场调研数据。包括竞争对手的定价政策、用户意见和反馈数据等信息，能够帮助商超更合理地做出定价政策。

六、 模型的评价、改进和推广

6.1 模型的优点

（一）ARIMA 模型可以适用于各种不同类型的时间序列数据，包括非平稳的和具有趋势、季节性的序列。它能够自动处理这些具有特征性的数据，并提供相对准确的预测结果，并且建模过程相对简单，需要提供的参数较少。

6.2 模型的缺点

（一）如果时间序列具有较长的滞后性或包含大量的噪声，ARIMA 模型可能需要较高阶的自回归和移动平均项来捕捉模式。这会增加模型的复杂性，并增加参数估计和预测的不确定性，降低预测的概率。

6.3 模型的改进

（一）可以使用贝叶斯信息准则（BIC）。BIC 是 AIC 的改进版本，通过引入更严格的复杂性惩罚项来平衡拟合优度和模型复杂度。相比于 AIC，BIC 更加倾向于选择更简单的模型。在选择模型时，可以在多个模型之间进行比较和综合考量，综合模型的结果，以获得更全面的模型评价结果。

（二）可以引用 SARIMA 模型。SARIMA 模型是 ARIMA 模型的扩展，用于处理具有季节性模式的时间序列。它引入了季节差分和季节自回归、移动平均项，能够充分地考虑季节性带来的影响，带来更准确的建模和季节性序列的预测。

6.4 模型的推广

本论文使用的 ARIMA 模型作为一种灵活性和解释性强的模型，具有简单通用、基于信息理论和简单易分析的特点。在本题中我们借助 AIC 来确定使用的 ARIMA 模型，得到更为准确的销售量、进货量等预测数据，这对商超未来的运营提供了新的思路，能够使用这种数学模型提高商超的利润率。

七、 参考文献

[1]张晓莉,王彦蕊,王国庆等.基于主成分分析与聚类分析的百合品质评价[J].农产品加工,2023(14):49-54.DOI:10.16693/j.cnki.1671-9646(X).2023.07.042.

[2]周成龙,陈玉明,朱益冬.粒 K 均值聚类算法[J].计算机工程与应用,2023,59(13):317-324.

[3]张平.傅里叶变换的性质探讨[J].科技资讯,2020,18(18):255-c256.DOI:10.16661/j.cnki.1672-3791.2020.18.255.

附录

问题一 附录 2 数据处理

```
import pandas as pd
import re

data = pd.read_excel("附件 2.xlsx")

# 重复行检查
duplicate_rows = data[data.duplicated()]
if duplicate_rows.shape[0] > 0:
    print("附件 2 中存在重复的行：")
    print(duplicate_rows)
else:
    print("附件 2 中没有重复的行。")

# 时间异常值检查：正则表达式
pattern = r'^([0-1][0-9]|2[0-3]):([0-5][0-9]):([0-5][0-9](\.\d{1,3})?)$'
invalid_times = data[~data['扫码销售时间'].astype(str).str.match(pattern)]
if invalid_times.shape[0] > 0:
    print("附件 2 中存在异常时间数据：")
    print(invalid_times)
else:
    print("附件 2 中没有异常时间数据。")

# 给"附件 2 预处理.xlsx"加上单品名与分类名
df1 = pd.read_excel('附件 1.xlsx')
df2 = pd.read_excel('附件 2 预处理.xlsx')
# 创建字典
mapping_dict1 = dict(zip(df1['单品编码'], df1['单品名称']))
mapping_dict2 = dict(zip(df1['单品编码'], df1['分类名称']))
# 字典映射
df2['单品名称'] = df2['单品编码'].map(mapping_dict1)
df2['分类名称'] = df2['单品编码'].map(mapping_dict2)
df2.to_excel('按每年的月份分类.xlsx', index=False)
```

问题一 绘制热力图

```
import pandas as pd
import numpy as np
import seaborn as sns
```

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei'] #  显示中文
plt.rcParams['axes.unicode_minus'] = False #  正常显示负号

#  从 Excel 读取数据
data_frame1 = pd.read_excel('品类热力图.xlsx')
#  提取矩阵数据
matrix_data = data_frame1.values
#  创建热力图
sns.heatmap(matrix_data, annot=True, cmap='RdBu')
#  展示热力图
plt.show()

#  从 Excel 读取数据
data_frame2 = pd.read_excel('聚类的相关系数矩阵.xlsx')
#  提取矩阵数据
matrix_data = data_frame2.values
#  创建热力图
sns.heatmap(matrix_data, annot=False, cmap='RdBu')
#plt.title("18 类单品的斯皮尔曼相关系数热力图")
#  展示热力图
plt.show()
```

问题二 数据预处理

```
import pandas as pd

#  读取 Excel 文件
df = pd.read_excel('附 2 按秒记录删去退货补充品名的完整数据.xlsx')

#  将销售日期转换为日期类型
df['销售日期'] = pd.to_datetime(df['销售日期'])

#  按日期、单品编码和品类分组，对销量进行求和，保留单价和品类
df_grouped = df.groupby(['销售日期', '单品编码', '品类']).agg({'销量(千克)': 'sum', '销售单价(元/千克)': 'first'}).reset_index()

#  将累加后的数据写回 Excel 文件
df_grouped.to_excel('完整数据 2.xlsx', index=False)
```

销售定价散点图

```
import pandas as pd
import matplotlib.pyplot as plt
```

```
import matplotlib.dates as mdates
import datetime
plt.rcParams['font.sans-serif'] = ['SimHei'] #  显示中文
plt.rcParams['axes.unicode_minus'] = False #  正常显示负号

pricing_data = pd.read_excel('weighted_prices.xlsx')
sales_data = pd.read_excel('总销售.xlsx')

#  茄类散点图
pricing_tomato = pricing_data['水生根茎类']
sales_tomato = sales_data['水生根茎类']
plt.scatter(pricing_tomato, sales_tomato, s=5)
plt.xlabel('定价')
plt.ylabel('销量')
plt.title('')
plt.show()

#  辣椒类散点图
pricing_pepper = pricing_data['辣椒类']
sales_pepper = sales_data['辣椒类']
plt.scatter(pricing_pepper, sales_pepper)
plt.xlabel('定价')
plt.ylabel('销量')
plt.title('辣椒类')
plt.show()

#  花叶类散点图
pricing_leafy = pricing_data['花叶类']
sales_leafy = sales_data['花叶类']
plt.scatter(pricing_leafy, sales_leafy)
plt.xlabel('定价')
plt.ylabel('销量')
plt.title('花叶类')
plt.show()

#  花菜类散点图
pricing_cauliflower = pricing_data['花菜类']
sales_cauliflower = sales_data['花菜类']
plt.scatter(pricing_cauliflower, sales_cauliflower)
plt.xlabel('定价')
plt.ylabel('销量')
plt.title('花菜类')
plt.show()
```

处理进货价

```
import pandas as pd

# 读取 Excel 数据到 DataFrame
df = pd.read_excel("进货价.xlsx")

# 按日期和单品分组，并计算批发价格的均值
result = df.groupby(["日期", "单品"])["批发价格(元/千克)"].mean()

# 按日期和分类名称分组，并计算批发价格的均值
result = df.groupby(["日期", "分类名称"])["批发价格(元/千克)"].mean().unstack()

# 将结果保存到新的 Excel 文件
result.to_excel("品类的进货价.xlsx", index=True)

# 将结果转换为 DataFrame 格式
result_df = result.reset_index()
result_df.to_excel("处理后的进货价.xlsx", index=True)
```

加权平均

```
import pandas as pd

# 读取单品销售数据
product_sales = pd.read_excel('完整数据 2.xlsx')
# 读取品类总销售量数据
category_sales = pd.read_excel('总销售.xlsx')

# 创建一个空的 DataFrame 用于保存每天每个品类的加权平均定价
weighted_prices = pd.DataFrame(columns=category_sales.columns[1:])
# 遍历每一天的销售数据
for date in category_sales['销售日期']:
    # 获取该天的品类销售量数据
    daily_sales = category_sales[category_sales['销售日期'] == date].iloc[:, 1:]
    # 获取该天的单品销售数据
    daily_products = product_sales[product_sales['销售日期'] == date]
    # 遍历每个品类
    for category in daily_sales.columns:
        # 获取该品类的销量
        category_sales_volume = daily_sales[category].values[0]
        # 获取该品类的单品数据
        category_products = daily_products[daily_products['品类'] == category]
        # 计算该品类单品的加权平均定价
```

```
        weighted_price = sum(
            category_products['销售单价(元/千克)'] * (category_products['销量(千克)'] /
category_sales_volume))
        #  添加该品类的加权平均定价到 DataFrame 中
        weighted_prices.loc[date, category] = weighted_price

#  将结果保存到 Excel 文档中
weighted_prices.to_excel('weighted_prices.xlsx', index_label='销售日期')
```

arima AIC

```
import sys
import os
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
import statsmodels.tsa.api as smt
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.arima_model import ARIMA
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.graphics.api import qqplot
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pylab as plt
from matplotlib.pylab import style

style.use('ggplot')
from arch.unitroot import ADF
import warnings

warnings.filterwarnings("ignore")
pd.set_option('display.float_format', lambda x: '%.5f' % x)
np.set_printoptions(precision=5, suppress=True)
pd.set_option('display.max_columns', 100)
pd.set_option('display.max_rows', 100)
import datetime as dt
import seaborn as sns

plt.rcParams['font.sans-serif'] = ['SimHei'] #  设置中文字体
plt.rcParams['axes.unicode_minus'] = False #  解决负号显示问题

dataframe = pd.read_excel('品类销量按天汇总.xlsx')
# dataframe=dataframe[0:19]
```

```
columns = dataframe.columns
data = dataframe.set_index('销售日期')
data.index = pd.to_datetime(data.index)


def choose(df, Ddf, col):
    df = df.astype(float)
    #  定阶
   pmax = int(len(Ddf) / 10) #  一般阶数不超过 length/10
    qmax = int(len(Ddf) / 10) #  一般阶数不超过 length/10
    bic_matrix = [] # bic 矩阵
   for p in range(pmax + 1):
        tmp = []
        for q in range(qmax + 1):
            try: #  存在部分报错，所以用 try 来跳过报错。
                tmp.append(sm.tsa.arima.ARIMA(df, order=(p, 1, q)).fit().aic)
            except:
                 tmp.append(None)
        bic_matrix.append(tmp)
    bic_matrix = pd.DataFrame(bic_matrix) #  从中可以找出最小值
   p, q = bic_matrix.stack().idxmin() #  先用 stack 展平，然后用 idxmin 找出最小值位置。
   print(u'BIC 最小的 p 值和 q 值为：%s、%s' % (p, q))
    return


for col in columns[1:]:
    df_d1 = data.diff(1).dropna()
    # choose_model(data,col)
    choose(data[str(col)], df_d1[str(col)], col)
```

arima

```
import statsmodels.api as sm
import pandas as pd
from statsmodels.tsa.arima_model import ARIMA
from statsmodels.stats.diagnostic import acorr_ljungbox
import warnings
warnings.filterwarnings("ignore")


dataframe=pd.read_excel('品类销量按天汇总.xlsx')
dataframe=dataframe[936:1086]
columns = dataframe.columns
```

```
data = dataframe.set_index('销售日期')
data.index = pd.to_datetime(data.index)
P=[1, 3, 3, 3, 4, 1]
Q=[6, 0, 1, 0, 2, 1]

def AR(data,p,q):
    model = sm.tsa.arima.ARIMA(data, order=(p,1,q))
    result = model.fit()
    print(result.summary()) #给出模型报告
   yc=result.forecast(7)#作为期 7 天的预测
   print("作为期 7 天的预测:")
    print(yc)
    cc=result.resid
    #白噪声检验
   print(u'残差序列的白噪声检验结果为：', acorr_ljungbox(cc, lags=1)) #返回统计量和 p 值
   return yc

def score(X,Y):
    return

for i in range(1,7):
    col=columns[i]
    p=P[i-1]
    q=Q[i-1]
    print(str(col))
    AR(data[str(col)],p,q)
```

背包问题

```
import numpy as np
import pandas as pd
import math
data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='水生根茎类']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 7#单品种类
def comlete_1(n,m):
        dp = np.zeros((n+1,m+1))
        for i in range(1,n+1):
                for j in range(1,m+1):
                        dp[i][j] = dp[i - 1][j]
```

```
                if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                    dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("水生根茎类")
comlete_1(n,m)

data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='食用菌']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 5#单品种类
def comlete_1(n,m):
    dp = np.zeros((n+1,m+1))
    for i in range(1,n+1):
        for j in range(1,m+1):
            dp[i][j] = dp[i - 1][j]
            if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("食用菌")
comlete_1(n,m)

data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='茄类']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 5#单品种类
def comlete_1(n,m):
```

```
    dp = np.zeros((n+1,m+1))
    for i in range(1,n+1):
        for j in range(1,m+1):
            dp[i][j] = dp[i - 1][j]
            if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("茄类")
comlete_1(n,m)

data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='辣椒类']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 9#单品种类
def comlete_1(n,m):
    dp = np.zeros((n+1,m+1))
    for i in range(1,n+1):
        for j in range(1,m+1):
            dp[i][j] = dp[i - 1][j]
            if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("辣椒类")
comlete_1(n,m)

data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='花叶类']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
```

```
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 5#单品种类
def comlete_1(n,m):
    dp = np.zeros((n+1,m+1))
    for i in range(1,n+1):
        for j in range(1,m+1):
            dp[i][j] = dp[i - 1][j]
            if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("花叶类")
comlete_1(n,m)

data=pd.read_excel('第三问单品利润及销量.xlsx')
data=data[data['分类名称']=='花菜类']
w = list(map(math.ceil, data['平均销量'].values))#单品销量
v = list(map(math.ceil, data['平均利润'].values))#单品利润
m = int(data['类别需求量'].values[0])#背包总载重
n = 2#单品种类
def comlete_1(n,m):
    dp = np.zeros((n+1,m+1))
    for i in range(1,n+1):
        for j in range(1,m+1):
            dp[i][j] = dp[i - 1][j]
            if j >= w[i-1] and dp[i][j] < dp[i-1][j-w[i-1]]+v[i-1]:
                dp[i][j] = dp[i-1][j - w[i-1]] + v[i-1]
    j = m
    x = [False for i in range(n)]
    for i in range(n,0,-1):
        if dp[i][j]>dp[i-1][j]:
            j-=w[i-1]
            x[i - 1] = True
            print('i=',i+1)#记录选择的物品
    return 0
print("花菜类")
comlete_1(n,m)
```