生鲜商超最佳销售决策的优化分析

摘　要

生鲜商超作为冷链物流供应链上不可或缺的一环。它所做出的销售决策对于农副市场有着举足轻重的地位。然而传统私营商超往往存在着销售信息不对称的问题：脱离市场分析的补货方案造成新鲜蔬菜供大于求被迫打折，“菜贱伤农”；忽略销售流水的随意定价决策导致原本供不应求的热销蔬菜一再抬价，“菜贵伤民”。作为**产供销一体化**链条中的一环，基于现有销售流水数据建立合适的数学模型实现市场需求分析意义重大。本文旨在分析商超现有数据实现补货和定价的双重优化，结合实际考虑打破信息壁垒亟待给出的相关数据，打破农产品供需不平衡局面。

首先对附件中的数据进行预处理。基于 VLOOKUP 函数将附件 1 商品的基本信息与后续批发价、损耗率合并同类项，使用滑动平均对附件 2 的大量销售流水数据进行插补得出最终汇总表格；通过 Max-min 标准化法处理数据，以成本加成定价的方式确定实时预期标价，由此实现销售明细与蔬菜单品的信息捆绑。

针对问题一，利用蔬菜品类及单品汇总信息，在不考虑退货量的前提下得到近 3 年来总销售量的数据，并使用 Python 代码进行图片可视化，通过对比图分析不同品类销售排行；除去品类上的横向比较，分布规律的研究也考虑到了蔬菜的时序分布，利用 Matlab 绘制蔬菜品类在不同月份的销量变化，并在此基础上建立三维散点图反映其时序曲线的相关关系。相互关系方面，对于六大类蔬菜品类，通过 **Pearson 相关分析**，对总体数据求取协方差得到 Pearson 相关系数，经检验，各品类数据均近似正态分布。

针对问题二，选用**多项式逼近**方法，基于寻优参数 R^2 通过 EXCEL 对数据进行多种基本类型的拟合，以多项式函数拟合历史销售数据，通过 Matlab 直观地给出各类蔬菜拟合曲线，得出销售总量与成本加成定价之间的模式；采用**时间预测** ARIMA 模型，基于蔬菜在供应端和需求端都与时间具有强烈规律性，通过对过往大量数据的分析得出销售量与时间的规律性，从而预测未来一周各大类蔬菜的预计销售量；基于蔬菜在销售中的需求弹性曲线，探究其需求价格在现实经济生活中表现出的一定鲁棒性，从而预计销售量并作为日补货总量进行进货，然后综合多方面因素考虑制定定价策略。

针对问题三，依据引入的全新变量“单位利润”评判各蔬菜单品参与补货的价值，利用 c++代码将过去一周数据取均值作为预测销售量，通过**贪心算法**实现对多达 251 种蔬菜单品的分布式规划，保证单位利润的情况下优先预测销售量更大的商品以避免商品滞销。

针对问题四，不再局限于附件获取信息，通过分析前三问模型的局限性提取出可供参考的数据信息，相较于商超的原始分拣单数据，多层面的考虑更利于模型的构建和准确预测。

关键词：产供销一体化；Pearson 相关分析；多项式逼近；时间预测；贪心算法

一、 问题重述

1.1 问题背景

生鲜蔬菜主要包括花叶类、花菜类、水生根茎类、茄类、辣椒类和食用菌等，在当前市场经济环境下，产品的营销模式发生了极大的转变，传统门店对于补货和定价决策的不合理规划往往导致了资源冗余及浪费[1]。正基于此，利用数据建模优化生鲜商超补货策略，确定未来蔬菜品类的补货数量和预估定价，并结合近期蔬菜销售流水明细得出短期内的单品进货策略将有助于提高供应链上商户的决策能力，在信息交流的基础上，最大限度地发挥生鲜蔬菜供应链的效益，从而增强生鲜商超的市场竞争力。

1.2 问题提出

问题一：根据附件 2 提供的各商品销售流水明细，我们需要对蔬菜各品类的总销售量进行概括和分析。通过表格数据处理将不同蔬菜品类按销售量降序排列，从而了解各品类销售量的分布情况，知道哪些品类销售量较高，哪些品类销售量较低。

同时，结合附件 3 中的月度销售流水数据，可以建立一个时序表格在纵轴记录各个蔬菜品类的销售量。这样可以分析蔬菜品类在时间上的分布规律，判断销售量是否存在季节性的变化趋势。

对于蔬菜的 251 种单品，也可以采用类似的方法建立表格，按销售量降序排列，并考虑时间上的分布规律。然而，需要注意的是，单纯从品类或单品的角度可能无法建立起自变量和因变量的明确关联函数。蔬菜品类与销售量的大小不一定具有明显的逻辑相关性。因此，我们可以分析蔬菜品类与销售量随月份变化的关联性，并进行相关性检验，以确定销售量与时间的关系是否具有统计显著性。

问题二：考虑商超以蔬菜的品类为单位进行补货计划和定价决策，我们的目标是分析不同蔬菜品类的销售总量与成本加成定价之间的关系，以便在未来一周（2023 年 7 月 1 日至 7 月 7 日）制定最优的日补货总量和定价策略，从而最大化商超的收益。通过建立适用于各蔬菜品类的数学模型，包括多项式逼近和 ARIMA 时间序列分析，我们旨在为商超提供基于数据的决策支持，以实现销售优化和成本R^2控制。在此背景下，我们将研究蔬菜品类的销售趋势、定价策略与销售总量之间的相互关系，以确定最佳的补货计划和定价策略，以最大程度地提高商超的经济效益。

问题三：根据附件 2、3、4 提供的以往商品数据，我们需要筛选得到 2023 年 6 月 24-30 日的数据，并对筛选出来的数据进行一定的分析和处理。通过 python 代码数据处理，将销售流水明细根据单品类名统计数量，借助公式计算单位利润的平均值，并保留成本加成定价平均值方便后续计算。

同时，为了更好地计算单品补货量和定价策略的最优解，考虑到整体最优解可以通过局部最优解实现，建立起基于贪心算法的预测模型。

问题四：在前三问的研究基础上，基于商超决策者出发的补货和定价决策趋于成熟，问题四的提出便是跳出原有的商超视角考虑外部因素带来的影响，旨在摆脱理想数学模型假设，更多地结合时代考虑传统商超在大数据时代生存的支撑。

二、 问题分析

2.1 问题一的分析

为了分析蔬菜各品类及单品销售量的分布规律及相互关系，我们可以将这两个数据集合并，创建一个数据框，以便同时查看每个蔬菜单品的销售情况和其所属的品类。考虑销售量与其他变量的逻辑关系时，我们仅从附件 1 和附件 2 出发，只考虑品类与销量而不考虑价格，在这种前提下，因运损和品相变差导致的打折商品与正常商品计算无异。

对于分布规律的总结，我们尽可能使用柱状图、折线图、扇形图进行对比分析，将蔬菜六大品类和主要的单品销量进行可视化呈现，并通过 Python 代码在 vscode 平台复现各品类随月份变化的时序图，得出其时间分布规律。

对于相互关系的提取，我们基于相关系数进行判断，显然给出的日销售流水数据在 3 年的庞大基数下近似连续，且在时序图的辅助下我们可以轻松的判断其线性关系，因此最为要紧的是对蔬菜品类销量和蔬菜单品销量分布进行正态性检验，满足正态分布的数据可以采用效率更高的 Pearson 相关系数，基于数据庞大的单品销量分布则更适宜采用 Spearman 相关系数分析。

2.2 问题二的分析

在生鲜商超中，蔬菜类商品的销售与定价策略密切相关。本研究旨在分析销售总量与成本加成定价之间的关系，并通过时间序列分析预测未来一周的日补货总量。通过多项式逼近和 ARIMA 模型，我们建立了适合不同蔬菜品类的销售总量与成本加成定价的数学模型，进一步提供了未来一周的补货计划。本研究结果为商超的补货和定价决策提供了重要参考。同时根据题干信息，蔬菜的供应品种在 4 月至 10 月较为丰富，同时蔬菜的损耗率与其周转情况是对应的，周转越快，销量越大，损耗率则越低，因此需要制定更符合市场需求以及收益的定价策略。

2.3 问题三的分析

为了探究 7 月 1 日的单品补货量和定价策略,我们首先要对 2023 年 6 月 24-30 日的可售品种进行分析。我们引入一个新的变量“单位利润”，来表示单位质量单品可以获得的利润大小。将 6 月 24-30 日单位利润的平均值作为评价各品类蔬菜 7 月 1 日单位利润的标准，由此判断各个商品是否值得进货。

对于单品补货量和定价策略最优解的求解，我们借助贪心算法，运用 c++代码，在单位利润更大的前提下，优先考虑预测销售量大于 2.5 的商品，以避免销售不出去的情况。

2.4 问题四的分析

仅从商超本身销售数据分析的优化策略显然是不全面的，针对生鲜蔬菜的补货方案，我们更多需要从蔬菜品类方面获取更多数据，如客户对于蔬菜产品的反馈情况，这将为商超提供蔬菜品类的进货参考；针对蔬菜产品的定价策略，我们则应该获取整体市场方面的销售数据，如对于进货商品牌与客户的关系，产品的品牌效益也是决定价格定价一大因素。

三、 模型假设

1. 假设商超对于生鲜蔬菜品相有相对严苛的判断标准，对于蔬菜单品的实时品相有较为灵敏的价格调整策略，即面对运损和品相变差的商品

2. 假设附件 1 涵盖的所有商品均保持独立出售，各蔬菜单品间不存在捆绑销售的情况，同时各个蔬菜摆放位置没有明显的区别，均处在同等的销售空间出售

3. 假设附件 4 所涉及的蔬菜类商品近期损耗率具有一般性，针对我们研究课题中的未来一周具有潜在的普遍规律，同时我们认为这种普遍规律在过去的 3 年任何一个月份不会有过大的偏差。

4. 假设蔬菜销售不会长期受到特殊环境影响，例如由于供应量、天气条件或节假日等原因，某些蔬菜在特定时期的销售量可能会呈现周期性的波动。这是因为蔬菜的供应、品质和消费需求都会随着季节的变化而发生变化，从而影响其销售情况。

5. 假设蔬菜商品陈列状况长期不变，商超保证每个商品最小陈列量为 2.5 千克，且蔬菜类商品陈列位置和陈列空间在一段时间内保持不变。

四、 符号说明

符号	说明
σ_x,σ_y	用于计算 Pearson 相关系数的标准差
$\rho_{x,y}$	表示适用具有正态性变量之间相关性的 Pearson 相关系数
R_i,S_i	分别代表第i个 x 值，y 值的秩
θ	表示适用非线性和非正态分布的变量之间的相关性的 Spearman 相关系数
$E(a_0,a_1,a_2,...,a_n)$	求解最小二乘法的残差函数
Y_t	数据化为的时间序列
$Y_t^{'}$	差分后的时间序列
$\rho\ (x)$	两个时间点的观测值之间的相关性的自相关函数 ACF
$Cov(X_t,X_{t+k})$	时间点 t和时间点 $t+k$的观测值的协方差
$Var(X_t)$	时间序列X_t的方差
$\psi\ (x)$	当前时间点的观测值与某滞后观测值间时间相关性的偏自相关函数 PACF
(d)	表示进行多少次差分操作得到平稳序列的差分的阶数
(p)	描述了模型中使用的观测值的滞后值的的自回归阶数
(q)	描述了模型中使用的错误项的滞后值的移动平均阶数
ϕ_i	ARIMA 模型的自回归系数
θ_i	ARIMA 模型的移动平均系数
(e_t)	均值为常数，方差为常数的序列，且序列中不同时点的数据之间的协方差为 0 的白噪声误差

五、 数据预处理

在回答问题之前，由于单日蔬菜商品流水数据较杂，我们首先进行数据预处理，从而提高数据质量，减少异常值和错误对分析结果的干扰，确保分析的准确性和可靠性。

5.1 合并同类项

由于附件 1 提供了蔬菜的单品编码、单品名称、分类编码和分类名称，而附件 2 提供了销售日期、销售时间、单品编码、销售量、销售单价、销售类型以及是否打折销售的信息。为了综合考虑蔬菜的单品信息和销售信息，我们既需要得到单品编码与销售价格的对应关系，也不得不依靠附件 1 中蔬菜品类的商品信息对应寻找不同蔬菜品类对应的销售流水。显然，自 2020 年 7 月 1 日起累计 3 年的数据并不适宜同时将两表的数据直接代入 Python 求解。在合并数据集时，我们优先根据单品编码进行匹配。可以使用编程语言 Python 中的合并函数 merge、join 来实现初步合并。（文件名 Connected.py，详见代码合并附件）使用转化工具将得到的数据集汇总为表格，通过 EXCEL 自带的 VLOOKUP 函数外加 COUNTIF 功能实现对于各品类蔬菜某时段销售量的查找，具体公式如下：

单品类：=VLOOKUP(C2,[附件 1.xlsx]Sheet1!A$2:D$252,2,FALSE)

蔬菜品类: =VLOOKUP(C2,[附件 1.xlsx]Sheet1!A$2:D$252,4,FALSE)

图 1VLOOKUP 函数操作演示

有了 VLOOKUP 的加持，我们建立起一个庞大的蔬菜单品信息汇总表，为了通过销售流水进一步计算蔬菜单品单批次所获取净利润，我们结合附件 3 同步获取售出蔬菜单品的批发单价，横向的比较也便于后续可视化的进行。

批发价格: =VLOOKUP(C2,[附件 3.xlsx]Sheet1!B$2:C$55983,2,FALSE)

在先前的假设下，我们可以将附件 4 中的数据与商品编码进行横向比较。在假设商品的损耗率大体不变的情况下，我们可以将其与实时变化的商品销售额信息对应起来。这样一来，我们可以免去内部汇总表格查找所需要的时间成本，并简化代码的编写难度。通过在 Excel 表格中完成这个过程，我们可以复现成本加成定价策略。

单品损耗率: =VLOOKUP(H2,[附件 4.xlsx]Sheet1!B$2:CS252,2,FALSE)

通过将附件 1 和附件 2 合并，我们可以获得一个包含完整单品信息和销售信息的数据集，为后续的蔬菜销售量分布规律和相互关系分析提供更全面和准确的数据基础。

5.2 数据标准化

数据标准化是在数据分析和建模过程中常用的一种预处理技术，用于将不同尺度和范围的数据转化为具有统一标准的数据，以消除不同变量之间的量纲影响，便于比较和分析。在本研究中，销售流水数据是关于生鲜蔬菜在不同时段的销售情况，我们将对附件二中给出的生鲜蔬菜销售流水明细数据进行数据标准化处理，以便更好地分析蔬菜各品类及单品之间的销售量分布规律和相互关系。我们将使用 Min-max 标准化和 z-score 标准化两种方法对数据进行预处理。

1. Min-max 标准化：

Step 1.找出销售流水数据中的最小值（min）和最大值（max）。

Step 2.对于每个数据点，使用下述公式进行标准化：

$$x_{ij}^{'} = \frac{x_{ij} - \min x_{ij}}{\max x_{ij} - \min x_{ij}}$$

Step 3.标准化后的数据的取值范围在 0 到 1 之间，数据越接近 1 表示销售流水较高，越接近 0 表示销售流水较低。

我们将对销售流水数据进行转换，使其在最小值和最大值之间映射。这样可以确保所有销售流水数据都在相同的范围内，并且保留了数据的原始分布特征。通过这种标准化方法，我们可以比较不同时间段和不同品类的蔬菜销售流水，并揭示它们之间的差异和趋势。

2. Z-score 标准化：

Step 1.计算销售流水数据的均值（mean）和标准差（standard deviation）。

Step 2.对于每个数据点，使用下述公式进行标准化：

$$x_{i}^{'} = \frac{X_i - \overline{X}}{\sigma}$$

Step 3.标准化后的数据会围绕均值为 0，标准差为 1 的分布，正值表示销售流水高于平均水平，负值表示销售流水低于平均水平。

我们将对销售流水数据进行转换，使其符合标准正态分布。这样可以消除不同品类和单品之间的尺度差异和偏差，并使得数据均值为 0，标准差为 1。通过这种标准化方法，我们可以比较不同销售流水数据与平均水平的偏离程度，进一步揭示销售量的高低、变化和波动情况。

借助 python 代码对销售流水数据进行 Min-max 标准化和 z-score 标准化处理（文件名 pre-handle.py，详见），我们将得到标准化后的数据，使其更适合用于分析不同品类和单品之间的销售量分布规律和相互关系。这些标准化后的数据将为我们提供更准确、可比较的指标，以及更深入的市场需求分析，进而为补货决策和定价决策提供有力的支持和参考。

5.3 蔬菜各品类销售量时序图

已知六大蔬菜品类在一年不同月份的累计销量图，在承认季节性变化给销量带来影响的前提下，我们利用 Matlab 拟合出品类与时间的变化曲线。（文件名 tu.m，详见 Matlab 绘图附件）值得注意的是，仅仅考虑销量的时序变化不需要与退货量的负值进行加和。

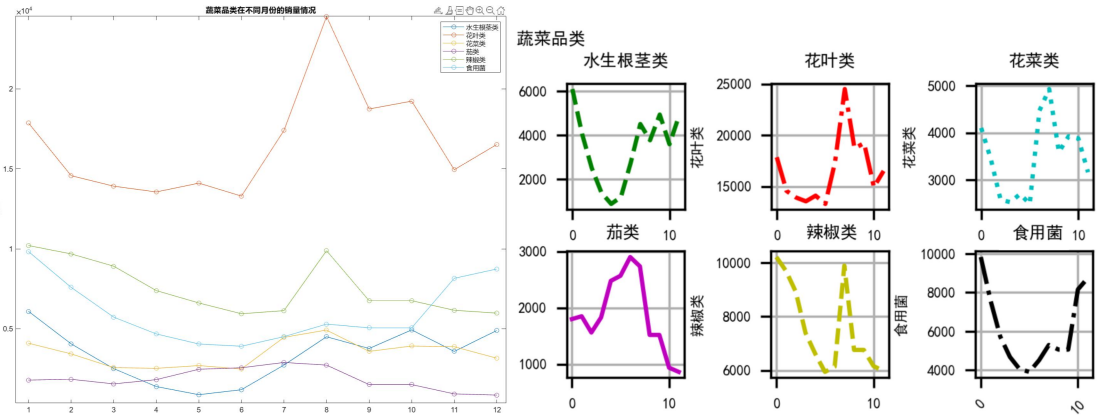

图 2 各蔬菜品类销量对比图（左）及对应曲线图（右）

可以观察到，在蔬菜供应品种繁多的 4 月至 10 月期间，不同蔬菜品类的销量差距仍然相当明显。例如，花叶类蔬菜无论是在蔬菜供应旺季还是淡季都保持较好的销量。而一些蔬菜，比如茄类，具有明显的季节性，人们更倾向于在气温回暖的春夏季节购买。这种现象可能与人们对于季节性蔬菜的偏好有关。

我们也可以看出一年中最炎热的 8 月份是人们购买生鲜蔬菜的高峰期。其中的原因可能在于高温天气不便于蔬菜的储存，大部分购买的蔬菜都会在当天销售以供应顾客，同时夏季也是消费者更加注重摄取食物中水分的时候，所以生鲜蔬菜成为当仁不让的选择。

另外，一年中的 1 月份也是蔬菜销量的第二个高峰期，这与我们之前提到的节假日春节息息相关。在这个时期，人们在准备和庆祝春节时会购买大量的蔬菜，从而推动了销量的增长。由此我们可以看出蔬菜销量在不同季节和特定节假日中存在着一定的差异，这为我们此后关于时序分布的研究提供了可视化依据。

六、 模型的建立与求解

6.1 蔬菜不同品类或单品间的关联度分析

6.1.1 关联度分析求解思路

关联度分析需要完成以下步骤，来说明蔬菜品类/单品间的关联性：

1. 销量分布：

- 将附件 2 中的销售流水明细按照品类进行汇总和分析。创建一个表格，将不同蔬菜品类按销售量降序排列，以了解各品类销售量的分布情况，知道哪些品类销售量较高，哪些品类销售量较低。

- 结合月度销售流水数据，建立一个时序表格，以月份为横轴，在纵轴上记录各个蔬菜品类的销售量。通过分析蔬菜品类在时间上的分布规律，可以确定销售量是否存在季节性的变化趋势。

2. 相互关系：

- 对于蔬菜品类，分析其与销售量随月份变化的关联性。可以绘制蔬菜品类与销售量的散点图，并计算 Pearson 相关系数，以确定销售量与时间的关系是否具有统计显著性。

- 对于蔬菜单品，同样可以采用类似的方法，建立表格，按销售量降序排列，并考虑时间上的分布规律，然后计算单品间的 Spearman 相关系数，以分析不同蔬菜单品间的销量关联度。

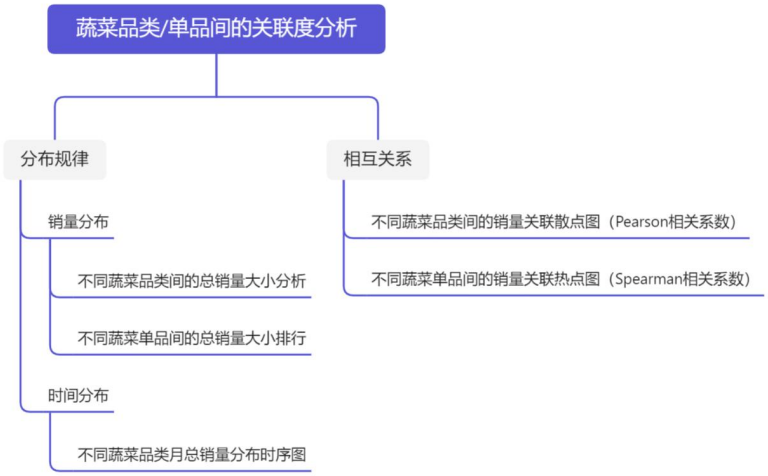

图 3 关联度分析示意

6.1.2 分布规律

在完成数据预处理后，我们可以首先对蔬菜各品类及销售量的分布规律进行探究。

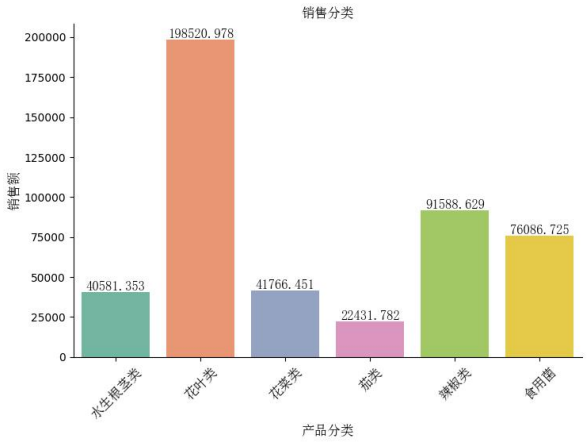

图 4 销售量分布-蔬菜品类

从先前得到蔬菜各品类销售量汇总表中，我们可以简单地通过 Python 绘制销售量分布柱状图（文件名 histogram.py，详见柱状图绘图附件）。显然在过去三年内，蔬菜的销售量各品类之间存在明显的差异。“花叶类”品类的销售量最高，销售量远超其他各类，达 198520.978 千克，显示出较高的市场需求和消费者偏好。其次是“辣椒类”和“食用菌”，两者销售量接近 78,000 千克，表明这两个品类在市场上具有一定的竞争力和稳定的销售表现。“水生根茎类”、"花菜类"和"茄类"的销售量相对较低。在这些品类中，"水生根茎类"的销售量最低，为 40,581.353 千克；"茄类"的销售量为 22,431.782 千克；"花菜类"的销售量为 41,766 千克，这可能意味着这些品类在市场上的需求相对较低或者面临其他竞争品类的挤压。

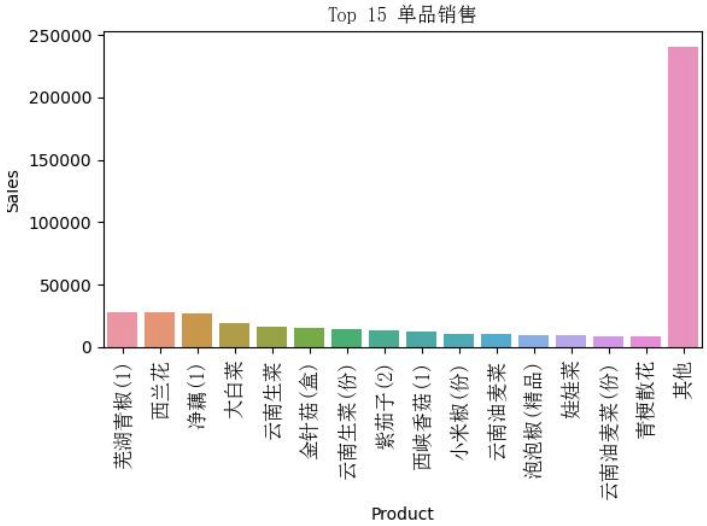

图 5 单品销量分布-TOP15

根据提供的图表数据，可以看出在按商品分类排序的销售量中，芜湖青椒（1）、西兰花和净藕（1）是销售量排名前三的产品。令人意外的是，即使花叶类蔬菜的销量高居第一，其涵盖的 100 类单品并没有在销量上占据明显的优势，由此体现消费者对于花叶类蔬菜的选取上没有特别的偏好，而是更看重花叶类普遍的营养价值和日常摄入。

我们判断辣椒类的芜湖青椒（1）作为销量最高的单品一定具有突出于其他辣椒类单品的口感，对于我们所研究的这家商超所在地，当地的人们口味更加倾向于芜湖青椒（1）所补充的辣度，使得这一类辣椒在当地的菜品制作上起到百搭的效果，成为居民日用的一种调料品；西兰花和净藕（1）分别是花菜类和水生根茎类的代表，即便这两种蔬菜品类总销量较少，这两类单品的销售量不容忽视，由此可见消费者对于花菜类和水生根茎类蔬菜有着明显的偏好倾向，人们已经形成了较为明显的消费习惯，在该类商品的选择上不会有太大的变动[2]。

有了前 15 位高销量蔬菜单品的表格数据，我们已经提炼出如上的分布规律信息，为了服务于后续多达 27-33 种单品的进货策略，我们同样使用 Python 代码按照销售量大小排列出前 25 种蔬菜单品的扇形分布图（文件名 fanshaped.py，详见扇形图绘制附件）。

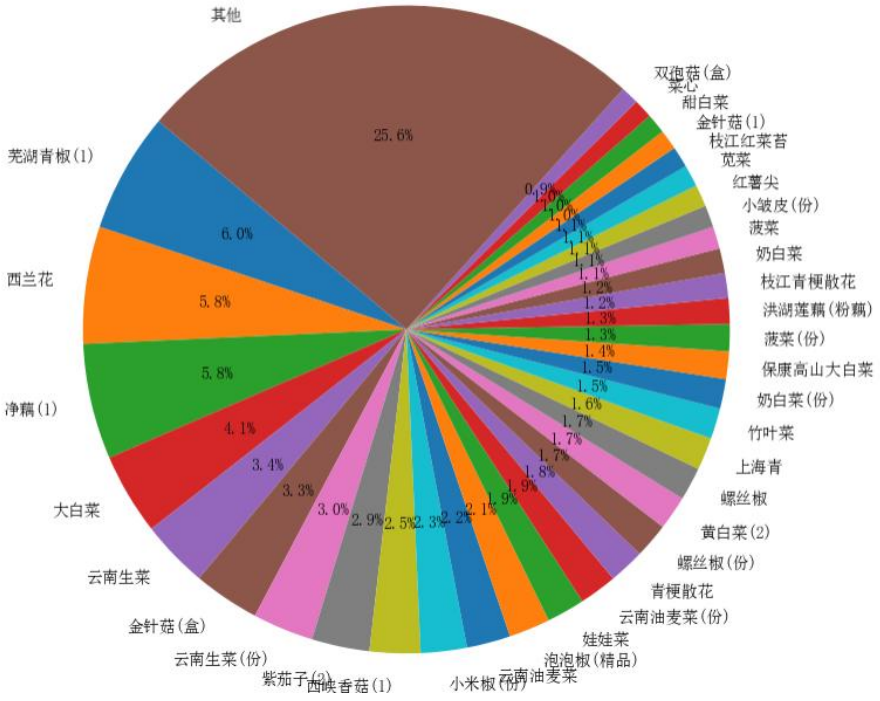

图 6 单品销售分布-TOP25

有了数据层面的扩展，我们改进并得出新的结论，在前 25 种单品中，包含单品最多的花叶类依旧占据了 9 种，而辣椒类单品有足足 6 种销量可观，体现出人们在调味品的选择方面更注重菜品的需求以及个人的口味。除此之外，包含单品较少的食用菌和花菜类分布有 4 种和 3 种单品销量靠前，在市场供货品种较少的情况下，这两类蔬菜的需求量保存着较高的水准，当然这可能也与市面上其他同类单品的稀缺有关。

6.1.3 相互关系

在统计学中，相关分析主要用于评估线性变量之间的相关程度[3]。通过计算相关系数等统计指标，可以确定相关关系的密切程度和线性相关性的方向。

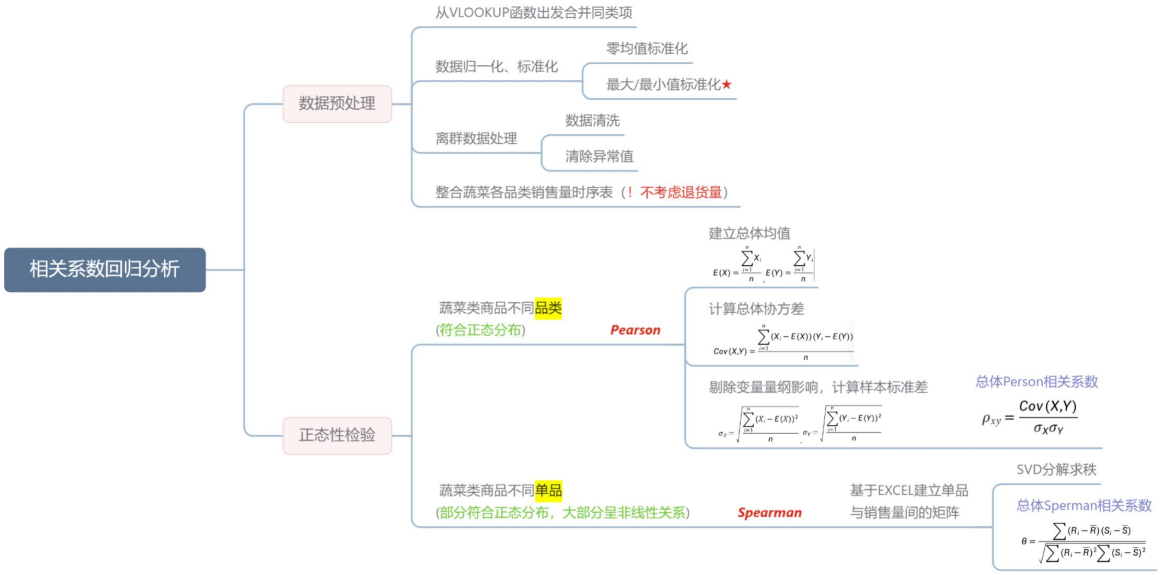

图 7 相关系数模型示意

对于蔬菜类商品品类的研究，我们需要先对随机变量 X 进行正态性检验，文中即

蔬菜类商品的不同品类，通过 Python 代码针对不同品类间的关联曲线逐一计算其对应的 p 值，各变量之间关联曲线的正态性检验结果在附录中体现（文件名 normality.txt，详见正态性检验附件）。

符合正态：Pearson 相关系数

用于度量两个变量 X 和 Y 之间的相关性（线性相关），其值介于-1 与 1 之间。系数的值为 1 意味着 X 和 Y 可以很好的由直线方程来描述，所有的数据点都很好的落在一条直线上，且 Y 随着 X 的增加而增加

我们首先将六个蔬菜品类过去总销售量一一计入两组数据：

$$X:\{X_1,X_2,\cdots,X_n\}\text{和}Y:\{Y_1,Y_2,\cdots,Y_n\}\text{（其中 n=1，2，…，6）}$$

这些数据共同作为相关系数分析的总体数据，这涵盖了商超所能补货的蔬菜品类的全部。

那么总体均值：$E(X)=\dfrac{\sum_{i=1}^{n}X_i}{n}$，$E(Y)=\dfrac{\sum_{i=1}^{n}Y_i}{n}$

由于总体样布仅有 6 个，对于标准差的计算我们可以直接依托公式得到：

$$\sigma_X=\sqrt{\frac{\sum_{i=1}^{n}(X_i-E(X))^2}{n}},\quad \sigma_Y=\sqrt{\frac{\sum_{i=1}^{n}(Y_i-E(Y))^2}{n}}$$

总体协方差指蔬菜六大品类两两之间的协方差，它可以告诉我们这两个变量的变化趋势是否一致。

总体协方差：$Cov(X,Y)=\dfrac{\sum_{i=1}^{n}(X_i-E(X))(Y_i-E(Y))}{n}$

有了前面公式的铺垫，我们可以利用总体数据得到皮尔逊相关系数，评估两个变量之间的线性相关程度，从而判断它们的关系强度和方向。

总体 Person 相关系数：$\rho_{xy}=\dfrac{Cov(X,Y)}{\sigma_X\sigma_Y}$

皮尔逊相关系数的范围在-1 到 1 之间，当系数接近 1 时，表示两个变量具有强正相关关系；当系数接近-1 时，表示两个变量具有强负相关关系；当系数接近 0 时，表示两个变量无线性相关关系。

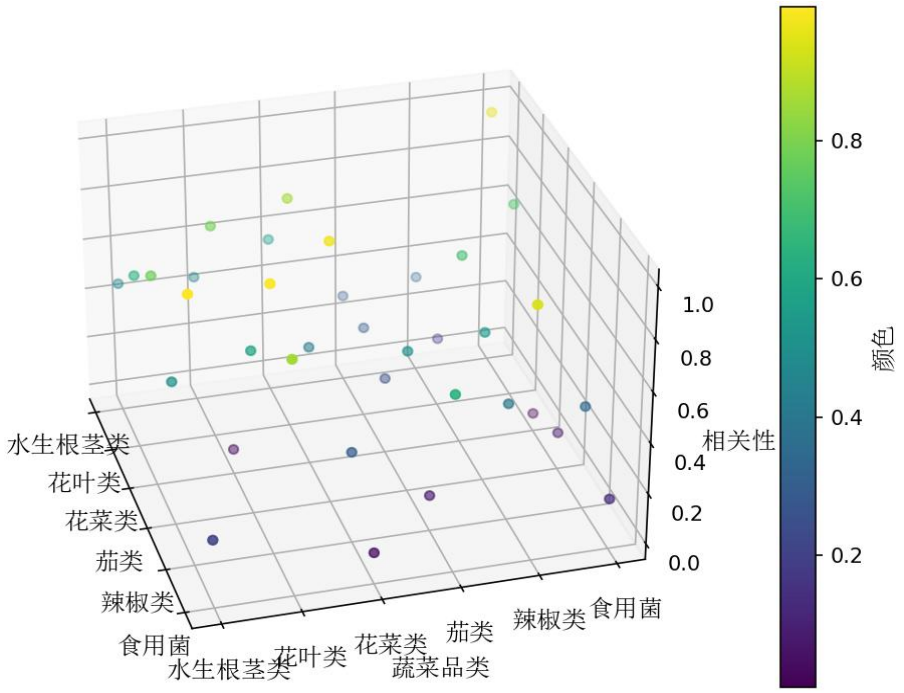

图 8 蔬菜各品类相关关系的三维散点图

通过 Python 读取相关系数图表（文件名 kind.py，详见三维散点图附件），我们使用三维散点图进行可视化处理，由此可观直观地展示蔬菜各品类之间的相关关系。通过将不同品类的数据点在三维坐标系中表示出来，我们可以更清楚地看到它们的分布模式和趋势，从而发现可能存在的相关模式。例如“花菜类”和“花叶类”存在较高的正相关性，即花菜类销售量增加的月份，消费者对于花叶类蔬菜的选取热情也同样增加了，此时作为商超的决策者则可以考虑将它们放在一起销售或联合推广，以提高销售效果。

非线性关系：Spearman 相关系数

斯皮尔曼等级相关分析又称为“等级差数法”，以分析变量之间相关关系的依据为等级资料，适用于数据为非线性或者非正态分布的情况。当其绝对值越接近 1，相关性越强其具体公式如下:

$$\theta = \frac{\sum (R_i - \overline{R})(S_i - \overline{S})}{\sqrt{\sum (R_i - \overline{R})^2 \sum (S_i - \overline{S})^2}}$$

该公式中，R_i, S_i分别代表第i个 x 值，y 值的秩。两个变量的平均值则分别用$\overline{R}, \overline{S}$来表示。针对 251 种不同的单品总体，我们选择抽取样本的方式分析茄类各单品的相关系数，以下是使用 Matlab 绘制的相关度热力图（文件名 Spearman.m，斯皮尔曼相关系数热点图绘制）：

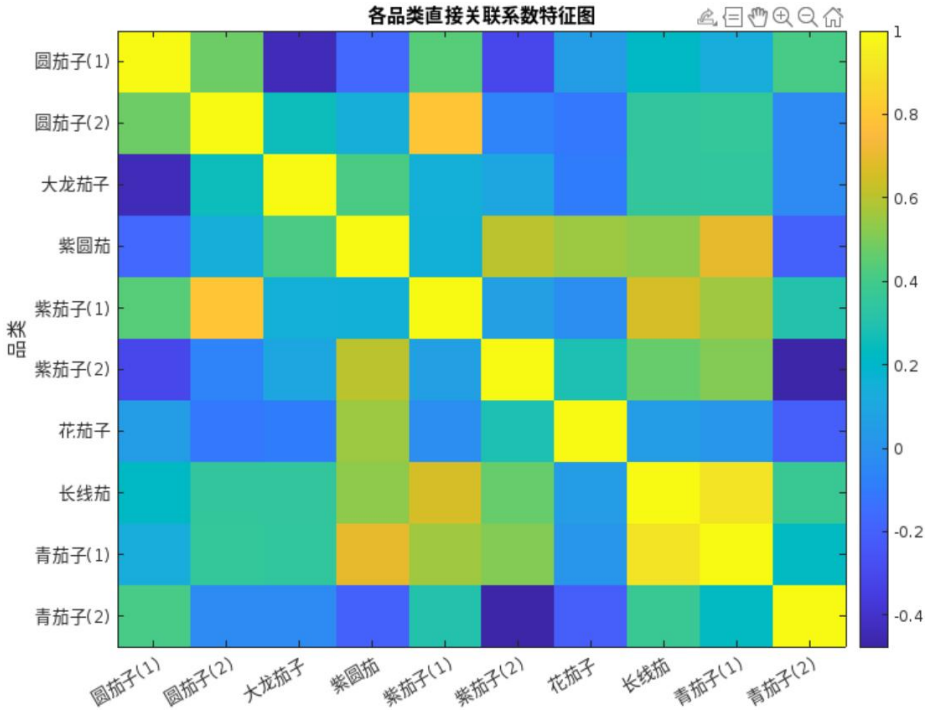

图 9 蔬菜不同单品间的相关系数热力图

同样针对各单品的相关系数我们可以得到两两之间的相互关系，如“青茄子”和长线茄就具有近似于 1 的关联系数，两种单品在外形和口感的近似可能在一定程度上诱导了顾客选择作为替代品，商超的决策者也可以考虑将它们捆绑销售，推出相关套餐等营销方式。

6.2 问题二模型的建立与求解

在缺乏市场数据的情况下，商超可以利用先前的分拣单原始数据，根据下列成本加成定价公式确定最终价格：

$$\text{成本加成定价} = \text{单位成本} \times \ (1 + \text{利润率})$$

确定预估定价，我们通过对数据进行拟合来探究销售总量与成本加成定价的关系，对于每个蔬菜品类，多项式逼近可以使用多项式函数来拟合实验数据或观测数据的方法，通过找到一个多项式函数，使其与数据点之间的误差最小。这种方法在回归分析和数据拟合中广泛应用。

多项式逼近的一般形式如下：

$$[P(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n]$$

其中，$(P(x))$ 是多项式函数，(x)是自变量，$(a_0,a_1,a_2,...,a_n)$ 是待定的系数，(n) 是多项式的次数（阶数）。

为了找到最佳的多项式系数 $(a_0,a_1,a_2,...,a_n)$，我们需要对上述**最小二乘法**公式进行求导并令导数等于零。这将导致一个系数方程组，可以使用线性代数方法求解。具体步骤如下：

Step 1.构建误差函数（残差函数）：

$$[E(a_0,a_1,a_2,...,a_n) = \sum_{i=1}^{N}(P(x_i) - y_i)^2]$$

Step 2.对误差函数分别对 $(a_0,a_1,a_2,...,a_n)$ 求偏导数：

$$[\frac{\partial E}{\partial a_0},\frac{\partial E}{\partial a_1},\frac{\partial E}{\partial a_2},...,\frac{\partial E}{\partial a_n}]$$

Step 3.令所有偏导数等于零，得到系数方程组：

$$[\frac{\partial E}{\partial a_0}=0,\frac{\partial E}{\partial a_1}=0,\frac{\partial E}{\partial a_2}=0,...,\frac{\partial E}{\partial a_n}=0]$$

Step 4.求解系数方程组，得到最佳的多项式系数 $(a_0,a_1,a_2,...,a_n)$。

Step 5.分析得到各蔬菜品类销售总量与成本加成定价的最终拟合函数：

花菜类销售总量

$P(X)=-1148.809+936.667x_2+43.523x_3-1.909x_4-0.31x_5+0.054x_6-0.004x_7$

花叶类销售总量

$P(X)=-19.553+39.311x-29.205x_2+11.396x_3-2.633x_4+0.379x_5-0.034x_6+0.002x_7$

辣椒类销售总量

$P(X)=-5.126+9.512x-5.49x_2+1.577x_3-0.257x4+0.025x5-0.002x6$

茄类销售总量

$P(X)=3681.525-3838.06x+1746.925x_2-455.298x_3+74.853x_4-8.049x_5+0.566x_6-0.025x_7+0.001x_8$

食用菌销售总量

$P(X)=0.917+0.17x-0.131x_2+0.051x_3-0.011x_4+0.001x_5$

水生根茎销售总量

$P(X)=127.231-98.487x+32.405x_2-5.929x_3+0.666x_4-0.048x_5+0.002x_6$

图 10 最终拟合函数汇总

各蔬菜品类未来一周(2023 年 7 月 1-7 日)的日补货总量

由于我们选取的时间序列预测法属于一种定量的回归预测，通过较为简单的思路快速得出未来顾客消费趋势，在缺乏外生变量的情况下，使用 ARIMA（自回归综合移动平均模型可以很多地处理季节性、趋势性、周期性的时间序列数据特性。生鲜蔬菜对季节时令敏感的特性恰恰与这一模型相匹配[4]。

Step 1.平稳性检验

ARIMA 模型要求时间序列是平稳的，即均值和方差在时间上保持恒定。平稳性是一个关键假设，因为 ARIMA 模型的基础是建立在平稳时间序列上的。我们使用单位根检验，ADF 检验的原假设是时间序列具有单位根，即不平稳。如果我们不能拒绝原假设，那么时间序列可能是不平稳的。

Step 2.差分操作

如果时间序列不是平稳的，我们需要进行差分操作，直到序列满足平稳性要求。差分操作是计算相邻时间点之间的差异，目的是消除趋势和季节性。

一阶差分操作可以用以下方程表示：

$$[Y_t'=Y_t-Y_{t-1}]$$

其中，(Y_t) 是原始时间序列，(Y_t') 是差分后的时间序列。

对于高阶差分，可以依次执行差分操作，直到时间序列变得平稳。

Step 3.自相关分析和偏自相关分析

自相关分析（ACF）和偏自相关分析（PACF）用于估算 ARIMA 模型中的自回归阶数（p）和移动平均阶数（q）。这些分析帮助确定模型的阶数，其中 ACF 反映时间序列自相关性，而 PACF 反映时间序列偏自相关性。

ACF:

$$\rho(k) = Cov(X_t, X_{t+k}) / Var(X_t)$$

PACF:

$$\psi(k) = Cov(X_t - E[X_t|X_{t-1},\cdots,X_{t-k+1}], X_{t-k} - E[X_{t-k}X_{t-k+1},\cdots,X_{t-1}]) / Var(X_t)$$

Step 4.模型选择

一旦确定了差分阶数(d)、自回归阶数(p)和移动平均阶数(q)，我们可以构建 ARIMA 模型。ARIMA 模型的具体形式如下：

$$[Y_t' = c + \phi_1 Y_{t-1}' + \phi_2 Y_{t-2}' + \ldots + \phi_p Y_{t-p}' + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \ldots + \theta_q e_{t-q} + e_t]$$

Step 5.模型检验

我们对 ARIMA 模型的残差进行白噪声检验。白噪声是一种随机序列，没有明显的模式或趋势。白噪声检验用于验证模型的残差是否符合白噪声假设。

作为商家，我们希望我们所进的货能够全部售出达到最大利益，因此我们将预测销售量作为我们的日补货总量来实现收益最大化。

各蔬菜品类未来一周(2023 年 7 月 1-7 日)的日补货总量：

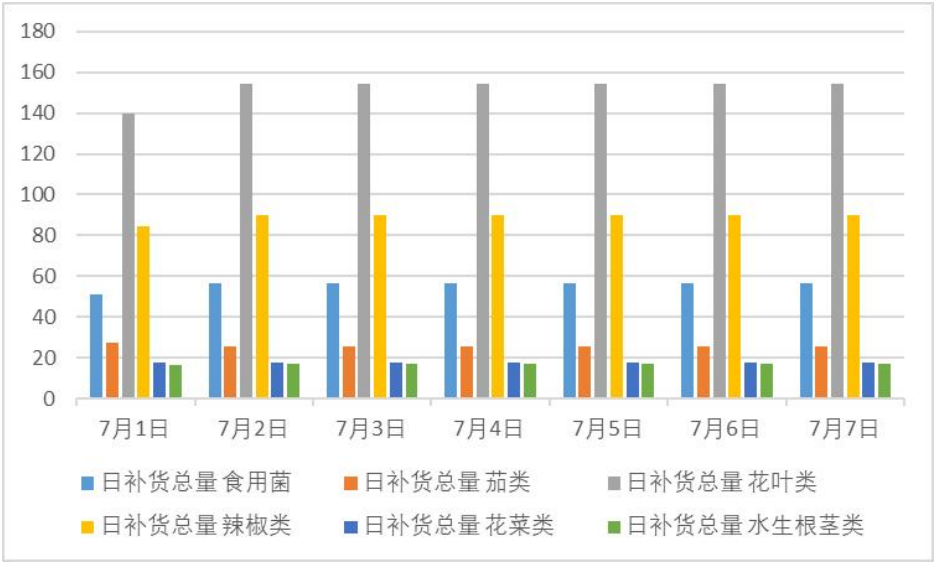

图 11 各蔬菜品类未来一周日补货总量柱状图

定价策略

结合题干信息，蔬菜的供应品种在 4 月至 10 月较为丰富，同时蔬菜的损耗率与其周转情况是对应的，周转越快，销量越大，损耗率则越低，因此对损耗率高的商品应当减少利润增加周转，减少损耗率造成的成本。而蔬菜作为生活必需品，根据生活必需品的需求弹性曲线得知，蔬菜的需求缺乏弹性，尤其对需求量低的蔬菜，对其需求量的影响具有鲁棒性。

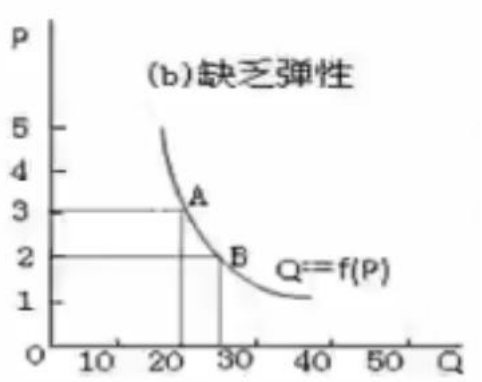

图 12 生活必需品弹性曲线

因此：

★损耗率超过 10%且预计日补货总量低于 60kg：加成率为 40%

★损耗率超过 10%且预计日本货总量高于 60kg：加成率为 30%

★损耗率低于 10%且预计日补货总量低于 60kg：加成率为 80%

★损耗率低过 10%且预计日本货总量高于 60kg：加成率为 65%

6.3 基于贪心算法补货策略优化

为了实现对未来某天单品补货量和定价的决策，商家需要根据过去一周的销售状况，在每日凌晨并不了解进货单品及进货成本的情形下迅速做出最大化利润的决策。因此，通过贪心算法短期内提供 27-33 种的进货单品对于商超决策者颇具裨益。

Step 1.对数据进行预处理，基于先前得到的年份汇总表，筛选出 2023 年 6 月 24-30 日的数据，借助如下单位利率计算公式计算出每笔交易中单个品类商品的单位利润：

$$\text{单位利润} = \sum \text{预测销量} \times (\text{销售单价} - \text{批发价格}) \times (1 - \text{单品损耗率})$$

值得注意的是，由于同种品类蔬菜的单位利润基本接近，有很多重合的数据，所以将平均值作为预测值是合理的。

我们将单位利润的平均值作为对各品类蔬菜 7 月 1 日当天单位利润的预测值，在使用 Python 代码整合计算之后(coonect.py,详见整合代码附件)，我们借助 Excel 自带绘图工具绘制各品类单位利润的图表，实现所求数据可视化：

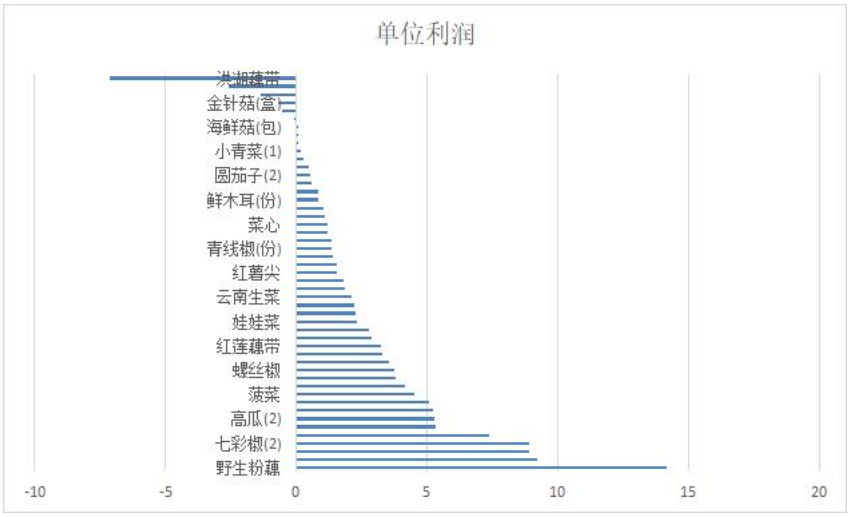

图 13 单位利润统计图

Step 2.通过 Python 代码整合已有数据，得到各品类销售量和加成定价平均值。

贪心算法，即在每一步问题求解时均做出当前的最优选择。在本题中，我们事先引进“**单位利润**”这一新变量作为是否值得补货的判断标准，单位利润越高意味着在同等进货量下，售出等量商品可以获得更高的利润，在每一种商品的选择中，都是选择当前单位利润最高的商品进货，符合贪心策略中的最优情况，所以使用贪心算法可以统计出各品类中最值得补货的商品。

与此同时，我们仍需要考虑到部分品类蔬菜预计销售量不能达到题目中陈列量不小于 2.5 千克的要求，所以我们采取先选择预计销售量大于 2.5 千克品类蔬菜的贪心策略，尽量保证进货的蔬菜全部售出。

由于题目要求，可售单品总数需要控制在 27-33 个，所以在选取完所有销售量大于 2.5 千克的蔬菜之后，我们使用如下公式计算销售量小于 2.5 千克蔬菜种类的加成利润值：

$$\text{加成利润值} = \text{单位利润} \times \text{预计销售量} - \text{未售出量}) \times \text{成本加成定价平均值}$$

对于销售量小于 2.5 千克的蔬菜单品，我们根据加成利润值的多少排序，将加成利润值大作为贪心策略，重新遍历商品以完善贪心算法。

在实际的操作中，选完销售量大于 2.5 千克品类蔬菜之后选取的蔬菜单品量已达到 32 个，符合题目中 27-33 个可售单品总数的要求，所以代码中未进行加成利润值的排序计算。

Step 3.使用 c++代码复现上述操作，（文件名 greedy.cpp，详见贪心算法附件）基于运行框数据，我们最终得到：

（1）当天商超最大收益为 188 元

（2）2023 年 7 月 1 日单品补货量和定价策略

	单品名	数量	定价策略
1	红椒（2）	2.5	19.09731858
2	七彩椒（2）	2.5	19.18255755
3	枝江青梗散花	2.5	13.06013793
4	菱角	2.5	14.16447433
5	净藕（1）	2.5	14.48864

6	菠菜	2.5	14.30595041
7	菠菜（份）	2.5	5.90088998
8	螺丝椒	2.5	11.45323411
9	紫茄子（1）	2.5	9.167727273
10	云南油麦菜	2.5	7.468116279
11	红莲藕带	2.5	9.488671698
12	娃娃菜	2.5	6.580374142
13	姜蒜小米椒组合装（小份）	2.5	4.916914892
14	云南生菜	2.5	9.408779762
15	云南生菜（份）	2.5	4.638277478
16	白玉菇（袋）	2.5	5.799864
17	红薯尖	2.5	5.458074074
18	双孢菇（盒）	2.5	5.032794444
19	云南油麦菜（份）	2.5	4.30762078
20	虫草花（份）	2.5	3.76323396
21	菜心	2.5	6.178695652
22	小米椒（份）	2.5	5.889167724
23	木耳菜	2.5	5.428687601
24	上海青	2.5	8.16420526
25	外地茼蒿	2.5	12.79433166
26	圆茄子（2）	2.5	6.207619735
27	竹叶菜	2.5	3.935085923
28	小皱皮（份）	2.5	2.693981685
29	小青菜（1）	2.5	5.307003984
30	木耳菜（份）	2.5	3.644351795
31	奶白菜	2.5	4.952552479
32	海鲜菇（包）	2.5	2.748387097

6.4 脱离模型假设进行的补货优化和定价决策

补货优化策略相关数据

Data1.市场总体补货倾向（包含本地市场和网络市场）

本地市场的生鲜蔬菜供货情况能在一定程度上反映出当地居民对于菜肴的口味和对蔬菜品类的偏好；网络市场的蔬菜供给量又在威胁传统商超的供货情况，面对当地市场不供货的生鲜蔬菜，许多顾客也会选择线上采购。

Data2.消费者反馈和评价数据。

消费者的满意度直接决定了单品的销售额，消费者满意的单品可以增加补货，不够满意的则要相应减少补货。特别地，对于同一销售单品，不同供货商供应的也可能给顾客带来不同的消费体验，在电商平台竞争激烈的当下，精品化蔬菜供应是传统商超需要考虑的一种策略[5]。

Data3.当地天气情况

蔬菜类商品极具时效性，需要保鲜，同时糟糕的天气也会降低消费者出门前往商超的欲望，只有具体了解当天天气才能更准确补货。

定价决策优化所需数据

Data1.本地交通通达情况

当地的交通通达状况将直接影响供应商运送哪些单品以及他们运送单品的时间、运送费用等；随着国内交通运输能力的不断提高，冷链产品的运输效率不断提高，网商平台也成为了当地传统商超的一大竞争对手。

Data2.当地节日及促销情况数据

节日优惠也是影响单品定价的一大因素，顾客总是希望在购买品质相当的蔬菜产品时支付相对更少的价格，面对节日促销，适当采取打折优惠是必要的。

Data3.生鲜产品企业品牌关系

了解当地顾客对于品牌的依赖性，发掘自身对接生鲜产品企业与其他同类生鲜平台下产品的独特性，了解消费者与品牌关系的远近，如消费者关注品牌的动态且持有积极正面的态度时，说明二者保持着近品牌关系[6]。

七、 贪心算法参与预测的合理性评估

7.1 贪心模型的优点

1.贪心算法的实现过程简单高效，在有限的代码量下使用 C++编程对少量数据进行贪心计算，逻辑清晰，易于理解和实现，是问题解决简单且迅速的选择。

2.有效利用空间：贪心算法求解问题不需要大量额外的内存空间，因此即使在内存受限的环境下也能运行，减轻了程序运行的压力，使得出结论变得更加便捷。

3. 贪心算法的成功取决于选择合适的贪心策略，在选择了正确的贪心策略之后，可以很好地实现对单品补货量和定价的推测。

7.2 贪心模型的缺点

1.局部最优解限制：贪心算法每次只考虑当前步骤的最优选择，而不考虑全局最优解。这可能导致在某些情况下得到次优解或不可行解。因此，贪心算法不能保证总是找到全局最优解。

2.缺乏回溯：贪心算法通常不具备回溯的能力，即一旦做出决策就无法撤销。这意味着如果在后续步骤中发现之前的选择是错误的，算法可能无法修复这个错误。

7.3 贪心模型的实际推广

对于生鲜蔬菜未来补货的策略安排，基于集中规划的粒子群算法、模拟退火算法[7]往往难以适应于多达 251 种蔬菜单品的协调分配，从 7 月 1 日过去一周的可售单品出发，每一次抉择都基于最大化“单位利润”的获取。在不具备其他相关数据的情况下，贪心算法对于分拣单原始交易数据的处理确有混乱，由此我们给出贪心算法的优化建议：

在处理完数据之后，借助 dfs、bfs 搜索算法，通过深度优先搜索和广度优先搜索的方式，有效地解决了贪心算法可能无法取得最优解的问题，无论是深搜的递归语句还是广搜全局遍历的思想都在一定程度上弥补了这一问题。

八、 参考文献

[1] 周业付.数字化背景下生鲜农产品供应链优化研究[J].科技风,2023(24):151-153.DOI:10.19392/j.cnki.1671-7341.202324050.

[2] 钟瑶,聂莹,郭建鑫等.北京市蔬菜价格波动特征及其规律研究——数据互联提升超大型城市“菜篮子“保障能力[J/OL].价格理论与实践:1-5[2023-09-07].http://kns.cnki.net/kcms/detail/11.1010.F.20230601.0845.002.html

[3] 欧阳轶慧. 复式足彩投注策略的算法研究与实例分析[D].中国地质大学（北京）,2011.

[4] 丁明,张立军,吴义纯.基于时间序列分析的风电场风速预测模型[J].电力自动化设备,2005(08):32-34.

[5] 郑文军,王勇.新零售背景下生鲜超市的演化博弈分析[J].运筹与管理,2022,31(06):48-55.

[6] 惠贤芳.生鲜市场创新对消费者新产品采纳意愿的影响研究——基于双元理论视角的分析[J].价格理论与实践,2022(08):120-123.DOI:10.19851/j.cnki.cn11-1010/f.2022.08.457.

[7] 刘华玲, et al."双边交易中的智能决策全局优化——以改进贪心算法的订单分配系统为例".第十八届（2023）中国管理学年会暨“一带一路”十周年研讨会论文集.Ed.. , 2023, 899-906.

附录

附录一		
支撑文件清单		
文件夹名	文件名	含义
代码 1	Connected.py	合并附件 1 和附件 2 代码
代码 2	pre-handle.py	数据预处理代码
代码 3	histogram.py	柱状图绘制
代码 4	fanshaped.py	扇形图绘制
代码 5	kind.py	蔬菜各品类三维散点图绘制
代码 6	coonect.py	各品类销售月份曲线规律图绘制
代码 7	123.py	数据整合处理代码
代码 8	greedy.cpp	贪心预测代码
代码 9	tu.m	图表绘制
代码 10	Spearman.m	斯皮尔曼相关系数热点图绘制
数据 1	normality.txt	正态性检验结果

代码 1

问题 1 程序 1 Connected.py <python> 合并附件 1 和附件 2 代码

```python
import pandas as pd

# 读取附件 1.xlsx 和附件 2.xlsx 文件
df1 = pd.read_excel('C:/Users/92579/Desktop/final/附件 1.xlsx')
df2 = pd.read_excel('C:/Users/92579/Desktop/final/附件 2.xlsx')
# 使用 merge 函数,根据单品编码合并
merged_df = pd.merge(df2, df1, on='单品编码', how='left')
# 保存合并后的数据为新的 Excel 文件 conform.xlsx
merged_df.to_excel('conform.xlsx', index=False)
```

代码 2

问题 1 程序 2 pre-handed.py <Python> 数据预处理代码

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler
# 读取保存好的 comform 文件
data = pd.read_csv('conform.csv')
# 缺失值处理（用均值填充缺失值）
data.fillna(data.mean(), inplace=True)
# 3. 数据归一化
scaler = MinMaxScaler()
data_normalized = scaler.fit_transform(data)
# 4. 数据标准化
scaler = StandardScaler()
```

```
    data_standardized = scaler.fit_transform(data)
    # 5. 保存预处理后的数据到新的 CSV 文件
    pd.DataFrame(data_normalized,
columns=data.columns).to_csv('one.csv', index=False)
    pd.DataFrame(data_standardized,
columns=data.columns).to_csv('two.csv', index=False)
```

代码 3

问题 1 程序 3 histogram.py <Python> 柱状图绘制

```
    import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt
    from matplotlib.font_manager import FontProperties

    # 设置中文字体
    font = FontProperties(fname="C:/Windows/Fonts/simsun.ttc",
size=12)
    desktop_path = 'C:/Users/92579/Desktop/'
    csv_filename = 'coonected.csv'
    # 构建完整的文件路径
    csv_file_path = desktop_path + csv_filename
    # 使用 pandas 读取 CSV 文件
    df = pd.read_csv(csv_file_path)
    category_corr =
df.groupby('Category')['Sales'].sum().reset_index()
    # 绘制柱状图
    plt.figure(figsize=(10, 6)) # 增加图形大小
    sns.barplot(x='Category', y='Sales', data=category_corr,
palette='Set2') # 调整颜色和样式
    plt.title('销售分类', fontproperties=font) # 使用中文标题
    plt.xticks(rotation=45, fontproperties=font) # 使用中文刻度标签
    plt.xlabel('产品分类', fontproperties=font) # 添加坐标轴标签
    plt.ylabel('销售额', fontproperties=font) # 添加坐标轴标签
    plt.gca().spines['top'].set_visible(False) # 去除顶部边框
    plt.gca().spines['right'].set_visible(False) # 去除右侧边框
    # 添加数据标签
    for index, row in category_corr.iterrows():
      plt.text(index, row['Sales'], f'{row["Sales"]}', ha='center',
va='bottom', fontproperties=font)
    plt.show()
```

代码 4

问题 1 程序 4 fanshaped.py <python> 各单品总销售额扇形图绘制

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

    # 设置中文字体
    plt.rcParams['font.sans-serif'] = ['SimSun']  # 指定中文字体
    plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题
    desktop_path = 'C:/Users/92579/Desktop/'
    csv_filename = 'coonected.csv'
    # 构建完整的文件路径
    csv_file_path = desktop_path + csv_filename
    # 使用 pandas 读取 CSV 文件
    df = pd.read_csv(csv_file_path)
    # 计算不同单品之间的相关性
    product_corr = df.groupby('Product')['Sales'].sum().reset_index()
    # 选择销售量较高的前 N 个单品
    top_n = 35  # 您可以根据需要更改 N 的值来选择前 N 个单品
    top_product_corr = product_corr.nlargest(top_n, 'Sales')
    # 计算剩余单品的销售总量，并将其添加到 top_product_corr 中
    other_sales =
product_corr[~product_corr['Product'].isin(top_product_corr['Product']
)]['Sales'].sum()
    other_df = pd.DataFrame({'Product': ['其他'], 'Sales': [other_sales]})
    top_product_corr = pd.concat([top_product_corr, other_df],
ignore_index=True)
    # 绘制扇形图
    plt.figure(figsize=(8, 8))  # 设置图形大小
    plt.pie(top_product_corr['Sales'],
labels=top_product_corr['Product'], autopct='%1.1f%%', startangle=140)
    plt.title('Top {} 单品销售比例'.format(top_n))  # 使用中文标题
    plt.axis('equal')  # 保持图形圆形
    plt.show()
```

代码 5

问题 1 程序 5 kind.py <python> 蔬菜各品类三维散点图绘制

```
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    from mpl_toolkits.mplot3d import Axes3D
    import matplotlib
    import seaborn as sns
    import warnings

    # 设置字体为宋体
```

```
matplotlib.rcParams['font.family'] = 'sans-serif'
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
# 忽略警告信息
warnings.filterwarnings("ignore")
# 使用绝对文件路径读取 Excel 文件
file_path = r"C:\Users\92579\Desktop\final\蔬菜品类时序图.xlsx"
df_1_data1 = pd.read_excel(file_path)
# 创建三维散点图
fig = plt.figure(figsize=(10, 8), dpi=300)
ax = fig.add_subplot(111, projection='3d')
# 提取蔬菜品类数据
vegetable_categories = df_1_data1.columns
x = np.arange(len(vegetable_categories))
# 为 y 轴生成一个等差数列，以便在三维图中将点分散开
y = np.arange(len(df_1_data1))
y, x = np.meshgrid(y, x)
y = y.flatten()
x = x.flatten()
# 提取关联系数作为 z 轴数据
z = df_1_data1.values.flatten()
# 创建散点图
scatter = ax.scatter(x, y, z, c=z, cmap='coolwarm', s=50, alpha=0.7)
# 设置轴标签
ax.set_xlabel('蔬菜品类', fontsize=12)
ax.set_ylabel('蔬菜品类', fontsize=12)
ax.set_zlabel('关联系数', fontsize=12)
# 调整标题位置
ax.title.set_position([0.5, 1.05])
# 添加颜色条
cbar = fig.colorbar(scatter)
cbar.set_label('关联系数', fontsize=12)
# 调整颜色条标签字体大小
cbar.ax.tick_params(labelsize=10)
# 设置标题
plt.title('蔬菜品类关联系数三维散点图', fontsize=14)
# 设置坐标刻度的字体大小
ax.tick_params(axis='x', labelsize=10)
ax.tick_params(axis='y', labelsize=10)
ax.tick_params(axis='z', labelsize=10)
# 显示图形
plt.show()
```

代码 6

问题 1 程序 6 coonect.py <python> 绘制蔬菜各品类销售月份曲线规律图

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import warnings

# 设置字体为宋体
matplotlib.rcParams['font.family'] = 'sans-serif'
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
# 设置表格字体大小
plt.rcParams['font.size'] = 5  # 进一步减小标签字体大小为 5
# 使用绝对文件路径读取 Excel 文件
df_1_data1 = pd.read_excel(r"C:\Users\92579\Desktop\final\蔬菜品类时序图.xlsx")
# 绘制蔬菜各品类时序图（销售月份曲线-规律）
plt.figure(figsize=(28, 12), dpi=300)  # 进一步增加图表宽度
plt.subplots_adjust(left=0.03, right=0.97, wspace=0.3, hspace=0.5)  # 增加左右间距和上下两行的间距
n = -1
line_styles = ['-', '--', '-.', ':']  # 不同的线条样式
colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k']  # 不同的颜色
for i, column in enumerate(df_1_data1.columns):
    s = df_1_data1[column]
    n += 1
    plt.subplot(2, 4, n + 1)
    sns.lineplot(data=s, linestyle=line_styles[i % len(line_styles)], color=colors[i % len(colors)])
    plt.title(column, fontsize=6)  # 进一步减小标签字体大小
    plt.grid()
plt.xticks(rotation=45)  # 旋转 x 轴标签
plt.savefig('蔬菜各品类销售-月份.png')  # 保存图片
plt.show()
```

代码 7

问题 3 程序 1 123.py <python> 数据整合处理代码

```python
import pandas as pd

# 读取 Excel 文件 123.xlsx
df = pd.read_excel('C:/Users/92579/Desktop/123.xlsx')
# 根据"单品类"分组并计算单位利润的平均值
average_profit = df.groupby('单品类')['单位利润
```

```
'].mean().reset_index()
    # 统计每种单品类的数量
    count_per_category = df['单品类'].value_counts().reset_index()
    count_per_category.columns = ['单品类', '数量']
    # 将数量除以 7 并添加到 average_profit 中
    average_profit['数量/7'] = count_per_category['数量'] / 7
    # 计算成本加成定价的平均值
    average_cost_plus_pricing = df.groupby('单品类')['成本加成定价
'].mean().reset_index()
    average_cost_plus_pricing.columns = ['单品类', '成本加成定价平均值']
    # 合并平均利润和成本加成定价平均值的数据
    merged_df = pd.merge(average_profit, average_cost_plus_pricing,
on='单品类')
    # 根据单位利润列进行排序
    merged_df_sorted = merged_df.sort_values(by='单位利润',
ascending=False)
    # 创建一个新的 Excel 文件并将结果保存到其中
    with pd.ExcelWriter('average_profit_with_count2.xlsx',
engine='xlsxwriter') as writer:
        merged_df_sorted.to_excel(writer, sheet_name='平均利润',
index=False)
```

代码 8

问题 3 程序 2 greedy.cpp <c++> 借助贪心算法求解最优解

```
    #include<iostream>
    using namespace std;
    //因为可售单品总数控制在 27-33 个，所以建立数组存放 33 个数据的单位利润
    const double
a[50]={14.179144,9.240230077,8.92,8.918126,7.390512,5.336660088,5.2672
36174,5.260698,5.10084,4.506397,4.151063,3.786010837,3.764660315,3.568
458,3.278344,3.25143,2.883753678,2.783518,2.351300712,2.293943203,2.20
3069041,2.1018,1.885218558,1.821885,1.587386667,1.56408,1.42714,1.3766
64,1.375813007,1.22495925,1.2082,1.1134072,1.045924528,0.8558865,0.830
029,0.608652571,0.553520667,0.504419023,0.295326987,0.161406,0.105665,
0.0944384,0.078387097};
    const double
b[50]={1.857142857,8.142857143,2,38.85714286,11,0.428571429,0.28571428
6,3.285714286,21.28571429,4.285714286,0.571428571,3.285714286,2.571428
571,9.285714286,30.71428571,7.285714286,2.142857143,0.857142857,14.285
71429,2.428571429,15.14285714,25.14285714,21.42857143,10,7,1.142857143
,20.14285714,0.571428571,28.42857143,2.857142857,4.428571429,12.857142
86,11.28571429,0.285714286,32.28571429,17.14285714,18.14285714,9.42857
1429,12.85714286,11.57142857,11.28571429,16.14285714,10.42857143};
```

```cpp
int main(){
    int sum=0;
    int ans=0;
    for(int i=0;i<43;i++){
        if(b[i]>2.5){
            sum+=a[i]*2.5;
            ans++;
            cout<<i<<endl;
        }
        if(ans==32)break;
    }
    if(ans>=27)cout<<sum;
    else cout<<"1";

    return 0;
}
```

代码 9
问题 1 程序 6　tu.py　<matlab>　图表绘制

```
% 给定数据
data = [
1 6092.273 17857.644 4103.945 1799.989 10205.421 9827.298
2 4065.084 14572.199 3437.325 1850.267 9684.989 7594.469
3 2516.075 13915.576 2579.784 1561.455 8915.809 5732.216
4 1395.518 13553.797 2533.346 1828.698 7392.322 4684.421
5 887.232 14104.847 2704.914 2477.506 6618.543 4054.72
6 1198.477 13297.117 2495.018 2568.888 5944.559 3913.554
7 2740.535 17398.938 4471.245 2900.921 6140.154 4514.817
8 4507.477 24516.036 4926.903 2736.189 9886.956 5298.255
9 3773.188 18738.506 3594.168 1518.467 6760.253 5063.409
10 4936.266 19229.364 3927.983 1518.234 6760.654 5063.432
11 3592.22 14944.403 3884.538 937.245 6162.623 8159.762
12 4901.188 16521.331 3165.887 862.461 5983.823 8740.956
];

% 提取月份和蔬菜品类数据
months = data(:, 1);
vegetables = data(:, 2:end);

% 绘制蔬菜品类在不同月份的变化情况
figure;
```

```
    plot(months, vegetables, '-o');
    xlabel('月份');
    ylabel('销量');
    legend('水生根茎类', '花叶类', '花菜类', '茄类', '辣椒类', '食用菌');
    title('蔬菜品类在不同月份的销量情况');

    % 两两比较蔬菜品类销量
    figure;
    pairs = nchoosek(1:size(vegetables, 2), 2);
    numPairs = size(pairs, 1);
    for i = 1:numPairs
    subplot(ceil(numPairs/2), 2, i);
    plot(vegetables(:, pairs(i, 1)), vegetables(:, pairs(i, 2)), 'o');
    xlabel('销量');
    ylabel('销量');
    title(sprintf('蔬菜品类%d vs 蔬菜品类%d', pairs(i, 1), pairs(i, 2)));
    end
```

代码 10
问题 1 文档 1 Spearman.m 斯皮尔曼相关系数热点图绘制

```
    % 定义数据
    data = [
    1.000000 0.480384 -0.437479 -0.171968 0.438286 -0.305699 0.045283
0.218357 0.131014 0.408318;
    0.480384 1.000000 0.255692 0.137684 0.795396 -0.062937 -0.108767
0.349650 0.356643 -0.032692;
    -0.437479 0.255692 1.000000 0.413777 0.152327 0.098074 -0.094430
0.350263 0.346761 -0.036388;
    -0.171968 0.137684 0.413777 1.000000 0.153534 0.605811 0.552031 0.532379
0.697600 -0.200250;
    0.438286 0.795396 0.152327 0.153534 1.000000 0.062384 -0.016172 0.655032
0.561456 0.299740;
    -0.305699 -0.062937 0.098074 0.605811 0.062384 1.000000 0.282794
0.468531 0.510490 -0.475845;
    0.045283 -0.108767 -0.094430 0.552031 -0.016172 0.282794 1.000000
0.050758 0.014502 -0.214690;
    0.218357 0.349650 0.350263 0.532379 0.655032 0.468531 0.050758 1.000000
0.909091 0.374138;
```

```
    0.131014 0.356643 0.346761 0.697600 0.561456 0.510490 0.014502 0.909091
1.000000 0.228841;
    0.408318 -0.032692 -0.036388 -0.200250 0.299740 -0.475845 -0.214690
0.374138 0.228841 1.000000
    ];

    % 绘制相关系数矩阵热图
    figure;
    imagesc(data);
    colorbar;
    title('各品类直接关联系数特征图');
    xticks(1:10); % 设置横坐标刻度
    yticks(1:10); % 设置纵坐标刻度
    xticklabels({'圆茄子(1)', '圆茄子(2)', '大龙茄子', '紫圆茄', '紫茄子(1)', '
紫茄子(2)', '花茄子', '长线茄', '青茄子(1)', '青茄子(2)'}); % 设置横坐标标签
    yticklabels({'圆茄子(1)', '圆茄子(2)', '大龙茄子', '紫圆茄', '紫茄子(1)', '
紫茄子(2)', '花茄子', '长线茄', '青茄子(1)', '青茄子(2)'}); % 设置纵坐标标签
    xlabel('品类');
    ylabel('品类');
```

附录十

问题 1 文档 1 normality.txt 正态性检验结果

水生根茎类：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.1353, pvalue=0.9597)).

花叶类：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.1893, pvalue=0.7163)).

花菜类：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.1657, pvalue=0.8448)).

茄类：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.1844, pvalue=0.7446)).

辣椒类：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.2644, pvalue=0.3136)).

食用菌：正态性检验结果为 True，即数据分布近似正态分布(KstestResult(statistic=0.2310, pvalue=0.4743)).