蔬菜类商品的自动定价与补货决策

摘 要：生鲜商超中蔬菜品类商品会因各种情况导致商品成本和销量等指标发生波动，进而影响商家的运营情况。因此，可靠的市场需求分析和商超销售空间的限制使得合理的销售组合变得极为重要。

针对问题一，首先对所提供的数据进行合并以及预处理，以获取有效且较为客观可靠的数据。接着，对不同蔬菜品类或单品数据使用 Python 进行可视化来分析得出销售量分布规律。随后针对数据进行 Pearson 相关性分析。从结果可以发现，蔬菜各品类及单品销售量之间存在着密切相关性。最后为了确保分析的准确性，选择使用 Apriori 模型来进行验证，通过置信度和提升度等属性来对结果进行筛选，对比印证分析结果的准确性。

针对问题二，构建了岭回归模型来对各蔬菜品类的销售总量与成本加成定价进行二元关系分析。在对数据进行聚合分类之后将其代入岭回归模型，从模型的输出情况中可以发现销售总量和成本加成定价之间存在极为密切的趋近关联关系。但是，通过 MSE 检验发现岭回归模型预测准确度不够，并且发现题目所给数据规模庞大还具有较强的时间序列关系，于是构建了 LSTM 预测模型。经过数据训练之后，模型在预测检测中表现很好，运用该模型完成了对销量的预测。最后则是根据销量确定不同品类的加价率再结合批发成本来制定定价策略。

针对问题三，首先通过在问题 2 的 LSTM 模型的基础上添加策略来预测 7 月 1 日的各单品销量并结合提出的成本加成定价策略计算得出预期利润。接下来考虑目标函数为收益最大以及添加题目中的约束条件来构建模拟退火模型，进而得出补货策略。根据补货策略和问题 2 的定价策略可以进一步预测利润，然后再次进入 SA 模型中进行求解，并设置自适应迭代过程中的终止条件，当结束自适应过程后就能分析得到使收益最大的 33 个单品的补货量及定价策略。

针对问题四，为了制定更完善的蔬菜商品补货和定价策略，选择收集蔬菜季节性销量、市场竞争、天气、库存量和相关销售促销活动等数据，并构建了多元回归模型来深入分析相关因素的影响。然后，将收集到的数据带入模型求解，从输出结果中可以发现所选择的相关因素对蔬菜商品的销量、成本以及利润等都具有较大的影响力，因此采集并考虑这些相关因素的数据可以为制定蔬菜商品的补货和定价决策等提供合理充分的依据。

关键词：定价与补货决策 岭回归模型 LSTM 时间序列预测 模拟退火

一、问题重述

1.1 问题背景

在生鲜商超中，蔬菜类商品的保鲜期有限，随着销售时间的增加，其品质也会逐渐变差。为此，商超通常根据历史销售和需求情况，每天进行蔬菜补货决策。考虑到蔬菜品种繁多且产地不同，商家需要在凌晨 3:00-4:00 进行进货交易，但并不能确定具体的单品和进货价格。因此，商超一般采用"成本加成定价"的方法，并对运损和品质下降的商品进行打折销售。可靠的市场需求分析对于补货和定价决策尤为重要。因此，商超需要合理的地制定蔬菜商品的补货和定价决策来充分利用有限的销售空间。

1.2 问题的提出：

题目中已知：各蔬菜品类的商品信息、销售流水明细、批发价格与各商品近期损耗率等相关数据。

问题 1 蔬菜类商品不同品类或不同单品之间可能存在一定的关联关系，请分析蔬菜各品类及单品销售量的分布规律及相互关系。

问题 2 考虑商超以品类为单位做补货计划，请分析各蔬菜品类的销售总量与成本加成定价的关系，并给出各蔬菜品类未来一周(2023 年 7 月 1-7 日)的日补货总量和定价策略，使得商超收益最大。

问题 3 因蔬菜类商品的销售空间有限，商超希望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，且各单品订购量满足最小陈列量 2.5 千克的要求。根据 2023 年 6 月 24-30 日的可售品种，给出 7 月 1 日的单品补货量和定价策略，在尽量满足市场对各品类蔬菜商品需求的前提下，使得商超收益最大。

问题 4 为了更好地制定蔬菜商品的补货和定价决策，商超还需要采集哪些相关数据，这些数据对解决上述问题有何帮助，请给出你们的意见和理由。

二、问题分析

2.1 问题一的分析

针对问题 1，由于蔬菜类商品不同品类或不同单品之间可能存在一定的关联关系，首先需要对蔬菜类商品的销售数据进行整理与清洗，以获取有效可用的数据。然后，可以通过饼状图、柱状图、折线图等可视化方式展示不同蔬菜品类或单品的销售量分布情况。对可视化结果进行观察，以寻找潜在的分布规律。接下来，通过 Pearson 相关性分析[1]，来评估不同蔬菜品类或单品之间的关联程度，判断它们之间的相互关系。最后，分析结果解释蔬菜各品类及单品销售量的分布规律和相互关系，再通过 Apriori 算法来验证我们分析结果[2]的准确性。这样做能够对销售数据进行全面的分析，得出准确的结论。

2.2 问题二的分析：

成本加成定价法是按产品单位成本加上一定比例的利润制定产品价格的方法，为了分析各蔬菜品类的销售总量与成本加成定价的关系，可以选择在问题 1 处理后的数据基础上进行分析。针对六类蔬菜品类使用岭回归模型分析销售总量和成本加成定价之间的二元关系。

而岭回归是一个线性模型[3]，预测效果较差，并不适用于预测各蔬菜品类未来一周的情况。同时，我们观察数据可以发现所给出各蔬菜品类的数据具有极强的时间序列关系以及极大的数据量。因此可以使用历史数据来对时间序列预测模型 LSTM 进行训练，并对训练的模型进行 MSE 等检验来验证模型的准确性，进而预测未来一周各蔬菜品类的销售量。在得出预测销售量后，就可以根据各个品类的成本、损耗率以及预期销售量来制定定价策略。

2.3 问题三的分析：

针对问题 3，为了得出在多个限制条件下的最优解，我们可以构建自适应模型来通过不断的循环迭代进而求出最优解。通过 LSTM 来预测当日的销量并与问题 2 构建的岭回归模型结合来得出初期预测利润，接着将数据喂入模拟退火优化模型之中考虑题目中所给出的约束条件并得出应补货的数量，根据补货的数量、定价策略以及销量重新计算预期利润。重复上述自适应模型，并设置最大迭代值，记录局部和全局最优解。即可根据混合模型运行的结果分析出单品补货量和定价策略。

2.4 问题四的分析：

为了更准确地预测市场需求、优化库存并制定更有效的定价策略，我们选择采集了季节性销量、市场竞争、天气、库存和销售促销活动等数据，并将其用于构建多元回归模型进行分析，通过对结果的分析评估，我们可以确定不同因素对蔬菜销售量的影响力大小，并分析它们之间的差异性。最后，通过结果确定不同因素对销售量的影响差异，并利用这些信息更精准地进行需求预测，避免过量或缺货的情况出现，进一步制定科学合理的蔬菜商品的补货计划和定价策略。

三、模型假设

假设 1：题目中所给数据完全真实，不存在虚假数据。

假设 2：蔬菜品类不会产生较大的波动。

假设 3：蔬菜损耗的概率与题目中的数据一致。

假设 4：各蔬菜品类在供给上不会出现问题。

假设 5：模型进行分析预测的时候未出现极端特殊情况。

四、符号说明

为使文章前后一致，特地对文中符号做如下说明：

表 4.1 符号说明

符号	符号说明
S_A	效应平方和
$Z-Score_{ij}$	零均值法对数据进行处理
S_T	多因素方差法中的误差表示
s_p^2	独立样本t检验中的样本方差
$Cov(X,Y)$	样本协方差
r_{ij}	构建的协方差矩阵
$\sigma_X(sigmaX)$	X 的标准差

这里只列出论文各部分通用符号，个别模型单独使用的符号在首次引用时会进行说明。

五、模型建立与求解

5.1 问题一的建模与求解

5.1.1 合并数据

根据提供的附件 1 和附件 2，我们进行了数据合并。附件 1 包含蔬菜的单品编码、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。通过合并数据，我们可以得到一个新的完整数据集，其中包含销售日期、销售时间、单品编码、单品名称、分类编码、分类名称、销售量、销售单价、销售类型以及是否打折销售的信息。合并后的数据集将帮助我们更全面地了解每个单品在销售过程中的表现，并且可以支持各种分析需求，比如统计不同单品的销售量、计算销售额、分析销售趋势等。为进一步的数据分析和业务决策提供有力支持。

5.1.2 预处理数据

1）数据量化处理

数据量化是将原始数据转换为可供分析和建模的数值形式的过程。它能够使不具体、无法用于算法的数据得以具体化、可计算，从而实现数据计算分析的目的。在问题一中，我们只需要选取附件 2 中所需的销售类型和是否打折销售等变量，并对其进行数值化处理。通过数据量化，原本难以处理的信息得以清晰明了地呈现出来，有助于深入了解数据的特征和规律，提高模型的建立和分析效率[4]。

2）异常值处理

异常值指的是在数据集中与其他观测值明显不同或偏离预期模式的值。异常值也被称为离群值（Outliers）。异常值可能对数据分析和建模产生负面影响，它们可能扰乱统计分析、降低模型的准确性，甚至导致错误结论。因此，对于异常值的处理对结果分析是尤为关键的。因此我们选用较为常见的 IQR 异常值识别方法来完成异常值的识别并将其去除，从而来消除异常值对数据分析和建模的不良影响。

3）数据标准化处理

由于每个影响因素的维度不同，因此对每个量纲进行标准化。标准化方法很多，如 Z-Score 法（每个数据减去均值再除以方差）、平均值法（每个数据除以该子序列的均值）、MaxMin 方法（每个数据减去最小值再除以最大值与最小值的差）等等[5]。因为附件所提供的数据差值明显，因此采用 MaxMin 方法：

$$A_i(j)=\left[\left(A_i(j)-A_i(j)_{\min}\right)/\left(A_i(j)_{\max}-A_i(j)_{\min}\right)\right]$$

在标准化处理后，可以确保在进行数据分析和建模时，不会因为相关数据的度量单位差异而导致结果偏差。

5.1.3 数据可视化分析

根据处理后的数据，我们使用饼状图、柱状图、折线图等可视化方式展示了不同蔬菜品类或单品的销售量分布情况。然后，我们对这些可视化结果进行观察，以寻找潜在的分布规律。通过可视化，我们可以更清晰地了解销售量在不同品类或单品之间的差异。

饼状图可以清晰地显示各个品类或单品在整体销售中所占比例，柱状图可以直观地比较不同品类或单品之间的销售量大小，而折线图则可以显示销售量随时间的变化趋势。观察可视化结果有助于我们发现潜在的分布规律。通过深入分析可视化结果和观察到的规律，我们可以更好地理解销售数据，并根据这些洞察做出相应的蔬菜商品的补货和定价决策。

这里给出部分结果与分析，剩余放置在附录：

图 5.1 不同蔬菜品类的销售量分布

根据图 1 所显示的结果，可以观察到销售量最高的品类是'花叶类'蔬菜，其次是'辣椒类'和'食用菌'。而相比其他品类的蔬菜，'茄类'的销售量相对较低。这些结果反映了各个品类蔬菜之间的销售差异。'花叶类'的高销售量可能意味着它在市场上的受欢迎程度较高，而'茄类'的低销售量则可能表明它的市场需求相对较低。

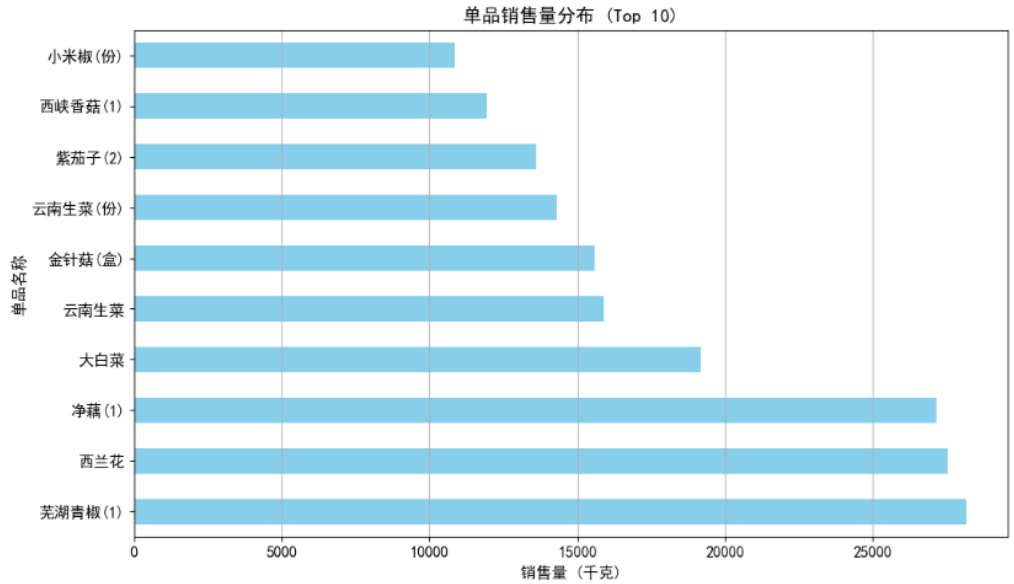

图 5.2 单品销售量分布(Top10)

根据图 2 显示的结果，我们可以观察到销售量排名前三的单品是芜湖青椒（1）、西兰花和净藕（1）。这些单品在销售中表现出较高的需求，也进一步说明这些蔬菜单品在消费者中享有广泛的认可度和受欢迎程度。

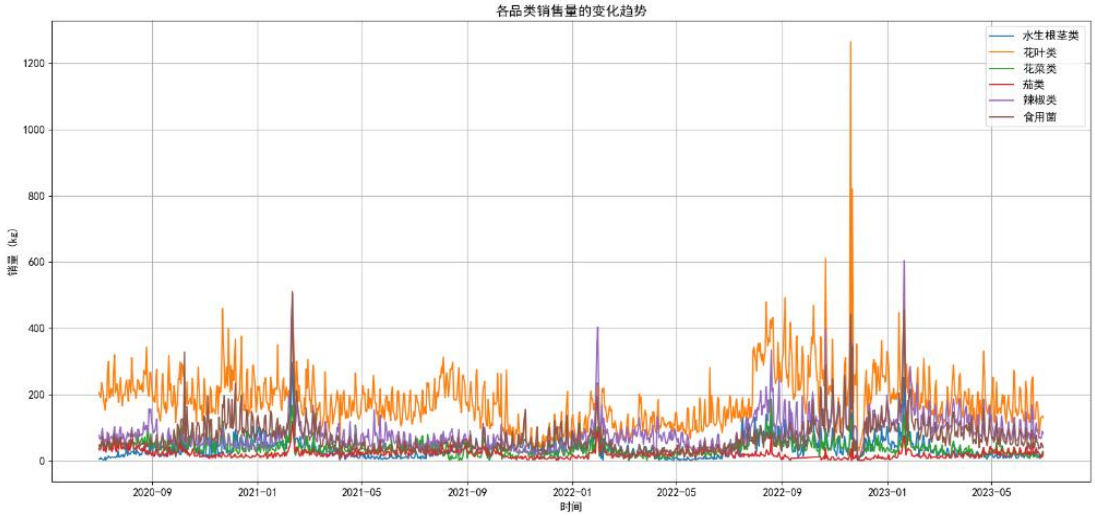

图 5.3 各品类销售量的变化趋势

根据图 3 的图表，观察销售量时间趋势可以发现：

1）不同蔬菜品类的销售量在一年内呈现明显的季节性变化，出现了高峰和低谷，也进一步说明了不同蔬菜品类的销售量。

2）在花叶类品类中，每年都会出现销售量的高峰期，这可能与某些特定节日或季节性事件密切相关。

5.1.4 Pearson 关联性分析

为了深入探索不同蔬菜品类之间的关联关系，我们将进行相关性分析。具体而言，我们使用 Pearson 相关性分析来计算各个品类销售量之间的相关性。Pearson 相关性分析是一种常用的统计方法，用于评估两个连续变量之间的线性关系强度和方向[1]。它基于 Pearson 相关系数，其范围在-1 和 1 之间，用于衡量两个变量之间的线性相关程度。

表 5.1 相关系数解释说明

相关系数	解释说明
相关系数接近 1	两个变量之间呈较高的正相关性；
相关系数接近-1	两个变量之间呈较高的负相关性；
相关系数接近 0	两个变量之间彼此独立，没有相关性

通过计算相关系数，我们可以了解哪些品类的销售趋势是相关的，即当一个品类的销售量增加时，另一个品类的销售量也可能会增加。这种相关性分析将帮助我们揭示蔬菜品类之间的潜在关联和相互作用。

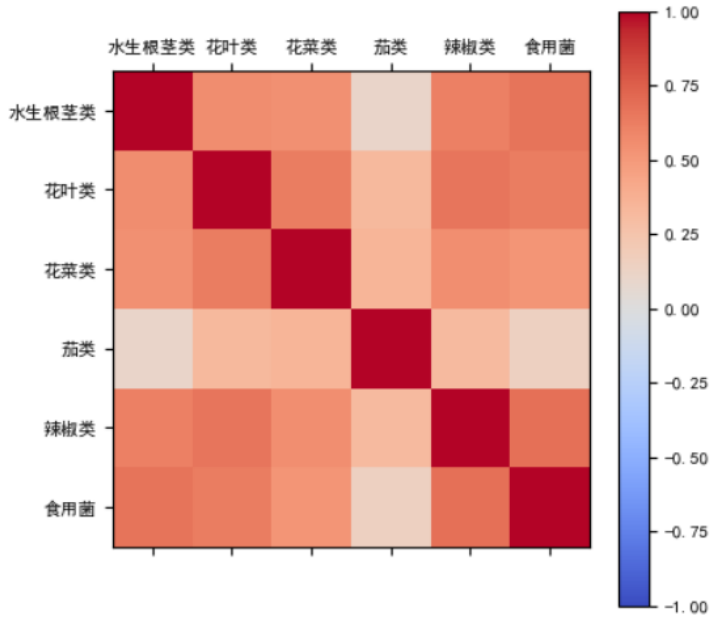

图 5.4 相关系数热力图

根据相关系数热力图，我们可以发现：

1、高相关性：“辣椒类”和其他类之间都存在有相对较高的正相关性。具体来说，当其他类的蔬菜的销售量增加时，“辣椒类”销售量也可能增加，反之亦然。这表明这“辣椒类”和其他类蔬菜之间存在着密切的关系，并且它们可能会受到相似的市场因素的影响。

2、低或无相关性：相较于其他蔬菜类别而言，"茄类"蔬菜与其他类别之间的相关性较低。这意味着"茄类"蔬菜与其他蔬菜类别之间的联系相对较少，它们可能有着独立的销售模式。它们将很少受到相似的市场因素的影响。

同时我们还得到了单品销量之间相关系数的热力图。这张图能够展示不同蔬菜单品之间的相关性程度。

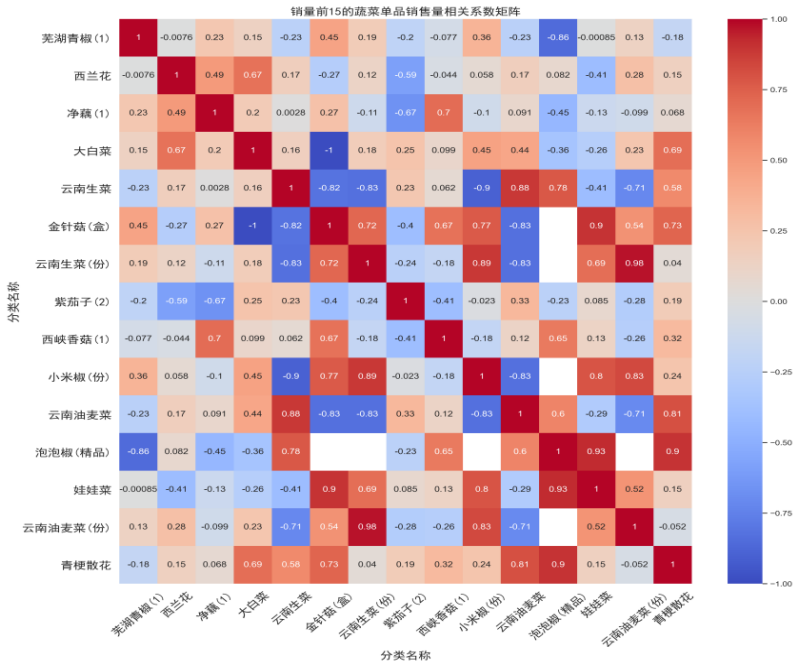

图 5.5 单品销量之间相关系数的热力图

根据图上出相关性大于 0.75 的这几组单品结果分析可知，这几组单品组在数量上和同意时间被购买，我们据此推测，它们可能将被组成某种组合或套餐的一部分来进行销售。基于这一推测，我们可以进一步研究这些组合或套餐的构成和销售策略，以更好地了解消费者的购买偏好和市场需求。通过深入了解消费者行为，我们可以更好地制定蔬菜商品补货和定价策略，从而最大化销售和收益。

5.1.5 Apriori 验证分析结果

在验证结果方面，我们选用 Apriori 算法来对我们的结果进行验证：

1、支持度（support）：在所有的商品交易中商品组合出现的频率。

2、置信度（confidence）：购买商品 A 时同时购买商品 B 的概率。

3、提升度（lift）： 商品 A 和商品 B 作为一个组合销售的概率与它们分别独立销售的概率之间的比值。

我们为了捕获更多的商品组合，将设定一个较低的支持度阈值。然后，我们通过使用置信度和提升度来对结果进行筛选。结果与上图所示结果保持相一致，根据分析，这些几组单品很有可能构成某些商品组合的一部分。

5.2 问题二的建模与求解

5.2.1 岭回归模型分析蔬菜品类二元关系

在这里我们需要构建出一个岭回归模型来对每个蔬菜品类的销售总量与成本加成定价进行二元分析，进而得出他们之间的关系。

1）数据处理

由于我们考虑的是六大蔬菜品类分别的销售总量与成本加成定价之间的关系，因此我们需要先对题目所给数据进行处理来得到每个类别的销售总量、销售单价和成本等。经代码处理后的数据如下：

2）模型的建立

构建代价函数

$$\mathrm{Cos}\,t(w)=\sum_{i=1}^{\mathrm{N}}(y_i-w^Tx_i)^2+\lambda\|w\|_2^2$$

其中λ表示惩罚项的系数，人为的控制惩罚项的大小

求使得岭回归的代价函数最小时 w 的大小：

$$w=\arg\min_w(\sum_{i=1}^{\mathrm{N}}(y_i-w^Tx_i)^2+\lambda\|w\|_2^2)$$

代价函数通过求导直接得到w的解析解，

$$w=(X^{\mathrm{T}}X+\lambda I)^{-1}X^{\mathrm{T}}y\quad\lambda\in\mathbb{R}$$

$$X=\begin{bmatrix}X_1^T\\X_2^T\\\vdots\\X_N^T\end{bmatrix}=\begin{bmatrix}X_{11}&X_{12}\cdots X_{1M}\\X_{21}&X_{22}\cdots X_{2M}\\\vdots\\X_{\mathrm{N}1}&X_{N2}\cdots X_{NM}\end{bmatrix}\quad y=\begin{pmatrix}y_1\\y_2\\\vdots\\y_N\end{pmatrix}$$

其中 X 为 $N*M$ 矩阵， y 为 N 维列向量， λ 属于实数集， I 为M*M 的单位矩阵。

3）模型的求解

在数据代入附录所给出的回归模型代码之后，可以得出各蔬菜品类的销量与价格之间的分析结果如下：

表 5.2 岭回归分析结果

K=0.12	非标准化系数		标准化系数	t	P	R^2	调整 R^2	F
	B	标准误	Beta					
常数	1011010418.871	172.894	-	5847577.958	0.000***	0.507	0.179	1.543(0.346)
定价	0	0	0.619	1.393	0.258			
销量	-0.003	0.002	-0.813	-1.83	0.165			

注：***、**、*分别代表 1%、5%、10%的显著性水平

从表格中我们可以发现基于 F 检验显著性 P 值为 0.346，水平上不呈现显著性，接受原假设拟合优度 R^2为 0.507，模型表现为较为较好，足以支撑进行后续的结果分析。

4）结果分析

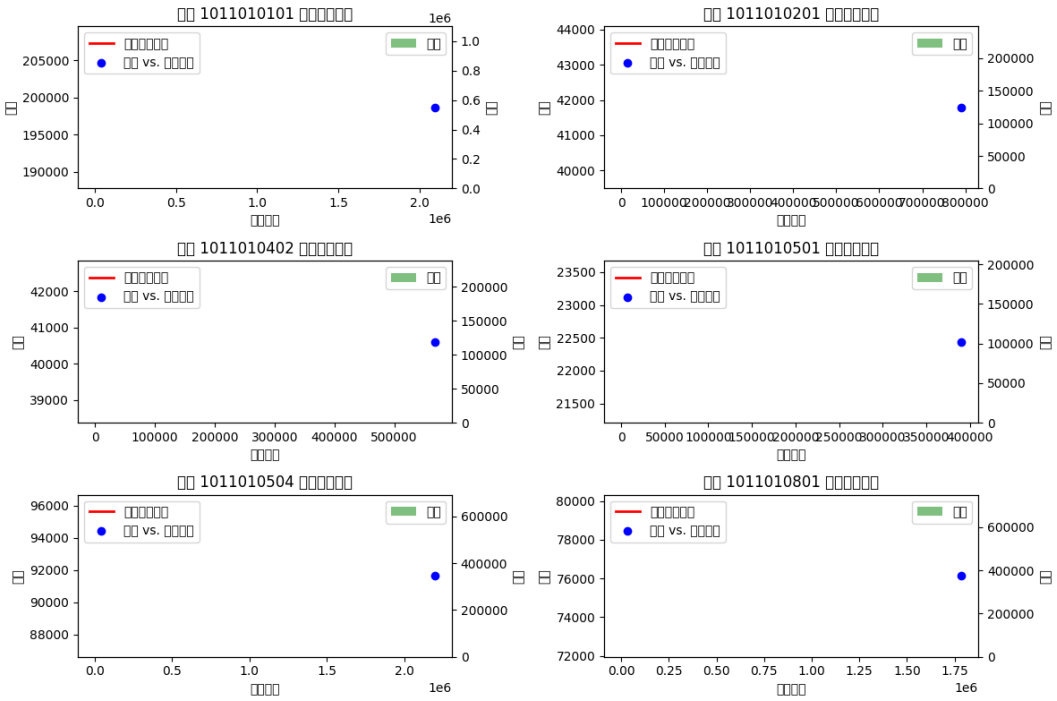

图 5.6 岭回归输出图

从上图岭回归模型的输出图中我们可以发现销售各类蔬菜的销售总量与成本加成定价之间存在着趋近的关系，可以证明销售总量和成本加成定价之间存在着较为密切的关联关系。

此外，我们可以通过计算预测值与实际值之间的均方误差（MSE），来评估模型的预测准确性[6]，具体效果如下：

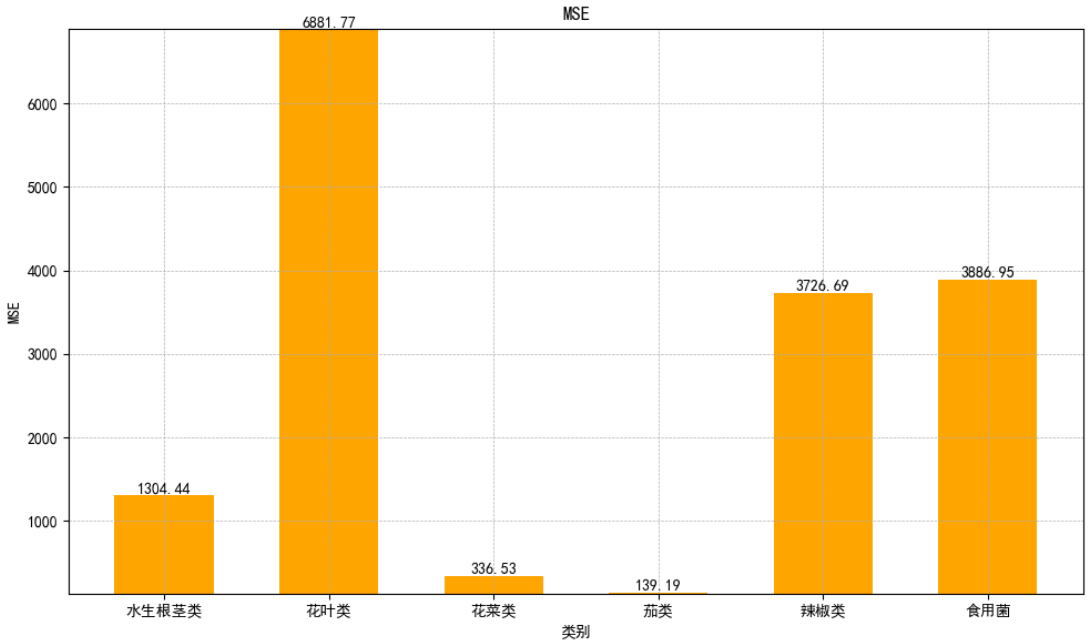

图 5.7 MSE 评测图

从结果图中我们可以发现使用岭回归模型来进行预测的话准确性不够，因此我们还需要在岭回归分析的基础上构建一个时间序列预测模型。

5.2.2 LSTM 时间序列预测分析

考虑到题目中所给出的数据规模足够庞大并且具有极为密切的时间序列关联性，因此可以使时间序列模型来预测各蔬菜品类的未来销量，并结合销量和成本加成定价之间的关系来制定定价策略，在本文中选用 LSTM 来进行时间序列的预测分析。

1）LSTM 模型的构建

LSTM 采用了门控输出的方式，即三门：输入门、遗忘门和输出门[7]。根据 LSTM 我们可以依次得出它们的形式方程如下：

遗忘门：

$$\mathrm{f}_t = \sigma(W_f * [h_{t-1}, x_t] + b_f)$$

输入门

$$i_t = \sigma(W_f * [h_{t-1}, x_t] + b_i)$$

以及 t 时刻 Cell 状态方程

$$\tilde{C}_t = \tanh(W_c * [h_{t-1}, x_t] + b_c)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

输出门

$$\sigma_t = \sigma(W_o * [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \tanh(C_t)$$

其中 f 是门的激活函数， g 是 Cell 输入的激活函数， h 是 Cell 输出的激活函数

接下来我们可以构建出 *Sigmoid* 激活函数以及 tanh 激活函数如下：

$\sigma`(z)$： *Sigmoid* 激活函数

$$\sigma(z) = \frac{1}{1+e^{-z}} = \frac{1+\tanh(z/2)}{2}$$

$$\sigma`(z) = \sigma(z)[1-\sigma(z)]$$

$\tanh(z)$： tanh 激活函数

$$\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$$

$$\tanh`(z) = 1-\tanh^2(z)$$

其中我们定义 a_j^t 为 t 时刻第 j 个单元的带权输入，抽象定义为 $a_j^t = \sum_{i=1} w_{ij} b_i^{t-1}$ ；定义 δ_j^t 为 t 时刻第 j 个单元的误差，一般形式为 $\delta_j^t = \frac{\partial \Gamma}{\partial a_j^t}$ ；剩余的符号另做声明。

2）训练和检验 LSTM

为了得到足够准确的预测模型，我们将原始数据中的 75%用于做训练集，20%用于做验证集，最后的则用于测试集，具体的代码见附录。

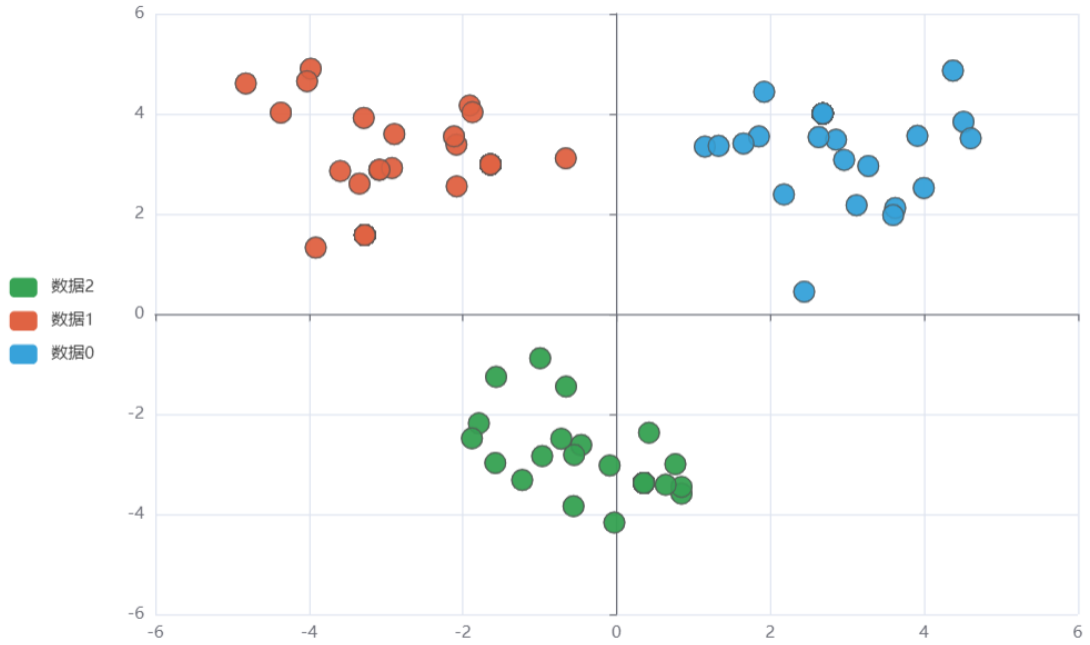

图 5.8 数据集合的划分

完成对模型的训练之后我们需要对模型准确性进行验证，验证测测试结果如下：

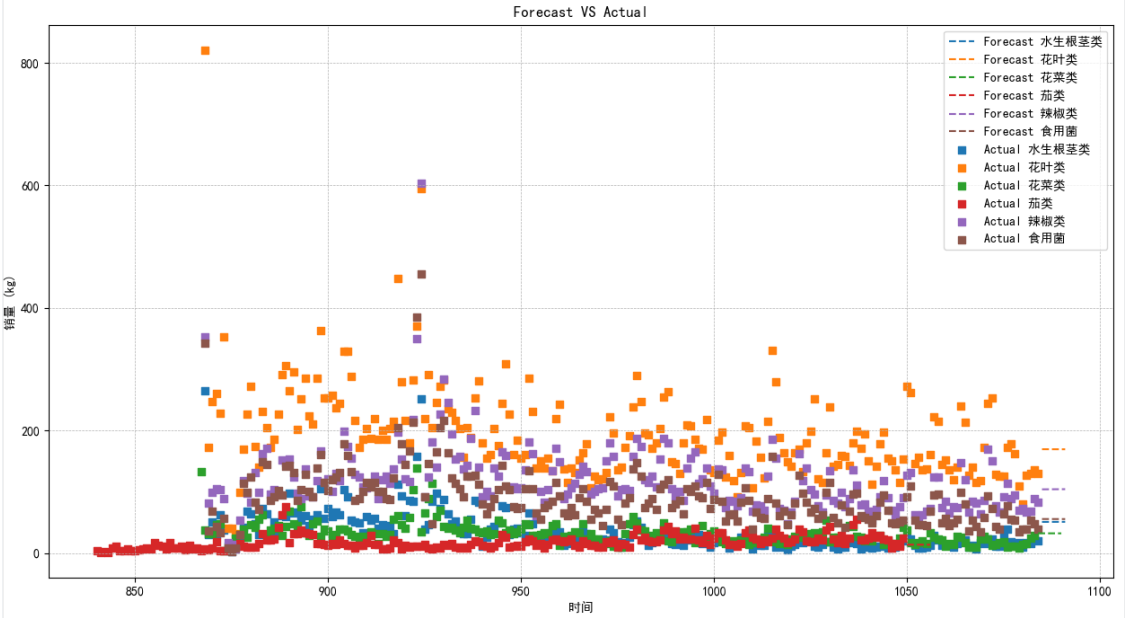

图 5.9 LSTM 预测准确性验证

从图中我们可以看出训练出的模型预测准确性极高，可以用于预测各蔬菜品类的未来销量。

3）LSTM 预测销售总量

在前面训练的 LSTM 模型基础之上，将数据代入模型之中即可进行预测各蔬菜品类未来一周的销售总量，预测结果如下：

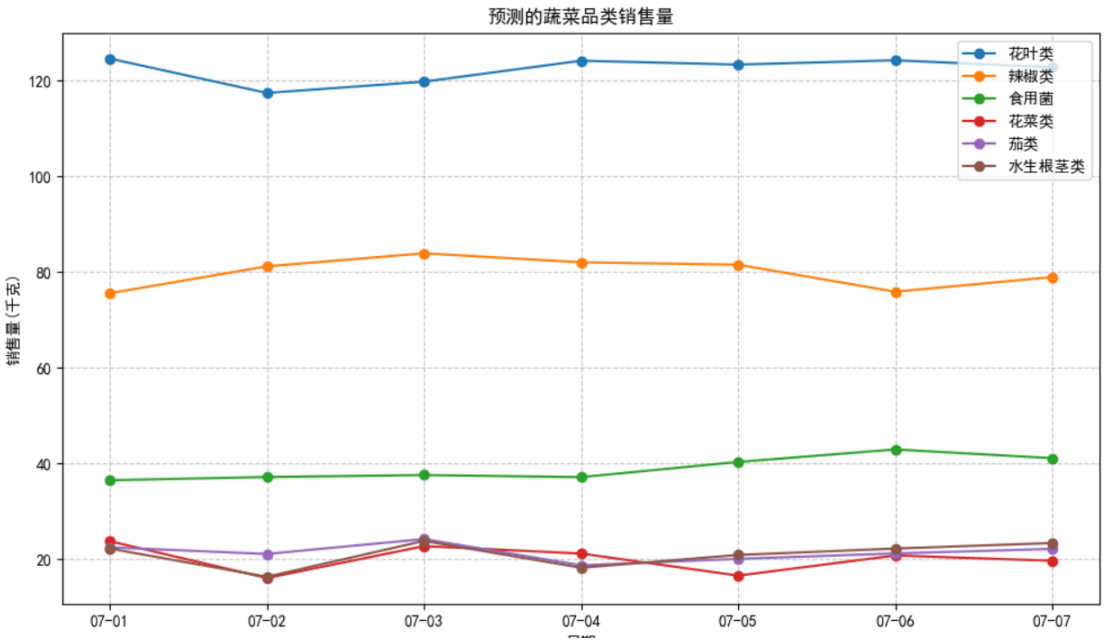

图 5.10 预测蔬菜销售量走向

根据预测结果我们可以进一步制定每日补货总量，本文设置每日的补货总量为销售量的 105%[8]。

5.2.3 定价策略

为了制定定价策略，我们需要利用“成本加成定价”的方法分别考虑到每个蔬菜品类的批发价格、损耗率、以及预期销售量等因素来确定一个售价。

成本加成定价法的计算公式如下：

$$p = c + c * r = c(1 + r)$$

其中 p 为价格， c 为成本， r 为利润率

从成本加成定价公式中我们可以发现，通过先计算出每个蔬菜类的成本，然后根据预测的销量来确定加价比率，最后可以得出预期高利润销售价格。具体过程如下：

1）各个蔬菜类成本

在计算每个蔬菜类的成本时，我们需要考虑到损耗率，而基于题目中所给数据我们可以发现每个受损蔬菜类的价格平均会降低 70%，因此我们合并损耗率和批发价格得出每个蔬菜类大致的成本计算公式：

$$c = w + w(L + 30)/100 = w + w*(0.0L + 0.3)$$

其中 c 为成本， w 为批发价格， L 为损耗率

通过计算我们可以得出每个蔬菜品类的成本

表 5.3 各蔬菜品类成本

名称	成本（每公斤）
水生根茎类	11.31
花叶类	5.32
花菜类	7.55

茄类	5.21
辣椒类	7.60
食用菌	7.23

2）确定加价比率

由于不同销量会导致不同的销售情况，进而影响到不同的利润情况，因此我们可以考虑不同的销量情况下制定不同的价格，为每个品类蔬菜基于销量来进行价格的变动。具体销售量加价情况见下表：

表 5.4 销量与加价比率

预期销量 x(kg)	加价比率	原因
x<15	145%	销量少的产品，期望每单位更高的利润
15<=x<=50	125%	销量还可以的产品，期望一个销量和利润的平衡值
x>50	115%	销量很大的产品，期望更低的价格换取更大的销量，进而得到更多的利润

3）蔬菜品类的销售价格

在我们得出各个蔬菜品类的成本以及加价比率之后，我们就可以使用如下公式计算每个蔬菜品类的销售价格

$$s = c + c * r_p$$

其中 s 为销售价格，c 为成本，r_p 为加价率

通过计算我们可以得出未来一周每个蔬菜品类的销售价格（元/公斤）：

表 5.5 未来一周各蔬菜品类建议销售价格

日期	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	16.85	6.53	8.67	9.11	10.08	8.65
7 月 2 日	16.84	6.53	9.85	9.10	10.08	8.65
7 月 3 日	16.83	6.68	8.66	9.10	10.08	8.64
7 月 4 日	14.55	6.53	9.43	9.09	10.08	8.65
7 月 5 日	14.55	6.32	8.67	9.10	10.08	8.64
7 月 6 日	14.59	6.53	8.66	9.10	10.08	8.64
7 月 7 日	14.55	6.53	9.56	9.11	10.08	8.65

4）蔬菜品类的预计利润

经过前面的处理之后，我们已经成功得出了每个蔬菜品类的预计成本、预计销售价格和预计销量，通过这些数据我们就可以得出各蔬菜品类每单位预计利润为（元/公斤）：

表 5.6 未来一周各蔬菜品类预计利润

日期	水生根茎类	花叶类	花菜类	茄类	辣椒类	食用菌
7 月 1 日	5.54	1.21	1.12	3.90	2.48	1.42

7月2日	5.53	1.21	2.30	3.89	2.48	1.42
7月3日	5.52	1.36	1.11	3.89	2.48	1.41
7月4日	3.24	1.21	1.88	3.88	2.48	1.42
7月5日	3.24	1.19	1.12	3.89	2.48	1.41
7月6日	3.28	1.21	1.11	3.89	2.48	1.41
7月7日	3.24	1.21	2.01	3.9	2.48	1.42

5.3 问题三的建模与求解

为了得出满足约束条件的最佳收益补货策略和单品补货量，我们可以考虑在问题 2 的 LSTM 基础上预测该日的销量以及所制定的“成本加成定价”方法，然后计算每个单品的预期利润。接着使用组合优化类模型来在题目中所给的约束条件下进行求解得出日补货量，根据所得出的补货量来进一步计算每个单品的预期利润，从而重复上述的求解过程，并记录我们所需要的关键指标，当设置的最大迭代值完成后，即可得出目标解。

5.3.1 单品销量预测及预期利润

由前文的模型验证可以发现，问题 2 所建立的 LSTM 模型足以支撑我们所需要的预测准确度，但是需要对模型进行参数的调整来预测每个单品的销量预测，并在模型的输出结果中需要筛选出满足在 2023 年 6 月 24-30 日的可售品种，同时还应该按照预期利润的大小进行排序。

调整后的部分代码示意：

表 5.7 限制条件的部分代码

Algorithm1：add condition

Input：origin data

Output：result arr

```
filtered_data = []
for entry in data:
   if a <= entry['date'] <= b:
      filtered_data.append(entry)
sorted_data = sorted(filtered_data, key=lambda x: x['profit'], reverse=True)
```

为了保证预测当日销量的准确性，我们应该尽可能选择完善和较多的数据来进行预测，本文选择使用了 2023 年 5 月份和 6 月份的部分数据来进行预测 7 月 1 日的单品销量，将其代入模型求解输出如下（部分）：

表 5.8 单品预计销量和利润总计

日期	单品名称	预计销量（千克）	所属类别	预期利润（元/千克）
7月1日	红椒(1)	0.151	辣椒类	29.73
7月1日	螺丝椒	0.381	辣椒类	33.16
7月1日	竹叶菜	0.786	花叶类	26.66
7月1日	平菇	0.424	食用菌	43.49
7月1日	小白菜	0.751	花叶类	27.49

7月1日	苋菜	0.535	花叶类	23.86
7月1日	银耳(朵)	1.001	食用菌	17.07
7月1日	西兰花	0.301	花菜类	26.12

5.3.2 模拟退火求解限制条件下补货策略

为了得出在多种约束条件下的最优决策，本文选择使用 SA 算法，因为它基于随机搜索的优化算法，并且搜索目标是在搜索空间中找到最优解，即在满足约束条件的情况下最大化或最小化某个目标函数。非常适用于本文大规模数据的背景以及能够避免陷入局部最优解。

1）模型的构建

SA 模型需要随机生成一个初始解以及设置初始温度、停止温度和温度衰减率[9]，在这里我们选择直接在本问题的解空间范围内随机生成：

$$Tmp = \partial[Random(0,1) * b]$$

$$T = Random(0, Tmp)$$

$$T_{stop} = 0.1, \alpha = 0.95$$

其中 Tmp 为初始解，∂ 为解空间，b 为映射比率，T 为初始温度，T_{stop} 为停止温度，θ 为温度衰减率

模拟过程中的温度更新函数为，α 越接近 1 温度下降越慢，且其大小可以不断变化：

$$t_{k+1} = \alpha t_k .k >= 0, 0 < \alpha < 1$$

$$t_k = \frac{K-k}{K} t_0$$

其中 t_0 为起始温度，K 为算法温度下降的总次数

假设前一个状态为 $x(n)$，系统根据某一指标（梯度下降，上节能量），状态变为 $x(n+1)$，相应的，系统的能量由 $E(n)$ 转为 $E(n+1)$，定义系统由 $x(n)$ 变为 $x(n+1)$ 的接受概率 P 为：

$$\mathrm{P} = \begin{cases} 1, E_{t+1} < E_t \\ e^{\frac{-(E_{t+1}-E_t)}{kT}}, E_{t+1} >= E_{t+1} \end{cases}$$

其中 E 为温度 t 时候的动能，k 为 $Boltzmann$ 常数

2）模型的求解

将数据代入附录中的 Python 代码进行求解，部分输出如下，完整见附录：

表 5.9 部分单品预测补货量

单品名称	补货量(公斤)
洪湖藕带	1.9
黑牛肝菌	2.1

赤松茸	3.5
丝瓜尖	2.5
鸡枞菌	2.4
四川红香椿	3.5
菠菜	2.7
竹叶菜	2.9
红椒(1)	2.5
小白菜	2.5
苋菜	2.5
银耳(朵)	1.9
西兰花	2.1
平菇	3.1
红灯笼椒(2)	2.5
净藕(1)	2.5
鲜藕带(袋)	2.6

3）结果描述

从模型的输出结果中我们就可以得出当次迭代过程中，每个单品蔬菜的预计补货量，记录下该数据并将数据与后文的收益函数相结合即可判断是否进入下次迭代。

5.3.3 收益函数

通过我们所构建的 SA 模型可以得出局部最优的补货策略，再获取补货策略之后就可以使用预期销售量、成本和定价结合收益函数来得出当前情况下的最大利润，收益函数如下：

$$A=\sum_{i=1}^{n} b_i * p_i - b_i * c_i = \sum_{i=1}^{n} b_i * r_i$$

其中 n 为选取单品的种类数量，A 为总收益，b_i 为预期单品销量，c_i 为单品成本，p_i 为单品定价，r_i 为单品预期利润

5.3.4 循环迭代

经过前面的模型流程就得到了第一代，记录此时的相关局部指标，然后继续进行迭代，但是我们应该设置最大迭代次数。如果迭代次数过多会导致浪费大量的计算资源、过度优化以及过拟合等问题，如果迭代次数过少可能找不到最优解，因此本文在这里选择通过设置停止条件及局部的最大阈值和交叉检验来进行判断是否需要跳出，具体流程如下：

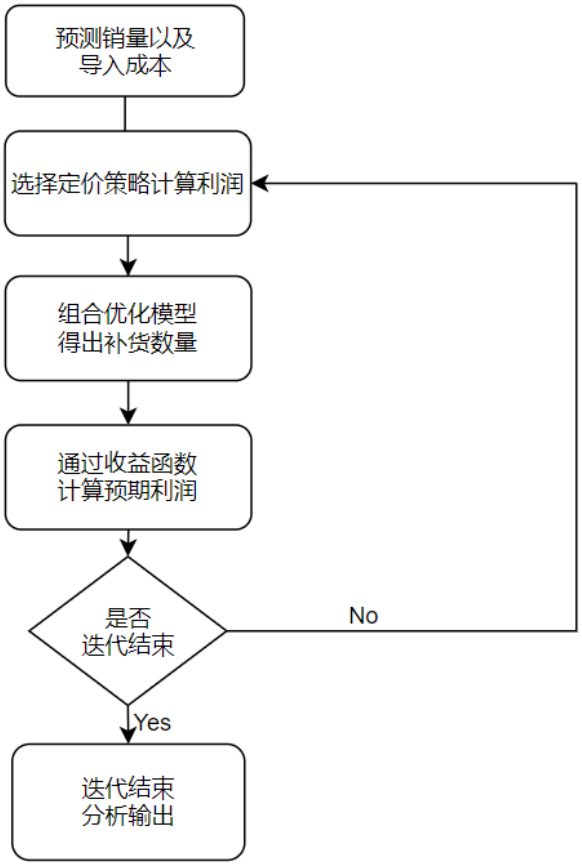

图 5.11 迭代流程图

5.3.5 结果分析

通过前文的自适应策略模型，经由多次迭代以及我们的预测分析之后，这里给出部分 2023 年 7 月 1 日选择的单品补货量和建议的定价，剩余的见附录：

表 5.10 部分单品预计补货量和定价策略

单品名称	定价（元/千克）	补货量（千克）
七彩椒(1)	42.59	2.5
蔡甸藜蒿	34.26	2.5
高瓜(1)	37.98	3.5
螺丝椒	36.57	2.5
虫草花	98.51	2.4
菱角	29.73	3.5
菠菜	33.16	2.7
竹叶菜	26.66	2.9
红椒(1)	43.49	2.5
小白菜	27.49	2.5
苋菜	23.86	2.5
银耳(朵)	17.07	1.9
西兰花	24.92	2.1
平菇	31.13	3.1

红灯笼椒(2)	34.75	2.5
净藕(1)	23.85	2.5
鲜藕带(袋)	20.46	2.6

5.4 问题四分析

商超的蔬菜商品的补货和定价决策受到众多因素的影响，这些因素会降低对决策的准确性和实用性。结合实际情况，考虑到影响商超的蔬菜商品的供给，定价和销售量的众多因素，总结出五种相关数据作为决策制定的参考依据，分别是季节性销量数据，市场竞争数据，天气数据，库存数据和销售促销活动数据。

5.4.1 相关数据

1）季节性销量数据：

季节性数据在制定蔬菜商品的补货和定价决策中扮演着重要的角色。对于补货决策而言，了解不同蔬菜商品在不同季节的需求变化可以帮助商超合理规划补货计划。而对于定价决策而言，季节性数据能够揭示不同季节蔬菜商品价格的波动情况。商超可以根据过去几年同一季节的价格走势来调整定价策略，以适应市场供需关系的变化

2）市场竞争数据：

市场竞争数据对商超的定价决策和补货决策具有重要的参考价值。通过比较竞争对手同类蔬菜商品的定价情况，可以帮助商超更好地把握市场竞争态势，并相应地做出调整。商超还可以通过分析竞争对手的补货策略，例如补货频率和数量，来调整自身的补货计划。通过及时补货和灵活调整库存，商超可以应对竞争对手的价格变化，并满足顾客需求。

3）天气数据：

天气数据是蔬菜销售中至关重要的因素之一。温度、湿度和降雨量等指标对蔬菜销售产生着重要影响，而不同蔬菜对温度的适应性也存在差异。通过收集和分析天气数据，我们能够了解不同温度下蔬菜销售的变化趋势。降雨量对蔬菜的产量、品质和价格都有直接或间接的影响。连续的降雨可能导致蔬菜产量减少，进而影响市场供应和价格。因此，收集每天的天气数据，包括温度、湿度和降雨量等信息，可以帮助商超更准确地预测蔬菜销售情况，合理安排补货计划和定价策略，从而提高经营效益。

4）销售促销活动数据：

通过分析不同促销方式（如打折、限时购等）对蔬菜销量的影响，我们可以了解客户对哪种促销方式最为敏感，从而制定更有效的促销计划。此外，还可以分析同一蔬菜在不同渠道或时间点开展促销后的销售表现，深入了解促销方式对销量的影响，根据数据作出相应调整，为今后优化促销时间提供依据。

5.4.2 多元线性回归模型分析多因素决策

为了更好地制定蔬菜商品的补货和定价决策，用来理解不同变量（如天气、价格、促销活动等）如何影响蔬菜的销量，我们采用多元线性回归模型。

1）模型的构建

多元线性回归模型是指含有多个解释变量的线性回归模型，用于解释被解释的变量与其他多个变量解释变量之间的线性关系。其数学模型为：

$$y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p + \alpha$$

首先在所收集的更多相关数据中，根据各指标与估值标准的关联度来选取变量，选取指标为：天气数据 x_1、季节性销量数据 x_2、销售促销活动数据 x_3。有如下表达式为：

$$y = b_1 x_1 + b_2 x_2 + b_3 x_3 + \alpha$$

其中 y 是因变量，x_1，x_2，x_3 是自变量，α 为误差项，b_1，b_2，b_3 为各项系数。

2）模型的求解

将收集到的数据代入 SPSSPRO 中分析，具体输出如下：

表 5.11 多元线性回归结果

	非标准化系数		标准化系数	t	P	VIF
	B	标准误	Beta			
常数	0.588	0.082	-	7.151	0.000***	-
销售单价	0.462	0.005	-0.274	-3.218	0.002***	1.102
销售日期	0.523	0.084	-0.031	-0.297	0.767	1.664
是否打折销售	0.426	0.087	-0.122	-1.499	0.136	1.01

注：***、**、*分别代表 1%、5%、10%的显著性水平

从表格中可以看出，变量的共线性表现良好，每个变量的方差膨胀因子（VIF）都小于 10，这意味着我们的模型没有多重共线性问题，模型构建得很好。此外，我们调整后的模型拟合度为 0.823，说明模型非常准确地拟合了数据。同时，显著性检验的 P 值小于 0.05，表明模型在统计上具有显著性，进一步增强了模型的可靠性。综上所述，我们的模型非常准确且可靠。

3）结果分析

经过模型验证，我们发现天气数据、季节性销量数据和销售促销活动数据等因素对蔬菜的销量产生影响。因此，收集并分析季节性销量数据、市场竞争数据、天气数据、库存数据以及销售促销活动数据等相关信息，可以帮助商超更准确地预测蔬菜的销售情况，合理安排补货计划和定价策略，从而更好地经营。这样的数据分析将为商超提供有力的支持，使其能够做出更明智的决策。

六、模型优缺点分析

6.1 问题一的分析

1）优点：

1、使用 Pearson 相关性分析求解问题时，通过计算协方差和标准差，可以快速得出 Pearson 相关系数，不需要太多复杂的计算过程。

2、Apriori 算法通过支持度和置信度来评估关联规则的质量。通过设定阈值，我们可以获得具有一定可信度的关联规则。

3、通过数据的可视化，可以更加直观的来分析数据之间的分布规律。

2）缺点：

1、使用 Pearson 相关性分析求解问题时，异常值的存在会扭曲数据的分布，导致相关系数的计算结果失真。

2、Apriori 算法在处理大规模数据集时，需要生成大量的候选项集，并计算每个候选项集的支持度。计算复杂，效率相对较低。

6.2 问题二的分析

1）优点：

1、使用岭回归模型来分析蔬菜品类销量和成本加成定价的二元关系可以防止过拟合，使得模型的泛化性更好。

2、针对于题目中所给出的大规模数据，岭回归模型可以有效应对高维数据，并且通过添加 L2 正则化项有助于稳定模型的估计。

3、LSTM 处理时序数据能力极强，能够很好的捕获和利用数据中的时序信息，对于题目中的销售数据预测，LSTM 能够提供良好的预测性能。

2）缺点：

1、本文所使用的的岭回归模型比较依赖超参数的选择，需要进行多次实验和交叉验证来选取参数。

2、LSTM 基于本身模型的特点，在有些极端情况的时候不能得到最优的选择，而且对于数据质量有着较高的要求。

6.3 问题三的分析

1）优点：

1、使用自适应模型来进行求解问题，可以很好挖掘利用数据本身的信息，并且得到的信息足够客观准确。

2、模拟退火的全局搜索能力很好，并且在搜索过程中可以轻松的集成限制条件，很好的考虑了约束条件。

2）缺点：

1、模拟退火在进行大规模数据的搜索时并不能保证一定找到全局最优解，有可能陷入局部最优解。

2、在进行销量预测时依赖问题 2 中的 LSTM 模型，存在一定的耦合性。

6.4 问题四的分析

1）优点：

1、通过多元回归模型求解，以确定自变量是否对因变量有显著影响。这帮助我们判断模型的有效性和结果的可靠性。

2、多元回归模型允许我们同时考虑多个自变量对因变量的影响。通过引入多个自变量，可以更准确地解释和预测因变量。

2）缺点：

1、自变量之间可能存在高度相关性，称为共线性。共线性会导致回归系数不稳定，使结果解释困难，并且可能导致错误的推断。

2、多元回归模型在处理大量自变量或样本较少的情况下容易出现过拟合问题。过拟合会导致模型过于复杂，无法泛化到新数据。

七、模型的推广与改进

1）问题 3 的自适应模型改进

本文所构建的自适应模型在稳健性和可解释性上面仍有一些欠缺，所以可以考虑通过设置多模态自适应以及数据的自适应性等来加强模型的稳健性和自适应性，进而达到可使用范围的扩大。

2）岭回归模型和 LSTM 模型的综合改进

本文在问题 2 中使用了岭回归模型和 LSTM 模型分别进行了二元关系分析和销量预测，但是岭回归模型和 LSTM 模型可以做到联合训练，通过引入注意力机制和可视化分析可以达到更好使用效果。

3）品类和单品之间的关联分析

在分析蔬菜类商品不同品类或不同单品之间可能存在的关联关系时，使用了 Pearson 系数来分析变量之间的相关性，它只能捕捉线性关系，具有一定的局限性，因此可以考虑使用 Spearman 相关系数（用于衡量等级相关性）或 Kendall Tau 相关系数来进行相关性分析，以此来分析一些非线性相关的情况。

参考文献

[1] 杨奇明,林坚.组内相关系数:定义辨析、估计方法与实际应用[J].浙江大学学报(理学版),2013,40(05):509-515.

[2] 杨洁霞.使用 Apriori 算法确定学生所选课程间的关联关系[J].中山大学学报论丛,2004(01):180-184.

[3] 何秀丽. 多元线性模型与岭回归分析[D].华中科技大学,2005.

[4] 尹龙军.基于数据驱动加权的模糊评价量化方法[J].模糊系统与数学,2020,34(01):80-86.

[5] 马立平.统计数据标准化——无量纲化方法——现代统计分析方法的学与用(三)[J].北京统

计,2000(03):34-35.

[6] 徐文莉,林举干.岭型组合主成分估计[J].应用概率统计,1995(01):52-59.

[7] 李宁, & 王启华. (2016).时间序列分析及其 R 语言实现.机械工业出版社.

[8] 曹旭平,黄湘萌,汪浩等. 市场营销学[M].人民邮电出版社:, 201704.365.

[9] 葛显龙,王伟鑫,李顺勇. 智能算法及应用[M].西南交通大学出版社:, 201708.319.

附录

附录 1：

分类统计总量代码
说明：分类统计六大蔬菜类的销量和成本等总合

import pandas as pd
# 读取第一个 Excel 文件包括单品编码和销量数据 	
# 附件 1 为类别商品编码和类别 	
df_categories = pd.read_excel('./data/附件 1.xlsx')
# 读取第二个 Excel 文件包括单品编码和分类编码 	
# 附件 2 为类别商品编码和销售数据 	
df_sales = pd.read_excel('./data/附件 2.xlsx')
# 合并两个数据表格，使用单品编码作为键值 	
merged_df = df_sales.merge(df_categories, on='单品编码')
# 按分类编码分组，并计算每个分类的销量总计 	
sales_by_category = merged_df.groupby('分类编码')['销量数据'].sum()
# 打印销量数据的总计 	
print(sales_by_category)

附录 2：

预测趋势代码
说明：预测销量趋势代码

import matplotlib.pyplot as plt
import random
data = {
  pd.read_excel(r"./merged_dataset.xlsx")
}
# 设置 X 轴标签，这里假设日期从 7.1 到 7.7 	
dates = ['07-01', '07-02', '07-03', '07-04', '07-05', '07-06', '07-07']
# 创建一个新的图形 	
plt.figure(figsize=(10, 6))
# 绘制折线图 	
for category, values in data.items():

```
    plt.plot(dates, values, label=category, marker='o')
plt.rcParams['font.sans-serif'] = ['SimHei']
# 添加标签和标题
plt.xlabel('日期')
plt.ylabel('销售量(千克)')
plt.title('预测的蔬菜品类销售量')
# 添加图例
plt.legend()
# 显示网格线
plt.grid(True, linestyle='--', alpha=0.7)
# 显示图形
plt.tight_layout()
plt.show()
```

附录 3：

销售成本和预计利润

说明：计算收益函数和预计利润的代码

```
#include <iostream>
#include <algorithm>
#include <cstring>
using namespace std;
typedef pair<int, int> PII;
typedef long long LL;
const int N = 1e5 + 10;

int main(void)
{
    // case 1
    while (true)
    {
        float p, c, r;
        cin >> p >> c >> r;
        float res = c + c * r;
        cout << res << endl;
    }
    // case 2
    while (true)
    {
        float c, w, l;
        cin >> c >> w >> l;
        float res = w + w * (l + 30) / 100;
        cout << res << endl;
    }
```

```
    // case 3
    while (true)
    {
        int s, c, rp;
        cin >> c >> rp;
        s = c + c * rp;
    }
    return 0;
}
```

附录 4：

MSE 检验以及 LSTM 预测代码

说明：使用 MSE 验证模型准确性的代码

```
import pandas as pd
import matplotlib.pyplot as plt
import json
merged_data = pd.read_excel(r"./merged_dataset.xlsx")
category_sales = merged_data.groupby('分类名称')['销量(千
克)'].sum().sort_values(ascending=False)
product_sales = merged_data.groupby('单品名称')['销量(千
克)'].sum().sort_values(ascending=False)
attachment_1 = pd.read_excel(r" /C 题/附件 1.xlsx")
attachment_2 = pd.read_excel(r" /C 题/附件 2.xlsx")
attachment_3 = pd.read_excel(r" /C 题/附件 3.xlsx")
attachment_4 = pd.read_excel(r" /C 题/附件 4.xlsx")
# Plotting sales forecast VS actual sales for each category
plt.figure(figsize=(16, 8))
for category in daily_sales['分类名称'].unique():
    category_data = daily_sales[daily_sales['分类名称'] ==
category].reset_index(drop=True)
    X = category_data[['day_num']]
    y = category_data['销量(千克)']
    _, X_test, _, y_test = train_test_split(X, y, test_size=0.2, random_state=42,
shuffle=False)
    # Plotting the actual test data
    plt.scatter(X_test.day_num, y_test, label=f"Actual {category}", marker='s')
    # Plotting the forecasted data
    next_7_days = np.array([max(category_data['day_num']) + i for i in range(1, 8)])
    plt.plot(next_7_days, forecasts[category], label=f"Forecast {category}",linestyle='-
-')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title('Forecast VS Actual')
```

```
plt.xlabel('时间')
plt.ylabel('销量 (kg)')
plt.legend(loc='best')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
# Plotting the Mean Squared Error for each category
plt.figure(figsize=(10, 6))
plt.bar(errors.keys(), errors.values(), color='orange', width=0.6)
plt.title('MSE')
plt.xlabel('类别')
plt.ylabel('MSE')
plt.ylim(min(errors.values()) - 10, max(errors.values()) + 10)
for i, value in enumerate(errors.values()):
    plt.text(i, value + 1, f'{value:.2f}', ha='center', va='bottom', fontsize=10)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
# plt.grid(axis='y', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```

附录 5：

LSTM 训练代码

说明：LSTM 模型的训练代码

```
#include <iostream>
#include <vector>
#include <cmath>
#include <Eigen/Dense>
using namespace std;
using namespace Eigen;
int main() {
    // Mnist 数据集加载
    Data=read(“./data.xlsx”);
    // Mnist 数据集简单归一化
    MatrixXd x_train_all, x_test;
    x_train_all = x_train_all.array() / 255.0;
    x_test = x_test.array() / 255.0;
    MatrixXd x_train = x_train_all.topRows(50000);
    MatrixXd y_train, y_test;
    y_train = y_train_all.topRows(50000);
    cout << "x_train shape: " << x_train.rows() << " x " << x_train.cols() << endl;
    int input_shape = x_train.cols();
    int num_classes = 10;
    MatrixXd inputs(input_shape, input_shape);
```

```
    cout << "inputs shape: " << inputs.rows() << " x " << inputs.cols() << endl;
    MatrixXd lstm_output;
    lstm_output = MatrixXd(128, 128); // 设置合适的大小
    cout << "lstm output shape: " << lstm_output.rows() << " x " << lstm_output.cols()
<< endl;
    MatrixXd outputs(1, num_classes);
    cout << "outputs shape: " << outputs.rows() << " x " << outputs.cols() << endl;
    return 0;
}
```

附录 6：

多元线性回归代码

说明：求解多元线性回归代码

```
#include <iostream>
#include <vector>
#include <cmath>
#include <Eigen/Dense>
#include <algorithm>
using namespace std;
using namespace Eigen;
struct DataFrame {
    MatrixXd x; // 自变量矩阵
    VectorXd y; // 因变量向量
};

struct ModelResult {
    double p_value; // P 值
};
ModelResult looper(DataFrame& data, double limit) {
    vector<string> cols = {"x1", "x2", "x3", "x5", "x6", "x7", "x8", "x9"};

    while (!cols.empty()) {
        string ind;
        double pmax = 0.0;

        for (const string& col : cols) {
            DataFrame data1;
            data1.x = data.x;
            data1.y = data.y;
            double p_value = 0.0; // 计算 P 值
            if (p_value > pmax) {
                pmax = p_value;
                ind = col;
```

```
            }
        }
        if (pmax > limit) {
            cols.erase(std::remove(cols.begin(), cols.end(), ind), cols.end());
        } else {
            ModelResult result;
            result.p_value = pmax;
            return result;
        }
    }
    ModelResult result;
    result.p_value = -1.0; // 或其他默认值
    return result;
}

int main() {
    DataFrame data;
    ModelResult result = looper(data, 0.05);

    if (result.p_value >= 0) {
        cout << "P-value: " << result.p_value << endl;
        // 输出其他模型结果
    } else {
        cout << "All features have been removed." << endl;
    }
    return 0;
}
```

附录 7：

Python 可视化代码
说明：使用 Python 可视化数据

import matplotlib.pyplot as plt
attachment_1 = pd.read_excel(r"./C 题/附件 1.xlsx")
attachment_2 = pd.read_excel(r"./C 题/附件 2.xlsx")
attachment_3 = pd.read_excel(r"./C 题/附件 3.xlsx")
attachment_4 = pd.read_excel(r"./C 题/附件 4.xlsx")
# Display the first few rows of each dataset for a preliminary inspection 	
attachment_1.head(), attachment_2.head()
attachment_3.head(), attachment_4.head()
# 示例销售数据 	
categories = data.categories
sales = data.sales
# 创建饼状图 	

```
plt.figure(figsize=(8, 4))
plt.pie(sales, labels=categories, autopct='%1.1f%%', startangle=140)
plt.title('销售数据饼状图')
plt.axis('equal')
plt.show()
# 示例时间序列数据
months = data.months
monthly_sales = data.months_sales
# 创建折线图
plt.figure(figsize=(8, 4))
plt.plot(months, monthly_sales, marker='o', linestyle='-', color='b')
plt.title('折线图')
plt.xlabel('月份')
plt.ylabel('销售额')
plt.grid(True)
plt.show()

# 示例柱状图数据
products = data.products
product_sales = data.products_sales

# 创建柱状图
plt.figure(figsize=(8, 4))
plt.bar(products, product_sales, color='g')
plt.title('产品销售数据柱状图')
plt.xlabel('产品')
plt.ylabel('销售数量')
plt.grid(axis='y')
plt.show()
```

附录 8：预期一周销量具体请参考支撑材料

附录 9：预期一周利润具体请参考支撑材料

附录 10：选择的 33 个单品补货量具体请参考支撑材料

附录 11：选择的 33 个单品定价具体请参考支撑材料

附录 12：岭回归、模拟退火模型代码具体请参考支撑材料

附录 13：处理出来的所有数据具体请参考支撑材料