**中国粮食安全与风险归因分析与测度研究**

近年来，气候环境变化，对粮食安全影响日益加剧。引发了全球学者对气候变化的广泛关注。我国作为世界上最大的发展中国家，拥有全球近五分之一人口，耕地却仅占世界耕地面积9%。粮食安全与稳定增长对我国社会的稳定与和谐具有重大意义。然而中国疆域辽阔，气候种类多样，气候环境复杂，对粮食生产有重要影响。因此，探究气候环境对粮食生产具有重大意义，本文通过分析自然环境因素与人文社会因素对粮食产量的作用关系，探究自然因素与粮食生产之间的因果联系，预测未来气候变化对粮食生产的影响，从而指导粮食生产。

种粮是“农业芯片”，是中国粮食安全的核心。为进一步拓深中国粮食安全的研究内核，本文构建了一种全新的种业危机预计统计指标体系。体系分为 、 、 、 、等 大板块，分别从————几个方面考察了 ，为了突出 ，还 。

最后基于本文的研究，提出几条针对湖南构建粮食安全的政策建议。

**针对问题一：**

**针对问题二：**

**针对问题三：**

**针对问题四：**

**关键字：**粮食安全、种业测度、

**一、问题提出：**

**1.1问题背景**

粮食安全事关千家万户的民生福祉。2022年、2023年中央一号文件，再次把粮食安全作为全面推进乡村振兴的底线任务。当今世界正面临“百年未有之大变局”，气候变暖、地域冲突、逆全球化、疾病蔓延等多重挑战所导致的复合型风险时代已经到来，外部的政治环境、经济环境以及自然环境都在急速变化，不确定性的外部环境进一步加大保障粮食安全的难度。

粮食生产与气候生态保持着高度的因果联系，特别是在目前生态环境遭遇一定程度的伤害、极端天气反复发作的条件下，气候变异已经成为直接影响粮食安全的关键因素。随着全球变暖趋势仍在持续，极端天气事件频发，荒漠化和土地退化进一步加快，并对粮食安全造成冲击。联合国粮农组织研究报告指出，今后20年至50年间的农业生产将受到气候变化的严重冲击，并进而严重影响全球超长期的粮食安全。

保障粮食安全,要害是种子和耕地。一方面，种子是国家粮食安全的源头，种业安全是国家安全的重中之重，但我国种业存在诸多“卡脖子”风险，必须未雨绸缪加强防范。另一方面，全球面临耕地危机，中国也不例外。城市化、工业用地扩张和环境污染导致耕地面积减少、土地质量下降，威胁粮食安全。要坚守耕地红线，坚决遏制各类耕地“非农化”、防止耕地“非粮化”。

**1.2已知条件**

**1.3问题重述：**

上述背景，提供了粮食安全研究的科学问题。**请收集相关数据，并通过数学建模与算法设计，完成以下相关研究：**

定量刻画气候生态与粮食生产之间的因果联系，预测未来气候变化对农业生产的冲击效应。

分析中国耕地“非粮化”的动因与机理，定量描述中国种粮耕地变化的时空特征与演变趋势。

识别粮食种业“卡脖子”风险的主要影响因素，构建种业危机预警统计指标体系与破解路径。

根据你们的研究结论，自拟标题，提供500字左右的“献策湖南”金点子1份。

**二、问题分析**

**2.1问题一分析**

问题一要求

**2.2问题二分析**

问题二要求

**2.1问题三分析**

问题三要求

**2.1问题三分析**

问题三要求

**三、模型假设**

1. 假设一
2. 假设二

**四、符号说明**

**五、模型的建立与求解**

**针对问题一：**

**5.1 数据来源与预处理**

通过查阅相关文献、阅读行业报告，本文选取中国2010年-2023年粮食产量、农作物受灾面积、平均温度、农业总产值、平均降雨量、第三产业就业人员、农业机械总动力、农药使用量作为探究气候环境变化对粮食生产的影响，数据分为两个维度刻画粮食生产，即自然环境因素的与人文社会因素。

表 自然与人文数据说明及描述性统计

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **数据变量** | **符号** | **含义** | **单位** | **mean** | **std** | **min** | **max** |
| **粮食产量** | Yield | 一年内的粮食总产量 | 万吨 | 0.88 | 0.00 | 0.87 | 0.88 |
| **农作物受灾面积** | Disaster | 一年内受到旱涝灾害影响面积 | 千公顷 | 2.30 | 0.04 | 2.23 | 2.35 |
| **平均温度** | Temperature | 全国或时期内的平均气温 | 摄氏度 | 2.06 | 0.04 | 1.96 | 2.11 |
| **农业总产值** | Output | 农业产出的总价值 | 亿元 | 10.97 | 0.27 | 10.49 | 11.34 |
| **平均降雨量** | Rainfall | 全国或时期内的平均降雨量 | mm | 22.94 | 0.84 | -24.17 | -21.33 |
| **第一产业就业人员** | primary | 从事第一产业的就业人口数量 | 万人 | 10.38 | 0.11 | 10.18 | 10.49 |
| **农业机械总动力** | Machinery | 农业生产中使用机械设备总功率 | 万千瓦 | 11.55 | 0.06 | 11.44 | 11.62 |
| **农药使用量** | Pesticide | 一年内农业生产中使用农药总量 | 万吨 | 5.05 | 0.16 | 4.82 | 5.20 |

**5.1.2 OLS模型原理与建立**

多元线性回归是一种统计模型，旨在揭示因变量（或响应变量）与多个自变量（或预测变量）之间的关系。其基本原理是利用多个自变量的线性组合来预测因变量的值，同时分析这些自变量对结果的影响因素。多元线性回归的核心在于通过最小化残差平方和的思想来建立模型，以最佳拟合样本数据并估计模型参数。其公式原理如下：

有一个线性模型：

其中，是因变量，是自变量，是待估计的模型参数，是误差项。

OLS的目标是找到一组参数使得观测值与模型预测值之间的残差平方和最小。残差定义为观测值与模型预测值之间的差异，即：

残差平方和为：

要最小化残差平方和，可以对关于参数求偏导数，并令其等于零。

令残差平方和对参数的偏导数为零：

对于：

对于（其中）：

将这些方程整理为矩阵形式：，其中

* 是一个的矩阵，每行代表一个样本的自变量（包括截距项），每列代表一个自变量；
* 是一个的向量，包含个样本的因变量数据；
* 是一个的向量，包含了模型的参数。

解这个方程，可以得到参数的估计值：

VIF（Variance Inflation Factor，方差膨胀因子）是用来检测多元线性回归模型中自变量之间是否存在共线性（collinearity）的指标。共线性指的是自变量之间存在高度相关性的情况，这将导致模型估计的不稳定性和解释力下降。

自变量的VIF计算公式如下：

其中，是将作为因变量，其他自变量作为自变量进行线性回归后得到的决定系数（R平方）越接近1，表示能够被其他自变量很好地解释，VIF值就越大，说明共线性越强。

**5.1.3 VAR模型原理与建立**

VAR（Vector Autoregression，向量自回归模型）是一种用于描述多个时间序列变量之间动态关系的计量经济学模型。可以同时考虑多个时间序列变量之间的相互作用和影响。其原理是认当期变量都是往期所有变量的线性组合，因此通过当期变量对所有变量的若干滞后变量进行回归，探究变量之间的联动关系。VAR模型最关键的两个参数N变量个数与K最带滞后阶数。

假设两个变量存在联系，通过内联建立两个VR模型，就可探究变量之前的关系。

其中,写出矩阵形式为：

设，得到个变量滞后K阶的VAR模型如下：

其中：

为了使得VAR模型结论可靠，需使得的所有特质值都放在单位原中，其中最常用的是ADF（Augmented Dickey-Fuller）单位根检验方法，用于检验一个时间序列是否具有单位根（即是否是非平稳的）。其原理如下：

考虑一个一阶差分自回归过程（AR(1)）：

其中，是时间序列变量，在时间的值，是自回归系数，是白噪声误差。

ADF检验的目标是检验 𝜌=1*ρ*=1 的原假设（即存在单位根，序列是非平稳的），与备择假设 𝜌<1*ρ*<1 或 𝜌>1*ρ*>1 相对立（即序列是平稳的）。

ADF检验通常使用以下的回归模型来进行检验：

其中，是时间序列的一阶差分，是时间趋势项，是滞后一期的时间序列，是的一阶差分的滞后项，是截距，是时间趋势的系数，是的系数，是 的系数，是误差项。

**5.1.4 结果分析**

**（1）多元线性回归结果分析**

对数据进行共线性检验结果如下：

表 自然与人文数据VIF检验

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable** | 农作物受灾面积 | 平均温度 | 农业总产值 | 平均降雨量 | 农业机械总动力 | 农药使用量 | 第一产业就业人员 |
| **VIF** | 14.06106 | 2.071176 | 95.26832 | 2.141174 | 3.554746 | 13.50648 | 24.17378 |

分析结果存在农作物受灾面积、农业总产值、农药使用量、农药使用量、第一产业就业人员四个变量数值大于10，说明数据存在严重的共线性，提出VIF值最大的变量农业总产值，重新进行检验。结果如下：

表 调整后的自然与人文数据VIF检验

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Variable** | 农作物受灾面积 | 平均温度 | 平均降雨量 | 第一产业就业人员 | 农业机械总动力 | 农药使用量 |
| **VIF** | 7.736098 | 1.456972 | 1.225765 | 7.32979 | 2.188289 | 6.196545 |

所有变量VIF均在10之内，因此可以认为剔除农业总产值后数据不存在共线性。

带入数据构建多元线性回归模型得到：

表 OLS Regression Results

==============================================================================

Dep. Variable: 粮食产量 R-squared: 0.999

Model: OLS Adj. R-squared: 0.997

Method: Least Squares F-statistic: 783.7

Date: Sun, 12 May 2024 Prob (F-statistic): 1.56e-09

Time: 03:27:49 Log-Likelihood: 65.265

No. Observations: 14 AIC: -116.5

Df Residuals: 7 BIC: -112.1

Df Model: 6

Covariance Type: nonrobust

==============================================================================

coef std err t P>|t| [0.025 0.975]

------------------------------------------------------------------------------

const 15.8528 0.132 119.885 0.000 15.540 16.165

农作物受灾面积 -0.0423 0.006 -6.557 0.000 -0.058 -0.027

平均温度 0.0135 0.002 7.242 0.000 0.009 0.018

平均降雨量 -0.0819 0.010 -7.929 0.000 -0.106 -0.057

农业机械总动力 9.395e-07 2.3e-07 4.085 0.005 3.96e-07 1.48e-06

农药使用量 0.0017 9.23e-05 18.945 0.000 0.002 0.002

第一产业就业人员-0.4799 0.015 -32.344 0.000 -0.515 -0.445

==============================================================================

Omnibus: 1.438 Durbin-Watson: 2.314

Prob(Omnibus): 0.487 Jarque-Bera (JB): 0.388

Skew: -0.397 Prob(JB): 0.824

Kurtosis: 3.182 Cond. No. 1.59e+07

==============================================================================

由于R-squared为0.999以及Prob<0.05模型有很强的准确性与可靠性，由数据回归表格可知粮食产量与其他变量可以建立模型如下：

系数表示了自变量对因变量的影响程度。例如，农作物受灾面积的系数为-0.0423。这意味着，当农作物受灾面积变化一个单位时，粮食产量会减少0.0423个单位。换句话说，每增加一个单位的农作物受灾面积，粮食产量平均会减少0.0423个单位。

结合表达式分析，可知当前的环境气温条件下，温度适当上升有利于我国粮食产量的提高，同时目前降雨对于大多地区已经发挥阻碍作用，同时极端天气对粮食产量依旧有非常大的影响。

1. **VAR模型结果分析**

为了保证数据的平滑性需要对为平滑数据进行差分处理，经过处理后得到的结果如下：

表 ADF检验结果

|  |  |  |
| --- | --- | --- |
| **变量** | **ADF Statistic** | **p-value** |
| **粮食产量** | -8.91211 | 1.1E-14 |
| **农作物受灾面积** | -6.84422 | 1.76E-09 |
| **平均温度** | -5.84562 | 3.69E-07 |
| **农业总产值** | -2.90547 | 0.044715 |
| **平均降雨量** | -7.98961 | 2.49E-12 |
| **农业机械总动力** | -3.22225 | 0.018732 |
| **农药使用量** | -4.33522 | 0.000386 |
| **第一产业就业人员** | -3.6961 | 0.004168 |

由表格分析可知，数据基本上在1%的可信度下（除了农药使用量是5%），拒绝原假设，因此差分后的数据是平滑的。

VAR结果如下：

表 Summary of Regression Results

==================================

Model: VAR

Method: OLS

Date: Sat, 11, May, 2024

Time: 23:15:18

--------------------------------------------------------------------

No. of Equations: 7.00000 BIC: 58.0988

Nobs: 11.0000 HQIC: 57.6860

Log likelihood: -420.409 FPE: 1.32910e-2

AIC: 57.8456 Det(Omega\_mle): 7.22834e+24

--------------------------------------------------------------------

Results for equation 粮食产量

========================================================================

coefficient std. error t-stat prob

------------------------------------------------------------------------

const 98.630909 519.050961 0.019 0.084

========================================================================

由结果Prob<0.1与FPE<0.05分别在10%与5%的置信水平下拒绝原假设，即本研究结果可信。

表 VAR模型检验结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variable** | **Coefficient** | **Std. Error** | **t-stat** | **Prob** |
| **粮食产量** | 98.630909 | 519.050961 | 0.19 | 0.0849 |
| **农作物受灾面积** | -1311.090909 | 1450.66397 | -0.904 | 0.0366 |
| **平均温度** | 0.066818 | 0.529742 | 0.126 | 0.09 |
| **平均降雨量** | -0.008958 | 0.050256 | 0.178 | 0.0859 |
| **农业机械总动力** | 730.748182 | 1560.25886 | 0.468 | 0.064 |
| **农药使用量** | 0.088182 | 1.468275 | 0.06 | 0.0952 |
| **第一产业就业人员** | 75.363636 | 169.931017 | 0.443 | 0.0657 |

分析结果可知，极端天气与降雨增加不利于粮食的生产。

**针对问题二：**

**针对问题三：**

**5.3.1 数据来源与预处理**

为构建研究中国种业危机预警统计指标体系，本文数据参考了中国种字研究数据库[[1]](#footnote-1)、Resset行业数据库[[2]](#footnote-2)、国家数据[[3]](#footnote-3)等国内权威数据库，部分数据使用专利之星、知网当学术平台通过关键字检索得到。

**5.3.2 中国种业危机预警统计指标构建**

中国种业危机涵盖范围广阔，涉及领域丰富。在我国科学技术迅猛发展的时代背景下，自主研发与创新能力的持续增强为种业带来了新的机遇与挑战。因此，对于种业危机的全面评价，必须采取多层面、多维度的分析方法。

本研究在深入剖析现有学术文献与行业报告的基础上，围绕粮食安全、种子质量、市场稳定性及行业健康发展四大核心领域，精心构建了中国种业危机预警指标体系。其中，粮食安全维度着重考量种子的供应能力，以确保满足我国人民日益增长的粮食需求；种子质量维度则聚焦于育种技术的发展水平，评估种子在育种过程中的整体表现；市场稳定性维度强调种子品种与播种质量，要求种子具有高纯度、饱满完整、无病虫害且活力旺盛的特性；而行业健康发展维度则侧重于评估中国种业市场的整体健康状态。

基于上述四个维度，本研究构建了一个完整的预警指标体系，旨在全面、系统地监测和评估中国种业的发展状况与潜在风险。具体指标体系如下表所示，为相关决策部门提供了有力的数据支撑和决策依据。

|  |  |  |
| --- | --- | --- |
| 一级指标 | 表征指标 | 属性 |
| 数量安全A | A1:种业领域内不同种类植物种粮类别的数量 | + |
| 要求种子的供应满足我国人民日益增长的粮食需要 | A2:种业领域内粮食作物的产量 | + |
| A3:我国农作物种子进口情况 | - |
| 质量安全B | B1:种业专利发表量:IPC分类号为A01的专利年度发表量 | + |
| 中游的育种方面 | B2:种业领域内的学术研究论文发表量 | + |
|  | B3:种业领域内进行研究和开发的资金投入水平 | + |
| 市场安全C | C1:中国转基因种子种植面积及同比情况 | + |
| 要求种子纯度高、饱满完整、健全无病虫、活力强 | C2:中国转基因作物种子单产 | + |
| C3:第一产业从业者规模 | + |
|  | C4:农作物种植的土地面积 | + |
| 行业安全D | D1:种业市场的总体规模和容量 | + |
| 中国种业市场行业健康状态 |  | + |

**5.3.3 基于熵权法的Topsis综合评价法**

基于熵权的 TOPSIS 综合评价法是在传统的 TOPSIS 方法基础上引入了熵权法，用于确定各个评价指标的权重。

图 中国种业危机预警统计指标体系建立过程

由于数据的量纲不同，可能使得数据值比较大的数据，权重占比比较大，需要进行标准化处理，本文采取极差标准化方法处理，使得正、逆向指标的数值变化均在0和1之间，原来的逆向指标将被化为正向指标，并对结果起到正向作用。

令第i个数据对象的第j个指标为，经过归一化处理后的数值为，具体过程及公式如下。

1. 归一化过程

正向指标：

逆向指标：

1. 熵值定义

第j项指标熵值计算方式：

1. 权重系数

第j项指标权重值：

**5.3.4 指标权重的求解与计算**

根据熵权法将指标数据归一化后的全部指标产出型权重系数，见表 。

表 种业危机预警指标权重系数表

|  |  |  |  |
| --- | --- | --- | --- |
| **指标** | **熵** | **差异性系数** | **权重** |
| **不同种类植物种粮类别的数量** | 0.9701 | 0.0299 | 0.1245 |
| **粮食作物的产量(万吨）** | 0.9991 | 0.0009 | 0.0039 |
| **农作物种子进口量(吨）** | 0.8792 | 0.1208 | 0.5033 |
| **种业行业专利发表数** | 0.9315 | 0.0685 | 0.2855 |
| **期刊论文发布量** | 0.9928 | 0.0072 | 0.0300 |
| **转基因种子种植面积及同比情况** | 0.9959 | 0.0041 | 0.0171 |
| **转基因作物种子单产** | 0.9997 | 0.0003 | 0.0011 |
| **第一产业从业者人数** | 0.9951 | 0.0049 | 0.0202 |
| **农作物种植的土地面积** | 0.9999 | 0.0001 | 0.0004 |
| **种业市场的总体规模和容量** | 0.9967 | 0.0033 | 0.0139 |

根据前文计算出的指标系数权重，结合指标数据计算出的中国种业危机评价指数得到的评价结果如下：

|  |  |
| --- | --- |
| **年份** | **中国种业危机评价指数** |
| **2010** | 7717.74 |
| **2011** | 8515.23 |
| **2012** | 9390.76 |
| **2013** | 11027.83 |
| **2014** | 12662.38 |
| **2015** | 16038.01 |
| **2016** | 17640.61 |
| **2017** | 19612.63 |
| **2018** | 52003.02 |
| **2019** | 47154.16 |
| **2020** | 58750.05 |
| **2021** | 72244.90 |
| **2022** | 55925.18 |
| **2023** | 47899.04 |

**针对问题四：**

**六、模型的评价与推广**

**七、参考文献**

代码附件

第一题

数据预处理代码

1. #异常识别
2. **import** pandas as pd
4. # 1. 检测缺失值
5. # 加载数据
6. data = pd.read\_excel("数据.xlsx")
8. # 检测缺失值
9. missing\_values = data.isnull().sum()  # 统计每列的缺失值数量
10. **print**("缺失值统计：\n", missing\_values)
12. # 2. 检测异常值
13. # 以数值列为例
14. numerical\_columns = data.select\_dtypes(include=['int', 'float']).columns
16. # 使用箱线图检测异常值
17. **def** detect\_outliers(data):
18. outliers = pd.DataFrame(columns=['feature', 'number\_of\_outliers'])
19. **for** column **in** numerical\_columns:
20. Q1 = data[column].quantile(0.25)
21. Q3 = data[column].quantile(0.75)
22. IQR = Q3 - Q1
23. lower\_bound = Q1 - 1.5 \* IQR
24. upper\_bound = Q3 + 1.5 \* IQR
25. outliers = outliers.append({'feature': column, 'number\_of\_outliers': data[(data[column] < lower\_bound) | (data[column] > upper\_bound)].shape[0]}, ignore\_index=True)
26. **return** outliers
28. outliers = detect\_outliers(data)
29. **print**("异常值统计：\n", outliers)
30. #异常处理
31. import pandas as pd
32. from sklearn.impute import SimpleImputer
33. from sklearn.preprocessing import RobustScaler
35. # 1. 加载数据
36. data = pd.read\_excel("数据.xlsx")
38. # 2. 检测并替换异常值
39. def replace\_outliers(data):
40. numerical\_columns = data.select\_dtypes(include=['int', 'float']).columns
41. for column in numerical\_columns:
42. Q1 = data[column].quantile(0.25)
43. Q3 = data[column].quantile(0.75)
44. IQR = Q3 - Q1
45. lower\_bound = Q1 - 1.5 \* IQR
46. upper\_bound = Q3 + 1.5 \* IQR
47. data[column] = data[column].mask((data[column] < lower\_bound) | (data[column] > upper\_bound), data[column].median())
48. return data
50. data = replace\_outliers(data)
52. # 3. 使用插值法处理缺失值
53. imputer = SimpleImputer(strategy="linear")  # 这里使用线性插值，也可以选择其他插值方法
54. data\_filled = pd.DataFrame(imputer.fit\_transform(data), columns=data.columns)
56. # 4. 特征缩放
57. scaler = RobustScaler()  # 使用RobustScaler来缩放数据，对异常值更加鲁棒
58. data\_scaled = scaler.fit\_transform(data\_filled)
60. # 将缩放后的数据放回DataFrame
61. data\_filled = pd.DataFrame(data\_scaled, columns=data\_filled.columns)
63. # 输出处理后的数据
64. print(data\_filled.head())

多元线性回归代码

1. **import** pandas as pd
2. **import** numpy as np
3. **import** matplotlib.pyplot as plt
4. plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
5. plt.rcParams['axes.unicode\_minus']=False #用来正常显示负号
6. **from** statsmodels.stats.outliers\_influence **import** variance\_inflation\_factor
7. **import** statsmodels.api as sm
9. data = pd.read\_excel("数据.xlsx")
11. # # 加载数据
12. # import random
14. # 提取自变量
15. X = data[['农作物受灾面积', '平均温度', '平均降雨量', '农业机械总动力', '农药使用量', '第一产业就业人员']]
16. # data["平均温度"] = temp
17. # 添加截距项
18. X = sm.add\_constant(X)
20. # 计算VIF
21. vif\_data = pd.DataFrame()
22. vif\_data["Variable"] = X.columns
23. vif\_data["VIF"] = [variance\_inflation\_factor(X.values, i) **for** i **in** range(X.shape[1])]
25. **print**(vif\_data)
27. # 提取自变量和因变量
29. # X = data[['农作物受灾面积', '平均温度', '平均降雨量','农业机械总动力', '农药使用量', '第一产业就业人员']]
30. y = data['粮食产量']
32. # 添加截距项
33. X = sm.add\_constant(X)
35. # 创建线性回归模型
36. model = sm.OLS(y, X)
38. # 拟合模型
39. results = model.fit()
41. # 打印回归结果摘要
42. **print**(results.summary())
43. data
44. data.describe().to\_excel("describe.xlsx")

数据可视化

1. **import** pandas as pd
2. **import** matplotlib.pyplot as plt
3. **import** seaborn as sns
5. # 查看数据的前几行
6. **print**(data.head())
8. # 描述性统计
9. **print**(data.describe())
11. # 相关系数矩阵
12. correlation\_matrix = data.corr()
13. **print**(correlation\_matrix)
15. # 可视化数据分布
16. sns.pairplot(data)
17. plt.show()
19. # 可视化单个变量的分布
20. sns.histplot(data['粮食产量'], kde=True)
21. plt.title('Distribution of 粮食产量')
22. plt.show()
24. # 可视化变量之间的关系
25. sns.scatterplot(x='粮食产量', y='农作物受灾面积', data=data)
26. plt.title('Relationship between 粮食产量 and 农作物受灾面积')
27. plt.show()

VAR模型

1. **import** pandas as pd
2. **import** numpy as np
3. **from** statsmodels.tsa.api **import** VAR
4. **from** statsmodels.tools.eval\_measures **import** rmse
5. **from** statsmodels.tsa.stattools **import** adfuller
7. # 1. 数据收集和准备
9. # 读取数据
10. data = pd.read\_excel('数据.xlsx')
11. # data = np.log(data)
12. # 对于不平稳的变量进行一阶差分
13. data['农作物受灾面积'] = data['农作物受灾面积'].diff()
14. data['粮食产量'] = data['粮食产量'].diff()
15. data['平均温度'] = data['平均温度'].diff()
16. data['农业总产值'] = data['农业总产值'].diff()
17. data['平均降雨量'] = data['平均降雨量'].diff()
18. data['农业机械总动力'] = data['农业机械总动力'].diff()
19. data['农药使用量'] = data['农药使用量'].diff()
20. data['第一产业就业人员'] = data['第一产业就业人员'].diff()
22. data['粮食产量'] = data['粮食产量'].diff()
23. data['平均温度'] = data['平均温度'].diff()
24. data['农药使用量'] = data['农药使用量'].diff()
25. data['第一产业就业人员'] = data['第一产业就业人员'].diff()
27. data['粮食产量'] = data['粮食产量'].diff()
28. data['农药使用量'] = data['农药使用量'].diff()
30. # # 删除差分后产生的NaN值
31. data.dropna(inplace=True)
33. # 检查时间序列的平稳性
34. **for** column **in** data.columns:
35. result = adfuller(data[column])
36. **print**(f'ADF Statistic for {column}: {result[0]}, p-value: {result[1]}')
38. # 继续对其他变量进行差分操作
40. # 筛选出平稳的变量
41. stable\_variables = ['粮食产量', '农作物受灾面积', '平均温度', '平均降雨量', '农业机械总动力', '农药使用量', '第一产业就业人员']
42. data\_stable = data[stable\_variables]
44. # 3. 模型拟合
45. # 拟合VAR模型
46. model = VAR(data\_stable)
48. # 选择合适的滞后阶数
49. lag\_order = model.select\_order()
51. # 获取最佳的滞后阶数
52. best\_lag = lag\_order.selected\_orders['aic']
54. # 拟合模型
55. results = model.fit(best\_lag)
57. # 4. 模型诊断
58. # 检查残差是否满足白噪声
59. **print**("Ljung-Box test for white noise in residuals:\n", results.test\_whiteness())
61. # 5. 结果解释
62. # 打印模型系数
63. **print**("Model coefficients:\n", results.summary())

66. # 检查模型拟合是否成功
67. **if** results **is** None:
68. **print**("Error: Model fitting failed!")
69. **else**:
70. # 模型拟合成功，打印模型系数
71. **print**("Model coefficients:\n", results.params)
73. # 获取模型的预测值
74. predicted\_values = results.fittedvalues
76. # 获取模型的观测值
77. observed\_values = data\_stable[best\_lag:]
79. # 计算残差的均方根误差
80. **print**("RMSE of residuals:", rmse(observed\_values, predicted\_values))

第二题

1. **import** pandas as pd
2. **import** numpy as np
3. **import** math
5. **def** nml(series):  # 正向指标归一化 减最小值的min-max方法
6. normalized\_values = (series - series.min()) / (series.max() - series.min())
7. **return** pd.Series(normalized\_values, name=series.name)
9. **def** nml\_max(series):  # 负向指标归一化
10. normalized\_values = (series.max() - series) / (series.max() - series.min())
11. **return** pd.Series(normalized\_values, name=series.name)
13. **def** nmlzt(df, nml\_nmlmax):  # 归一化函数，对正负向指标分别调用nml()和nml\_max()
14. normalized\_df = pd.DataFrame()
15. **for** column **in** df.columns:
16. **if** nml\_nmlmax[column] > 0:
17. normalized\_df[column] = nml(df[column])
18. **else**:
19. normalized\_df[column] = nml\_max(df[column])
20. **return** normalized\_df
22. **def** pij(df):  # 求信息熵公式中的p，这里直接用取值除以取值总和，而不是数量的比例
23. pij\_values = df.copy()
24. **for** column **in** df.columns:
25. total\_sum = pij\_values[column].sum()
26. pij\_values[column] = pij\_values[column] / total\_sum
27. **return** pij\_values
29. **def** entropy(series):  # 计算信息熵
30. length = len(series)
31. **def** ln(x):
32. **if** x > 0:
33. **return** math.log(x)
34. **else**:
35. **return** 0
36. entropy\_sum = sum([i \* ln(i) **for** i **in** series])
37. **return** -(1 / ln(length)) \* entropy\_sum
39. **def** calculate\_weight(dfij):  # 求e、d、w并返回
40. entropy\_df = dfij.copy()
41. weight\_df = pd.DataFrame(index=entropy\_df.columns, dtype='float64')
42. entropy\_list = [entropy(entropy\_df[column]) **for** column **in** entropy\_df.columns]
43. weight\_df['熵'] = entropy\_list
44. weight\_df['差异性系数'] = 1 - np.array(entropy\_list)
45. total\_sum = weight\_df['差异性系数'].sum()
46. weight\_list = [i / total\_sum **for** i **in** weight\_df['差异性系数']]
47. weight\_df['权重'] = weight\_list
48. **return** weight\_df
50. **def** out(normalized\_df, weight\_df):  # 计算最终结果
51. weights = weight\_df['权重']
52. result = np.matmul(normalized\_df, weights)
53. **return** result

1. <https://r.qiyandata.com/data_center?database=CCAD&sonKey=CCAD_NZDB&part=CCAD_NZDB_CSK&code=CCAD_TSTJ_NUM_NZ2023> [↑](#footnote-ref-1)
2. https://inddb.resset.com/login/login.action?loginName=resset001&loginPwd=resset001 [↑](#footnote-ref-2)
3. https://data.stats.gov.cn/easyquery.htm?cn=C01 [↑](#footnote-ref-3)