《地学大数据基础》

实验指导书



资源与地球科学学院

地球信息科学系

2022年11月

目 录

[实验一 地学大数据分析基础 1](#_Toc120709276)

[一、实验目的 1](#_Toc120709277)

[二、实验要求 1](#_Toc120709278)

[三、实验内容 1](#_Toc120709279)

[1. python运行环境搭建 1](#_Toc120709280)

[2. Anaconda的使用 1](#_Toc120709281)

[3. Jupyter Notebook基础运用 1](#_Toc120709282)

[4. numpy、pandas、matplotlib的使用 1](#_Toc120709283)

[四、拓展练习 2](#_Toc120709284)

[实验二 探索性数据分析与降维 3](#_Toc120709285)

[一、实验目的 3](#_Toc120709286)

[二、实验要求 3](#_Toc120709287)

[三、实验内容 3](#_Toc120709288)

[1.数据清洗 3](#_Toc120709289)

[2.相关分析 3](#_Toc120709290)

[3.降维 4](#_Toc120709291)

[四、拓展练习 6](#_Toc120709292)

[实验三 分类与预测 7](#_Toc120709293)

[一、实验目的 7](#_Toc120709294)

[二、实验要求 7](#_Toc120709295)

[三、实验内容 7](#_Toc120709296)

[实验四 数字图像处理 10](#_Toc120709297)

[一、实验目的 10](#_Toc120709298)

[二、实验要求 10](#_Toc120709299)

[三、实验内容 10](#_Toc120709300)

[1.skimage数字图像处理基础方法 10](#_Toc120709301)

[2.使用skimage进行数字图像处理 14](#_Toc120709302)

[实验五 网络数据爬取与时间序列分析 15](#_Toc120709303)

[一、实验目的 15](#_Toc120709304)

[二、实验要求 15](#_Toc120709305)

[三、实验内容 15](#_Toc120709306)

[1. 网络数据爬取与分析 15](#_Toc120709307)

[2.时间序列数据处理 16](#_Toc120709308)

[四、拓展练习 19](#_Toc120709309)

# 实验一 地学大数据分析基础

## 一、实验目的

掌握python数据分析环境的配置，能够熟练使用python进行基础数据读写、可视化，能够进行数据的探索性分析。

## 二、实验要求

1.能够成功进行python数据分析的运行环境配置。

2.掌握使用python进行数据分析的基本工具包的特点。

3.掌握jupyter notebook的使用技巧，在jupyter notebook中完成实验。

4.能够正确使用numpy、pandas、matplotlib等基本工具进行数据的读写、可视化，并进行探索性数据分析。

5.**实验报告要求：**实验内容各部分的实现过程、运行结果截图均要在提交的实验报告中进行体现，实验完成后尽快提交纸质实验报告。实验报告格式见本实验指导书的附件，后续各实验的报告要求与实验一相同，不再重复。

## 三、实验内容

### 1. python运行环境搭建

安装anaconda，结合课程前期作业完成本环节。

### 2. Anaconda的使用

在anaconda环境下运行调试环境的方法见课程资料的相关文档。

### 3. Jupyter Notebook基础运用

Jupyter Notebook是一个强大的交互式数据科学项目的工具，其中包含代码、解释说明文字、代码运行结果、数学公式等等，是非常流行的开发和程序调试工具。关于Jupyter notebook的使用方法参见课程资料的相关文档。

### 4. numpy、pandas、matplotlib的使用

现有智利Pulacayo矿山沿脉坑道中以2米的采样间距在含闪锌矿石英脉采集的118个矿石样品Zn的化学分析值数据（见practice1.xlsx文件中Sheet1工作表），使用numpy、pandas、matplotlib对该数据进行读取、分析和可视化，计算该数据地区Zn元素的最大、最小值、平均值、四分位数、标准差，并分别绘制该数据的箱线图、折线图。

主要步骤如下：

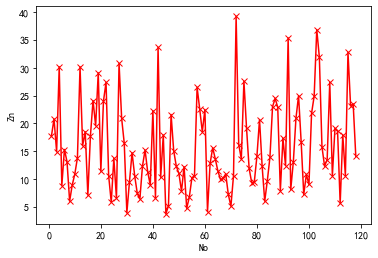
1）导入模块numpy、pandas、matplotlib.pyplot模块；

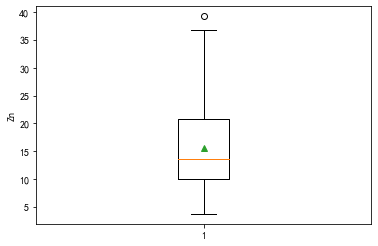
2）读取并观察数据：使用pandas的read\_excel()方法读取实验数据，同时利用DataFrame的head()方法观察读入的数据体的结构，进而使用describe()方法获取的特征；

3）对读入的数据进行索引、切片等提取操作，构造matplotlib绘图的数据；

4）运用matplotlib绘制Zn元素的折线图（plot）、箱线图（boxplot），如下图所示。

注意：如对matplotlib绘图参数设置有疑问，要掌握查询matplotlib官网示例程序的方法。





## 四、拓展练习

现有近1个月全球地震监测记录数据（数据位于practice1.xlsx文件的earthquakes工作表），试运用本实验中的方法对数据进行读取，对数据中的'time', 'place', 'latitude', 'longitude', 'depth', 'mag'等主要参数获取其统计学特征，如对震源深度'depth'、震级'mag'等获取其最大值、中位数等，并结合所学进行可视化分析。

# 实验二 探索性数据分析与降维

## 一、实验目的

掌握使用python进行数据清洗、数据预处理和探索性数据分析方法，能够正确使用python进行数据的探索性分析与降维处理。

## 二、实验要求

1.能够正确使用所需的第三方库（如numpy\pandas\matplotlib\scikit-learn）等基本工具进行数据的读写、可视化，并对数据进行探索性数据分析。

2.能够正确使用PCA等方法对高维数据进行降维处理。

3.实验报告要求同实验一。

## 三、实验内容

### 1.数据清洗

对所给数据进行描述，分别求其中位数、众数、上四分位数、下四分位数、最大值、最小值、下内界值、上内界值、远外值、方差、偏度及峰度，结合数据本身给出分析结论。

实验数据文件名：“movie.txt”

实验步骤：

（1）使用numpy.loadtxt()与numpy.savetxt读取数据

（2）使用numpy或scipy中的quantile()、median()、mean()、mode()、var()、cov()、std()、kurt()、skew()，基于公式定义计算实验数据的均值、中位数、众数、上四分位数、下四分位数、最大值、最小值、下内界值、上内界值、远外值、方差、偏度及峰度，并基于这些统计结果对数据本身的特征进行描述，给出描述性结论。

（3）基于3-sigma方法与箱型图方法，确定实验数据中的异常值。并写出实验过程。

### 2.相关分析

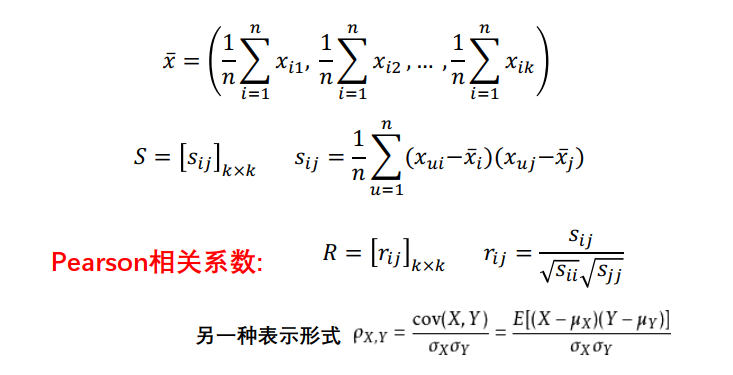
基于相关分析的定义，import合适的第三方库（scipy等）并求出实验数据的Pearson相关系数（公式如下），厘定其相关性，并说明总体相关系数和样本相关系数的联系、差别以及转化方式。

实验数据：“practice2.xlsx”文件中的exam工作表

实验步骤：

（1）需要使用scipy库（pip install或直接下载安装）

（2）import scipy中的统计函数stats，并利用现有的函数求出实验数据的协方差矩阵，并在协方差矩阵的基础上，求出两组数据的Pearson相关系数。Pearson相关系数的计算公式如下图所示。



### 3.降维

利用主成分分析对给定的实验数据进行降维，掌握主成分分析的基本步骤并使用Python进行编程实现。

实验数据：“practice2.xlsx”文件中的“PCA”工作表

主要实验步骤：

（1）计算相关系数矩阵，检验待分析的变量是否适合做主成分分析（如无相关性则不做）。

（2）数据标准化，求解协方差矩阵。

（3）求协方差矩阵的特征根及对应的标准化特征向量。

（4）确定主成分个数，将特征值按照从大到小的排序，选择其中最大的k个。

（5）写出主成分表达式，计算各样品的主成分的得分。

实验步骤参考：

（1）读取数据

import ……

plt.rcParams['font.sans-serif'] = ['SimHei'] # 用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False # 用来正常显示负号

df = pd.read\_excel(r'.\practice2.xlsx',sheet\_name= 'PCA', header= 0)

fields = df.columns.values #记录数据列名称

（2）观察数据的特点，确定计算方法

X= np.array(df.values)[:,1:]#获取数据体，去掉编号列

print("输出X")

print(X[:5,:])#输出X前5行，观察数据特征

（3）计算原始数据相关系数矩阵，判断变量的相关性

rho = np.corrcoef(X.T)# 使用numpy的corrcoef函数计算原始数据的相关系数矩阵

print("输出相关系数矩阵")

print(rho) #输出相关系数矩阵

（4）对原始数据进行标准化

Z=np.zeros\_like(X)

for j in range(X.shape[1]):

Z[:,j]=(X[:,j]-X[:,j].mean())/X[:,j].std()

print("输出标准化后的结果:")

print(Z)

（5）计算协方差矩阵，计算矩阵的特征值和特征向量

S = np.cov(Z.T) # 计算协方差矩阵

print("输出协方差矩阵:")

print(S) #输出协方差矩阵

eig,eigv = np.linalg.eigh(S) #求协方差矩阵的特征值及其特征向量

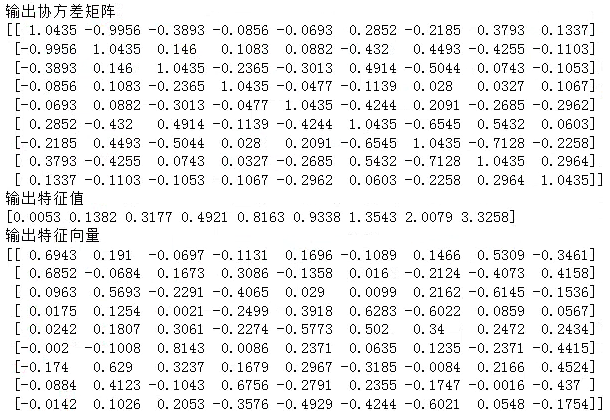
print("输出特征值:")

print(eig) #注意观察值的大小顺序，为升序排序

print("输出特征向量:")

print(eigv) #顺序与特征值对应

运行结果参考：



(6)主成分分析结果:

* 主成分标准差
* 主成分贡献率(方差比例)
* 主成分累积贡献率(方差累积比例)
* 主成分载荷
* 计算主成分贡献率，确定主成分个数

eig1=eig[::-1] #对特征值进行倒序排序

eigv1=eigv[::-1,:] #对特征向量进行倒序排序

com\_std=eig1\*\*0.5 #主成分标准差

com\_prop=eig1/np.sum(eig1) #主成分贡献率(方差比例)

com\_cumprop=np.cumsum(eig1)/np.sum(eig1) #主成分累积贡献率(方差累积比例)

print('特征值排序后：')

print(eig1)

print('特征向量排序后：')

print(eigv1)

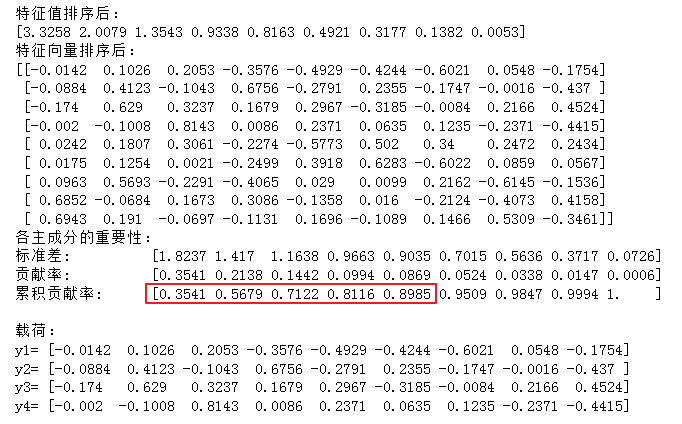
print('各主成分的重要性：')

print('标准差: ',com\_std)

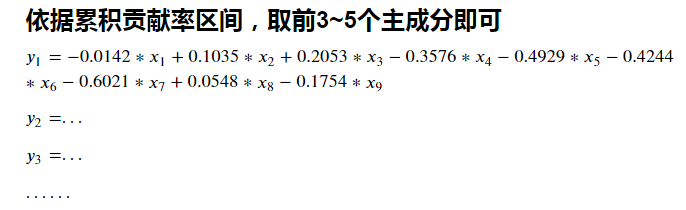
print('贡献率：',com\_prop)

print('累积贡献率：',com\_cumprop)

运行结果参考：



（7）写出主成分表达式：



## 四、拓展练习

对实验内容3的数据降维操作，利用sklearn.decomposition库实现主成分分析，并对不同方法实现数据降维的特点和分析结果进行对比。

# 实验三 分类与预测

## 一、实验目的

掌握使用python进行线性回归分析的具体方法，熟悉聚类的方法流程。

## 二、实验要求

1.掌握一元、多元线性回归分析的方法原理和分析过程。

2.熟悉聚类分析的操作过程。

## 三、实验内容

**1.线性回归分析**

现有某煤田的煤层气钻井采样分析数据如下表（实验数据文件为CBM.csv），分别记录了23口钻井样品的煤层气相关测试或实验分析数据，如煤层埋深（Depth）、最大镜质组反射率（Romax）、煤层厚度（Thick）、空气干燥基水份（Mad）、灰分（Aad）、挥发分（Vad）、含气量（Gas），以及煤样吸附气体中的甲烷（CH4）、二氧化碳（CO2）和氮气（N2）的比例。试分别采用一元线性回归和多元线性回归分析方法，分析参数之间的线性关系，具体要求如下。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| FID | Well | Depth | Romax | Thick | Mad | Aad | Vad | Gas | CH4 | CO2 | N2 |
| 0 | M-15 | 693.2 | 1.37 | 2.75 | 0.86 | 7.53 | 22.37 | 8.23 | 95.15 | 0.66 | 4.19 |
| 1 | M-32 | 764.8 | 1.43 | 2.94 | 0.71 | 6.79 | 21.37 | 11.84 | 96.54 | 1.06 | 2.4 |
| 2 | M-37 | 681.2 | 1.29 | 4.16 | 0.64 | 7.41 | 24.09 | 9.18 | 96.2 | 0.57 | 1 |
| 3 | M-47 | 680.6 | 1.21 | 2.4 | 0.68 | 7.95 | 25.79 | 6.07 | 92.24 | 1.3 | 5.56 |
| 4 | M-55 | 648.5 | 1.29 | 2.54 | 0.7 | 7.94 | 24.14 | 5.96 | 87.51 | 1.44 | 7.87 |
| 5 | M-149 | 547.4 | 1.67 | 1.7 | 1 | 7.39 | 17.17 | 6.08 | 83.56 | 0.8 | 15.64 |
| 6 | M-169 | 592.7 | 1.37 | 1.1 | 0.84 | 29.23 | 22.54 | 6.75 | 95.42 | 0.46 | 3.87 |
| 7 | M-193 | 554.2 | 1.33 | 1.2 | 0.68 | 35.84 | 23.32 | 4.57 | 60.19 | 3.34 | 36.46 |
| 8 | M-215 | 522.5 | 1.58 | 2.3 | 0.79 | 28.87 | 18.66 | 7.9 | 69.08 | 2.15 | 27.84 |
| 9 | M-227 | 364.8 | 1.7 | 1.5 | 1.02 | 37.08 | 16.66 | 6.19 | 66.12 | 0.44 | 33.44 |
| 10 | T-7 | 675.1 | 1.48 | 2.04 | 0.52 | 14.48 | 20.4 | 7.96 | 97.33 | 1.14 | 1.53 |
| 11 | T-15 | 652.3 | 1.49 | 1.1 | 0.6 | 9.86 | 20.3 | 8.31 | 96.14 | 0.81 | 2.46 |
| 12 | T-17 | 717.4 | 1.38 | 1.5 | 0.91 | 21.48 | 22.27 | 5.74 | 77.81 | 1.6 | 18.66 |
| 13 | T-19 | 528.3 | 1.62 | 3.3 | 0.68 | 9.9 | 17.97 | 11.58 | 83.26 | 1.81 | 14.93 |
| 14 | T-46 | 513.7 | 1.67 | 2.2 | 0.82 | 6.48 | 17.23 | 13.34 | 98.54 | 0.89 | 0.57 |
| 15 | T-80 | 429.5 | 1.62 | 2.75 | 0.48 | 9.3 | 18.06 | 9.2 | 98.79 | 0.29 | 0.92 |
| 16 | T-119 | 382.7 | 1.41 | 2.5 | 0.85 | 40.24 | 21.64 | 4.25 | 88.11 | 0.48 | 11.41 |
| 17 | T-123 | 367.7 | 1.62 | 1.7 | 0.56 | 14.54 | 18.02 | 6.17 | 81.98 | 0.23 | 17.8 |
| 18 | T-126 | 295.7 | 1.57 | 2.1 | 0.3 | 18.72 | 18.84 | 6.64 | 91.01 | 0.75 | 7.4 |
| 19 | T-155 | 389.3 | 1.47 | 3.8 | 0.84 | 27.73 | 20.61 | 7.77 | 98.01 | 0.18 | 1.81 |
| 20 | D-6 | 400.4 | 1.59 | 0.6 | 0.76 | 26.32 | 18.49 | 3.83 | 84.37 | 0.59 | 14.61 |
| 21 | D-26 | 472.8 | 1.6 | 1.3 | 0.42 | 17.74 | 18.29 | 7.94 | 97.69 | 0.68 | 1.62 |
| 22 | D-51 | 419.6 | 1.7 | 1.4 | 0.44 | 12.88 | 16.75 | 8.81 | 92.65 | 0.59 | 6.75 |
| 23 | D-78 | 358.3 | 1.41 | 0.95 | 0.61 | 37.38 | 21.61 | 7.88 | 89.12 | 0.27 | 10.61 |

**实现方法：**

1）采用一元线性回归，分析最大镜质组反射率与挥发分的关系，写出主要实现代码、回归方程及检验结果。

2）采用多元线性回归分析方法，分析最大镜质组反射率与灰分、含气量等参数之间有无线性关系，分析含气量与其他参数间有无线性关系。

注意：

1）先绘制散点图，观察个参数之间的分布情况；

2）一元线性回归分析时，可使用最小二乘法进行回归系数计算、检验，也可以使用sklearn.LinearRegression进行计算，具体步骤参考课堂示例，在实验报告中将分析过程及回归方程、显著性检验等体现出来。

**2.聚类分析**

继续对实验内容1中的煤层气样品数据，使用系统聚类方法，对样品进行分类，主要步骤如下：

1）引入模块:

import scipy.stats as st

import scipy.cluster as sc

import scipy.spatial.distance as dist

2）读取数据:

pd.read\_csv()；

3）丢弃数据中的'FID','Well','CH4','CO2','N2','Depth’参数字段:

df.drop([‘FID’],axis=1)；

4）对数据使用Well重建索引:

X.index = df['Well']

以上两步主要是删除非数值参数字段，并保留Wellname字段作为索引，可以使用多种方法完成，可以参考我的示例代码

5）对原始数据进行标准化:

X\_std = (X - np.mean(X,axis=0))/st.tstd(X,axis=0)

6）计算欧式距离矩阵:

dis =dist.pdist(X\_std,'euclidean')；

7）采用最远距离法进行聚类:

Z =sc.hierarchy.linkage(dis,'complete')

8）绘制谱系图：

row\_dendr=sc.hierarchy.dendrogram(Z,labels=X.index)

参考上述步骤写出实现代码，将各步骤的运行结果写在实验报告中。

# 实验四 数字图像处理

**（选做实验）**

## 一、实验目的

掌握使用python进行图像处理的方法；了解使用python进行数据爬取的方法，综合运用本课程所学方法对所爬取的数据进行分析。

## 二、实验要求

1. 掌握skimage常用的数字图像处理的方法和实现过程。

2. 实验报告要求同实验一。

## 三、实验内容

熟悉skimage的模块功能，如表4-1。

表4-1 skimage主要模块及其功能

|  |  |
| --- | --- |
| **子模块名称** | **主要实现功能** |
| io | 读取、保存和显示图片或视频 |
| data | 提供一些测试图片和样本数据 |
| color | 颜色空间变换 |
| filters | 图像增强、边缘检测、排序滤波器、自动阈值等 |
| draw | 操作于NumPy数组上的基本图形绘制，包括线条、矩形、圆和文本等 |
| transform | 几何变换或其它变换，如旋转、拉伸和拉东变换等 |
| morphology | 形态学操作，如开闭运算、骨架提取等 |
| exposure | 图片强度调整，如亮度调整、直方图均衡等 |
| feature | 特征检测与提取等 |
| measure | 图像属性的测量，如相似性或等高线等 |
| segmentation | 图像分割 |
| restoration | 图像恢复 |
| util | 通用函数 |

### 1.skimage数字图像处理基础方法

1）图像校正是对图像进行基本的亮度和对比度调整，常用gamma校正和log调整。

|  |  |  |
| --- | --- | --- |
| **校正方法** | **所属模块** | **调用格式** |
| gamma | skimage.exposure | gam1=exposure.adjust\_gamma(image,2)#调暗  gam2=exposure.adjust\_gamma(image,0.5)#调亮 |
| log | gam1=exposure.adjust\_log(image)#对数调整 |

2）图像的滤波通常有两种效果：一种是平滑滤波，用来抑制噪声；另一种是微分算子，可以用来检测边缘和特征提取。

|  |  |  |
| --- | --- | --- |
| **滤波算子** | **所属模块** | **调用格式** |
| sobel | skimage.filters | edges=filters.sobel(img) |
| roberts | edges=filters.roberts(img) |
| scharr | edges=filters.scharr(img) |
| prewitt | edges=filters.prewitt(img) |
| gabor | filt\_real,filt\_imag=filters.gabor(img,frequency=0.6) |
| gaussian | edges2=filters.gaussian(img,sigma=5) |
| median | edges1=filters.median(img, disk(5))  #disk是由skimage.morphology设置的滤波器形状 |
| canny | skimage.feature | edges2=feature.canny(img,sigma=3) |

3）直方图均衡化

如果一副图像的像素占有很多的灰度级而且分布均匀，那么这样的图像往往有高对比度和多变的灰度色调。直方图均衡化就是一种能仅靠输入图像直方图信息自动达到这种效果的变换函数。它的基本思想是对图像中像素个数多的灰度级进行展宽，而对图像中像素个数少的灰度进行压缩，从而扩展取值的动态范围，提高了对比度和灰度色调的变化，使图像更加清晰。

from skimage import data,exposure

import matplotlib.pyplot as plt

img=data.moon()

plt.figure("hist",figsize=(8,8))

arr=img.flatten()

plt.subplot(221)

plt.imshow(img,plt.cm.gray)#原始图像

plt.subplot(222)

plt.hist(arr,bins=256,edgecolor='None',facecolor='red')#原始图像直方图

img1=exposure.equalize\_hist(img)

arr1=img1.flatten()

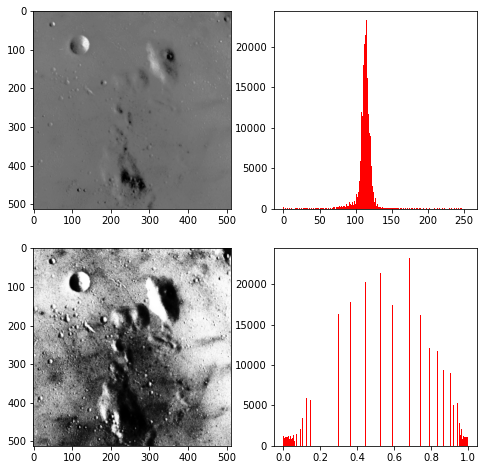
plt.subplot(223)

plt.imshow(img1,plt.cm.gray)#均衡化图像

plt.subplot(224)

plt.hist(arr1,bins=256,edgecolor='None',facecolor='red')#均衡化直方图

plt.show()



4）自动阈值分割

在skimage库中，阈值分割的功能在filters模块中。我们可以手动指定一个阈值，从而来实现分割。也可以让系统自动生成一个阈值，下面几种方法用来自动生成阈值。

|  |  |  |
| --- | --- | --- |
| threshold\_otsu | image=data.camera()  thresh=filters.threshold\_otsu(image)  #返回阈值102  dst=(image<=thresh) #根据阈值进行分割  plt.figure('thresh',figsize=(8,8))  plt.subplot(121)  plt.title('originalimage')  plt.imshow(image,plt.cm.gray)  plt.subplot(122)  plt.title('binaryimage')  plt.imshow(dst,plt.cm.gray)  plt.show() |  |
| threshold\_yen | image=data.camera()  thresh=filters.threshold\_yen(image)  #返回阈值为146  dst=(image<=thresh)  plt.figure('thresh',figsize=(8,8))  plt.subplot(121)  plt.title('originalimage')  plt.imshow(image,plt.cm.gray)  plt.subplot(122)  plt.title('binaryimage')  plt.imshow(dst,plt.cm.gray)  plt.show() |  |
| threshold\_li | thresh=filters.threshold\_li(image)  #返回阈值78.91288426606151  dst=(image<=thresh)  plt.figure('thresh',figsize=(8,8))  plt.subplot(121)  plt.title('originalimage')  plt.imshow(image,plt.cm.gray)  plt.subplot(122)  plt.title('binaryimage')  plt.imshow(dst,plt.cm.gray)  plt.show() |  |
| threshold\_isodata | thresh=filters.threshold\_isodata(image) #返回阈值102  dst=(image<=thresh)  plt.figure('thresh',figsize=(8,8))  plt.subplot(121)  plt.title('originalimage')  plt.imshow(image,plt.cm.gray)  plt.subplot(122)  plt.title('binaryimage')  plt.imshow(dst,plt.cm.gray)  plt.show() |  |
| threshold\_local | dst1=filters.threshold\_local(image,31,'mean')  dst2=filters.threshold\_local(image,5,'median')  plt.figure('thresh',figsize=(8,8))  plt.subplot(121)  plt.title('block = 31')  plt.imshow(dst1,plt.cm.gray)  plt.subplot(122)  plt.title('block = 5')  plt.imshow(dst2,plt.cm.gray)  plt.show() |  |

5）基本形态学滤波

对图像进行形态学变换,变换对象一般为灰度图或二值图，功能函数放在morphology子模块内。

|  |  |  |
| --- | --- | --- |
| **滤波操作** | **调用格式** | **作用** |
| 膨胀（dilation) | dilation(image,selem=None） | 一般对二值图像进行操作。找到像素值为1的点，将它的邻近像素点都设置成这个值。1值表示白，0值表示黑，因此膨胀操作可以扩大白色值范围，压缩黑色值范围。一般用来扩充边缘或填充小的孔洞。 |
| 腐蚀（erosion) | erosion(image,selem=None） | 和膨胀相反的操作，将0值扩充到邻近像素。扩大黑色部分，减小白色部分。可用来提取骨干信息，去掉毛刺，去掉孤立的像素。 |
| 开运算（opening) | openning(image,selem=None） | 图像的开运算等同于先腐蚀再膨胀，可以消除小物体或小斑块。 |
| 闭运算（closing) | closing(image,selem=None） | 图像的闭运算等同于先膨胀再腐蚀，可用来填充孔洞。 |
| 白帽运算（white-tophat) | white\_tophat(image,selem=None） | 白帽运算是将原图像减去它的开运算值，返回比结构化元素小的白点。 |
| 黑帽运算（black-tophat) | black\_tophat(image,selem=None） | 黑帽运算将原图像减去它的闭运算值，返回比结构化元素小的黑点，且将这些黑点反色。 |

### 2.使用skimage进行数字图像处理

现有煤岩、砂岩样品的CT扫描图像，运用上述数字图像处理方法分别提起煤岩图像中的裂隙、砂岩图像中的孔隙。图像数据在实验素材中的image文件夹中。

# 实验五 网络数据爬取与时间序列分析

## 一、实验目的

了解使用python进行大数据爬取的基本方法原理；掌握时间序列分析基础方法。

## 二、实验要求

1. 了解爬虫程序，掌握基于爬取数据使用python进行数据分析的方法。

2. 掌握时间序列数据的常用处理方法。

3. 实验报告要求同实验一。

## 三、实验内容

### 1. 网络数据爬取与分析

使用Python爬取全球新冠疫情的每日更新数据，并对所爬取的数据进行可视化分析。

下列代码从[www.worldmeters.info](http://www.worldmeters.info)网站上爬取公开的全球新冠肺炎实时数据，阅读并运行代码，了解程序代码的运行逻辑。综合运用本课程的相关内容，对程序运行后保存在本地的数据文件进行分析和可视化，分析全球新冠疫情的变化趋势。

# -\*- coding: utf-8 -\*-

"""

Created on Tue Apr 28 16:49:11 2020

Get COVID-19 realtime data from WorldMeters website

@author: LUO

"""

import requests

from bs4 import BeautifulSoup

import csv

import time

def get\_data(html\_text):

final = []

bs = BeautifulSoup(html\_text, "html.parser") # 创建BeautifulSoup对象

body = bs.body # 获取body部分

data = body.find('table', {'id': 'main\_table\_countries\_today'}) #找到对应id

thead=data.find('thead')

th\_list=thead.find\_all('th')

temp=[]

for th in th\_list:#get tabel header

if th.text != None:

temp.append(th.text)

final.append(temp)

tbody=data.find('tbody')

tr\_list=tbody.find\_all('tr')

for tr in tr\_list:

tmp=[]

td\_list=tr.find\_all('td')

for td in td\_list:

tmp.append(td.text)

final.append(tmp)

return final

def write\_data(data, name):

file\_name = name

with open(file\_name, 'a', errors='ignore', newline='') as f:

f\_csv = csv.writer(f)

f\_csv.writerows(data)

tm=time.strftime('%Y%m%d%H%M%S',time.localtime(time.time()))

url ='https://www.worldometers.info/coronavirus/'

html=requests.get(url)

result = get\_data(html.text)

write\_data(result, tm+'\_COVID\_19.csv')

### 2.时间序列数据处理

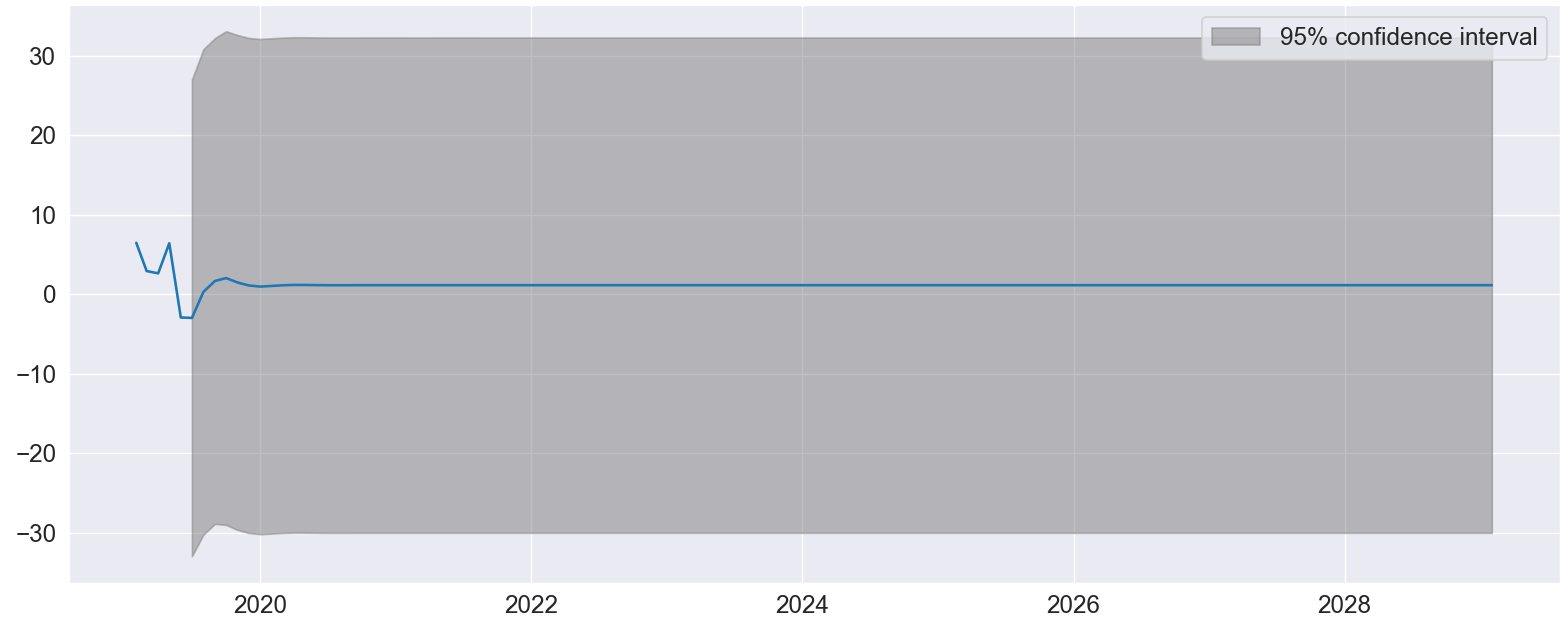
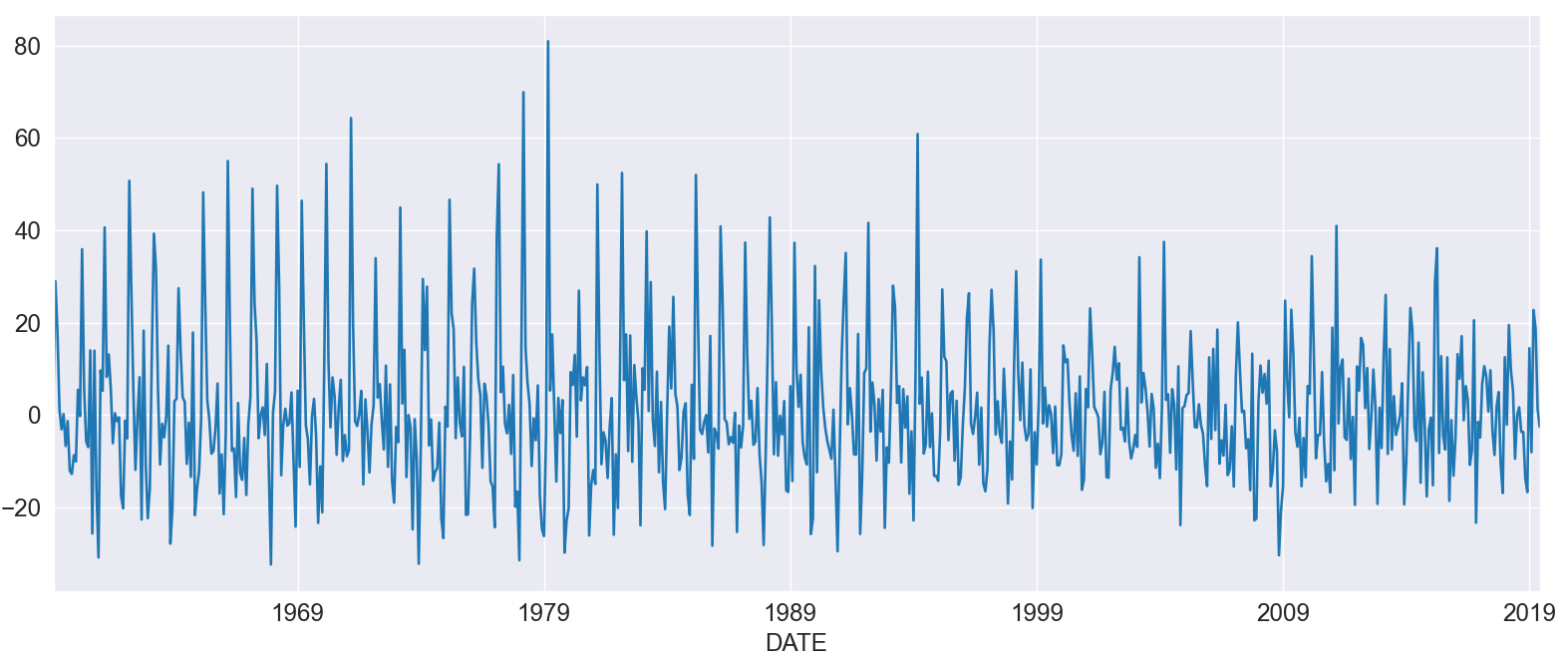
配置时间序列数据处理的运行环境，如numpy、pandas、stasmodels、seaborn、pandas\_datareader等，同时需要连接网络。

**1)案例一：自回归问题**

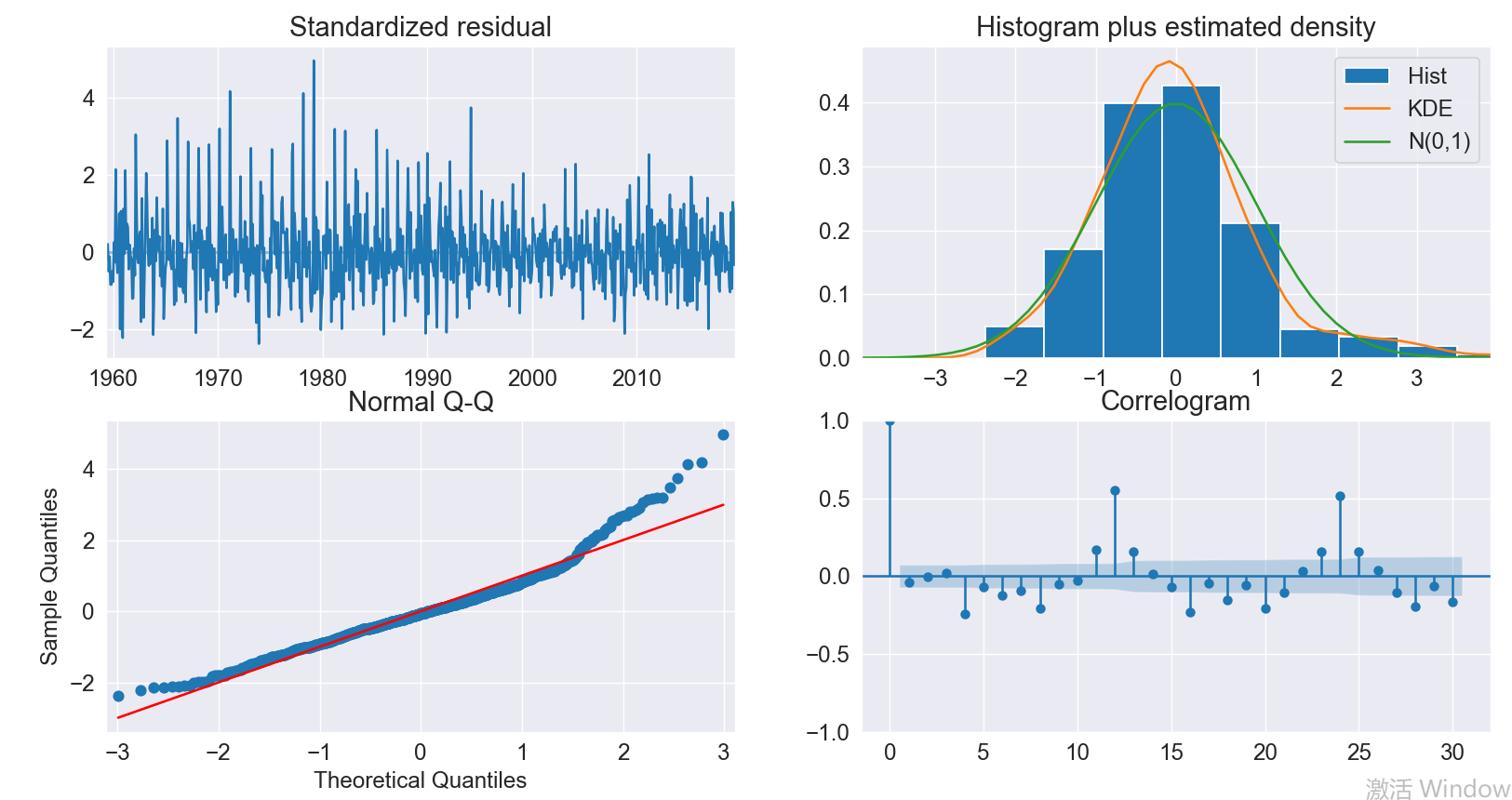
现有某城市的房价月度数据，针对房价波动问题，使用时间序列分析方法，预测房价的变化趋势。

import matplotlib.pyplot as plt  
import pandas as pd  
import pandas\_datareader as pdr  
import seaborn as sns  
from statsmodels.tsa.api import acf, graphics, pacf  
from statsmodels.tsa.ar\_model import AutoReg, ar\_select\_order  
  
sns.set\_style("darkgrid")  
pd.plotting.register\_matplotlib\_converters()  
# Default figure size  
sns.mpl.rc("figure", figsize=(16, 6))  
sns.mpl.rc("font", size=14)  
  
data = pdr.get\_data\_fred("HOUSTNSA", "1959-01-01", "2019-06-01")  
housing = data.HOUSTNSA.pct\_change().dropna()  
# Scale by 100 to get percentages  
housing = 100 \* housing.asfreq("MS")  
fig, ax = plt.subplots()  
assert isinstance(ax, object)  
ax = housing.plot(ax=ax)  
  
mod = AutoReg(housing, 3, old\_names=False)  
res = mod.fit()  
print(res.summary())  
  
fig = res.plot\_predict(720, 840)  
  
fig = plt.figure(figsize=(16, 9))  
fig = res.plot\_diagnostics(fig=fig, lags=30)  
  
plt.show()

**运行结果：**



**原始数据：**

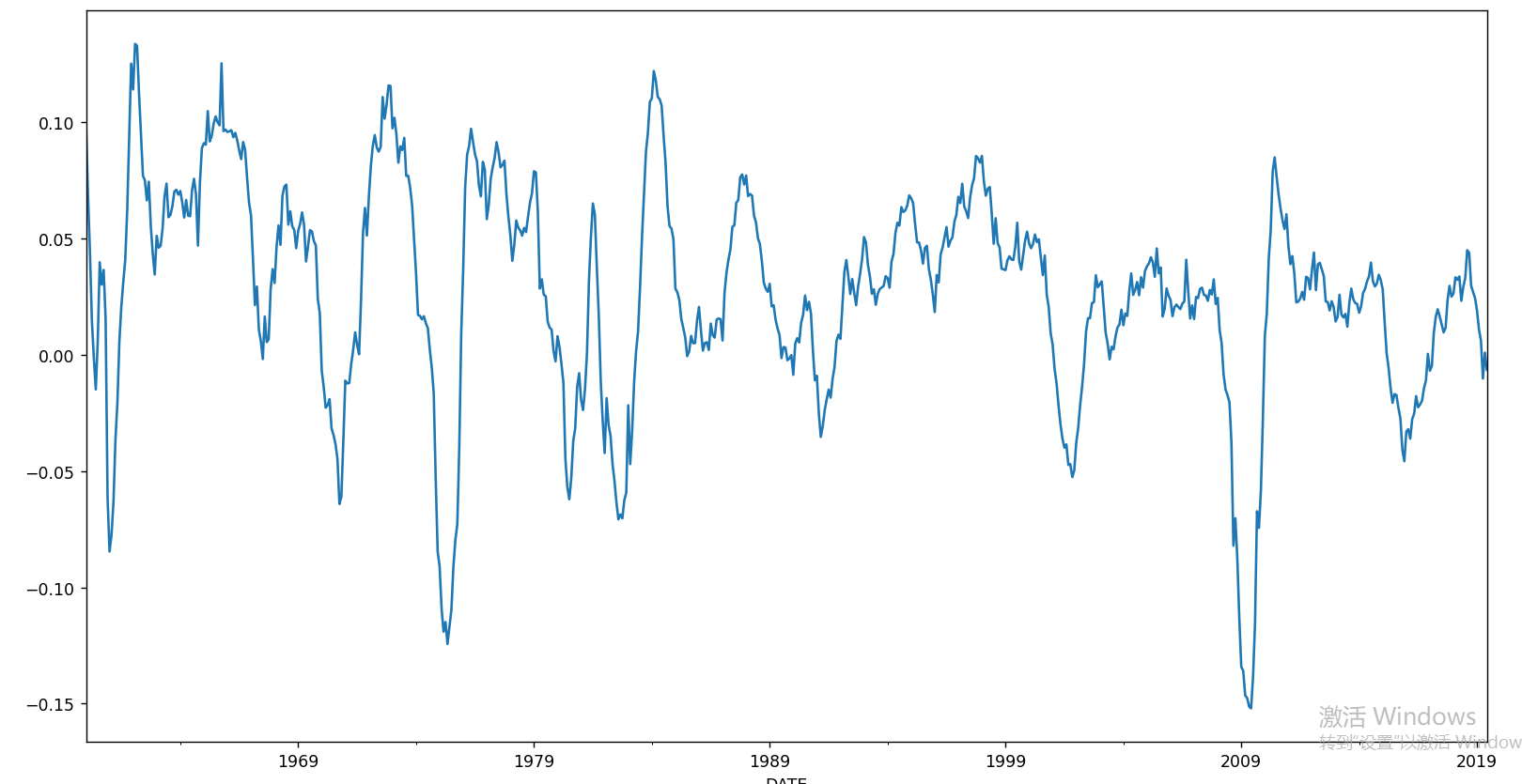
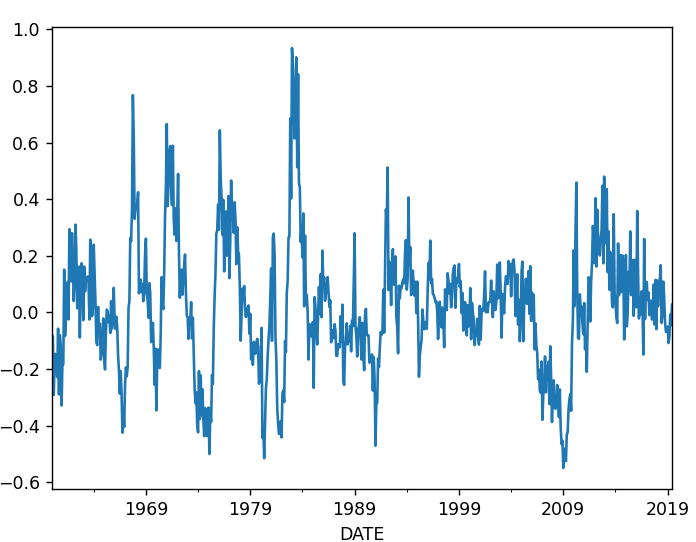


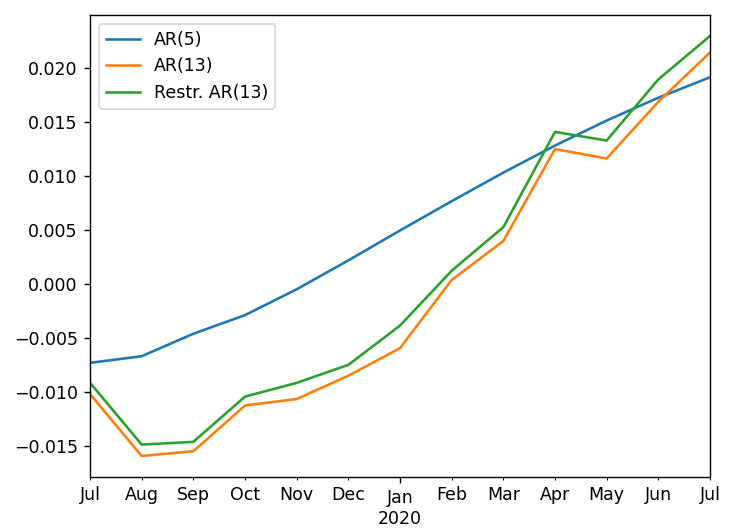
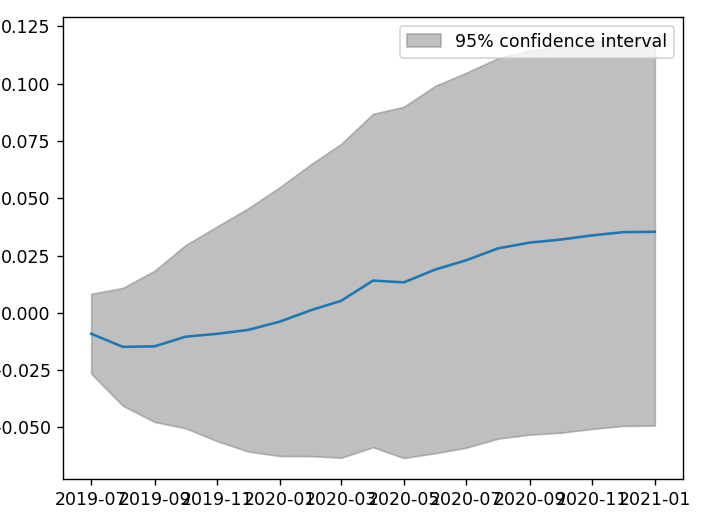
**2）案例二：季节动态数据分析**

分析房价数据的季节性波动特征及变化趋势。

import matplotlib.pyplot as plt  
import pandas as pd  
import pandas\_datareader as pdr  
import seaborn as sns  
from statsmodels.tsa.api import acf, graphics, pacf  
from statsmodels.tsa.ar\_model import AutoReg, ar\_select\_order  
  
data = pdr.get\_data\_fred("HOUSTNSA", "1959-01-01", "2019-06-01")  
  
yoy\_housing = data.HOUSTNSA.pct\_change(12).resample("MS").last().dropna()  
\_, ax = plt.subplots()  
ax = yoy\_housing.plot(ax=ax)  
  
sel = ar\_select\_order(yoy\_housing, 13, old\_names=False)  
sel.ar\_lags  
  
sel = ar\_select\_order(yoy\_housing, 13, glob=True, seasonal=True, old\_names=False)  
sel.ar\_lags  
res = sel.model.fit()  
print(res.summary())  
  
data = pdr.get\_data\_fred("INDPRO", "1959-01-01", "2019-06-01")  
ind\_prod = data.INDPRO.pct\_change(12).dropna().asfreq("MS")  
\_, ax = plt.subplots(figsize=(16, 9))  
ind\_prod.plot(ax=ax)  
  
sel = ar\_select\_order(ind\_prod, 13, "bic", old\_names=False)  
res = sel.model.fit()  
print(res.summary())  
  
sel = ar\_select\_order(ind\_prod, 13, "bic", glob=True, old\_names=False)  
sel.ar\_lags  
res\_glob = sel.model.fit()  
print(res.summary())  
  
ind\_prod.shape  
  
fig = res\_glob.plot\_predict(start=714, end=732)  
  
res\_ar5 = AutoReg(ind\_prod, 5, old\_names=False).fit()  
predictions = pd.DataFrame(  
 {  
 "AR(5)": res\_ar5.predict(start=714, end=726),  
 "AR(13)": res.predict(start=714, end=726),  
 "Restr. AR(13)": res\_glob.predict(start=714, end=726),  
 }  
)  
\_, ax = plt.subplots()  
ax = predictions.plot(ax=ax)  
  
plt.show()

**运行结果：**





## 四、拓展练习

**结合下列代码自由探索世界序列数据分析的相关方法。**

# -\*- coding: utf-8 -\*-  
import pandas as pd  
import numpy as np  
from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error  
import numpy.linalg as la  
import math  
# from statsmodels.tsa.arima\_model import ARIMA  
import time  
import statsmodels.api as sm  
  
def preprocess\_data(data, time\_len, rate, seq\_len, pre\_len):  
 data1 = np.mat(data)  
 train\_size = int(time\_len \* rate)  
 train\_data = data1[0:train\_size]  
 test\_data = data1[train\_size:time\_len]  
  
 trainX, trainY, testX, testY = [], [], [], []  
 for i in range(len(train\_data) - seq\_len - pre\_len):  
 a = train\_data[i: i + seq\_len + pre\_len]  
 trainX.append(a[0: seq\_len])  
 trainY.append(a[seq\_len: seq\_len + pre\_len])  
 for i in range(len(test\_data) - seq\_len - pre\_len):  
 b = test\_data[i: i + seq\_len + pre\_len]  
 testX.append(b[0: seq\_len])  
 testY.append(b[seq\_len: seq\_len + pre\_len])  
 return trainX, trainY, testX, testY  
  
###### evaluation ######  
def evaluation(a, b):  
 rmse = math.sqrt(mean\_squared\_error(a, b))  
 mae = mean\_absolute\_error(a, b)  
 F\_norm = la.norm(a - b) / la.norm(a)  
 r2 = 1 - ((a - b) \*\* 2).sum() / ((a - a.mean()) \*\* 2).sum()  
 var = 1 - (np.var(a - b)) / np.var(a)  
 return rmse, mae, 1 - F\_norm, r2, var  
  
time\_start = time.time()  
path = r'./432\_1800\_cql\_00001.csv'  
data = pd.read\_csv(path,encoding='utf-8')  
  
time\_len = data.shape[0]  
num\_nodes = data.shape[1]  
train\_rate = 0.8  
seq\_len = 7  
pre\_len = 1  
trainX, trainY, testX, testY = preprocess\_data(data, time\_len, train\_rate, seq\_len, pre\_len)  
method = 'ARIMA' ####HA or SVR or ARIMA  
  
######## ARIMA #########  
if method == 'ARIMA':  
 rng = pd.date\_range('1/3/2012', periods=1379, freq='D')  
 a1 = pd.DatetimeIndex(rng)  
 data.index = a1  
 num = data.shape[1]  
 rmse, mae, acc, r2, var, pred, ori = [], [], [], [], [], [], []  
 for i in range(3):  
 ts = data.iloc[:, i]  
 ts\_log = np.log(ts)  
 ts\_log = ts  
 ts\_log = np.array(ts\_log, dtype=np.float)  
 where\_are\_inf = np.isinf(ts\_log)  
 ts\_log[where\_are\_inf] = 0  
 ts\_log = pd.Series(ts\_log)  
 ts\_log.index = a1  
 model = sm.tsa.arima.ARIMA(ts\_log, order=(1, 1, 1))  
 # model = ARIMA(ts\_log, order=[1, 0, 0])  
 properModel = model.fit()  
 predict\_ts = properModel.predict(4, dynamic=True)  
 log\_recover = np.exp(predict\_ts)  
 log\_recover = predict\_ts  
 ts = ts[log\_recover.index]  
 path = './output/'  
 name = 'logrecover\_' + str(i) + '.csv'  
 logrecover = pd.DataFrame(log\_recover)  
 logrecover.to\_csv(path + name)  
 er\_rmse, er\_mae, er\_acc, r2\_score, var\_score = evaluation(ts, log\_recover)  
 rmse.append(er\_rmse)  
 mae.append(er\_mae)  
 acc.append(er\_acc)  
 r2.append(r2\_score)  
 var.append(var\_score)  
 # for i in range(109,num):  
 # ts = data.iloc[:,i]  
 # ts\_log=np.log(ts)  
 # ts\_log=np.array(ts\_log,dtype=np.float)  
 # where\_are\_inf = np.isinf(ts\_log)  
 # ts\_log[where\_are\_inf] = 0  
 # ts\_log = pd.Series(ts\_log)  
 # ts\_log.index = a1  
 # model = ARIMA(ts\_log,order=[1,1,1])  
 # properModel = model.fit(disp=-1, method='css')  
 # predict\_ts = properModel.predict(2, dynamic=True)  
 # log\_recover = np.exp(predict\_ts)  
 # ts = ts[log\_recover.index]  
 # er\_rmse,er\_mae,er\_acc,r2\_score,var\_score = evaluation(ts,log\_recover)  
 # rmse.append(er\_rmse)  
 # mae.append(er\_mae)  
 # acc.append(er\_acc)  
 # r2.append(r2\_score)  
 # var.append(var\_score)  
 acc1 = np.mat(acc)  
 acc1[acc1 < 0] = 0  
 print('arima\_rmse:%r' % (np.mean(rmse)),  
 'arima\_mae:%r' % (np.mean(mae)),  
 'arima\_acc:%r' % (np.mean(acc1)),  
 'arima\_r2:%r' % (np.mean(r2)),  
 'arima\_var:%r' % (np.mean(var)))  
  
time\_end = time.time()  
print(time\_end - time\_start, 's')