****

**东北大学秦皇岛分校**

**计算机与通信工程学院**

**工程训练报告**

|  |  |
| --- | --- |
| 专业名称 | 计算机科学与技术 |
| 班级学号 | 计科1802-20188068 |
| 学生姓名 | **孔天欣** |
| 校内指导教师 | 沈哲 |
| 报告时间 | 2021年9月10日 |

基于多元线性回归的秦皇岛PM2.5浓度预测模型

项目工程训练报告

1. 训练任务及要求

**1. 训练任务**

本次项目训练的任务是根据秦皇岛自2016年至2020年9月的每日气象数据预测出2020年9月至2021年8月秦皇岛的PM2.5浓度值。其中，气象数据包含日期、AQI（空气质量指数），PM10，SO2，NO2，CO，O3以及PM2.5的浓度值。本项目基于多元线性回归模型实现预测技术，并借助Python提供的Numpy，Matplotlib，Seaborn，Sklearn工具库来实现可视化图像的绘制和多元线性回归模型的实现。

**2. 训练要求**

本训练任务要求能够通过2020年9月至2021年8月所提供的每日气象数据，来推测出每天可能的PM2.5值，并且能够和当天实际的PM2.5浓度值做到近似，从而体现训练模型的正确预测能力。

1. 相关技术
   1. 线性回归模型

**1. 一元线性回归**

线性回归是回归问题中的一种，线性回归假设目标值与特征之间线性相关，即满足一个多元一次方程。通过构建损失函数，来求解损失函数最小时的参数w和b。通常可以表达成公式如（2-1）所示：

|  |  |
| --- | --- |
|  | （2-1） |

其中，为预测值，自变量x和因变量y是已知的，一元线性回归的目标是实现对于特定的x，获得其对应的预测值y。因此，为了构建该函数关系，要求通过已知数据点，求解线性模型中w和b两个参数。

**2. 多元线性回归**

多元线性回归和一元线性回归有相同的形式定义，但在多元线性回归中，一个因变量开始由多个自变量来决定，它的方程形式如（2-2）式所示。

|  |  |
| --- | --- |
|  | （2-2） |

将方程转变为向量形式，可得式（2-3）：

|  |  |
| --- | --- |
|  | （2-3） |

由此可得多元线性回归模型的矩阵形式（2-4）：

|  |  |
| --- | --- |
|  | （2-4） |

* 1. 损失函数

**1. 损失函数定义**

求解最佳参数，需要一个标准来对结果进行衡量，为此需定量化一个目标函数式，使得线性回归模型程序可在求解过程中不断地进行优化。针对线性回归模型求解问题，最终可以得到一组预测值 ，设已有的真实值为y，数据行数为n，可以将损失函数定义如（2-5）所示：

|  |  |
| --- | --- |
|  | （2-5） |

即预测值与真实值之间的平均的平方距离，统计中一般称其为MAE(Mean Square Error)均方误差。将（2-1）式代入（2-5）式，并且将需要求解的参数w和b看做是函数L的自变量，可得（2-6）式：

|  |  |
| --- | --- |
|  | （2-6） |

现在的任务是求解最小化L时w和b的值，即核心目标优化式为（2-7）式所示：

|  |  |
| --- | --- |
|  | （2-7） |

**2. 最小二乘法（Least Square Method）**

最小二乘法是一种数学优化技术。它通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。

最小二乘法求解w和b是使损失函数最小化的过程，将（2-6）式分别对w和b求导，得（2-8）和（2-9）式：

|  |  |
| --- | --- |
|  | （2-8） |
|  | （2-9） |

令（2-8）和（2-9）式的值为0，变形联立运算后可得到w和b最优解的闭式解（2-10），（2-11）：

|  |  |
| --- | --- |
|  | （2-10） |
|  | （2-11） |

同理可得，多元线性回归下的矩阵模式最小二乘法中W的最优解如式（2-12）所示。

|  |  |
| --- | --- |
|  | （2-12） |

* 1. Python工具库

**1. 科学计算工具Numpy**

NumPy（Numerical Python）是python语言的一个扩展程序库，支持大量的维度数组与矩阵运算，此外也针对数组运算提供大量的数学函数库。它支持大型多维数组对象和各种工具。各种其他的库，如Pandas、Matplotlib和Scikit-learn，都建立于此之上。

**2. 数据结构化工具Pandas**

Pandas是一个基于numpy的强大的分析结构化数据的工具集。它的命名衍生自术语"panel data"和"Python data analysis"，广泛应用在学术、金融、统计学等各个数据分析领域。它可以从各种文件格式比如 CSV、JSON、SQL、Microsoft Excel等数据文件中导入数据。Pandas还可以对各种数据进行运算操作，比如归并、再成形、选择，还有数据清洗和数据加工特征。

**3. 绘图工具Matplotlib**

Matplotlib是一个Python的2D绘图库，它以各种硬拷贝格式和跨平台的交互式环境生成出版质量级别的图形。通过Matplotlib，开发者可以仅需要几行代码，便可以绘制目标图像。一般可绘制折线图、散点图、柱状图、饼图、直方图、子图等等。Matplot使用Numpy进行数组运算，并调用一系列其他的Python库来实现交互。

**4. 增强绘图工具Seaborn**

Seaborn是基于matplotlib的图形可视化python包。它提供了一种高度交互式界面，便于用户能够做出各种有吸引力的统计图表。

Seaborn是在matplotlib的基础上进行了更高级的API封装，从而使得作图更加容易，在大多数情况下使用seaborn能做出很具有吸引力的图，而使用matplotlib就能制作具有更多特色的图。应该把Seaborn视为matplotlib的补充，而不是替代物。同时它能高度兼容numpy与pandas数据结构以及scipy与statsmodels等统计模式。

**5. 机器学习框架Sklearn**

scikit-learn，又写作sklearn，是一个开源的基于python语言的机器学习工具包。它通过NumPy, SciPy和Matplotlib等python数值计算的库实现高效的算法应用，并且涵盖了几乎所有主流机器学习算法。

在工程应用中，用python手写代码来从头实现一个算法的可能性非常低，这样不仅耗时耗力，还不一定能够写出构架清晰，稳定性强的模型。更多情况下，是通过分析采集到的数据，根据数据特征选择适合的算法，在sklearn包中调用算法，调整算法的参数，来获取需要的信息，从而实现算法效率和效果之间的平衡。

1. 任务实现
   1. 爬虫技术获取数据集

在进行线性回归前，首先需要获取模型训练所需的数据集，因此通过requests库提供的爬虫工具，从天气后报网站中提取秦皇岛从2016年8月到2021年8月的所有气象数据，这些数据包括日期、空气质量等级、AQI指数、PM2.5，PM10，So2，No2，CO，O3等，相关代码如代码清单3.1所示。

代码清单3.1 获取秦皇岛气象数据集

|  |
| --- |
| def get\_data():  url = 'http://www.tianqihoubao.com/aqi/qinhuangdao.html'  headers = {  'user-agent': "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36 Edge/18.18362", }  response = requests.get(url, headers=headers)  html = response.text  response = etree.HTML(html)  url\_list = response.xpath('//div[@class="box p"]//a/@href')  for url in url\_list:  url = 'http://www.tianqihoubao.com' + url  data = pd.read\_html(url, header=0, encoding='gbk')[0]  print(data)  time.sleep(1)  data.to\_csv("./pm2.5.csv", mode='a', header=False) |

数据爬取完毕后，在工程目录下生成了文件pm2.5.csv，之后通过pandas读取该文件，生成一个Dataframe对象，该对象包含了文件内的数据集，通过调用head()函数来查看数据集的部分数据信息，代码清单见3.2。

代码清单3.2 读取数据文件

|  |
| --- |
| # 读取数据文件  filename = "./pm2.5.csv"  df = pd.read\_csv  (filename, names=['date', 'quality', 'AQI', 'ranking', 'PM2.5(μg/m3)', 'Pm10(μg/m3)', 'So2(μg/m3)', 'No2(μg/m3)', 'Co(mg/m3)', 'O3(μg/m3)'])  # 显示部分数据  print(df.head()) |

输出结果如图3.1所示。

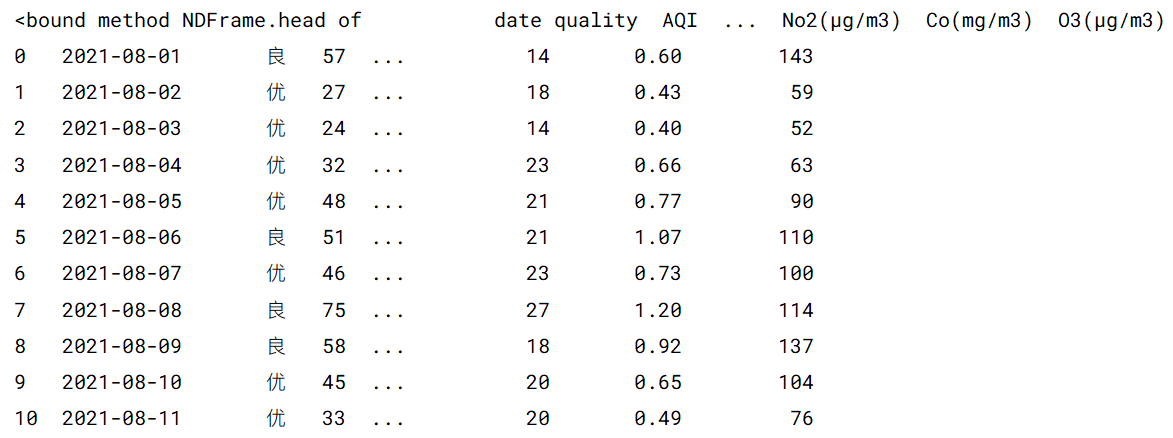


图3.1 Dataframe对象数据内容

* 1. 数据预处理

**1. 时间索引序列变换**

使用info函数对该对象的基本信息进行检视。如代码清单3.2所示。

代码清单3.3 时间序列索引变化

|  |
| --- |
| # 读取数据文件  filename = "./pm2.5.csv"  df = pd.read\_csv  (filename, names=['date', 'quality', 'AQI', 'ranking', 'PM2.5(μg/m3)', 'Pm10(μg/m3)', 'So2(μg/m3)', 'No2(μg/m3)', 'Co(mg/m3)', 'O3(μg/m3)'])  # 显示基本信息  df.info()  # 把日期转为索引  df['time\_index'] = pd.to\_datetime(df['date'])  # inplace = True 不生成新的 DataFrame 对象  df.set\_index('time\_index', inplace=True) |

通过df.info()获取到的数据文件基本信息如图3.2所示。

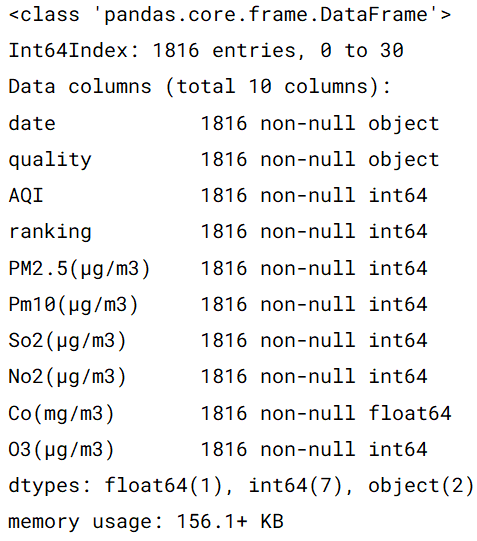


图3.2 Dataframe对象的基本信息

由图3.2可见，爬取的数据共有1816条，同时所有的数据没有空值（都是non-null），因此不需要进行空值预处理，同时，在csv文件的浏览中，也没有发现异常数据，因此无需进行相关处理。

此后将date一列的时间序列转变成索引，索引名为time\_index，同时，在set\_index函数的形参中，使用inplace=True参数来将改变后的Dataframe替换原来的Dataframe对象，而不是另外返回新的对象，这样就可以达到节约内存的目的。

**2. 数据可视化**

将2016年8月到2021年8月中每个月的PM2.5按照月份的平均值进行采样，使用matplotlib库进行绘制，规定x轴是年月时间，y轴是秦皇岛的月均PM2.5值，相关代码如代码清单3.4所示。

代码清单3.4 数据可视化

|  |
| --- |
| # 月均采样  df1 = df.resample('M').mean()  plt.figure(figsize=(25, 10))  data = df1['PM2.5(μg/m3)']  # 提取 x y轴数据  \_x = data.index  \_y = data.values  \_x = [i.strftime("%Y/ %m") for i in \_x]  # 绘制图像  plt.plot(range(len(\_x)), \_y, 'blue')  plt.xticks(range(len(\_x)), \_x, rotation=60)  plt.xlabel('日期')  plt.ylabel('PM2.5(ug/m3)')  plt.title('秦皇岛月均 PM2.5 (ug/m3) 从 2016-08 - 2021-08 的变化 ') |

绘制完毕的图像如图3.3所示，图中曲线可知秦皇岛月均PM2.5的变化历年来都呈现出一定的规律性，PM2.5的浓度基本都在相同的几个月达到了高峰期，以及在某些月份到达低值，因此适合使用多元线性回归模型进行预测未来某个时间点的PM2.5 浓度的大致数值范围。

将采样点改成年平均值后，再次绘制反映秦皇岛年均PM2.5变化的曲线图像，如图3.4所示。可见PM2.5的年均浓度是在逐年下降的。

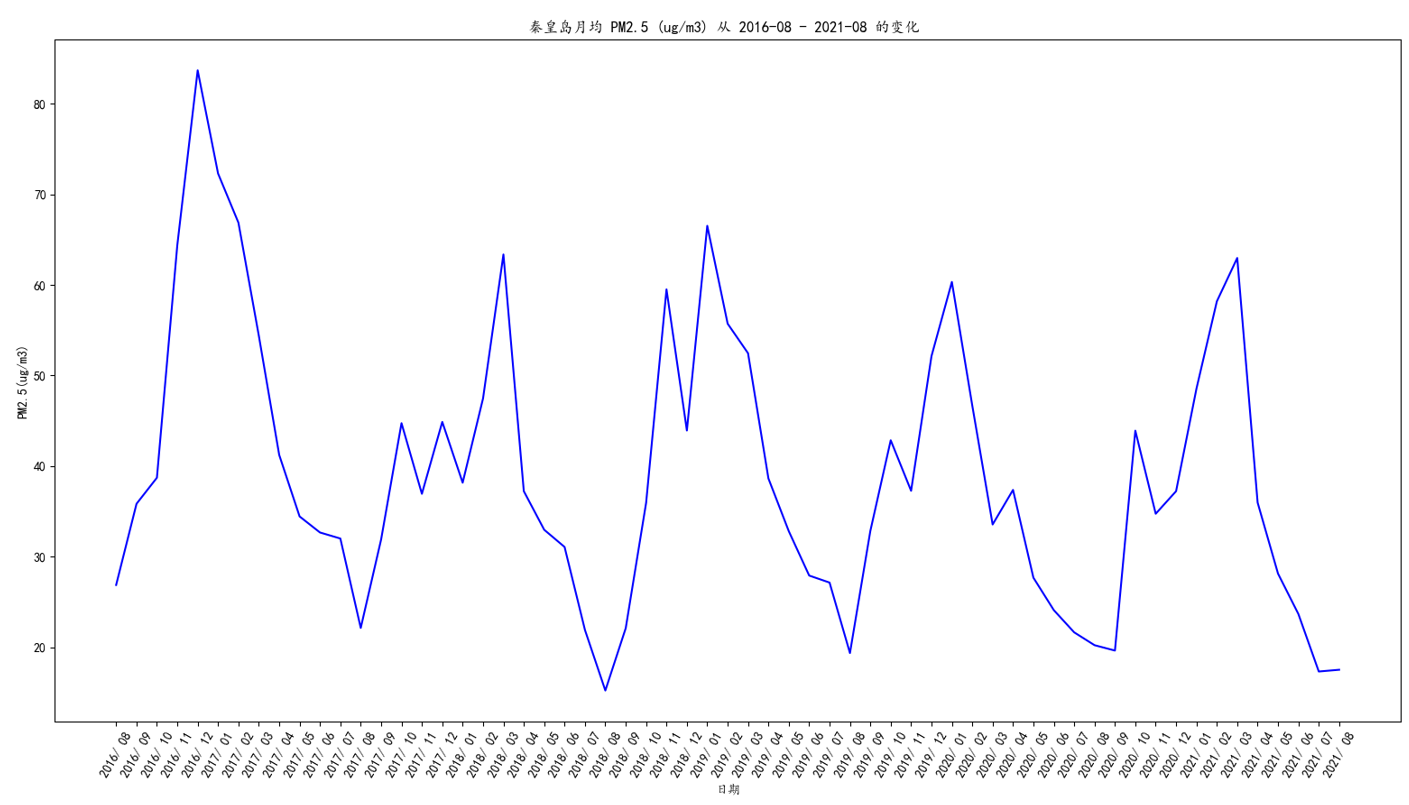


图3.3 秦皇岛月均PM2.5（ug/m3）从2016-08 到 2021-08的变化

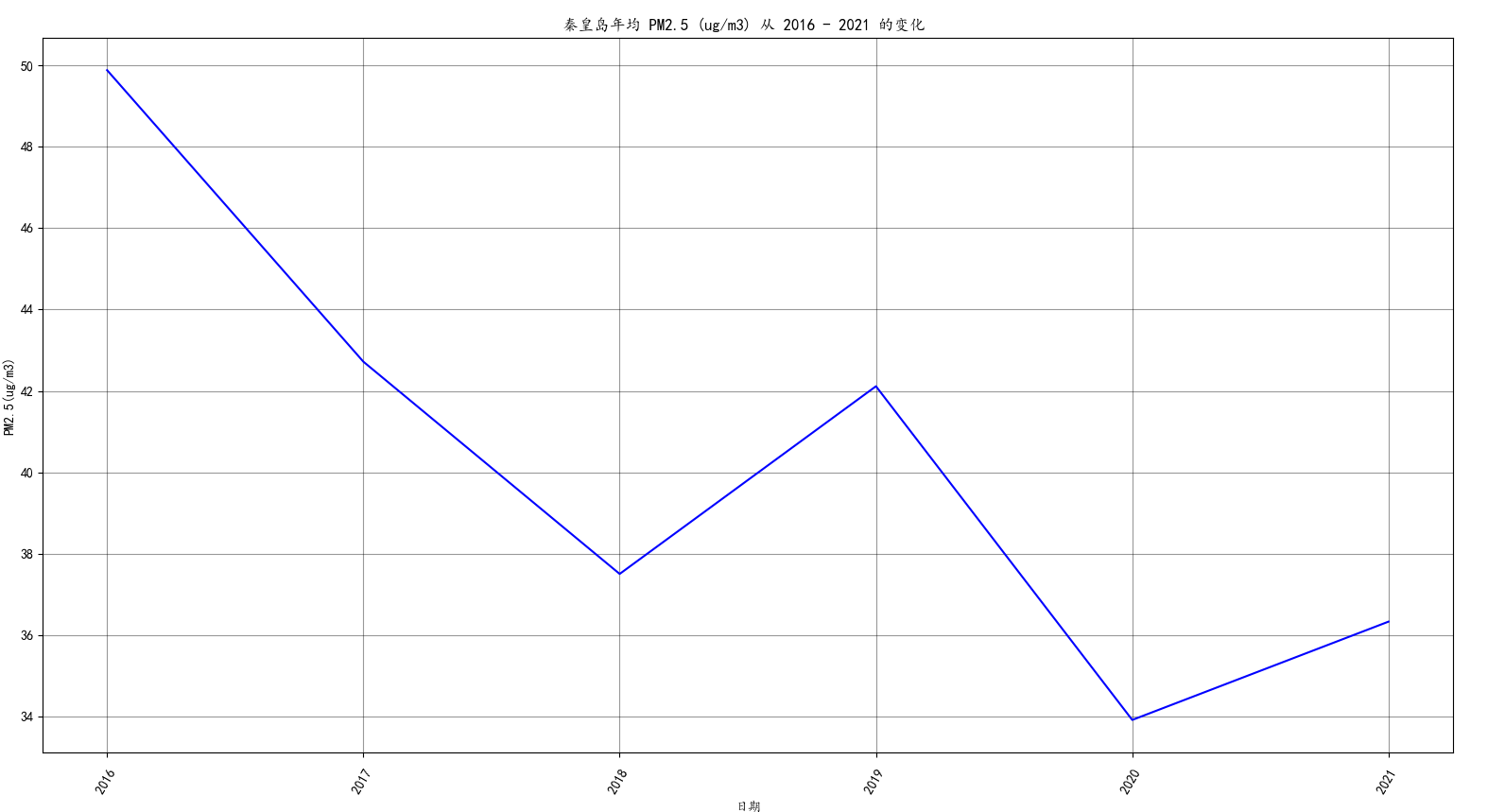


图3.4 秦皇岛年均PM2.5（ug/m3）从2016 到 2021年的变化

此后，再根据Dataframe的数据使用Seaborn库的heatmap函数绘制相关系数矩阵热力图，观察不同变量之间的相关性，绘制出的图像如图3.5所示，由图可知，PM2.5的值和AQI，PM10，NO2和CO的值呈显著的正相关性，和SO2的值呈现出一定的正相关性，和O3的值呈现负相关性。

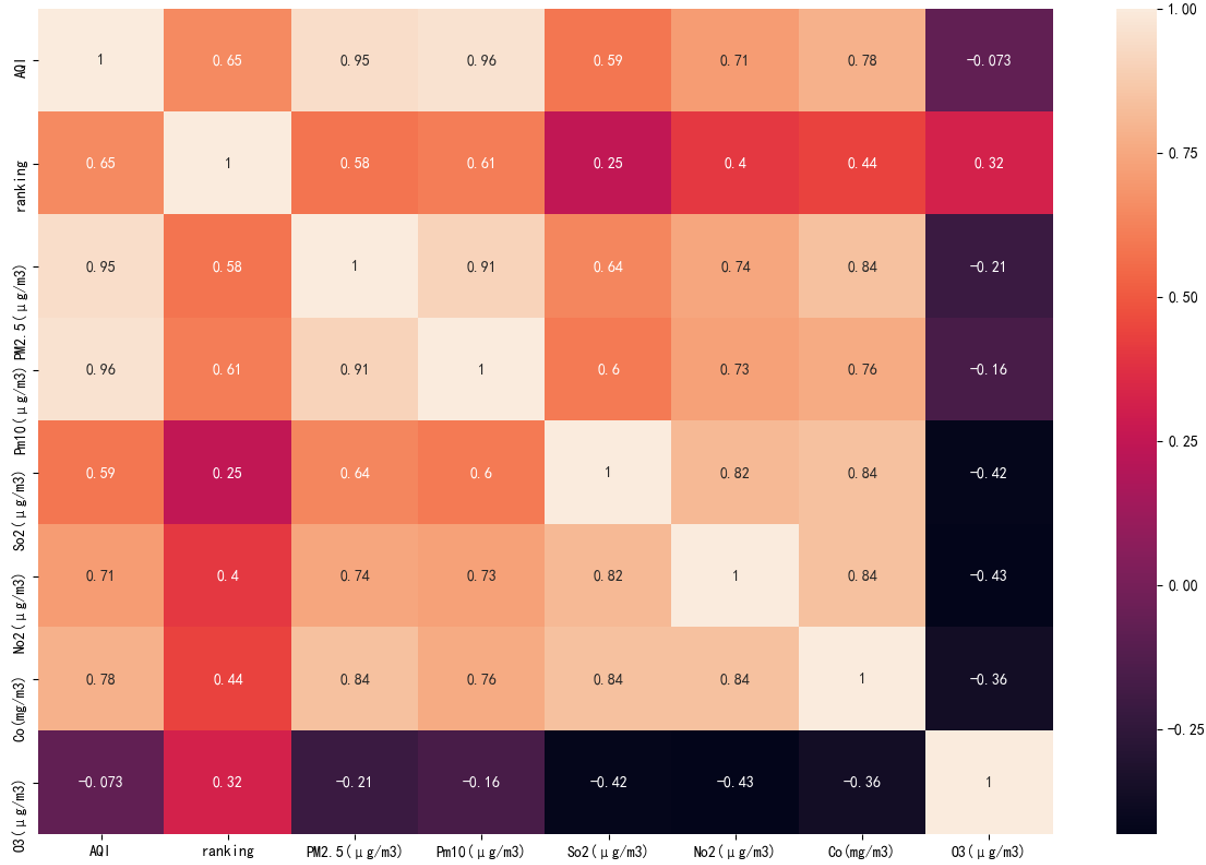


图3.5 数据集相关系数矩阵热力图

绘制相关系数矩阵热力图的相关代码如代码清单3.5所示。

代码清单3.5 绘制相关系数矩阵热力图

|  |
| --- |
| # 绘制热力图  corr = df.corr()  fig = plt.figure(figsize=(15, 15))  sns.heatmap(corr, annot=True)  plt.show() |

* 1. 模型训练

1. 数据集划分

在开始训练之前，首先将数据集进行划分，在该多元线性回归模型中，其中80%是训练集，20%是测试集，然后再剔除无关的因素字段date和quality，剩下的便是自变量AQI，PM10，SO2，NO2，CO，O3，因变量设置为PM2.5。然后将训练集和测试集的所有数据按照自变量和因变量提取成矩阵以及列向量，相关代码如代码清单3.6所示。

代码清单3.6 划分数据集和测试集

|  |
| --- |
| split\_percent = 0.2  # 80%训练集  data\_train = df.iloc[int(len(df) \* split\_percent):]  # 20%测试集  data\_test = df.iloc[:int(len(df) \* split\_percent)]  # 去除无关的字段  data\_train = data\_train.drop(['date', 'quality'], axis=1)  data\_test = data\_test.drop(['date', 'quality'], axis=1)  # 设置训练集的自变量矩阵（除date,quality的其他因素）和因变量y（PM2.5值）  y\_train = data\_train['PM2.5(μg/m3)'].values  x\_train = data\_train.drop('PM2.5(μg/m3)', axis=1).values  # 设置测试集的真实值y\_true和用于测试的自变量x\_test  y\_true = data\_test['PM2.5(μg/m3)'].values  x\_test = data\_test.drop('PM2.5(μg/m3)', axis=1).values |

在划分完毕后，通过输出测试集的列表，可以发现测试集的数据是2020年9月1日至2021年8月31日的气象数据。

**2. 使用多元线性回归模型训练**

通过使用sklearn包提供的回归模型LinearRegression，将训练集的自变量矩阵和因变量向量填入其中，就能方便地进行多元线性回归的训练，使用默认的最小二乘法，训练完毕后输出相关的模型评估指标，这些指标包含平均绝对误差、均方误差、中值绝对误差、可解释反差值、R方值，以及各个影响因素的权重等。相关代码如代码清单3.7所示。

代码清单3.7 多元线性回归模型

|  |
| --- |
| # 进行多元线性回归模型的训练  model = LinearRegression()  model.fit(x\_train, y\_train)  y\_pred = model.predict(x\_test)  # 输出模型评估指标  print('平均绝对误差为', mean\_absolute\_error(y\_true, y\_pred).round(3))  print('中值绝对误差为', median\_absolute\_error(y\_true, y\_pred).round(3))  print('可解释方差值为', explained\_variance\_score(y\_true, y\_pred).round(3))  print('R方值为', r2\_score(y\_true, y\_pred).round(3))  print('各个权重为', model.coef\_) |

输出结果后，首先制作一个预测值和真实值进行比对的Dataframe对象，然后再通过matplotlib绘图工具绘制测试集中预测值和真实值的对比曲线图，即可直观地看出训练后的模型对2020年9月至2021年8月PM2.5浓度预测的准确情况，相关代码如代码清单3.8所示。

代码清单3.8 绘制秦皇岛PM2.5浓度预测值和真实值比对图像

|  |
| --- |
| # 将真实值转换成Dataframe对象  time = test\_time.values  df\_true = pd.DataFrame({'date': time, 'PM2.5(μg/m3)': y\_true})  df\_true['time\_index'] = pd.to\_datetime(df\_true['date'])  df\_true.set\_index('time\_index', inplace=True)  # 将预测值转换为Dataframe对象  df\_pred = pd.DataFrame({'date': time, 'PM2.5(μg/m3)': y\_pred})  df\_pred['time\_index'] = pd.to\_datetime(df\_pred['date'])  df\_pred.set\_index('time\_index', inplace=True)  # 制作预测值和真实值比较的Dataframe对象  df\_cmp = pd.DataFrame({'date': time, 'PM2.5\_true': y\_true, 'PM2.5\_predict': y\_pred})  # 绘制预测值和真实值对比的图像  df\_true['PM2.5(μg/m3)'].plot(figsize=(25, 10), color='blue', grid=True, fontsize=15, label='预测值')  df\_pred['PM2.5(μg/m3)'].plot(figsize=(25, 10), color='red', grid=True, fontsize=15,label='真实值')  plt.xticks()  plt.legend(loc="upper right")  plt.title('回归模型预测秦皇岛 PM2.5 (ug/m3) 从 2020-9 - 2021-8 预测值和真实值的对比 ')  plt.show() |

1. 实验结果及性能分析

经过多元线性回归模型训练后，将秦皇岛2020年9月至2021年8月PM2.5浓度的真实值和预测值绘制对比曲线，得到实验结果如图4.1所示。

由图4.1可见，预测值和真实值的误差处于正常波动范围内，并且预测值变化的趋势和真实值也呈现一致，仅由部分时间点PM2.5的值差别较大，这也说明了本实验模型仍有亟待改进之处。

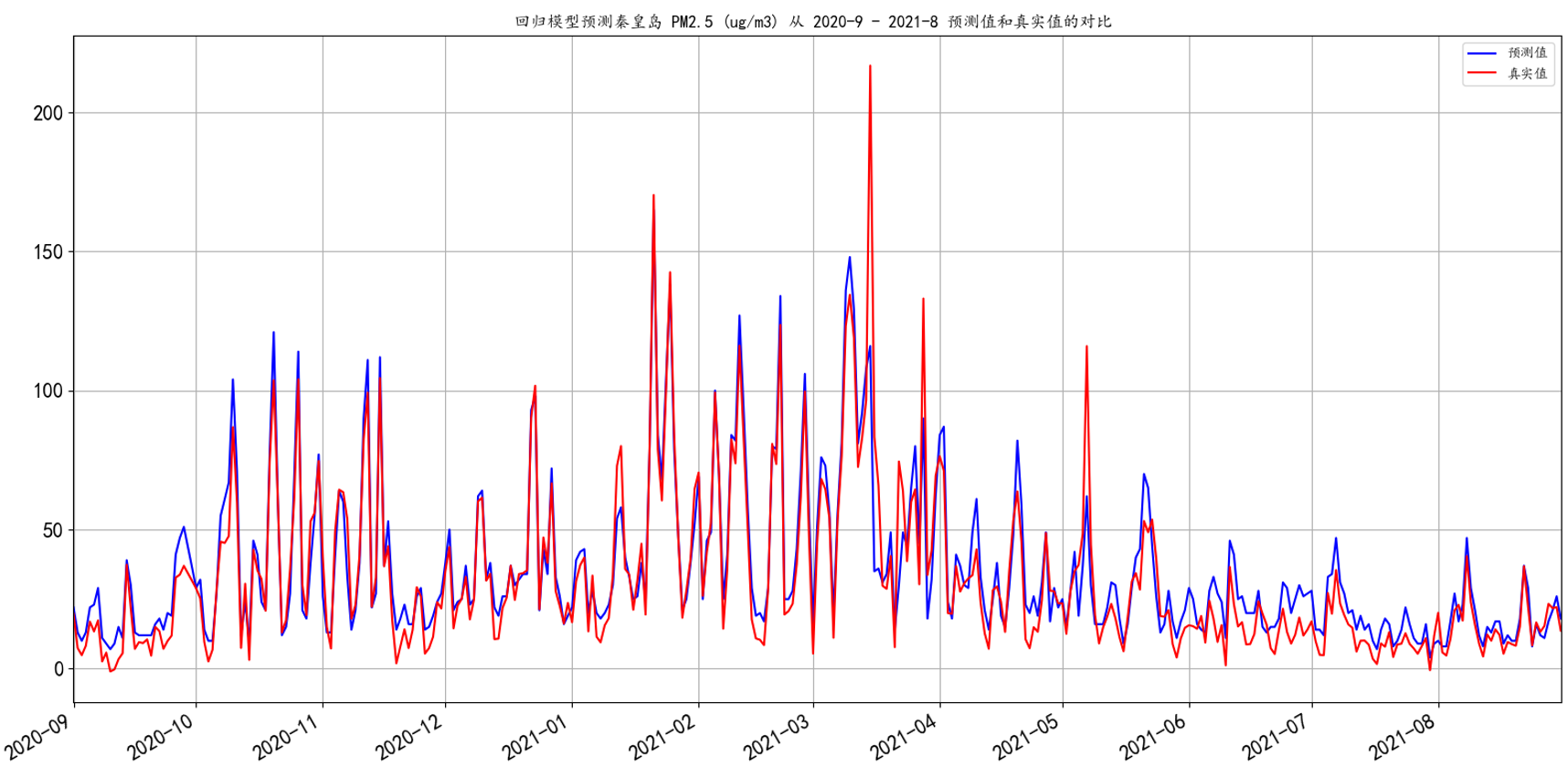


图4.1 秦皇岛PM2.5浓度从2020-9至2021-8预测值和真实值对比

为使数据能够看到精确的比对情况，将预测值和真实值与时间序列进行合并，生成一张对比表输出至控制台，控制台中该表的数值信息如图4.2所示。图中展示了从2021年8月1日至2021年8月16日的秦皇岛PM2.5浓度真实值（PM2.5\_true）和预测值（PM2.5\_predict）的对比情况。可见，预测值和真实值的误差基本都在10以内，因此该模型可以较为准确的进行预测。

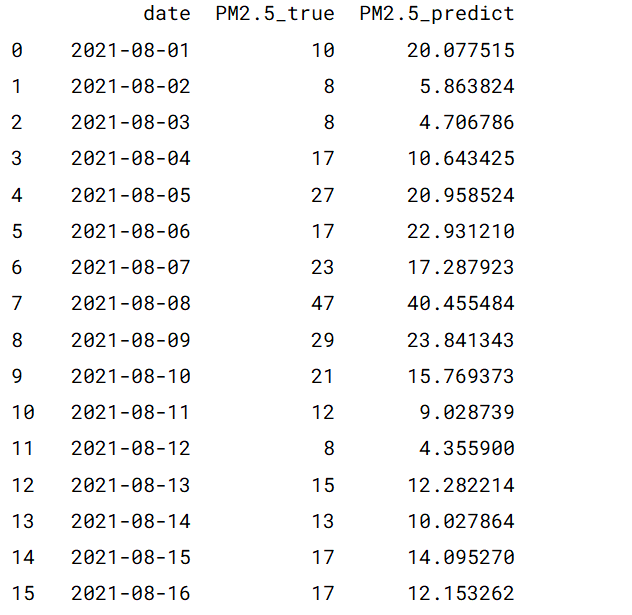


图4.2 秦皇岛PM2.5从2021-8-1日至2021-8-16预测值和真实值对比表

该模型的评估指标如表4.1所示。可见MAE的值在7.244，说明真实值和预测值绝对差的平均和处于较低的水平，同时R方值能够达到0.85，说明预测值和真实值的数据是非常接近的。在权重一列中，PM10，CO和AQI起到了相当积极的正面影响，而SO2的值对PM2.5浓度则起到了显著的负面影响，此外，NO2和O3则对PM2.5浓度值预测起的作用不是很大。

总体而言，该模型能够较为准确地从测试集中根据一系列影响因素来预测出秦皇岛某个时间点PM2.5浓度的大致数值，尽管仍有不足之处，大体上完成了工程训练的基本任务要求。

表4.1 秦皇岛PM2.5预测模型的评估指标表

|  |  |  |
| --- | --- | --- |
| 指标 | 值 | |
| 均方绝对误差（MAE） | 7.244 | |
| 中值绝对误差 | 5.746 | |
| 可解释反差值 | 0.861 | |
| R方值 | 0.850 | |
| 权重 | AQI | 7.52 |
| PM10 | 6.13 |
| SO2 | -9.45 |
| NO2 | -1.11 |
| CO | 8.38 |
| O3 | -1.37 |

1. **心得体会**

通过本次工程训练，在老师的教导下，本人初步学习并掌握了Numpy，Matplotlib，Seaborn，Sklearn工具库的使用方法，并能够将其应用到本人的项目当中。此外，老师还教授了机器学习入门的基本知识，例如线性回归、逻辑回归和神经网络，并能够使用Jupyter Notebook作为工具，在上面进行相关项目的实战。在学习和制作项目的过程中，本人虽然遇到了一些技术知识点上的困难，但在老师和同学的悉心帮助下，以及在网络上搜索解决方案，逐步探索并克服了相关问题，并最后能够实现一个简单的秦皇岛PM2.5多元线性回归预测模型。

在为时两周的工程训练中，尽管本人初步对人工智能的理论体系有所掌握，但由于人工智能相关的实战训练还是偏少，因此仍有较多理论和实践上的不足，例如算法的数学原理、工具包的使用技巧等。这是需要更多后续的系统训练提高经验来解决的。因此本人将在将来更多地深入学习人工智能相关技术，并将其和自己研究的领域相结合，碰撞交融出创新的思想火花。