**Python爬虫与地理可视化**

一．成员介绍

|  |  |  |
| --- | --- | --- |
| **成员姓名** | **学院** | **专业** |
| 黄煜鲲 | 数学与统计学院 | 统计学 |
| 吴宇辉 | 软件学院 | 软件工程 |
| 魏昕格 | 资源环境与地球科学学院 | 大气科学 |
| 胡鑫 | 化工 | 化学 |
| 王宇 | 数学与统计学院 | 信息与计算科学 |
| 张晴 | 数学与统计学院 | 统计学 |
| 李鸿潇 | 经济学院 | 经济学 |
| 刘华 | 软件学院 | 网络工程 |

二．作品简介

利用Python爬虫对天气网站的空气指标进行爬取，将爬取的数据实现可视化，反映不同地区不同空气指标的情况。

三．总体设计

(a)基本思路

(1)通过爬虫对天气预报网站中的空气指标进行爬取，将网站上每个城市天气的各项指标导入Excel；

(2)城市名调用百度地图API，实现经纬度转换；

(3)用pyechats实现数据可视化；

(4)针对不同的指标，展示出不同地区的不同空气指标的分布程度.

(b)主要技术难点和解决方案

技术难点：

(1)部分网页因为数据缺失，导致正则提取失败；

(2)调用百度地图API是仍有部分城市的经纬度结果出不来；

(3)定义获取网页的函数需要城市代码，而所有城市代码难获取；

(4)最终获取的地区代码有2600多个，需要从中提取出市级代码.

解决方案：

(1)将提取失败的网页地区代码设置报错并浏览该网页，发现有空气污染物缺失；

(2)和上面采用同样的方法并返回网页查明原因，得到百度地图没有该地区的结论，故进行删去；

(3)通过对网页的观察，发现有储存所有城市代码的字典，利用正则提取，得到代码；

(4)观察得到的城市代码有两个固定的数字，对此入手，将所有非市级地区代码去除.

四．特色和创新点

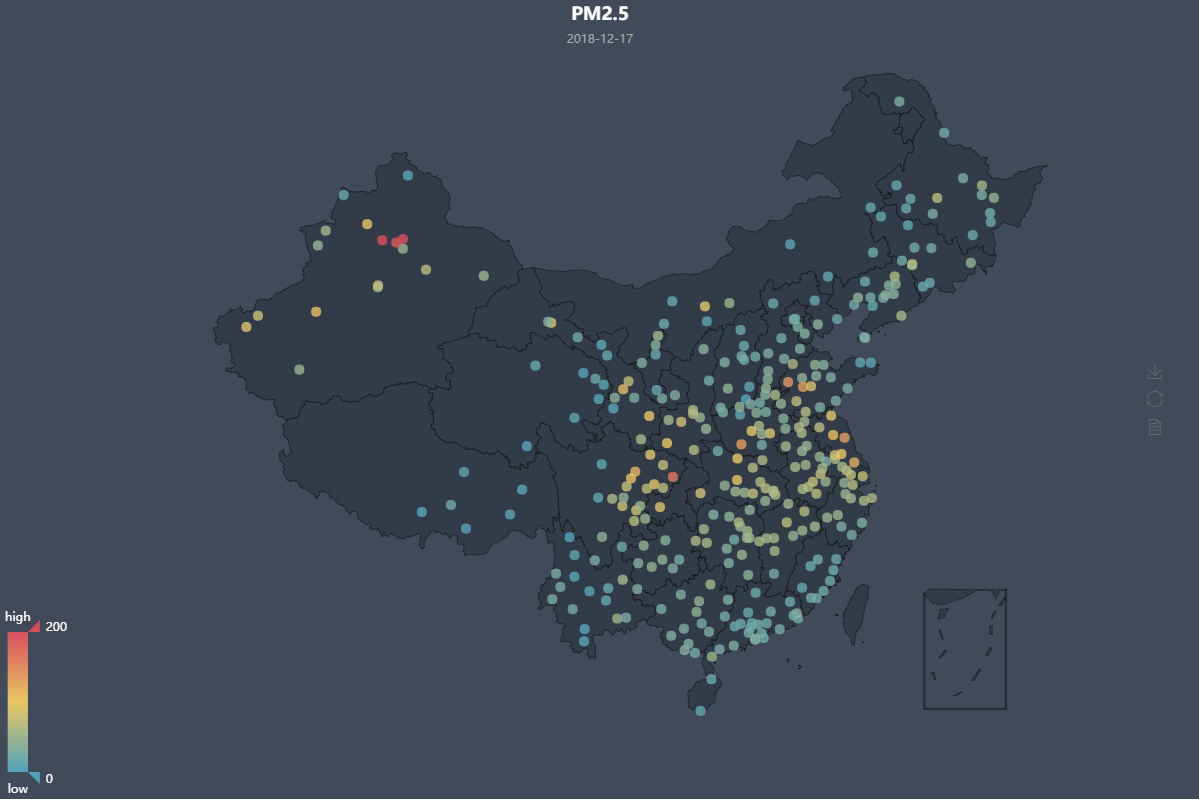
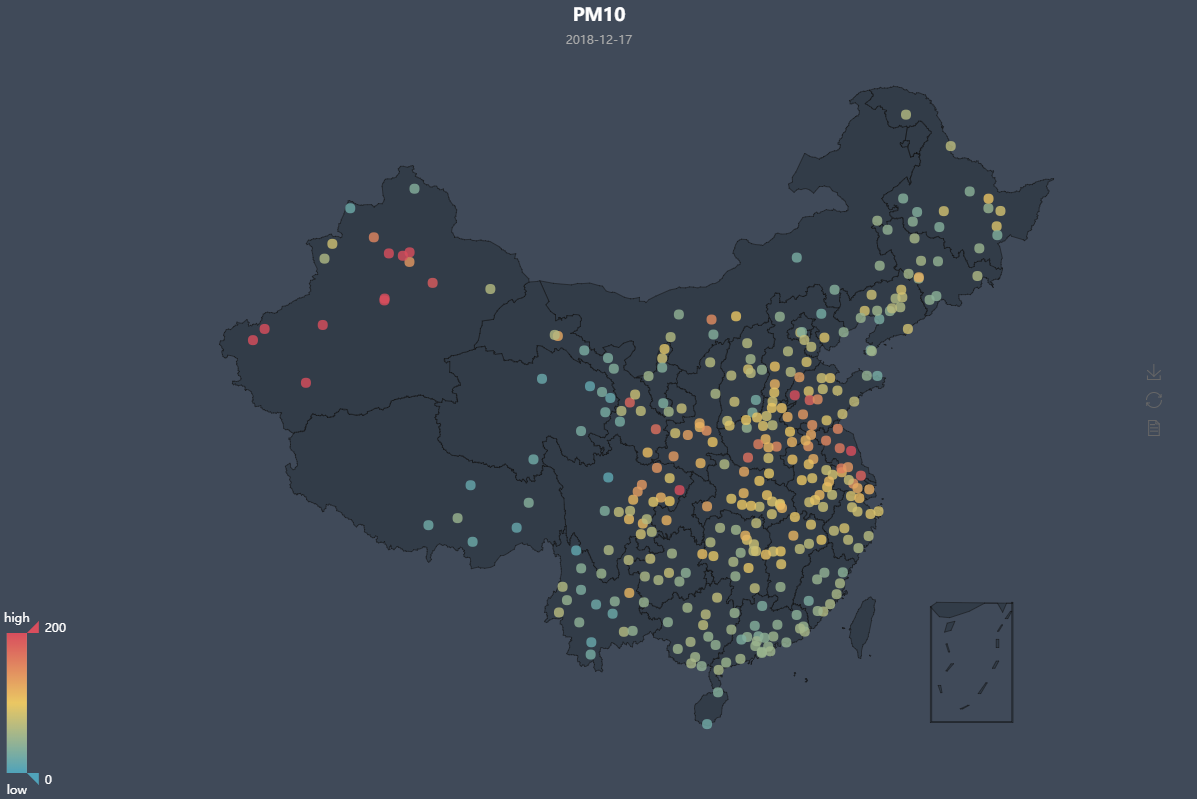
(a)特色

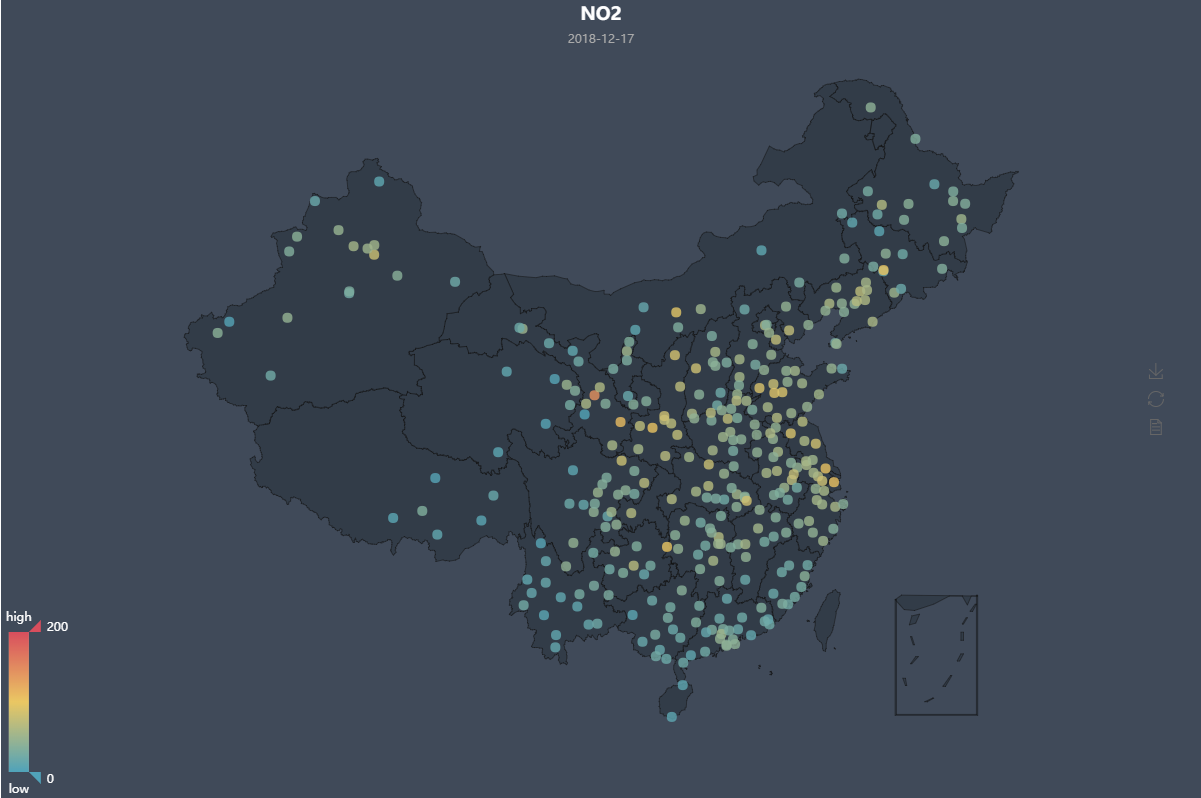
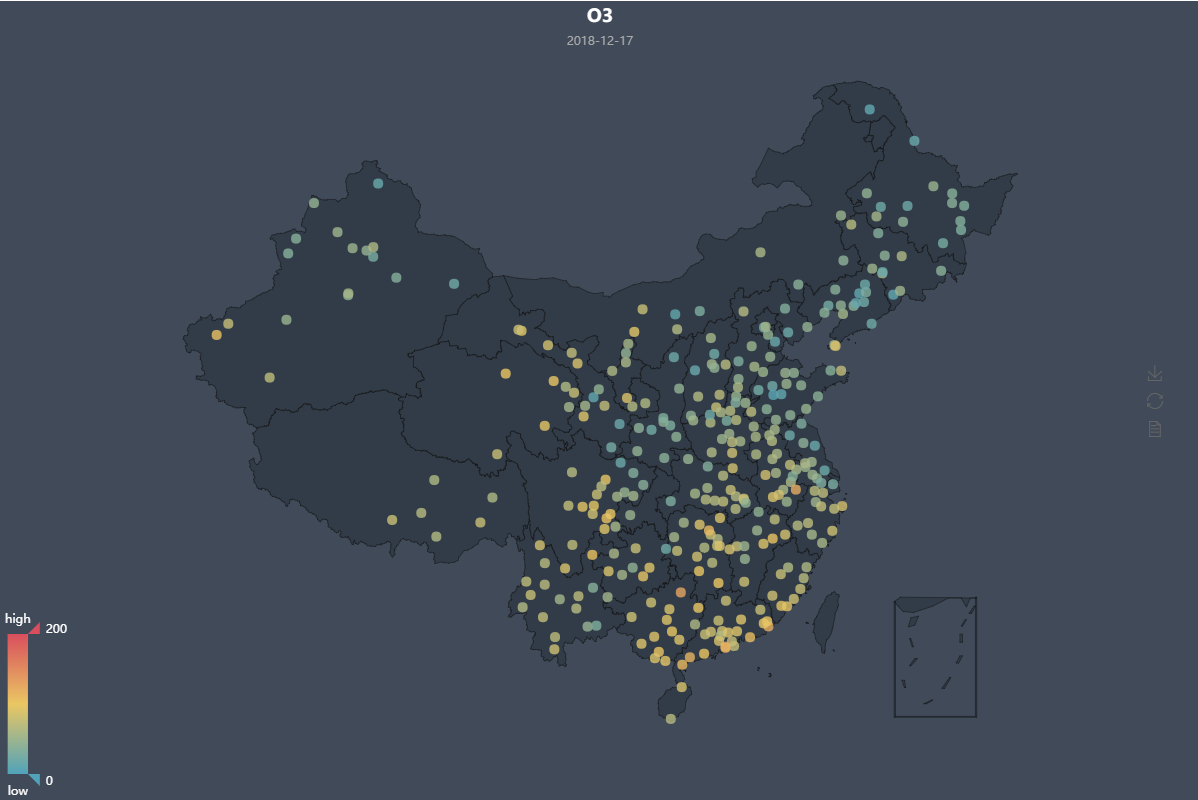
利用Python爬虫爬取300多个市级地区的空气指标情况，再对数据进行处理实现数据的可视化，可以让人们直观的看出不同地区不同空气指标的状况，具有现实意义.

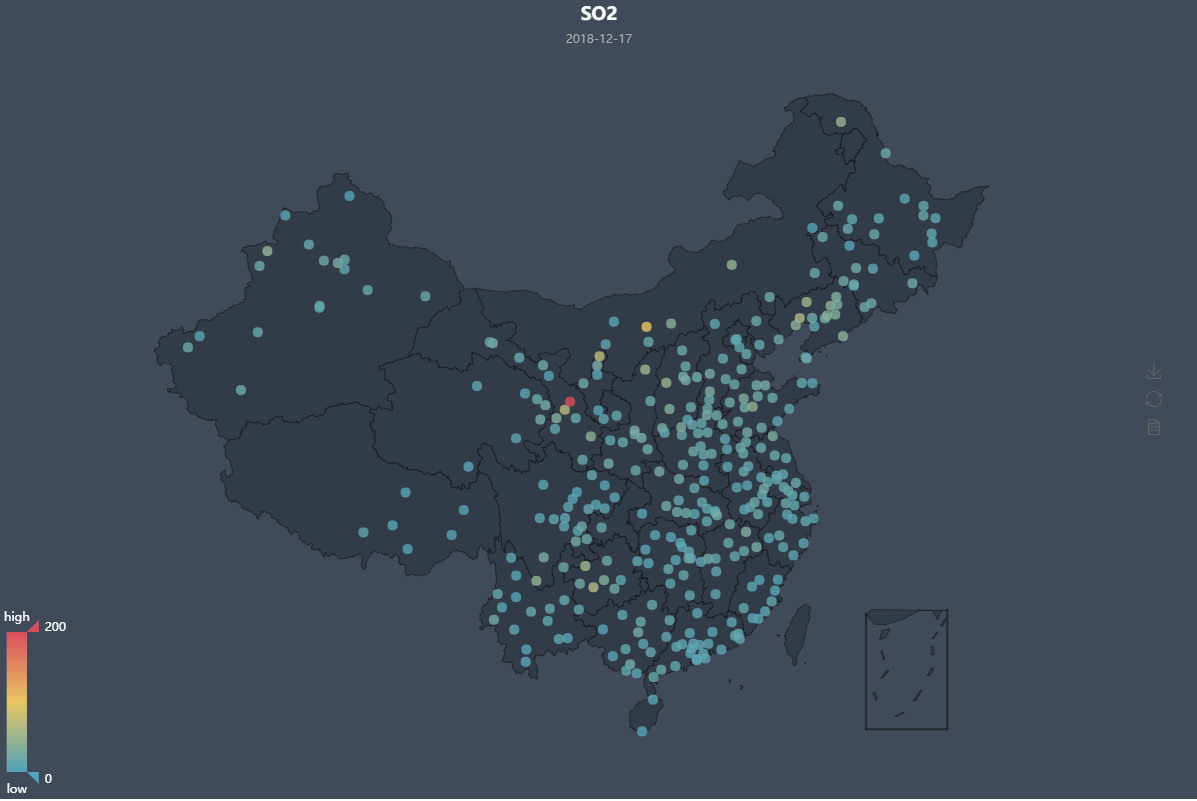
(b)创新点

Excel表格数据处理

利用Python实现数据可视化

五．运行截图





六．项目组成员的工作心得

成员一：黄煜鲲，2017级，项目负责人

学习能力方面：提升了自学能力，了解了该如何运用网络去解决实际中不会的问题

独立思考能力方面：通过对大量代码的理解，学会了在原代码上加以独立思考并改进

发现问题能力方面：在实践中不断的更正错误，不断的完善，已能从报错的类型迅速找出出现问题的代码行

其它方面：知识得到了拓展，学会了更多更实用的方法，函数及模块

成员二：张晴，2017级，项目负责人

学习能力方面：学习能力强，喜欢发现问题并解决

独立思考能力方面：擅于独立思考问题

发现问题能力方面：喜欢主动地去发现问题

其它方面：通过Python的数据可视化的初步学习，了解到Echarts这个强大的工具库使我们可以更加直观地观察数据

成员三：魏昕格，2017级，项目参与者

学习能力方面：一般，基本能跟上课程，课后有认真看线上视频。

独立思考能力方面：一般，只会运用课堂上讲的知识。课后能够独立完成作业。

发现问题能力方面：发现问题后会求助互联网和小组成员。

其它方面：无

成员四：李鸿潇，2017级，项目参与者

学习能力方面：学会了通过各种手段联合解决问题，包括书，搜索，查询

独立思考能力方面：能够自主解决很多问题

发现问题能力方面：能够主动找到一些问题

其它方面：各方面都有很大的提升

成员五：王宇，2016级，项目参与者

学习能力方面：能够独立解决问题，对python有浓厚的兴趣

独立思考能力方面：喜欢独立思考问题，对于解决不了的问题擅于网上学习

发现问题能力方面：喜欢主动地去发现问题

其它方面：学会了excel数据处理，学会了用pyecharts库实现数据可视化

成员六：胡鑫，2017级，项目参与者

学习能力方面：对python有浓厚的兴趣

独立思考能力方面：课后作业可以独立思考，集体作业依赖太多

发现问题能力方面：发现自己确实是零基础学python

其它方面：上课形式新颖，利于我们自由发挥

成员七：刘华，2016级，项目参与者

学习能力方面：参与项目进度，过程中发现新的知识，通过查阅各种资源培养了自主查找资料进行学习的能力

独立思考能力方面：遇到问题需要自己思考来解决，以此培养了独立思考能力

发现问题能力方面：项目的后期优化较项目的实现更为重要和困难，通过分析算法，锻炼了发现问题的能力

成员八：吴宇辉，2017级， 项目参与者

学习能力方面：通过这次项目的实践过程，在学习方面，我感觉我的自学能力有所提高。由于课堂上的作业很多都需要自己完成，需要自己在课下花费比较多的时间来完成。同时由于没有固定的教材，需要自己在网上搜索文献自学。在这样的过程中，我的自学能力得到了提高。

独立思考能力：这门课的特点是作业没有固定的答案，很多作业题目都是可以自己天马行空地想。同时，这门课地作业一部分是要求个人独立完成的，需要独立思考，因此我的独立思考能力、创新能力有所提高。

发现问题能力方面：在我们小组的项目完成过程中，我发现了一些我原有的编程方面的坏习惯，同时也纠正了一些原有的对于编程方面的错误理解。能更好地反思自己，更深入地思考，由此发现更多问题。

其他方面：本次地学习过程中，有一部分作业是需要小组合作完成的，组后的大项目也是由整个小组一起完成。在这个过程中我感觉我自己的团队合作能力有所提高，能够更好的表达自己的想法，和同学合作更加融洽。

七．存在的问题，建议及其他需要说明的情况

(1)请求多网页，设置爬取间隔和模拟浏览器，防止被反爬虫针对；

(2)定义请求函数时要标注每一个是否成功，方便定位问题的所在；

(3)要学会从网页中发现爬取数据以外的数据，发现其他有用的信息可以提高爬取效率；

(4)通过Echarts可实现数据可视化，得到的结果可交互，更方便人们分析数据.

八．附件：代码

指标爬取：

import re

import pandas as pd

import openpyxl

import requests

import bs4

import time

user\_agent="Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10\_6\_8; en-us) AppleWebKit/534.50 (KHTML, like Gecko) Version/5.1 Safari/534.50"

headers={'User-Agent':user\_agent}

page1=requests.get('https://tianqi.so.com/weather/101290101/?tdsourcetag=s\_pctim\_aiomsg',headers=headers)

page1.encoding='utf-8'

html1=page1.text

soup=bs4.BeautifulSoup(html1,'lxml')

results1=re.findall('{"namecn":"(.\*?)","area\_id":"(.\*?)"}',str(soup))

cities=[]

for i in results1:

if i[-1][-1]=='1' and i[-1][-2]=='0':

cities.append(i[1])

for i in results1:

if i[1] in ['101010100','101020100','101030100','101040100']:

cities.append(i[1])

cities=sorted(cities)

pages=[]

Names=[]

PM25=[]

PM10=[]

O3=[]

NO2=[]

SO2=[]

len(cities)

def get\_one(number):

user\_agent="Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10\_6\_8; en-us) AppleWebKit/534.50 (KHTML, like Gecko) Version/5.1 Safari/534.50"

headers={'User-Agent':user\_agent}

url='https://tianqi.so.com/air/'+str(number)

page=requests.get(url,headers=headers)

if page.status\_code==200:

page.encoding='utf-8'

soup=bs4.BeautifulSoup(page.text,'lxml')

sou=soup.select('span.g-fl')

name=re.search('【(.\*?)PM2.5实时查询】',str(soup)).group(1)

if len(sou)>=5:

PM25.append(sou[0].text)

PM10.append(sou[1].text)

O3.append(sou[2].text)

NO2.append(sou[3].text)

SO2.append(sou[4].text)

Names.append(name)

print(number+'OK!')

else:

print(number+'re Failed!')

else:

print(number+'get response Failed!')

for i in cities:

get\_one(i)

time.sleep(1)

Frame1=pd.DataFrame({'city':Names,'PM2.5':PM25,'PM10':PM10,'O3':O3,'NO2':NO2,'SO2':SO2})

Frame1.to\_excel('G:\\py\_pro\\pollution111.xlsx')

百度API:

import requests

import pandas as pd

import re

from urllib.parse import quote

import openpyxl

import time

ak='4IU3oIAMpZhfWZsMu7xzqBBAf6vMHcoa'

user\_agent="Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10\_6\_8; en-us) AppleWebKit/534.50 (KHTML, like Gecko) Version/5.1 Safari/534.50"

headers={'User-Agent':user\_agent}

lng=[]

lat=[]

cities=[]

data=pd.read\_excel('G:\\py\_pro\\pollution111.xlsx')

def get\_lng\_lat(place,data):

url='http://api.map.baidu.com/geocoder/v2/?address='+quote(place)+'&output=json&ak='+ak+'&callback=showLocation //GET请求'

response=requests.get(url,headers=headers)

if response.status\_code==200:

response.encoding='utf-8'

result=re.search('"location":{"lng":(.\*?),"lat":(.\*?)}',response.text)

if result==None:

print(place+' get wrong!')

data=data[data['city']!=str(place)]

else:

lng.append(result.group(1))

lat.append(result.group(2))

print(place+' OK!')

def test(place):

url='http://api.map.baidu.com/geocoder/v2/?address='+quote(place)+'&output=json&ak='+ak+'&callback=showLocation //GET请求'

response=requests.get(url,headers=headers)

response.encoding='utf-8'

result=re.search('"location":{"lng":(.\*?),"lat":(.\*?)}',response.text)

print(result)

for city in data['city']:

get\_lng\_lat(city.strip(),data)

time.sleep(0.5)

data=data[data['city']!='阿左旗']

data=data[data['city']!='昆玉']

data['lng']=lng

data['lat']=lat

data.to\_excel('G:\\py\_pro\\pollution222.xlsx')

Pyecharts可视化：

import pandas as pd

from pyecharts import Map

from pyecharts import Geo,Page

data=pd.read\_excel('G:\\py\_pro\\pollution222.xlsx')

cities=[]

PM25=[]

posi=[]

for i in data['city']:

cities.append(i)

for i in data['PM2.5']:

PM25.append(i)

i2=zip(cities,PM25)

z=zip(round(data['lng'],2),round(data['lat'],2))

for i in z:

posi.append(list(i))

z2=dict(zip(cities,posi))

pages=Page()

my\_map1=Geo('全国部分城市PM2.5情况分布',width=1200,height=600,background\_color="#404a59",title\_color="#fff",title\_pos="center")

my\_map1.add('',cities,data['PM2.5'],title\_color="#fff",

title\_pos="center",maptype='china',symbol\_size=10,geo\_cities\_coords=z2,

type='scatter',is\_visualmap=True,visual\_range=[min(data['PM2.5']),max(data['PM2.5'])],

is\_piecewise=True,visual\_text\_color="#fff",is\_label\_emphasis=True,label\_formatter='{b}')

pages.add\_chart(my\_map1)

my\_map2=Geo('全国部分城市PM10情况分布',width=1200,height=600,background\_color="#404a59",title\_color="#fff",title\_pos="center")

my\_map2.add('',cities,data['PM10'],maptype='china',symbol\_size=10,geo\_cities\_coords=z2,title\_color="#fff",

title\_pos="center",type='scatter',is\_visualmap=True,visual\_range=[min(data['PM10']),max(data['PM10'])],

is\_piecewise=True,visual\_text\_color="#fff",is\_label\_emphasis=True,label\_formatter='{b}')

pages.add\_chart(my\_map2)

my\_map3=Geo('全国部分城市O3情况分布',width=1200,height=600,background\_color="#404a59",title\_color="#fff",title\_pos="center")

my\_map3.add('',cities,data['O3'],maptype='china',symbol\_size=10,geo\_cities\_coords=z2,visual\_range=[min(data['O3']),max(data['O3'])],type='scatter',is\_visualmap=True,

is\_piecewise=True,visual\_text\_color="#fff",is\_label\_emphasis=True,label\_formatter='{b}')

pages.add\_chart(my\_map3)

my\_map4=Geo('全国部分城市NO2情况分布',width=1200,height=600,background\_color="#404a59",title\_color="#fff",title\_pos="center")

my\_map4.add('',cities,data['NO2'],maptype='china',symbol\_size=10,geo\_cities\_coords=z2,visual\_range=[min(data['NO2']),max(data['NO2'])],

type='scatter',is\_visualmap=True,title\_color="#fff",title\_pos="center",

is\_piecewise=True,visual\_text\_color="#fff",is\_label\_emphasis=True,label\_formatter='{b}')

pages.add\_chart(my\_map4)

my\_map5=Geo('全国部分城市SO2情况分布',width=1200,height=600,background\_color="#404a59",title\_color="#fff",title\_pos="center")

my\_map5.add('',cities,data['SO2'],maptype='china',symbol\_size=10,geo\_cities\_coords=z2,title\_color="#fff",title\_pos='center',

type='scatter',is\_visualmap=True,visual\_range=[min(data['SO2']),max(data['SO2'])],

is\_piecewise=True,visual\_text\_color="#fff",is\_label\_emphasis=True,label\_formatter='{b}')

pages.add\_chart(my\_map5)

pages.render()