**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称：** | **Python数据分析** |
| **学生姓名：** | **邱俊源** |
| **学生学号：** | **201830730406** |
| **学生专业：** | **金融学（汇丰精英班）** |
| **开课学期：** | **2020-2021年第一学期** |

**经济与贸易学院**

**2019年9月**

**目 录**

[实验一 数据抓取 2](#_Toc60866116)

[实验二 数据清洗 19](#_Toc60866117)

[实验三 地区实证 30](#_Toc60866118)

[实验四 企业实证 47](#_Toc60866119)

# 数据抓取

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | 楼 | 房； | **实验台号：** |  |
| **实验日期与时间：** |  | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** |  |

【实验目的及要求】

掌握Python 爬虫的使用。使用Python的Request、lxml、BeautifulSoup、selenium等工具包，从知识产权官方媒体的抓取相关数据

【实验内容】

第一部分，基础部分

1. 国家知识产权局（2008及以后）专利统计数据的收集
2. 从佰腾网完成各省不同类型专利数据的收集
3. 主要省份知识产权官方微博数据的抓取
4. 主要省机关报数据的抓取

**一、国家知识产权局（2008及以后）专利统计数据的收集**

打开国家知识产权局网站<https://www.cnipa.gov.cn/col/col61/index.html#mark>并滑动到如下页面

****

在国家知识产权局统计年报中搜索年份2018



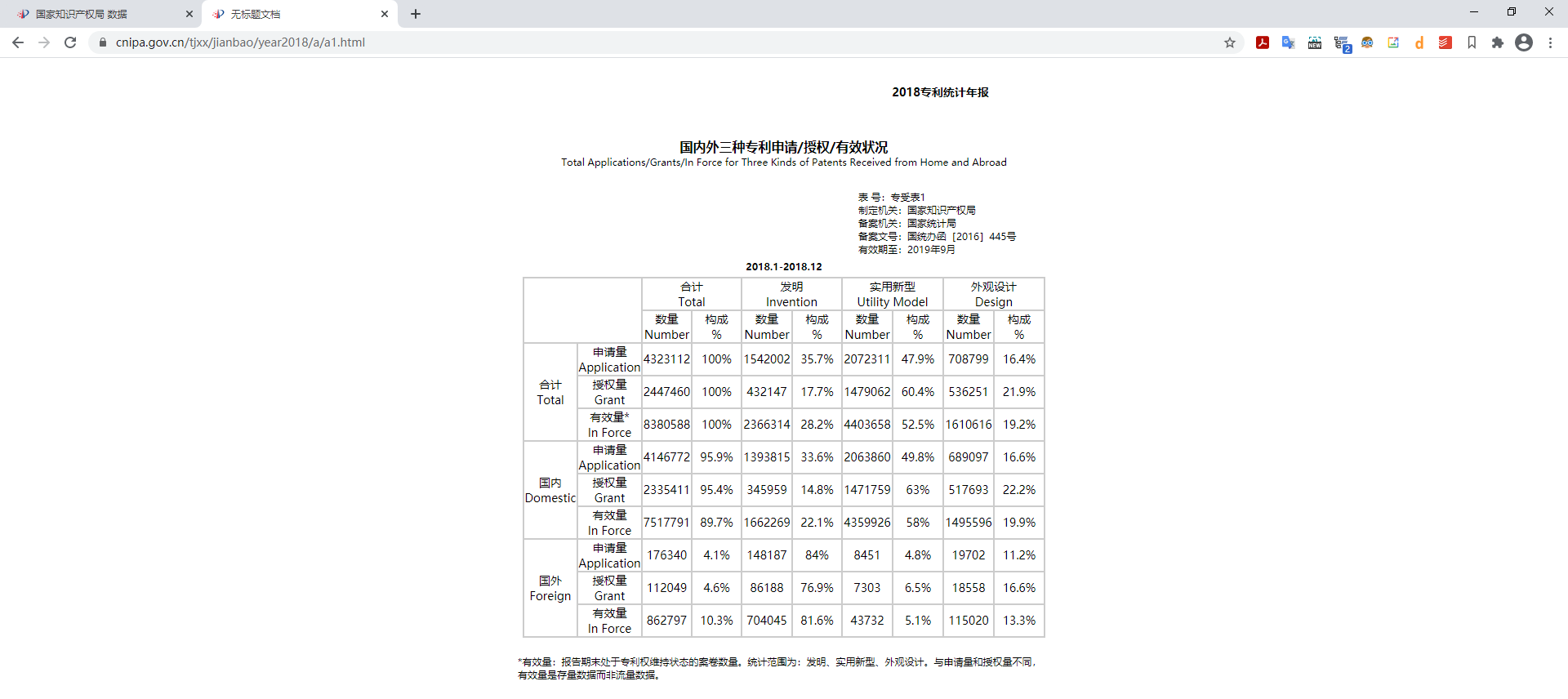
网址为：<https://www.cnipa.gov.cn/tjxx/jianbao/year2018/indexy.html>

点开一个板块后再次观察



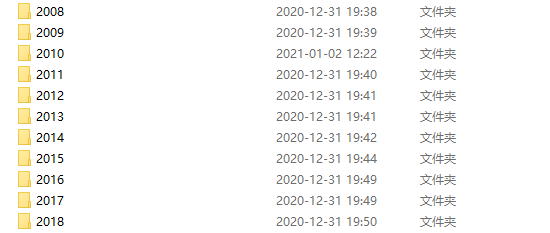
网址为：<https://www.cnipa.gov.cn/tjxx/jianbao/year2018/a.html>

再次点开一个板块观察



终于到了我们需要爬取的内容了，此网址为<https://www.cnipa.gov.cn/tjxx/jianbao/year2018/a/a1.html>

通过观察，我们发现可以通过修改网址来获取不同的页面。但我们需要提前知道一个网页到底有多少个模块才能通过修改部分网址名称来访问，比较麻烦。这里我使用了一个新的函数， requests\_html的HTMLSession，其可以直接获取网页中所有的链接，非常适合这种只有我们需要的链接而没有其他多于链接的网页的爬取。**但该函数在阿里云上运行时进入最初的页面都会被识别为爬虫而报错无法访问，所以无法在阿里云上爬取**。最后我们使用pd.read\_html()函数来获得表格的内容并把它拼接在一起最后输出到excel中。最后分年份保存如下



1. **import** pandas as pd
2. **import** numpy as np
3. **from** requests\_html **import** HTMLSession
4. **import** re
5. **import** time
6. **import** os
7. **import** xlwt
8. **from** tqdm **import** tqdm
10. # 伪装成浏览器访问
11. headers = {
12. 'Connection': 'close',
13. 'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.105 Safari/537.36'
14. }
16. # 获取请求对象
17. session = HTMLSession()
18. # 存储已经访问过的 html
19. has\_get = []
20. # 存储未访问成功的 html
21. not\_get = []
22. # 获取当前路径
23. cwd = os.getcwd()
24. # 步入实验一文件夹中
25. os.chdir(cwd+'/国家知识产权局专利统计数据')
26. # 获取当前路径
27. cwd\_ = os.getcwd()
29. # 采取遍历访问
30. **for** year **in** tqdm(range(2008, 2019)):
31. os.mkdir(cwd\_+'/{}'.format(year))     # 创建文件夹
32. os.chdir(cwd\_+'/{}'.format(year))     # 进入文件夹
33. url = "https://www.cnipa.gov.cn/tjxx/jianbao/year{}/indexy.html".format(year)     # 主页网址
34. respose = session.get(url, headers = headers, timeout=10, verify=False)     # 获取主页内容
35. regex = re.compile(r'[a-z].html\*')
36. html\_ = regex.findall(" ".join(respose.html.links))     # 解析主页上的所有链接（相对路径）
38. # 遍历访问主页上的所有链接（绝对路径+相对路径）
39. **for** suffix **in** sorted(html\_):
40. url = "https://www.cnipa.gov.cn/tjxx/jianbao/year{0}/{1}".format(year, suffix)     # 次主页网址
41. respose = session.get(url, headers = headers, timeout=10, verify=False)     # 获取次主页内容
42. regex = re.compile(r'[a-z]/[a-z]\d+.html')
43. html\_1 = regex.findall(" ".join(respose.html.links))     # 解析次主页上的所有链接（相对路径）
45. # 遍历访问次主页上的所有链接（绝对路径+相对路径）
46. **for** suffix\_ **in** sorted(html\_1):
47. url = "https://www.cnipa.gov.cn/tjxx/jianbao/year{0}/{1}".format(year, suffix\_)    # 次次主页网址
48. **if** url **not** **in** has\_get:    # 检查url是否被重复访问
49. has\_get.append(url)
50. **print**("访问"+url)
51. isOk = True
52. n = 0      # 记录没访问成功的次数
53. # 如果访问失败则再次尝试，直到访问成功或失败次数超过3次
54. **while** isOk:
55. **try**:
56. **if** n == 3:
57. not\_get.append(url)
58. **break**
59. data = pd.read\_html(url)    # 获取网页的table
60. isOk = False
61. **except**:
62. **print**("连接失败"+url)
63. n += 1
64. # 判断是否有标题
65. **if** ''.join(re.findall('[\u4e00-\u9fa5]', data[1].values[0,0])) != '':
66. # 将页面上的五个表格分别写入一个excel中
67. excelName = data[0].values[0,0][:4]+"  "+''.join(re.findall('[\u4e00-\u9fa5]', data[1].values[0,0]))+'.xlsx'
68. excelTabel= xlwt.Workbook()
69. excelTabel.add\_sheet("sheet")
70. excelTabel.save(excelName)
71. write = pd.ExcelWriter(excelName)
72. data[0].to\_excel(write, index=False, header=False)
73. data[1].to\_excel(write, startrow=data[0].shape[0]+2, index=False, header=False)
74. data[2].to\_excel(write, startrow=data[0].shape[0]+data[1].shape[0]+2, index=False, header=False)
75. data[3].to\_excel(write, startrow=data[0].shape[0]+data[1].shape[0]+data[2].shape[0]+2, index=False, header=False)
76. data[4].to\_excel(write, startrow=data[0].shape[0]+data[1].shape[0]+data[2].shape[0]+data[3].shape[0]+2, index=False, header=False)
77. write.save()
78. **else**:
79. **continue**
81. os.chdir(cwd\_)    # 返回实验一目录
83. os.chdir(cwd)    # 返回主目录

**二、各省知识产权官方微博数据的抓取（以北京知识产权官方微博为例）**

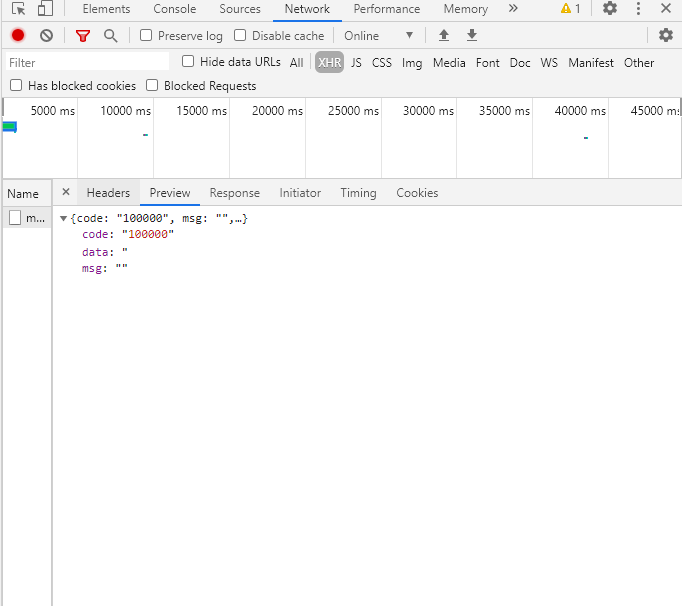
①使用selenium模拟登录获取cookie

1. userName = "17774858223"
2. password = "Qjy450211"
4. browser = webdriver.Chrome()
5. browser.get('https://weibo.com/')
6. browser.set\_window\_size(1920, 1080)
7. time.sleep(10)
8. user = browser.find\_element\_by\_id("loginname")
9. user.send\_keys(userName)
10. pwd = browser.find\_element\_by\_name("password")
11. pwd.send\_keys(password)
12. browser.find\_element\_by\_xpath('/html/body/div[1]/div[1]/div/div[2]/div[1]/div[2]/div/div[2]/div[1]/div[2]/div[1]/div/div/div/div[3]/div[6]/a').click()
14. cookies = browser.get\_cookies()
15. cookies\_l = []
16. cookie = {}
17. **for** d **in** cookies:
18. cookie[d['name']] = d['value']
19. cookies\_l.append(cookie)

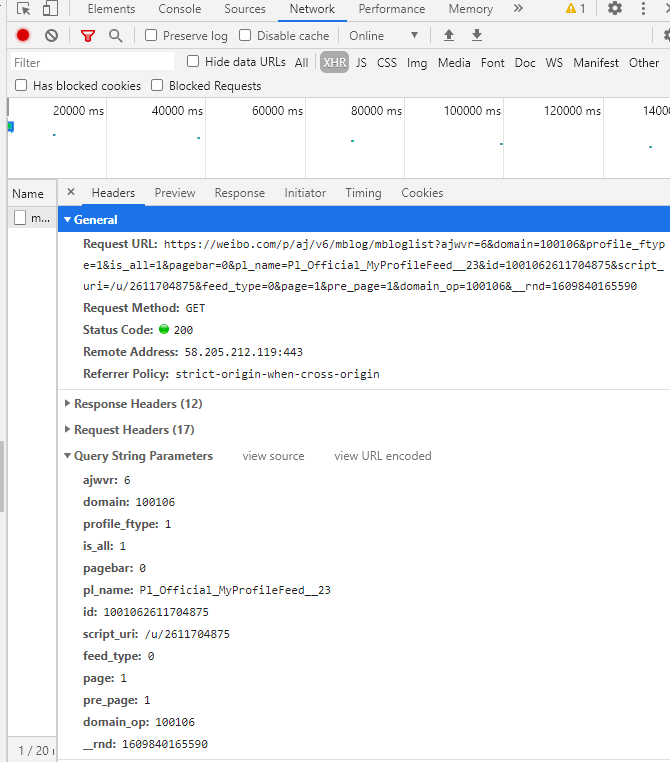
获取的cookie如下所示



②打开北京知识产权官方微博<https://weibo.com/u/2611704875?profile_ftype=1&is_all=1>打开开发者工具（F12），点击Network选项并且选择XHR。下拉网页直到刷到新的微博内容，观察XHR选项卡的”Preview”中的内容。



每条微博内容都存在了”data”中，再观察”Headers”



复制Headers的URL并传入（pagebar、page、pre\_page、\_\_rnd）这些变动的参数就可以获取”data”，通过分析拆解进而得到该微博的相关数据

1. headers = {
2. 'Connection': 'keep-alive',
3. 'X-Requested-With': 'XMLHttpRequest',
4. 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36',
5. 'Content-Type': 'application/x-www-form-urlencoded',
6. 'Accept': '\*/\*',
7. 'Sec-Fetch-Site': 'same-origin',
8. 'Sec-Fetch-Mode': 'cors',
9. 'Referer': 'https://weibo.com/u/2611704875?from=myfollow\_all&is\_all=1',
10. 'Accept-Encoding': 'gzip, deflate, br',
11. 'Accept-Language': 'zh-CN,zh;q=0.9'
12. }
14. all\_data = []
16. **for** i **in** tqdm(range(1, 101)):
17. **for** j **in** ('0', '1'):
18. url = 'https://weibo.com/p/aj/v6/mblog/mbloglist?ajwvr=6&domain=100106&from=myfollow\_all&is\_all=1&pagebar={}&pl\_name=Pl\_Official\_MyProfileFeed\_\_24&id=1001062611704875&script\_uri=/u/2611704875&feed\_type=0&page={}&pre\_page={}&domain\_op=100106&\_\_rnd={}'.format(j, i, 1, int(round(time.time() \* 1000)))
19. **try**:
20. sleep(random.uniform(0, 0.5))
21. response = requests.get(url, cookies=cookie, headers=headers)
22. **if** response.status\_code == 200:
23. json = response.json()
24. **else**:
25. **print**('访问失败'+response.status\_code)
26. **except** Exception as e:
27. **print**(sys.exc\_info())
28. **continue**
29. **if** json **is** **not** None:
30. soup = BeautifulSoup(json['data'], "html.parser")
31. **for** k **in** soup.find\_all('div',class\_="WB\_cardwrap WB\_feed\_type S\_bg2 WB\_feed\_like"):
32. data = {}
33. data['内容'] = re.sub('\s+','',k.find('div',class\_="WB\_text W\_f14").text.strip().replace('\u200b', ''))
34. data['时间'] = k.find('div',class\_="WB\_from S\_txt2").find('a').get('title')
35. **if** len(k.find('div',class\_="WB\_from S\_txt2").text.strip().split()) > 2:
36. data['来源'] = k.find('div',class\_="WB\_from S\_txt2").text.strip().split()[3]
37. **else**:
38. data['来源'] = ""
39. trans = k.find\_all('span',class\_="line S\_line1")[1]
40. **if** '转发' **in** str(trans):
41. data['转发'] = 0
42. **else**:
43. data['转发'] = trans.text.replace('\ue607','')
45. comment = k.find\_all('span',class\_="line S\_line1")[2]
46. **if** '评论' **in** str(comment):
47. data['评论'] = 0
48. **else**:
49. data['评论'] = comment.text.replace('\ue608','')
51. like = k.find\_all('span',class\_="line S\_line1")[3]
52. **if** '赞' **in** str(like):
53. data['点赞'] = 0
54. **else**:
55. data['点赞'] = like.text.replace('\nñ','').strip()
57. all\_data.append(data)

最后整理得到的表格（北京知识产权官方微博数据爬取.xlsx）如下图所示



**三、从佰腾网完成各省不同类型专利数据的收集**

①打开查询链接<https://www.baiten.cn/gjs.html>

找到如下两个条件并分别输入2018和广东

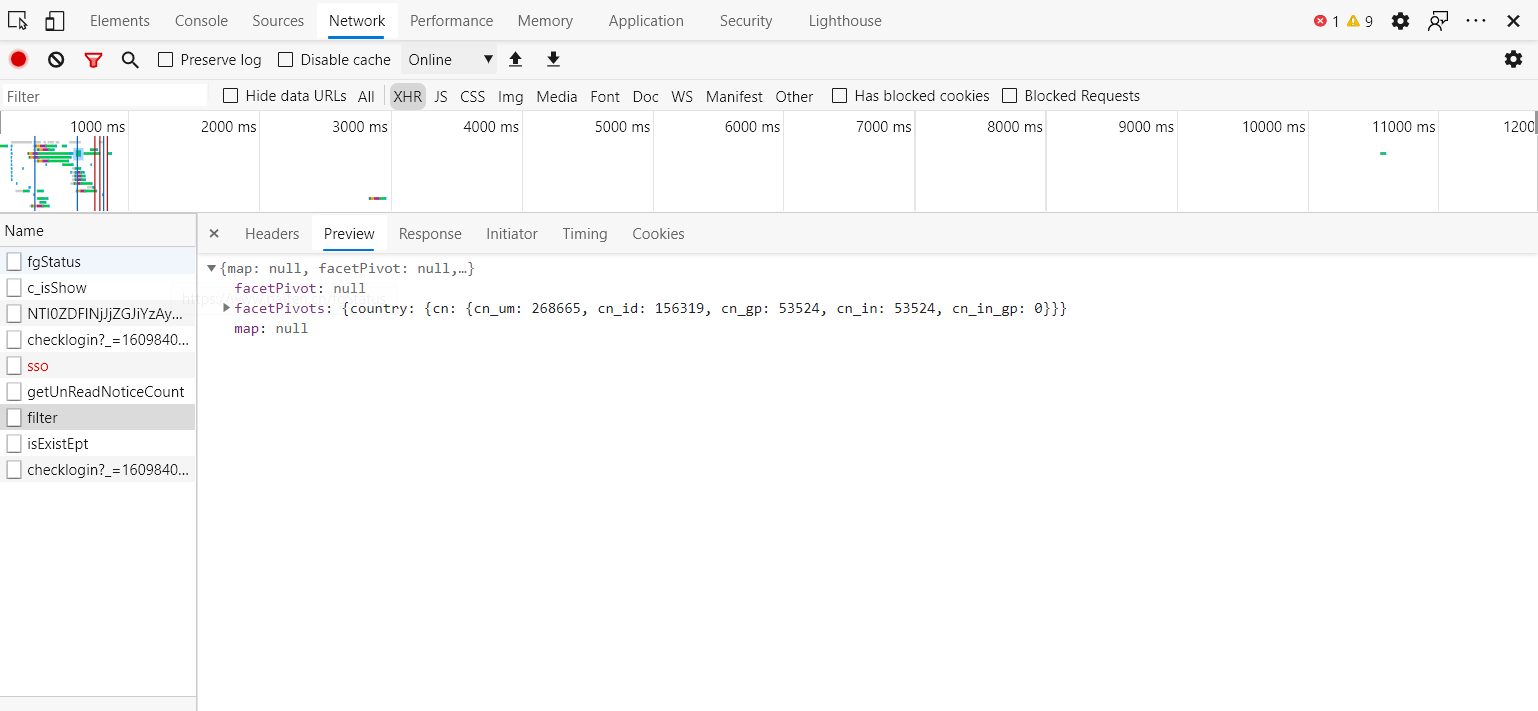




我们要爬取的就是侧边栏展开的这一部分内容的数据



②打开开发者工具，刷新网页，找到Network中XHR的名为filter的请求的Preview，发现我们要的数据就在filter中



那么我们只要模仿filter来访问并获得filter的Json即可。

首先获取filter的Headers的URL作为基础的URL

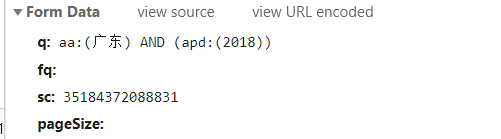
这里我们发现需要在他的后面加一个问号来访问获取，不然会导致失败

base\_url = 'https://www.baiten.cn/results/filter?'

根据filter的Headers中的General、Request Headers来构建我们的Headers

1. headers = {
2. 'Connection': 'keep-alive',
3. 'Accept': '\*/\*',
4. 'X-Requested-With': 'XMLHttpRequest',
5. 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36 Edg/87.0.664.66',
6. 'Origin': 'http://www.hshfy.sh.cn',
7. 'Referer': 'http://www.hshfy.sh.cn/shfy/gweb2017/ktgg\_search.jsp?zd=splc',
8. 'Accept-Language': 'zh-CN,zh;q=0.9',
9. 'Content-Type': 'application/x-www-form-urlencoded',
10. 'Cookie': 'yunsuo\_session\_verify=fa3ec82197b75c4fc9b421f6d9cc518b; JSESSIONID=0356E5924DFFCE14AAF0A3201A6A7180; Hm\_lvt\_7fc44f078bf7b5e19489428c362109a3=1609465896; UM\_distinctid=176bba49ab760-02ff8442486b7e-5a301e44-1fa400-176bba49ab8368; CNZZDATA1275904268=1445559182-1609462773-%7C1609468176; zlcp202101=true; BSESSION=3ba135cd396a5b96820206450bea0a9b3d82a9404a7a4964; PD=804b06c60a7659e10098c2f494dc28716c2e9f8cd5cbbf2c2ced7c0908b7fb63c1810531646f9438; zg\_did=%7B%22did%22%3A%20%22176bba49acc236-0485c245161fe8-5a301e44-1fa400-176bba49acd7e%22%7D; zg\_a965386b4f3b4914b0c2043d3472cde5=%7B%22sid%22%3A%201609468182289%2C%22updated%22%3A%201609469584052%2C%22info%22%3A%201609465895635%2C%22superProperty%22%3A%20%22%7B%5C%22%E5%BA%94%E7%94%A8%E5%90%8D%E7%A7%B0%5C%22%3A%20%5C%22%E6%95%B0%E6%8D%AE%E6%8C%87%E9%92%88%5C%22%7D%22%2C%22platform%22%3A%20%22%7B%7D%22%2C%22utm%22%3A%20%22%7B%7D%22%2C%22referrerDomain%22%3A%20%22%22%2C%22landHref%22%3A%20%22https%3A%2F%2Fwww.baiten.cn%2Fgjs.html%22%2C%22zs%22%3A%200%2C%22sc%22%3A%200%2C%22firstScreen%22%3A%201609468182289%2C%22cuid%22%3A%20%2217774858223%22%7D; Hm\_lpvt\_7fc44f078bf7b5e19489428c362109a3=1609469584'
11. }

观察filter的Form Data，我们发现只要改变参数q就可以实现不同年份和不同省份的查询，之后再获取filter得到我们需要的数据



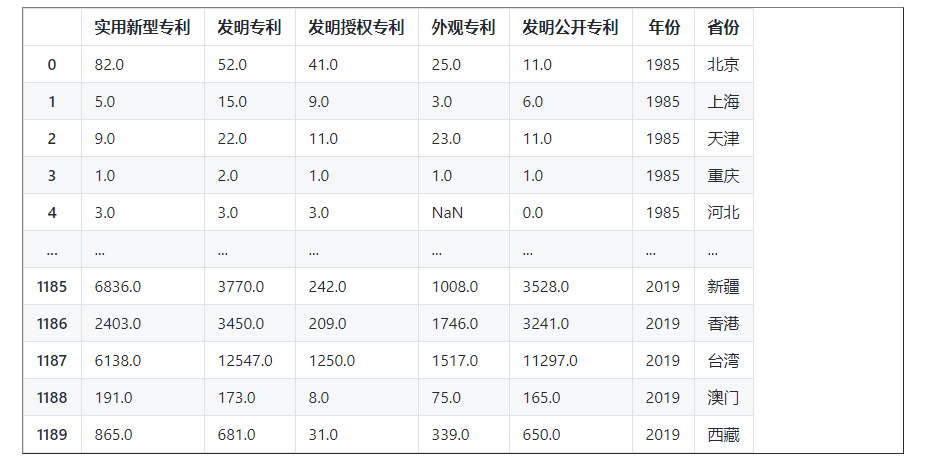
综上，获取filter Json的代码为

1. **def** get\_page(province, year):
2. n = 3
3. q = 'aa:({}) AND (pd:({}))'.format(province, year)
4. **while** True:
5. **try**:
6. sleep(random.uniform(1, 2))
7. data = {
8. 'q': q,
9. 'fq': '',
10. 'sc': '35184372088831',
11. 'pageSize': ''
12. }
13. url = base\_url + urlencode(data)
14. **try**:
15. response = requests.request("POST", url, headers = headers)
16. **if** response.status\_code == 200:
17. re = response.content.decode('gbk')
18. **return** re  # 解析内容
19. **except** requests.ConnectionError as e:
20. **print**('Error', e.args)
21. **except** (TimeoutError, Exception):
22. n -= 1
23. **if** n == 0:
24. **print**('请求3次均失败，放弃此url请求,检查请求条件')
25. **return**
26. **else**:
27. **print**('请求失败，重新请求')
28. **continue**
29. }

之后使用re库进行解析即可得到数据

1. with tqdm(total=len(year)) as pbar:
2. pbar.set\_description('Processing:')
3. data = []
4. **for** y **in** year:
5. **for** p **in** province:
6. html = get\_page(p, y)
7. columns = [x[:x.find(':')].replace('"''"','') for x in re.findall(r'"\w+".\d+',html)]
8. value = [int(x[x.find(':')+1:].replace('"','')) **for** x **in** re.findall(r'"\w+".\d+',html)]
9. dict\_data = dict(dict(zip(columns,value)),\*\*dict(zip(['年份','省份'],[y,p])))
10. data.append(dict\_data)
11. pbar.update(1)

最终得到的1985-2019年所有专利种类数量数据.xlsx为



**四、省机关报数据的抓取（以北京日报为例）**

①打开北京日报，进入第一个板块

<http://paper.people.com.cn/rmrb/html/2020-12/03/nbs.D110000renmrb_01.htm>



点开二个板块后，页面url变为

<http://paper.people.com.cn/rmrb/html/2020-12/03/nbs.D110000renmrb_02.htm>

有规律可循。

再点进一篇文章，观察url为

<http://paper.people.com.cn/rmrb/html/2020-12/03/nw.D110000renmrb_20201203_1-02.htm>

也有规律可循

但我们需要获取每个板块的名称来建立文件夹。

观察到有蛮多的版面，并且有重复，排序不规范，所以不能直接遍历。也不能直接用第一个问题中获得所有url来访问，因为这个页面中存在很多我们不需要的url链接。

通过爬虫返回一个网址的html后发现，这些板块的名称和页面上的文章名可以用re库进行解析，可以得到每个板块的名称，从而可以获得板块的数量或文章的数量并通过构造遍历来访问。

代码如下

1. headers = {
2. 'Connection': 'close',
3. 'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.105 Safari/537.36'
4. }
6. startDate = '2020-12-01'
7. endDate = '2020-12-31'
8. day\_list = [strftime('%Y-%m/%d', strptime(ts,'%Y-%m-%d')) **for** ts **in** pd.date\_range(start=startDate,end=endDate, freq='D').format()]
10. # 获取当前路径
11. cwd = os.getcwd()
12. # 获取请求对象
13. session = HTMLSession()
14. with tqdm(total=len(day\_list)) as pbar:
15. pbar.set\_description('Processing:')
16. **for** day **in** day\_list:
17. **print**("正在爬取"+day)
18. url = "http://paper.people.com.cn/rmrb/html/{}/nbs.D110000renmrb\_01.htm".format(day)
19. respose = session.get(url, headers = headers, timeout=5)
20. plateName = [name[name.find('：')+1:] **for** name **in** re.findall('\n\d+\w：\w+' ,respose.html.text)]
21. **for** plateNum **in** [str(i).rjust(2,'0') **for** i **in** np.arange(1, len(plateName)+1)]:
23. name = plateName[int(plateNum)-1]
24. **if** os.path.exists(name):
25. os.chdir(cwd+'\\{}'.format(name))     # 进入文件夹
26. **else**:
27. os.mkdir(cwd+'\\{}'.format(name))     # 创建文件夹
28. os.chdir(cwd+'\\{}'.format(name))     # 进入文件夹
30. url\_plate = "http://paper.people.com.cn/rmrb/html/{}/nbs.D110000renmrb\_{}.htm".format(day, plateNum)
31. #     print(url\_plate)
32. respose = session.get(url\_plate, headers = headers, timeout=5)
33. titleName = [title[2:] **for** title **in** re.findall(r'\n·\S+[ \S]\*', respose.html.text)]
35. **for** titleNum **in** [i **for** i **in** np.arange(1, len(titleName)+1)]:
37. cwd\_ = os.getcwd()
38. suffixName = day.replace('/','-')
39. **if** os.path.exists(suffixName) > 0:
40. os.chdir(cwd\_+'\\{}'.format(suffixName))     # 进入文件夹
41. **else**:
42. os.mkdir(cwd\_+'\\{}'.format(suffixName))     # 创建文件夹
43. os.chdir(cwd\_+'\\{}'.format(suffixName))     # 进入文件夹
45. url\_passage = "http://paper.people.com.cn/rmrb/html/{}/nw.D110000renmrb\_{}\_{}-{}.htm".format(day, day.replace('-','').replace('/',''), titleNum, plateNum)
46. #         print(url\_passage)
47. sleep(random.uniform(0, 0.5))
48. n = 3
49. isOk = True
50. **while** isOk:
51. **try**:
52. respose = session.get(url\_passage, headers = headers, verify=False)
53. isOk = False
54. **except**:
55. **print**("访问失败")
56. n -= 1
57. #         print(respose.text)
58. respose.encoding="utf-8"
59. content = []
60. title\_list = []
61. passageTitle = etree.HTML(respose.text).xpath('/html/body/div[2]/div[2]/div[3]/div[2]/h1/text()')      #获取文章标题
62. **for** i **in** range(len(passageTitle)):
63. title\_list.append(passageTitle[i])
64. text = etree.HTML(respose.text).xpath('//\*[@id="ozoom"]/p/text()')                # 获取文章内容
65. title = ''.join(title\_list)
66. **for** i **in** range(len(text)):
67. content.append(text[i])
68. content = re.sub('\xa0', '', ''.join(re.findall(r"[\u4E00-\u9FA5A-Za-z0-9\_]+[,]{0,}[^,;=?]{0,}", ','.join(content))))
69. **try**:
70. with open(title+'.txt','w') as f:
71. f.write(content)
72. **except**:
73. **print**("Error: 没有找到{}.txt文件或读取文件失败".format(title))
75. os.chdir(cwd\_)
76. os.chdir(cwd)
77. pbar.update(1)

# 数据清洗

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | 楼 | 房； | **实验台号：** |  |
| **实验日期与时间：** |  | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** |  |

【实验目的及要求】

掌握Python数据预处理技术。使用Python 的re、pandas库对抓取和下载的数据进行清洗和规整等操作

【实验内容】

第一部分，基础部分

1.知识产权年鉴数据清洗，包括各地区知识产权执法、确权等数据

2.机关报数据清洗，包括各地区机关报数据中相关性指标

3.知识产权微博数据清洗，包括各地区的关注度指标

**一、知识产权年鉴数据清洗（即实验一的国家知识产权局数据）**

①读取知识产权年鉴中各地区每年（这里以2008年为例）专利执法数（统计各个地区的立案、结案数） 数据为：h1.xls（各地区管理专利工作的部门专利执法统计表）

②筛选出年份这一列为2018的数据，将地区这一列数据改为对应的中文，并将地区设置为index, 计算结案与立案的比值，变量名为结案/立案（或结案率）并保存入h1清洗.xls中



相关代码如下：

1. **import** pandas as pd
2. **import** numpy as np
4. h1data = pd.read\_excel('./h1.xls', skiprows=[0,1], index\_col=0).iloc[2:,:4]
5. h1data.columns = ['地区', '年份', '立案', '结案']
6. h1data.index.name = ''
7. h1data
9. trans = [{h1data.iloc[i+1,0]: h1data.iloc[i,0]} **for** i **in** range(0,h1data.shape[0],2)]
10. trans
12. h1data\_ = h1data[h1data['年份']=='2008'].copy()
13. h1data\_['地区'] = h1data\_['地区'].map(**lambda** x: [d[x] **for** d **in** trans **if** x **in** d][0])
14. h1data\_.set\_index('地区', drop=True, inplace=True)
15. h1data\_['结案/立案'] = h1data\_['结案']/h1data\_['立案']
16. h1data\_
18. h1data\_.to\_excel('./h1清洗.xls')

循环清洗的代码：

1. **def** clear\_h1(name):
2. h1data = pd.read\_excel('./h1.xls', skiprows=[0,1], index\_col=0).iloc[2:,:4]
3. h1data.columns = ['地区', '年份', '立案', '结案']
4. h1data.index.name = ''
5. trans = [{h1data.iloc[i+1,0]: h1data.iloc[i,0]} **for** i **in** range(0,h1data.shape[0],2)]
6. h1data\_ = h1data[h1data['年份']==name].copy()
7. h1data\_['地区'] = h1data\_['地区'].map(**lambda** x: [d[x] **for** d **in** trans **if** x **in** d][0])
8. h1data\_.set\_index('地区', drop=True, inplace=True)
9. h1data\_['结案/立案'] = h1data\_['结案']/h1data\_['立案']
10. h1data\_.to\_excel('./h1清洗.xls')
11. **print**(name+"清理成功")
13. **import** os
14. **import** pandas as pd
15. **import** numpy as np
16. cwd = os.getcwd()
17. os.chdir(cwd+'\\2008-2018知识产权年鉴数据')
18. cwd\_ = os.getcwd()
19. **for** name **in** os.listdir(cwd\_):
20. os.chdir(cwd\_+'\\{}'.format(name)+'\\h')
21. clear\_h1(name)
22. os.chdir(cwd\_

③统计各地区每年查处执法假冒量读入数据，数据为h5.xls（各地区查处假冒专利执法统计表（一）），重命名列名，将地区设置为索引值，将数据保存到h5清洗.xls中

相关代码如下：

1. **import** pandas as pd
3. h5data = pd.read\_excel('./h5.xls',index\_col=0, header=1).iloc[1:, [0,2]]
4. h5data.columns = ['地区','假冒数量']
5. h5data.set\_index('地区', inplace=True)
6. h5data
8. h5data.to\_excel('./h5清洗.xls')

循环清洗的代码如下：

1. **def** clear\_h5(name):
2. h5data = pd.read\_excel('./h5.xls',index\_col=0, header=1).iloc[1:, [0,2]]
3. h5data.columns = ['地区','假冒数量']
4. h5data.set\_index('地区', inplace=True)
5. h5data.to\_excel('./h5清洗.xls')
6. **print**(name+"清理成功")
8. **import** os
9. **import** pandas as pd
10. **import** numpy as np
11. cwd = os.getcwd()
12. os.chdir(cwd+'\\2008-2018知识产权年鉴数据')
13. cwd\_ = os.getcwd()
14. **for** name **in** os.listdir(cwd\_):
15. os.chdir(cwd\_+'\\{}'.format(name)+'\\h')
16. clear\_h5(name)
17. os.chdir(cwd\_)

④地区专利情况： 各地区3种专利的申请量、有效量，发明专利授权量/申请量，发明专利/(发明+实用新型)。读入数据，用a4（国内三种专利申请量），b3（国内三种专利申请授权量），c2（国内三种专利有效量）并选择需要的行和列，分别存入df1、df2、df3，并将地区设置为索引，合并 三者为df



⑤根据这些数据计算相关的指标，指标计算如下

1. df['发明专利授权率'] = df['发明\_授权量'] / df['发明\_申请量']
2. df['实用新型专利授权率'] = df['实用新型\_授权量'] / df['实用新型\_申请量']
3. df['外观设计专利授权率'] = df['外观设计\_授权量'] / df['外观设计\_申请量']
4. df['专利授权率'] = (df['发明\_授权量'] + df['实用新型\_授权量'] + df['外观设计\_授权量']) / \
5. (df['发明\_申请量'] + df['实用新型\_申请量'] + df['外观设计\_申请量'])
7. df['发明专利申请比例'] = df['发明\_申请量'] / (df['发明\_申请量'] + df['实用新型\_申请量'] + df['外观设计\_申请量'])
8. df['实用新型专利申请比例'] = df['实用新型\_申请量'] / (df['发明\_申请量'] + df['实用新型\_申请量'] + df['外观设计\_申请量'])
9. df['外观设计专利申请比例'] = df['外观设计\_申请量'] / (df['发明\_申请量'] + df['实用新型\_申请量'] + df['外观设计\_申请量'])
11. df['发明专利授权比例'] = df['发明\_授权量'] / (df['发明\_授权量'] + df['实用新型\_授权量'] + df['外观设计\_授权量'])
12. df['实用新型专利授权比例'] = df['实用新型\_授权量'] / (df['发明\_授权量'] + df['实用新型\_授权量'] + df['外观设计\_授权量'])
13. df['外观设计专利授权比例'] = df['外观设计\_授权量'] / (df['发明\_授权量'] + df['实用新型\_授权量'] + df['外观设计\_授权量'])
15. df['发明专利有效比例'] = df['发明\_有效量'] / (df['发明\_有效量'] + df['实用新型\_有效量'] + df['外观设计\_有效量'])
16. df['实用新型专利有效比例'] = df['实用新型\_有效量'] / (df['发明\_有效量'] + df['实用新型\_有效量'] + df['外观设计\_有效量'])
17. df['外观设计专利有效比例'] = df['外观设计\_有效量'] / (df['发明\_有效量'] + df['实用新型\_有效量'] + df['外观设计\_有效量'])

⑥将计算好的数据保存入a4b3c2合并.xls中

循环清洗的代码如下：

1. **import** pandas as pd
2. **import** os
3. cwd = os.getcwd()
4. os.chdir(cwd+'\\2008-2018知识产权年鉴数据')
5. cwd\_ = os.getcwd()
6. **for** name **in** os.listdir(cwd\_):
7. os.chdir(cwd\_+'\\{}'.format(name))
8. merge\_a4\_b3\_c2(name)
9. os.chdir(cwd\_)

⑦将上面三个导出的excel合并成一个excel，以(年份,地区)作为二重行索引，把各年份数据整合，获得知识产权年鉴清洗后的数据命名为“所有合并.xls”

相关代码：

1. **import** pandas as pd
2. df1 = pd.read\_excel('./h1清洗.xls', index\_col=0)
3. df2 = pd.read\_excel('./h5清洗.xls', index\_col=0)
4. df3 = pd.read\_excel('./a4b3c2合并.xls', index\_col=0)
5. df = pd.merge(left=df1,right=df2,left\_index=True,right\_index=True)
6. df = pd.merge(left=df,right=df3,left\_index=True,right\_index=True)
7. df
8. df.to\_excel('./所有合并.xls')

循环代码：

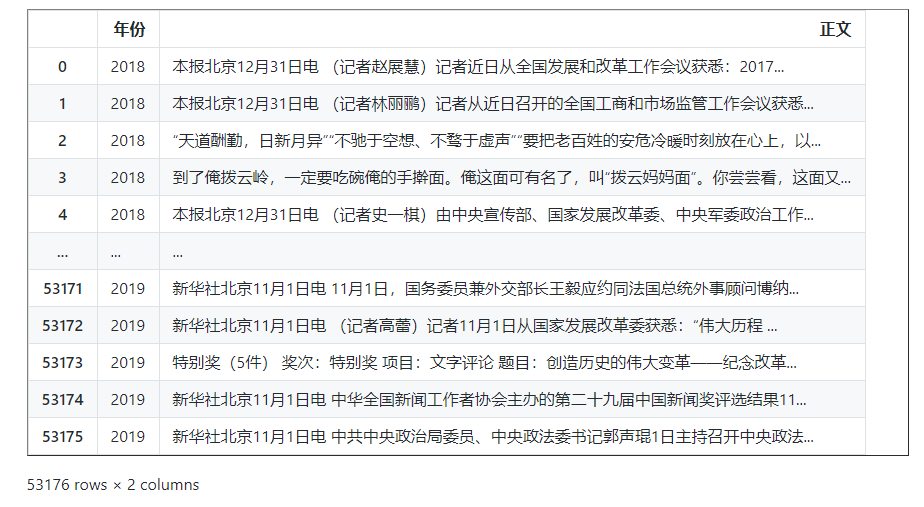
1. **def** merge\_all(name):
2. df1 = pd.read\_excel('./h/h1清洗.xls', index\_col=0)
3. df2 = pd.read\_excel('./h/h5清洗.xls', index\_col=0)
4. df3 = pd.read\_excel('./a4b3c2合并.xls', index\_col=0)
5. df = pd.merge(left=df1,right=df2,left\_index=True,right\_index=True)
6. df = pd.merge(left=df,right=df3,left\_index=True,right\_index=True)
7. df.to\_excel('./所有合并.xls')
8. **print**(name+'合并完成')
9. **import** os
10. **import** pandas as pd
11. cwd = os.getcwd()
12. os.chdir(cwd+'\\2008-2018知识产权年鉴数据')
13. cwd\_ = os.getcwd()
14. **for** name **in** os.listdir(cwd\_):
15. os.chdir(cwd\_+'\\{}'.format(name))
16. merge\_all(name)
17. os.chdir(cwd\_)

⑧把2008-2018年的每年的“所有和并.xls”合并为“总的数据合并.xls”

1. **import** os
2. **import** pandas as pd
3. cwd = os.getcwd()
4. os.chdir(cwd+'\\2008-2018知识产权年鉴数据')
5. cwd\_ = os.getcwd()
6. df\_list = []
7. **for** name **in** os.listdir(cwd\_):
8. df\_list.append(pd.read\_excel('./{}/所有合并.xls'.format(name), index\_col=0))
10. df = pd.DataFrame()
11. **for** d **in** df\_list:
12. df = df.append(d)
14. df.reset\_index(inplace=False).set\_index(['年份','地区'],inplace=False).to\_excel('./总的数据合并.xls')

**二、机关报数据清洗**

①读取数据：人民日报.xlsx，新增年份列，为标题前四个字符。然后除去标题列



1. **import** pandas as pd
2. **import** re
3. data = pd.read\_excel('./人民日报.xlsx', index\_col=0)
4. data['年份'] = data['标题'].str[:4]
5. **del** data['标题']
6. data = data[['年份','正文']]
7. data

②计算每年的知识产权文章数量以及每年知识产权文章字数和总文章字数。

使用正则表达式去掉文章中多余的空白以及，。？；：‘’、！（）《》等符号，

1. data['正文去除符号'] = data['正文'].map(**lambda** x: ''.join(re.findall(r'[\u4E00-\u9FA5A-Za-z0-9]+', str(x))))

遍历所有文章的内容，若文章内容中存在知识产权， 专利， 商标， 版权， 著作权五个关键词中的一个或者多个，即记为知识产权类文章。计算每年知识产权类文章的数量，字数， 以及该年份所有文章的数量，字数



1. keywords = ['知识产权','专利','商标','版权','著作权']
3. **def** find\_key(x):
4. **for** word **in** keywords:
5. **if** word **in** x:
6. **return** True
7. **return** False
9. data['是否为知识产权文章'] = data['正文去除符号'].map(**lambda** x:find\_key(x))
10. data['字数'] = data['正文去除符号'].map(len)

通过第2步中的数据计算出知识产权平均字数（每年知识产权类文章总字数除以知识产权类文章的数量），每年的知识产权类文章数量与全部新闻数量的比例， 每年的知识产权类文章的字数与全部新闻总字数的比例，然后将其放入DataFrame，并存入“知识产权数据清洗.xls”



1. result = {}
2. **for** year, value **in** data.groupby('年份'):
3. isTrue = value[value['是否为知识产权文章']]
4. isNum = sum(value['是否为知识产权文章'])
5. isWords = isTrue['字数'].sum()
6. allNum = value.shape[0]
7. allWords = value['字数'].sum()
8. result[year] = [allNum, isNum, allWords, isWords]
10. df = pd.DataFrame(result, index=['总发文量','知识产权文章数量','全部文章总字数','知识产权类文章总字数']).T
12. df['知识产权文章数量占比'] = df['知识产权文章数量'] / df['总发文量']
13. df['知识产权文章总字数占比'] = df['知识产权类文章总字数'] / df['全部文章总字数']
14. df['知识产权平均字数'] = df['知识产权类文章总字数'] / df['知识产权文章数量']
16. df.to\_excel('./知识产权数据清洗.xls')

**三、微博数据清洗**

以广东知识产权微博.xlsx为例，筛选字段['微博内容', '时间', '点赞数', '评论数', '转发数']

①首先对微博内容这一列进行处理，去掉其中一些不需要的字符串（如：'...'，'\u200b'，'O网页链接'，'\u3000'），对时间进行处理，只保留年份，计算这个微博每一年的发文量，在data1中新增发文量这一列，用对应年份的发文量进行填充。



1. **import** pandas as pd
2. data = pd.read\_excel('./广东官微.xlsx')
3. data1 = data[['微博内容','时间','点赞数','评论数','转发数']].copy()
4. data1['微博内容'] = data1['微博内容'].map(**lambda** x: str(x))
5. data1['微博内容'] = data1['微博内容'].str.replace(u'\u200b','').str.replace('网页链接 ','').str.replace(u'\u3000','').str.replace('.','')
6. data1['时间'] = data1['时间'].str[:4]
7. data1['发文量'] = data1.groupby('时间')['微博内容'].transform('count')

②分析该省微博对知识产权的重视程度，用知识产权平均字数（一年的知识产权类字数/这一年全部文章字数，知识产权类字数是指包含知识产权、专利、商标、版权和著作权5个关键字其中一个或多个的微博的总字数）衡量，在data1中新增知识产权平均字数这一列，用对应年份的知识产权平均字数进行填充。并保存为“微博数据清洗.xls”



1. keywords = ['知识产权','专利','商标','版权','著作权']
3. **def** find\_key(x):
4. **for** word **in** keywords:
5. **if** word **in** x:
6. **return** True
7. **return** False
9. data1['是否为知识产权文章'] = data1['微博内容'].map(**lambda** x:find\_key(x))
10. data1['字数'] = data1['微博内容'].map(len)
11. avgWords = data1.groupby('时间').apply(**lambda** x: sum(x[x['是否为知识产权文章']]['字数']) / sum(x['字数']))
12. avgWords.name = '知识产权平均字数'
13. data1 = pd.merge(left=avgWords, right=data1, left\_index=True, right\_on='时间')
14. data1 = data1[['微博内容', '时间', '点赞数', '评论数', '转发数', '发文量', '知识产权平均字数']]
15. data1.to\_excel('./微博数据清洗.xls')

# 地区实证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | 楼 | 房； | **实验台号：** |  |
| **实验日期与时间：** |  | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** |  |

【实验内容】

1） 清洗并整合中国统计年鉴数据，与实验二的数据进行整合

2） 构建省级知识产权保护指数

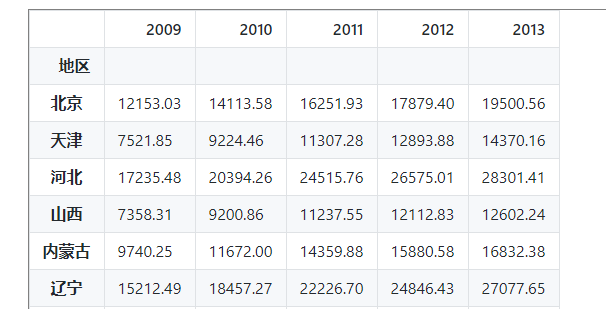
3）知识产权保护对地区创新产出的影响实证

一、清洗中国统计年鉴数据并与实验二的数据进行整合

1、中国统计年鉴数据清洗与整合（GDP数据）

①读取GDP中“2009-2013地区生产总值.xls”并清洗，得到2009-2013年的gdp数据df1。

1. df1 = pd.read\_excel('./GDP/2009-2013地区生产总值.xls', skiprows=[0,1,2,3,4], index\_col=0, header=0)
2. **del** df1['Unnamed: 1']
3. df1.drop(['2009.1','2010.1','2011.1','2012.1','2013.1'], axis=1, inplace=True)
4. df1.index.name = '地区'
5. df1



②分别读取2014-2017年的地区生产总值excel文件，并清洗，得到2014到2017年的数据df2。

1. df2 = []
2. **for** year **in** range(2014,2018):
3. df = pd.read\_excel('./GDP/{}各地区生产总值.xls'.format(year), skiprows=[0,1,2,3,4], index\_col=0, header=0).iloc[:,1:2]
4. df.columns = ['GDP']
5. df['年份'] = year
6. df.index.name = '地区'
7. df2.append(df)
8. df2 = pd.DataFrame().append(df2)
9. df2 = df2.reset\_index(drop=False).set\_index(['地区','年份']).unstack()['GDP']
10. df2.columns.name = ''



③读取“2018各地区生产总值.xls”，提取2018年的数据df3。

1. df3 = pd.read\_excel('./GDP/2018各地区生产总值.xls', skiprows=[0,1,2,3,4], index\_col=0, header=1).iloc[2:-3,1:2]
2. df3.columns = [2018]
3. df3.index = [''.join(i.split()) **for** i **in** list(df3.index)]



④将上述三个df合并得到综合2009-2018年的gdp数据,并存入“gdp.xlsx”（单位：亿元）

1. df = pd.concat([df1,df2,df3], axis=1)
2. df.index.name = '地区'



2、实验二和实验三数据清洗及整合

①读取“知识产权年鉴数据.xlsx”并存到df1，从知识产权年鉴数据中，选取需要的列['Unnamed: 0','Unnamed: 1','结案/立案','专利\_申请量','专利\_授权量','职务专利授权量百分比']，将列名重命名为['年份','地区','结案率','专利\_申请量','专利\_授权量','职务专利授权量百分比']，将年份这一列的数据按照对应的年份（利用fillna里面的method参数）填充好，将‘结案率’中的无穷大‘np.inf’替换为0，将大于1的结案率换为1

1. df1 = pd.read\_excel('./知识产权年鉴数据.xlsx')
2. df1 = df1.loc[:,['Unnamed: 0','Unnamed: 1','结案/立案','专利\_申请量','专利\_授权量','职务专利授权量百分比']]
3. df1.columns = ['年份','地区','结案率','专利\_申请量','专利\_授权量','职务专利授权量百分比']
4. df1['年份'] = df1['年份'].fillna(method='ffill').map(**lambda** x: int(x))
5. df1['结案率'] = df1['结案率'].replace(np.inf, 0)
6. df1['结案率'] = df1['结案率'].map(**lambda** x: 1 **if** x > 1 **else** x)



②读取“各省人口数（2009-2018）.xlsx”，得到各省人口数据df2，（单位：万人） 并用melt转换格式成如下所示

1. df2 = pd.read\_excel('./各省人口数（2009-2018）.xlsx')
2. df2 = df2.set\_index('地区').unstack().reset\_index()
3. df2.columns = ['年份','地区','人口数']
4. df2 = df2[['地区','年份','人口数']]
5. df2['人口数'] = df2['人口数'] \* 100000

③将以上两个dataframe整合得到df。删掉2008年的数据，因为缺少人口数据

1. df = pd.merge(left=df1,right=df2,on=['地区','年份'],how='inner')



④将整理好的GDP数据（单位是亿元），各省进出口总额（单位：亿美元），全社会固定资产投入（单位：亿元），外商直接投资（FDI）（单位：万美元）做同样处理，整合到df。（用的美元兑人民币汇率为6.5286）

1. df3 = pd.read\_excel('./gdp.xlsx')
2. df3 = df3.set\_index('地区').unstack().reset\_index()
3. df3.columns = ['年份','地区','GDP']
4. df3['年份'] = df3['年份'].map(**lambda** x: int(x))
6. df = pd.merge(left=df,right=df3,on=['地区','年份'],how='inner')
8. df4 = pd.read\_excel('./各省进出口总额.xlsx').iloc[:,:-1]
9. df4.columns = ['地区'] + list(range(2009, 2019))
10. df4 = df4.set\_index('地区').unstack().reset\_index()
11. df4.columns = ['年份','地区','进出口总额']
13. df = pd.merge(left=df,right=df4,on=['地区','年份'],how='inner')
15. df5 = pd.read\_excel('./全社会固定资产投入.xlsx').iloc[:,:-2]
16. df5.columns = ['地区'] + list(range(2009, 2019))
17. df5 = df5.set\_index('地区').unstack().reset\_index()
18. df5.columns = ['年份','地区','全社会固定资产投入']
20. df = pd.merge(left=df,right=df5,on=['地区','年份'],how='inner')
22. df6 = pd.read\_excel('./外商直接投资（FDI）.xlsx')
23. df6.columns = ['地区'] + list(range(2009, 2019))
24. df6 = df6.set\_index('地区').unstack().reset\_index()
25. df6.columns = ['年份','地区','外商直接投资']
26. df6['外商直接投资'] = df6['外商直接投资']
28. df = pd.merge(left=df,right=df6,on=['地区','年份'],how='inner')
30. df['GDP'] = df['GDP'] \* 100000000
31. df['进出口总额'] = df['进出口总额'] \* 100000000 \* 6.5286
32. df['全社会固定资产投入'] = df['全社会固定资产投入'] \* 100000000
33. df['外商直接投资'] = df['外商直接投资'] \* 10000 \* 6.52286
35. df.to\_excel('./整合数据.xlsx')



二、自变量的构建

（1）知识产权保护强度指数IPRs

***IPRs =Fs\*IPRc***

其中IPRs代表省份S修正后的知识产权保护水平，Fs代表省份s的执法力度，IPRc代表国家层面的知识产权保护水平,为4.86。

执法力度Fs有三个衡量指标：1）知识产权行政保护水平，用专利侵权案件的结案率（结案/立案）衡量；2） 产权社会保护水平，用国内专利申请中有专业代理机构的比例（职务专利授权量百分比）来度量；3）民众知识产权保护意识，用人均专利申请量（省专利申请量除以省总人口数）表示。采用主成分分析，根据三个指标的解释百分比加权计算出各省知识产权保护指数。

1)计算人均专利申请量，用每年各省专利申请量除以省人口数

1. data['人均专利申请量'] = data['专利\_申请量'] / data['人口数']
2. data['人均专利授权量'] = data['专利\_授权量'] / data['人口数']
3. # 填充结案率中的缺失值，使用向上填充
4. data['结案率'] = data['结案率'].fillna(method='ffill')

2）选出['结案率','职务专利授权量百分比','人均专利申请量']三列数据，利用PCA计算解释百分比,得到如下解释百分比即权重

1. df = data[['结案率','职务专利授权量百分比','人均专利申请量']]
2. scaler = StandardScaler()
3. scaler.fit(df)
4. trans\_data = scaler.transform(df)
5. pca = PCA(n\_components=3, random\_state=2021)
6. pca.fit(trans\_data)
7. ratio = pca.explained\_variance\_ratio\_



3) 对['结案率','职务专利授权量百分比','人均专利申请量']进行加权求和然后乘以4.86，得到省级知识产权保护指数。

1. data['知识产权保护指数'] = (data[['结案率','职务专利授权量百分比','人均专利申请量']] \* ratio).sum(axis=1)\*4.86



4)计算人均GDP、经济开放度（进出口总额/GDP）、GDP的哑变量：将数据按GDP的中位数分为两组，大于中位数的赋值为1，否则为0

1. data['人均GDP'] = data['GDP'] / data['人口数']
2. data['经济开放度'] = data['进出口总额'] / data['GDP']
3. data['GDP\_哑变量'] = data['GDP'].map(**lambda** x: 1 **if** x>data['GDP'].median() **else** 0)



三、知识产权保护对地区创新产出的影响实证

①使用describe函数进行描述性统计

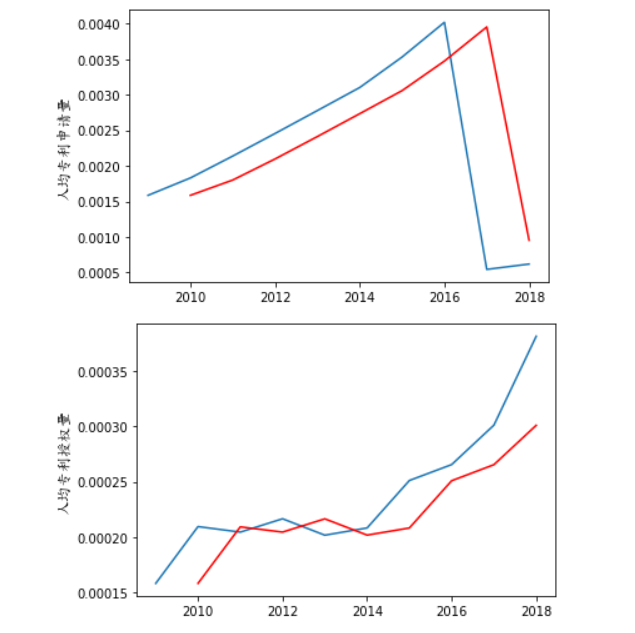


经过观察，所有统计变量都在正常的取值范围内，可以在一定程度上排除数据错误出现

②对每个省的人均专利申请量和人均专利授权量做时间序列预测

这里使用简单指数平滑函数SimpleExpSmoothing做时间序列预测，并以上海为例绘制图像

1. **from** statsmodels.tsa.api **import** SimpleExpSmoothing
2. **import** matplotlib.pyplot as plt
3. %matplotlib inline
4. **from** matplotlib.font\_manager **import** FontProperties
5. font\_set = FontProperties(fname="./华康楷体W5-A.TTF", size=12)
7. result = {}
8. **for** zone, value **in** df.groupby('地区'):
9. result[zone] = [value['人均专利申请量'].values,
10. value['人均专利授权量'].values,
11. SimpleExpSmoothing(value['人均专利申请量'].values).fit(),
12. SimpleExpSmoothing(value['人均专利授权量'].values).fit()]
14. **def** draw\_picture(data):
15. x = list(range(2009, 2019))
16. plt.plot(x, data[0])
17. plt.plot(x[1:], list(data[2].fittedvalues)[1:], color='red')
18. plt.ylabel('人均专利申请量',fontproperties=font\_set)
19. plt.show()
21. plt.plot(x, data[1])
22. plt.plot(x[1:], list(data[3].fittedvalues)[1:], color='red')
23. plt.ylabel('人均专利授权量',fontproperties=font\_set)
24. plt.show()
26. data = result['上海']
27. draw\_picture(data)

其中红色线为预测，蓝色线为实际情况，可以看出来，时间序列预测具有明显的滞后性。

③对变量进行正态性检验，检验变量为人均专利授权量、人均专利申请量、知识产权保护指数、人口数、全社会固定资产投入、gdp、经济开放度和外商直接投资

1. **from** scipy **import** stats
3. **for** x **in** ["人均专利授权量","人均专利申请量","知识产权保护指数","人口数","全社会固定资产投入","GDP","经济开放度","外商直接投资"]:
4. **print**(x)
5. data = df[x].values
6. **print**(stats.kstest(data,cdf='norm'))
7. **print**(stats.shapiro(data))
8. **print**('')

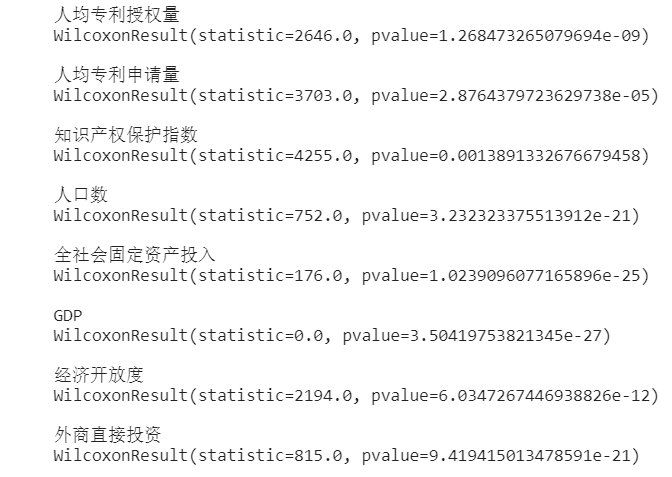


我们发现，所有变量的p值都小于0.001，也就是说，在0.1%的显著性水平下，可以拒绝零假设，即这些变量都不服从正态分布

④非参数检验：wilcoxons检验

根据gdp哑变量分组，检验两组数据在人均专利授权量、人均专利申请量、知识产权保护指数、人口数、全社会固定资产投入、GDP、经济开放度和外商直接投资有无显著差异。

1. **from** scipy **import** stats
3. **for** x **in** ["人均专利授权量","人均专利申请量","知识产权保护指数","人口数","全社会固定资产投入","GDP","经济开放度","外商直接投资"]:
4. **print**(x)
5. x1 = df[df['GDP\_哑变量']==0][x].values
6. x2 = df[df['GDP\_哑变量']==1][x].values
7. **print**(stats.wilcoxon(x1,x2,correction=True,alternative='two-sided'))
8. **print**('')



我们发现，所有变量的p值都小于0.005，也就是说，在0.5%的显著性水平下，可以拒绝零假设，即认为这些被检验的变量在GDP哑变量为0和1的两种情况下有显著差异。

⑤相关性分析：检验变量两两之间的相关性

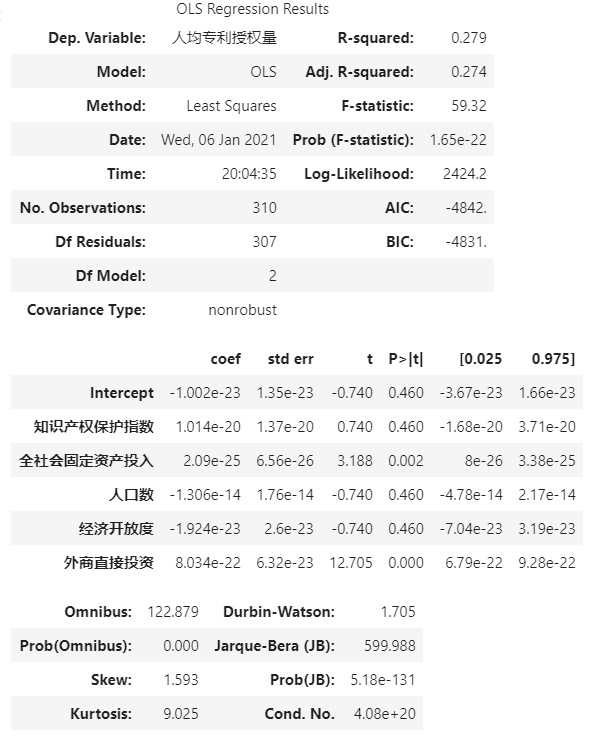
1. df[["人均专利授权量","人均专利申请量","知识产权保护指数","人口数","全社会固定资产投入","GDP","经济开放度","外商直接投资"]].corr(method='pearson')



⑥回归分析

以'人均专利授权量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

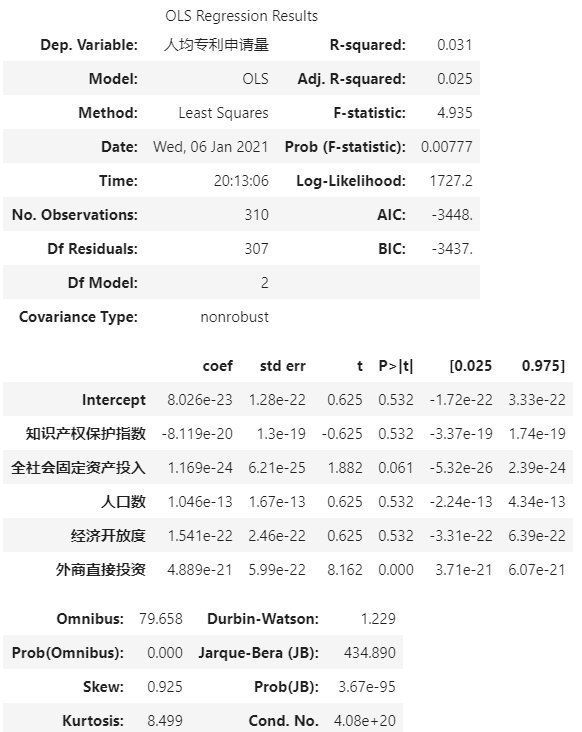
1. **import** statsmodels.formula.api as smf
3. result = smf.ols('人均专利授权量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资', data=df).fit()
4. result.summary()



该模型的R方为0.279，调整后的R方为0.274这些自变量中，说明该模型能解释人均专利授权量接近30%的变化。只有全社会固定资产投入和外商直接投资在10%的显著性水平下显著，其他的自变量都在该显著性水平下不显著。当全社会固定资产投入增加10亿元，人均专利授权量增加2.09e-15。当外商直接投资数额增加10亿元，人均专利授权量增加8.034e-12。

以'人均专利申请量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

1. result = smf.ols('人均专利申请量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资', data=df).fit()
2. result.summary()

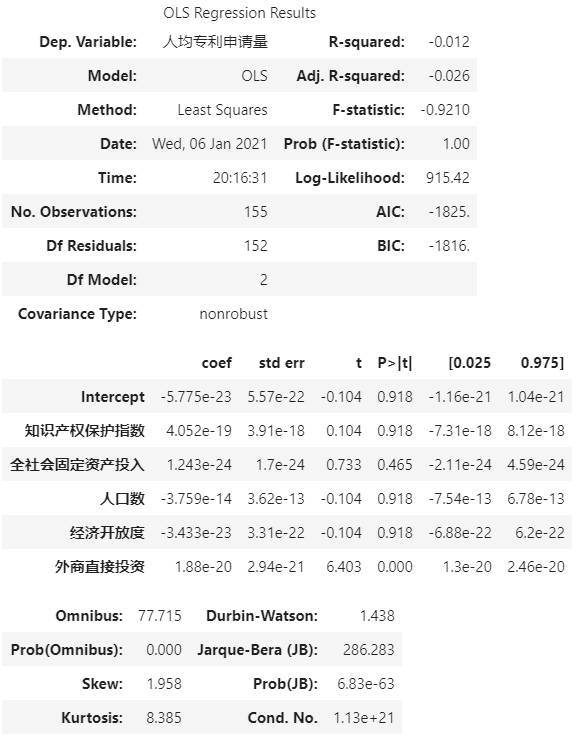


该模型的R方为0.031，调整后的R方为0.025这些自变量中，说明该模型仅能解释人均专利申请量接近3%的变化。只有全社会固定资产投入和外商直接投资在10%的显著性水平下显著，其他的自变量都在该显著性水平下不显著。当全社会固定资产投入增加10亿元，人均专利申请量增加1.169e-14。当外商直接投资数额增加10亿元，人均专利申请量增加4.889e-11。

⑦分组回归

根据GDP\_哑变量分组进行同上的回归分析

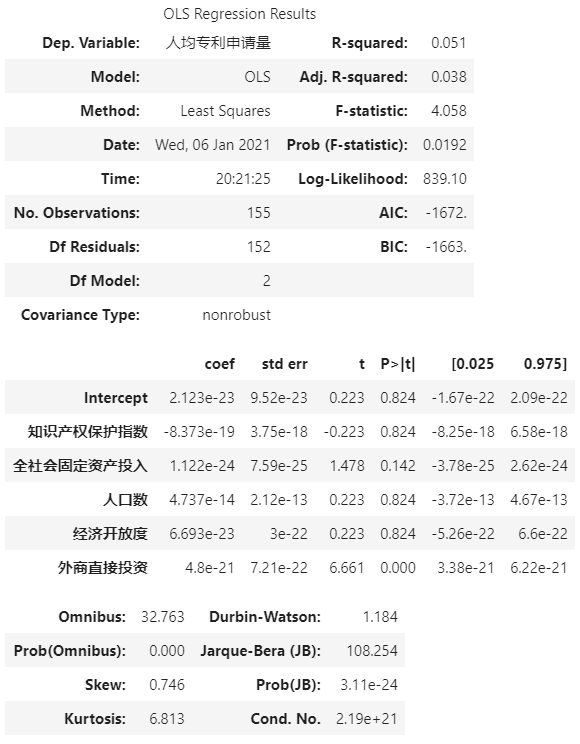
（1）GDP\_哑变量为0，以'人均专利申请量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

1. result = smf.ols('人均专利申请量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资',
2. data=df[df['GDP\_哑变量']==0]).fit()
3. result.summary()

该模型的R方和调整后的R方均为负数。在该模型中，只有外商直接投资这一自变量是在10%的显著性水平下显著，其他自变量均不显著。当外商直接投资增加10亿元时，人均专利申请量增加1.88e-10

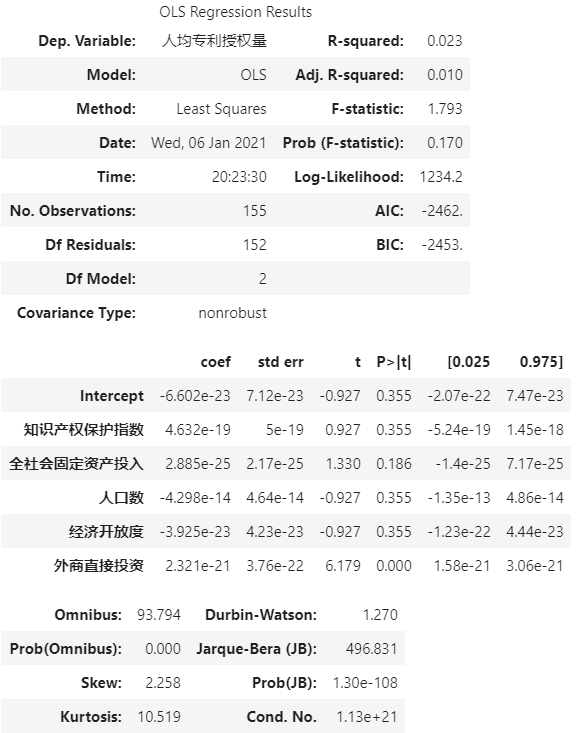
（2）GDP\_哑变量为1，以'人均专利申请量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

1. result = smf.ols('人均专利申请量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资',
2. data=df[df['GDP\_哑变量']==1]).fit()
3. result.summary()



该模型的R方为0.051，调整后的R方为0.038，说明该模型能解释人均专利申请量接近5%的变化，比没有分组是要高。在该模型中，只有外商直接投资这一自变量是在10%的显著性水平下显著，其他自变量均不显著。当外商直接投资增加10亿元时，人均专利申请量增加4.8e-11

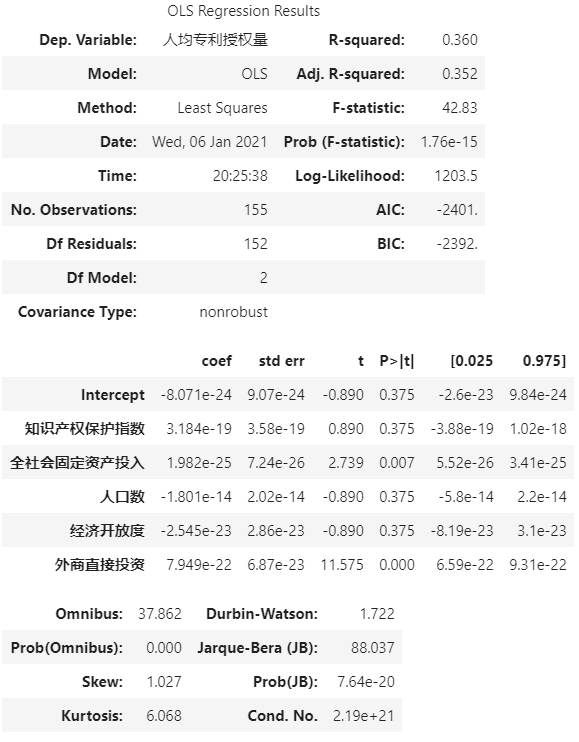
（3）GDP\_哑变量为0，以'人均专利授权量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

1. result = smf.ols('人均专利授权量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资',
2.                  data=df[df['GDP\_哑变量']==0]).fit()
3. result.summary()

该模型的R方为0.02.3，调整后的R方为0.01，说明该模型能解释人均专利授权量接近2%的变化，比没有分组是要低。在该模型中，只有外商直接投资这一自变量是在10%的显著性水平下显著，其他自变量均不显著。当外商直接投资增加10亿元时，人均专利授权量增加2.321e-11

（4）GDP\_哑变量为1，以'人均专利授权量',为因变量，以'知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资'为自变量，回归分析

1. result = smf.ols('人均专利授权量 ~ 知识产权保护指数 + 全社会固定资产投入 + 人口数 + 经济开放度 + 外商直接投资',
2. data=df[df['GDP\_哑变量']==1]).fit()
3. result.summary()



该模型的R方为0.36，调整后的R方为0.352，说明该模型能解释人均专利授权量接近35%的变化，比没有分组是要高得多得多。在该模型中，外商直接投资和全社会固定资产投入这两个自变量是在10%的显著性水平下显著，其他自变量均不显著。当外商直接投资增加10亿元时，人均专利授权量增加7.949e-12。当全社会固定资产投入增加10亿元时，人均专利授权量增加1.982e-15

# 企业实证

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | 楼 | 房； | **实验台号：** |  |
| **实验日期与时间：** |  | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** |  |

【实验目的及要求】

1. 企业基本面数据与知识产权数据可视化
   1. 从下载的不同格式数据中， 解释出包含但不限于企业知识产权数据、ROE/ROA数据、资产负债表率、总营业收入、无形资产、资产总额等数据，
   2. 以上数据与之前的数据进行整合，并选择合适的图例进行可视化
2. 回归实证

以企业的专利产出作为因变量，地区知识产权指数等其他数据作为自变量，对以上数据进行统计检验。实现截面数据回归

1. 企业创新100指数：

从专利密集型行业分类后，分别选择不同行业排名靠前的公司（共100家）作为企业创新100指数成分股，企业创新的绩效基准和股票价格指数

一、企业基本面数据与知识产权数据可视化

a)

ROE:：净资产收益率（Return on Equity，简称ROE），又称股东权益报酬率/净值报酬率/权益报酬率/权益利润率/净资产利润率，是净利润与平均股东权益的百分比，是公司税后利润除以净资产得到的百分比率，该指标反映股东权益的收益水平，用以衡量公司运用自有资本的效率。指标值越高，说明投资带来的收益越高。该指标体现了自有资本获得净收益的能力。

ROA：[资产](https://baike.baidu.com/item/%E8%B5%84%E4%BA%A7" \t "_blank)回报率= 税后净利润/[总资产](https://baike.baidu.com/item/%E6%80%BB%E8%B5%84%E4%BA%A7/9803630)，资产回报率，也叫[资产收益率](https://baike.baidu.com/item/%E8%B5%84%E4%BA%A7%E6%94%B6%E7%9B%8A%E7%8E%87/2429091)，它是用来衡量每单位资产创造多少[净利润](https://baike.baidu.com/item/%E5%87%80%E5%88%A9%E6%B6%A6/5352725)的指标。评估公司相对其总资产值的盈利能力的有用指标。计算的方法为公司的年度盈利除以总资产值，资产回报率一般以百分比表示。有时也称为[投资回报率](https://baike.baidu.com/item/%E6%8A%95%E8%B5%84%E5%9B%9E%E6%8A%A5%E7%8E%87/89993)。

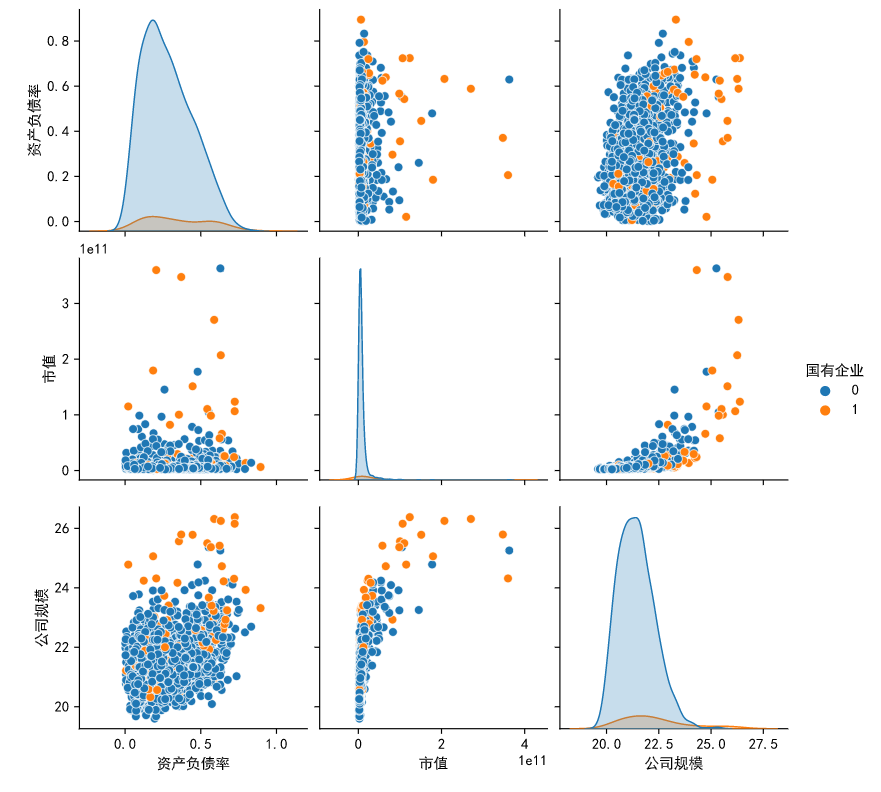
资产负债率：是指一家企业总资产中有多少比例是由债务形成的，是衡量企业财务状的一个关键指标。

总营业收入：营业总收入是指企业在从事销售商品，提供劳务和让渡资产使用权等日常经营业务过程中所形成的经济利益的总流入。

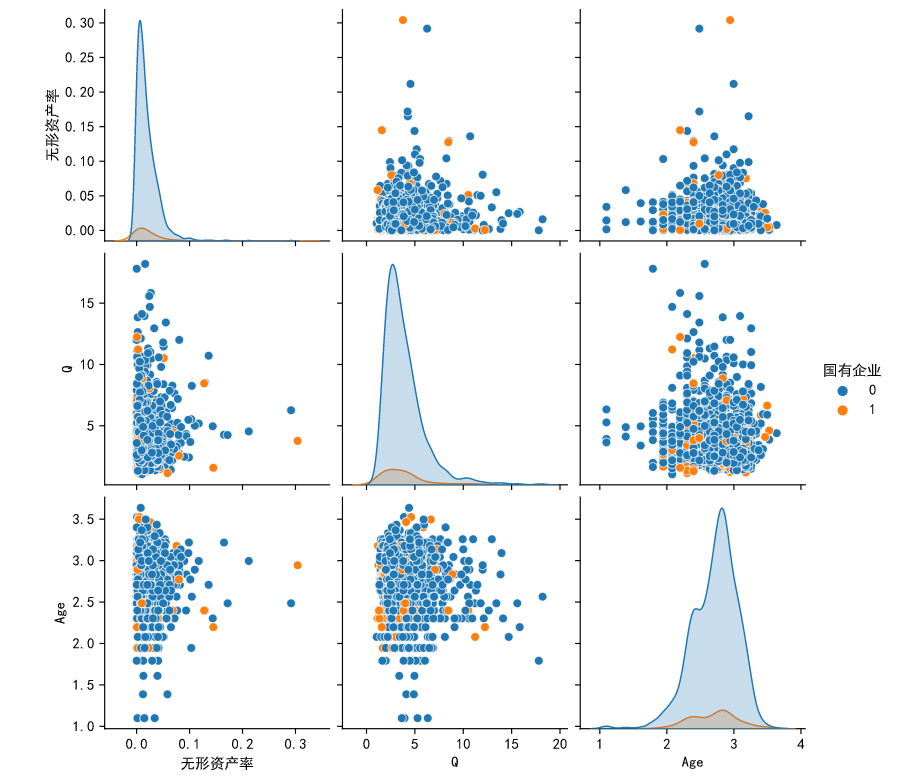
无形资产：无形资产（Intangible Assets）是指企业拥有或者控制的没有实物形态的可辨认非货币性资产。

资产总额：资产总额是指企业资产负债表的资产总计项。公司过去的交易、事项形成并由企业拥有或控制的资源，资产总额即是指企业拥有或控制的全部资产，这些资产包括流动资产、长期投资、固定资产、无形及递延资产、其他长期资产等。

b)可视化



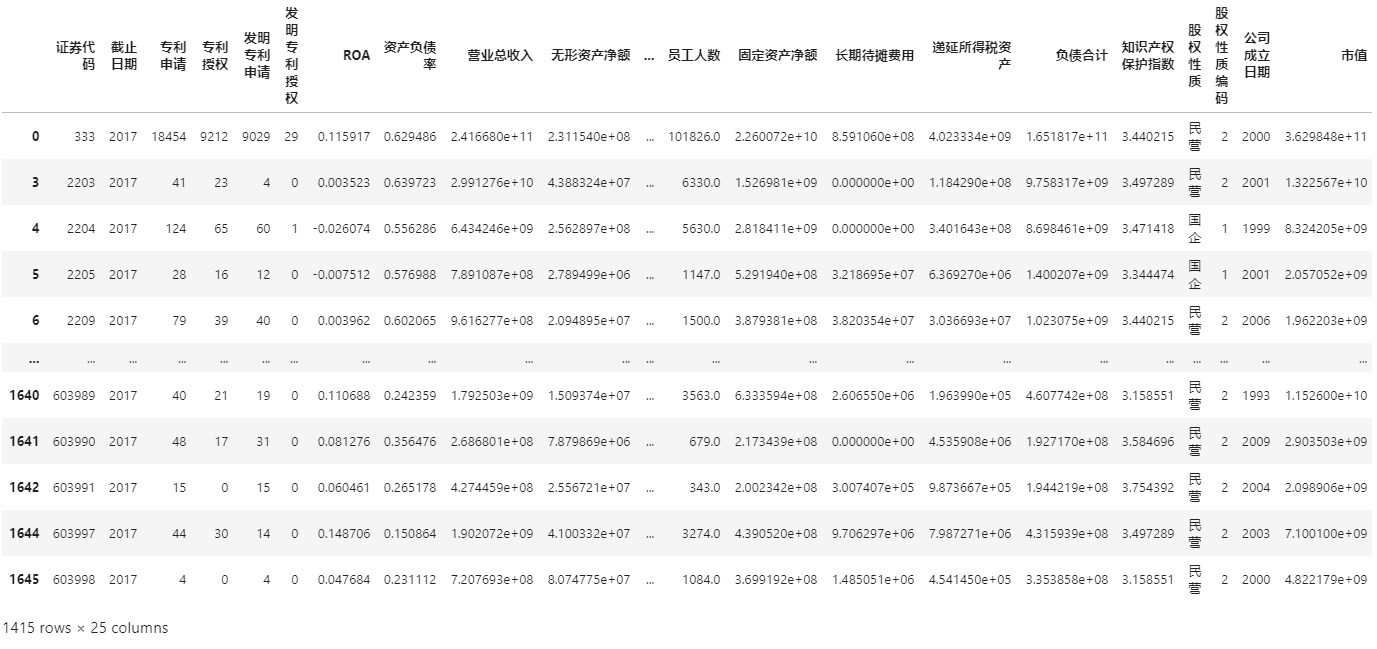
由上图我们可以发现，国企的公司规模总体上比非国有企业要大，随着公司规模的增大，市值呈现上升的趋势。



二、回归实证

①使用国泰企业数据.xlsx，将公司市值数据整合。读取文件公司成立日期.xlsx，整合到上面的df，并删除没有2017年数据的公司所有记录，

1. df1 = pd.read\_excel('./国泰安企业数据.xlsx',index\_col=0)
2. # 去重
3. df1 = df1.sort\_values(['证券代码','截止日期'])
4. df1.drop\_duplicates('证券代码', keep='last', inplace=True)
5. df1 = df1[df1['截止日期']==2017]
6. df2 = pd.read\_excel('./公司成立日期.xlsx')
7. df2['公司成立日期'] = df2['公司成立日期'].str[:4]
8. df = pd.merge(left=df1, right=df2, on='证券代码')
9. df3 = pd.read\_excel('./市值.xlsx')
10. df3 = df3[df3['2017'].notnull()]
11. df3['证券代码'] = df3['证券代码'].map(**lambda** x: int(x[:-3]))
12. df3 = df3[['证券代码','2017']]
13. df3.columns = ['证券代码','市值']
14. df = pd.merge(left=df,right=df3,on='证券代码')
15. df.dropna(inplace=True)
16. df



②**计算专利产出、Age、公司规模、托宾Q值、国有企业哑变量**

**专利产出：**用专利授权量除以资产总计，

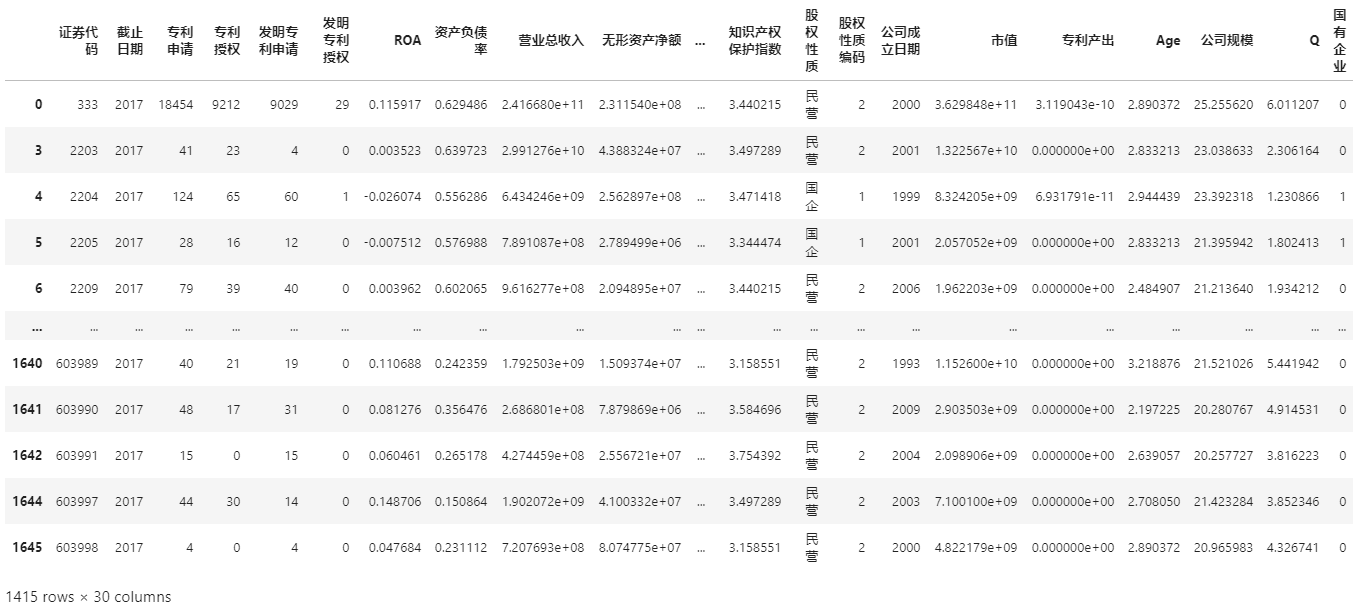
Age：公司成立年限+1取对数（成立年限：成立日到截至日期）

公司规模：为年末公司总资产取对数

托宾Q值：（股票市值+净债务）/有形资产现行价值，其中有形资产现行价值=资产总额-待摊费用-无形资产-递延税款借项）

**国有企业哑变量：1表示是，0表示否。根据股权性质编码，为1的是国有企业**

1. df['专利产出'] = df['发明专利授权'] / df['资产总计']
2. df['截止日期'] = df['截止日期'].map(**lambda** x: int(x))
3. df['公司成立日期'] = df['公司成立日期'].map(**lambda** x: int(x))
4. df['Age'] = np.log(df['截止日期'] - df['公司成立日期'] + 1)
5. df['公司规模'] = np.log(df['资产总计'])
6. df['Q'] = (df['市值'] + df['负债合计']) / (df['资产总计'] - df['长期待摊费用'] - df['无形资产净额'] - df['递延所得税资产'])
7. df['国有企业'] = df['股权性质'].map(**lambda** x: 1 **if** '国企' **in** x **else** 0)
8. df.to\_excel('./整合数据.xlsx')



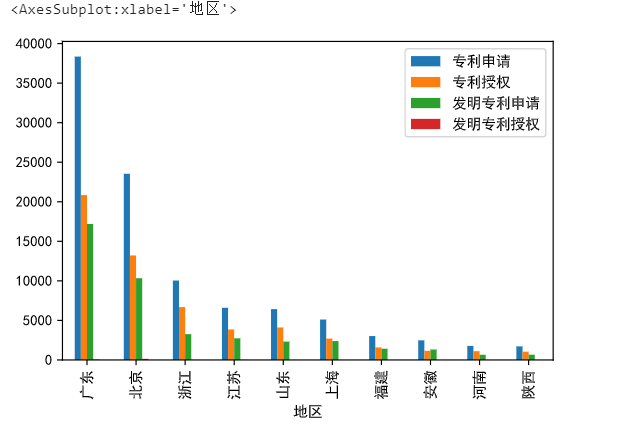
③使用describe函数进行描述统计



通过描述统计观察，所有数据的统计量分布都在正常范围内，可以认为没有严重的数据错误出现

④地区专利分布分析

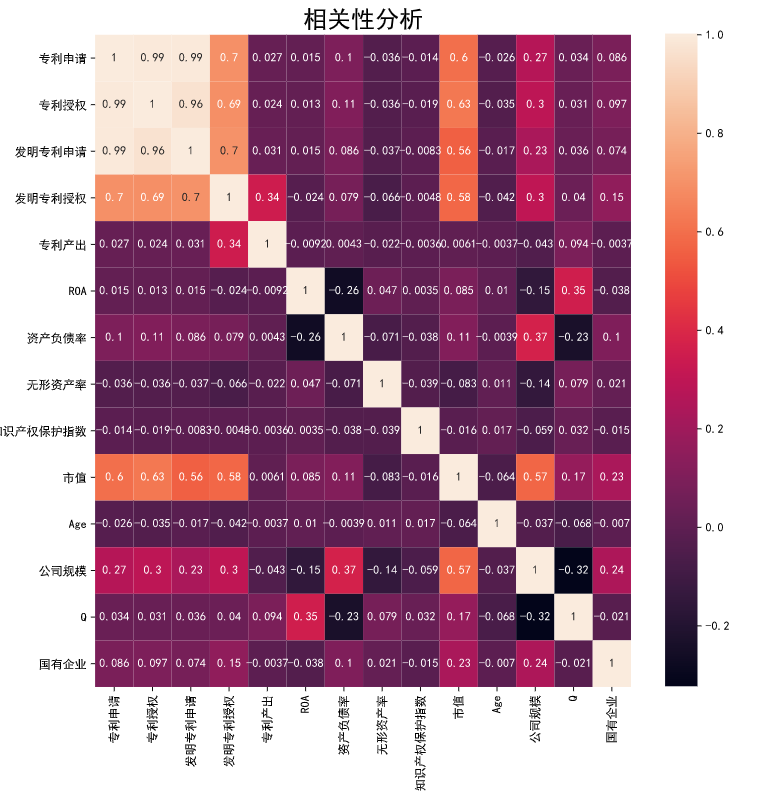
1. analyse = ['专利申请','专利授权','发明专利申请','发明专利授权']
2. data = []
3. **for** i **in** analyse:
4. data.append(df.groupby('地区')[i].sum())
6. data = pd.concat(data, axis=1)
7. data = data.sort\_values(['专利申请','专利授权','发明专利申请','发明专利授权'], ascending=False)
8. data[:10].plot.bar()



观察到，广东省无论是专利申请数量、专利授权数量还是发明专利申请数量，都位于全国第一。接下来到北京、浙江和江苏，专利申请数量和发明专利申请数量出现断崖式下降。

⑤变量的相关性分析

1. plt.figure(figsize=(10, 10))
2. ax = sns.heatmap(df[lie].corr(method='pearson'), annot=True)
3. ax.set\_title("相关性分析", fontdict={'weight':'normal','size': 20})
4. plt.show()



可以发现，专利申请、专利授权和发明专利申请三者之间具有高度的相关性。其次，市值专利申请、专利授权、发明专利申和+发明专利授权量四者关联关系相比其他而言也十分显著。

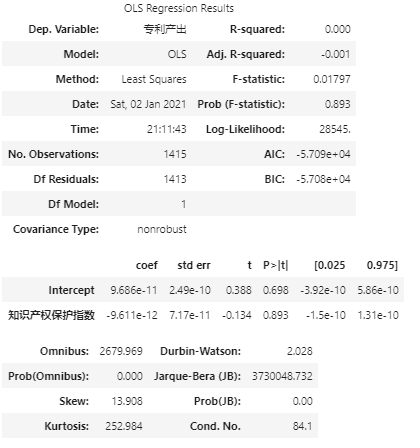
⑥正态性检验

1. **from** scipy **import** stats
3. **for** x **in** lie[:-1]:
4. **print**(x)
5. data = df[x].values
6. **print**(stats.kstest(data,cdf='norm'))
7. **print**(stats.shapiro(data))
8.     **print**('')

所有变量的P值都小于0.001，因此可以在0.1%的显著性水平下拒绝零假设，认为这些变量都不服从正态分布

⑦因变量为专利产出，与知识产权保护指数进行回归分析，分析相关性

1. result = smf.ols('专利产出 ~ 知识产权保护指数', data=df).fit()
2. result.summary()

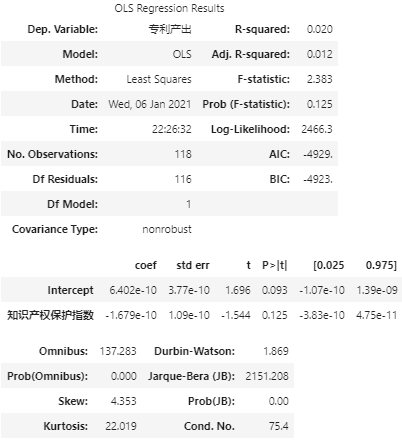


该模型的R方为0，调整后的R方为负。知识产权保护指数在10%的显著性水平下不显著。

按照国有变量进行分分类，再分析相关性。

当该企业是国有企业时

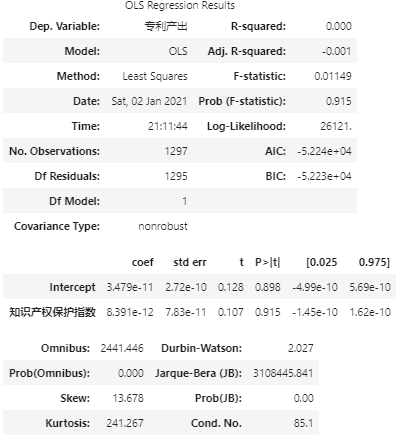
1. result = smf.ols('专利产出 ~ 知识产权保护指数', data=df[df['国有企业']==1]).fit()
2. result.summary()



该模型的R方为0.02，调整后的R方为0.012。知识产权保护指数在15%的显著性水平下显著。R方和调整后的R方均有所提高

当企业不是国有企业时

1. result = smf.ols('专利产出 ~ 知识产权保护指数', data=df[df['国有企业']==0]).fit()
2. result.summary()

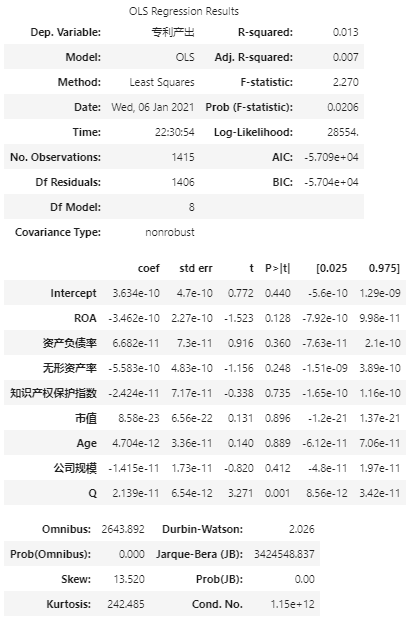


该模型的R方为0，调整后的R方为负，和之前的模型R方相同。知识产权保护指数的P值更大，几乎接近1。

综上，当企业是国有企业时，其知识产权保护指数与专利产出的相关性更大。

⑧以'专利产出'作为因变量，用'ROA', '资产负债率', '无形资产率', '知识产权保护指数', '市值', 'Age', '公司规模', 'Q'，作为因变量进行多元回归分析。

1. result = smf.ols('专利产出 ~ ROA+资产负债率+无形资产率+知识产权保护指数+市值+Age+公司规模+Q', data=df).fit()
2. result.summary()



该模型的R方为0.013，调整后的R方为0.007。自变量中只有Q在10%的显著性水平下显著，ROA和Q在15%的显著性水平下显著，其余自变量均不显著。当Q增加1，专利产出增加2.139e-11。当ROA增加0.1，专利产出减少3.462e-11。

⑨根据国有企业分组分别进行同6的回归分析

非国有企业回归

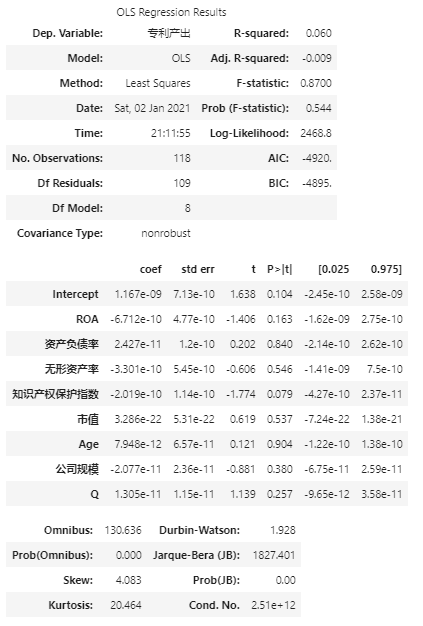
1. result = smf.ols('专利产出 ~ ROA+资产负债率+无形资产率+知识产权保护指数+市值+Age+公司规模+Q', data=df[df['国有企业']==0]).fit()
2. result.summary()



该模型的R方为0.013，调整后的R方为0.007。自变量中只有Q在10%的显著性水平下显著，ROA和Q在20%的显著性水平下显著，其余自变量均不显著。当Q增加1，专利产出增加7.34e-12。当ROA增加0.1，专利产出减少3.216e-11。

国有企业回归

1. result = smf.ols('专利产出 ~ ROA+资产负债率+无形资产率+知识产权保护指数+市值+Age+公司规模+Q', data=df[df['国有企业']==1]).fit()
2. result.summary()



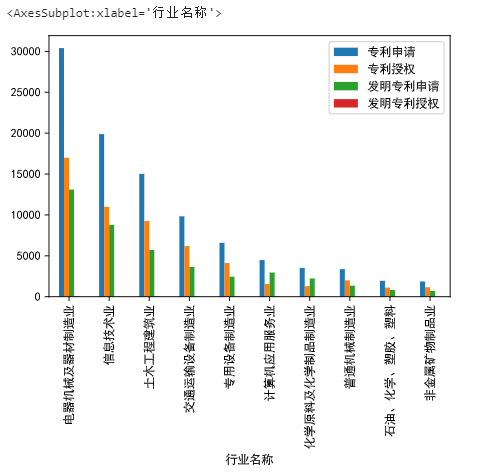
该模型的R方为0.06，调整后的R方为负。自变量中只有ROA在20%的显著性水平下显著，其余自变量均不显著。当ROA增加0.1，专利产出减少4.77e-11。

三、企业创新100指数

从专利密集型行业，分别选择不同行业排名靠前的公司（共100家）作为企业创新100指数成分股，构建企业创新的绩效基准和股票价格指数

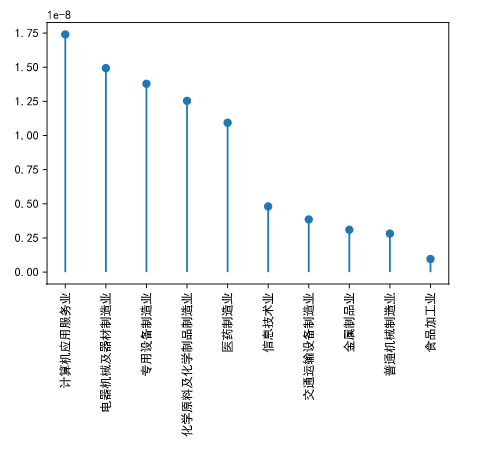
①总共63个行业，找出专利数据排名前10个行业数据，并画图

1. data = []
2. **for** i in analyse:
3. data.append(df.groupby('行业名称')[i].sum())
5. data = pd.concat(data, axis=1)
6. data = data.sort\_values(['专利申请','专利授权','发明专利申请','发明专利授权'], ascending=False)
7. data[:10].plot.bar()



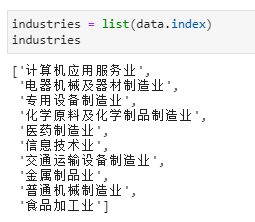
②找出专利产出前10名的行业，绘图

1. data = pd.DataFrame(df.groupby('行业名称')['专利产出'].sum()).nlargest(10,columns='专利产出')
2. plt.scatter(list(data.index), data.values)
3. plt.vlines(list(data.index),0,data.values)
4. plt.xticks(rotation=90)
5. plt.show()



③编写自己的选股策略

1）找到原来数据中行业名称为上面的10个行业的数据



2）找出每个行业专利产出前10的数据，按照每只股票2017年该股票市值占100只股票市值总和的比重构建权重，得到创新100指数

1. data = df[df['行业名称'].isin(industries)]
2. data = pd.DataFrame(data.groupby('行业名称')['专利产出'].nlargest(10)).reset\_index()
3. data.columns = ['行业名称','证券代码','专利产出']
4. index = list(data['证券代码'])
5. data = df.loc[index,['证券代码','截止日期','市值']]
6. data['权重'] = data['市值'].map(lambda x: x / sum(data['市值']))
7. data[['证券代码','权重']]

