

课题名称 51Job招聘数据挖掘

学 院 数学与信息科学学院

专 业 信息管理与信息系统

班 级 2016级1班

成 员 卢荣辉 1611240135

黄镘潼 1611240113

# 摘 要

网络爬虫是一种自动搜集互联网信息的程序。通过网络爬虫不仅能够为搜索引擎采集网络信息，而且可以作为定向信息采集器，定向采集某些网站下的特定信息，如招聘信息等。文章阐述了在PyCharm运行环境下利用简短的Python语句从Job51.com中采集时间段内广东/广西Java岗位招聘数据，并对采集的数据进行清洗、加工、可视化分析的一个过程。

关键词：数据 文本挖掘 招聘 网络爬虫 Python 可视化

目 录

[摘 要 2](#_Toc7908434)

[第一章 前言 4](#_Toc7908435)

[1.1研究背景和意义 4](#_Toc7908436)

[第二章 数据收集 5](#_Toc7908437)

[2.1 数据来源 5](#_Toc7908438)

[2.2 数据爬取工具 5](#_Toc7908439)

[2.3 简单信息数据爬取 5](#_Toc7908440)

[2.4 详细信息数据爬取 6](#_Toc7908441)

[第三章 数据概览 9](#_Toc7908442)

[3.1 数据读取 9](#_Toc7908443)

[3.2 描述性统计 9](#_Toc7908444)

[第四章 数据预处理 10](#_Toc7908445)

[4.1 数据清洗 10](#_Toc7908446)

[4.2 数据加工 11](#_Toc7908447)

[第五章 可视化分析 15](#_Toc7908448)

[5.1 公司类型可视化分析 15](#_Toc7908449)

[5.2 公司规模可视化分析 16](#_Toc7908450)

[5.3 学历和工作经验可视化分析 18](#_Toc7908451)

[5.4 薪资与地区的关系分析 19](#_Toc7908452)

[5.5 薪资与工作经验关系分析 20](#_Toc7908453)

[5.6 薪资与学历的关系分析 22](#_Toc7908454)

[5.7 公司性质词云分析 23](#_Toc7908455)

[5.8 福利词云分析 24](#_Toc7908456)

[5.9 岗位职责词云分析 26](#_Toc7908457)

[5.10 工作要求词云分析 27](#_Toc7908458)

[第六章 总结 30](#_Toc7908459)

[6.1 招聘信息总结 30](#_Toc7908460)

[6.2 数据分析结论的可靠性(不足) 30](#_Toc7908461)

# 前言

随着互联网的飞速发展，网络上的信息呈爆炸式增长。这使得人们在网上找到所需的信息越来越困难，这种情况下搜索引擎应运而生。搜索引擎搜集互联网上数以亿计的网页，并为每个词建立索引。在建立搜索引擎的过程中，搜集网页是非常重要的一个环节。

爬虫程序就是用来搜集网页的程序。以何种策略偏历互联网上的网页，也成了爬虫程序主要的研究方向。现在比较流行的搜索引擎，比如google，百度，它们爬虫程序的技术内幕一般都不公开。

爬虫程序是一个自动获取网页的程序。它为搜索引擎从互联网上下载网页，是搜索引擎的重要组成部分。爬虫程序的实现策略，运行效率直接影响搜索引擎的搜索结果。不同的搜索引擎，会根据对搜索结果的不同需求，选择最合适的爬行策略来搜集互联网上的信息。高效，优秀的爬虫程序可以使人们在互联网上寻找到更及时，更准确的信息。

实现网络爬虫的重点和难点有：多线程的实现；对临界资源的分配；遍历web图的遍历策略选择和实现；存储数据结构的选择和实现。

本文通过Python语言实现一个简单的单线程爬虫程序。通过实现此爬虫程序可以定点搜集某一站点的URL，如果需要搜集其他信息，可以在解析URL的同时，解析获取相应信息。

# 第二章 数据收集

## 2.1 数据来源

前程无忧招聘网站，爬取广西地区与广东地区有关Java岗位的招聘信息

## 2.2 数据爬取工具

Python、PyCharm

## 2.3 简单信息数据爬取

2.3.1 说明：爬取前程无忧网站上每一条有关广西/广东地区的Java招聘信息，包括职位、详细职位的url地址、公司名称、公司地址、薪资、日期。以下仅列出广西区的数据收集过程。

2.3.2 主要代码

(1) 打开目标网站：

def get\_content(page):  
 url = 'https://search.51job.com/list/140000,000000,0000,00,9,99,java,2,'+str(page)+'.html?lang=c&stype=&postchannel=0000&workyear=99&cotype=99&degreefrom=99&jobterm=99&companysize=99&providesalary=99&lonlat=0%2C0&radius=-1&ord\_field=0&confirmdate=9&fromType=&dibiaoid=0&address=&line=&specialarea=00&from=&welfare='  
 a = urllib.request.urlopen(url)*#打开网址* html = a.read().decode('gbk')*#读取源代码并转为unicode* return html

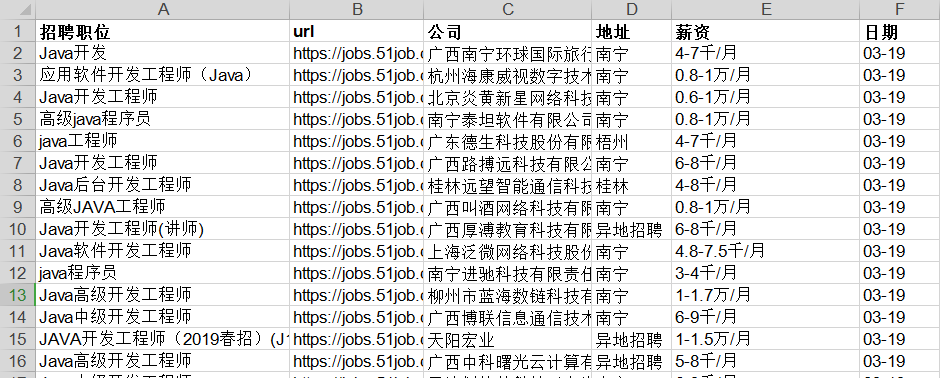
1. 正则表达式

def get(html):  
 reg = re.compile(r'class="t1 ">.\*? <a target="\_blank" title="(.\*?)".\*? href="(.\*?)".\*?<span class="t2"><a target="\_blank" title="(.\*?)".\*?<span class="t3">(.\*?)</span>.\*?<span class="t4">(.\*?)</span>.\*? <span class="t5">(.\*?)</span>',re.S)#匹配换行符  
 items = re.findall(reg,html)  
 return items

1. 写入excel表格

for item in items:#职位信息  
 for i in range(0,6):  
 #print item[i]  
 ws.write(index,i,item[i])#行，列，数据  
 print(index)  
 index+=1  
  
newTable="crawler\_jiemian.xls"#表格名称  
wb = xlwt.Workbook(encoding='utf-8')#创建excel文件，声明编码  
ws = wb.add\_sheet('sheet1')#创建表格  
headData = ['招聘职位','url','公司','地址','薪资','日期']#表头部信息  
for colnum in range(0, 6):  
 ws.write(0, colnum, headData[colnum], xlwt.easyxf('font: bold on')) # 行，列  
for each in range(1,11):  
 index=(each-1)\*50+1  
 excel\_write(get(get\_content(each)),index)  
wb.save(newTable)

2.3.3 获取到的部分初始数据



## 2.4 详细信息数据爬取

2.4.1 说明：根据简单信息数据爬取所得的url，爬取特定每条招聘的详细信息，包括职位、薪资、公司、公司信息、公司地址、地区、工作经验、学历、人数、时间、福利、岗位信息。以下仅列出广西区的数据收集过程。

2.4.2 主要代码

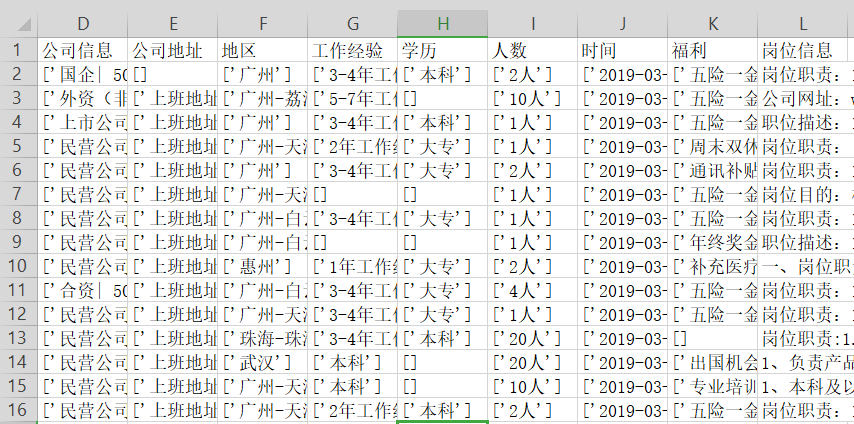
(1) 获取详细目标网站：

def getUrl(url):  
 print('New page')  
 res = requests.get(url)  
 res.encoding = 'GBK'  
 #print(res.text)  
 if res.status\_code == requests.codes.ok:  
 selector = etree.HTML(res.text)  
 urls = selector.xpath('//\*[@id="resultList"]/div/p/span/a/@href')  
 print(urls)  
 for url in urls:  
 parseInfo(url)  
 time.sleep(random.randrange(1, 4))

1. 正则表达式：

def parseInfo(url):  
 headers = {  
 'User-Agent': 'Opera/9.80 (Android 2.3.4; Linux; Opera Mobi/ADR-1301071546) Presto/2.11.355 Version/12.10'  
 }  
 res = requests.get(url, headers=headers)  
 res.encoding = 'utf-8'  
 selector = etree.HTML(res.text)  
 #职位  
 title = selector.xpath('//\*[@id="pageContent"]/div[1]/div[1]/p/text()')  
 #薪资  
 salary = selector.xpath('//\*[@id="pageContent"]/div[1]/p/text()')  
 #公司  
 company = selector.xpath('//\*[@id="pageContent"]/div[2]/a[1]/p/text()')  
 #公司信息  
 companyinfo = selector.xpath('//\*[@id="pageContent"]/div[2]/a[1]/div/text()')  
 #公司地址  
 companyplace = selector.xpath('//\*[@id="pageContent"]/div[2]/a[2]/span/text()')  
 #地区  
 place = selector.xpath('//\*[@id="pageContent"]/div[1]/div[1]/em/text()')  
 #工作经验  
 exp = selector.xpath('//\*[@id="pageContent"]/div[1]/div[2]/span[2]/text()')  
 #学历  
 edu = selector.xpath('//\*[@id="pageContent"]/div[1]/div[2]/span[3]/text()')  
 #人数  
 num = selector.xpath('//\*[@id="pageContent"]/div[1]/div[2]/span[1]/text()')  
 #时间  
 time = selector.xpath('//\*[@id="pageContent"]/div[1]/div[1]/span/text()')  
 #福利  
 fuli = selector.xpath('//\*[@id="pageContent"]/div[3]/div[3]/span/text()')  
 #岗位信息  
 info = selector.xpath('string(//\*[@id="pageContent"]/div[3]/div[2]/article)')  
 info = str(info).strip()

2.4.3 获取到的部分初始数据



# 第三章 数据概览

## 3.1 数据读取

以获取到的广东区的详细招聘数据为例，

df = pd.read\_csv(r'crawler\_clouds.csv')

df.info()

## 3.2 描述性统计

*#显示所有列*pd.set\_option('display.max\_columns', None)  
*#显示所有行*pd.set\_option('display.max\_rows', None)  
*#设置value的显示长度为100，默认为50*pd.set\_option('max\_colwidth',100)  
*# 设置字符显示宽度  
#pd.set\_option('display.width', 300)  
#2.大写的字母O,描述性分析*print(df.describe(include=['O']))



由上可以得出一些信息：

目前的数据维度11447行\*14列，职位名称中'Java开发工程师'最多, 多为民营公司,学历要求多为本科, 福利多含五险一金等信息。

# 第四章 数据预处理

## 4.1 数据清洗

目的是不让有错误或有问题的数据进入加工过程, 其主要内容包括: 重复值, 缺失值。以获取到的广东区的详细招聘数据为例。

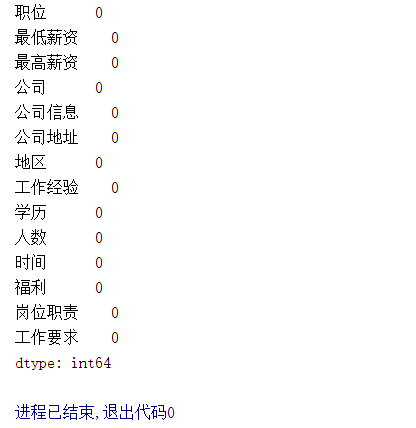
4.1.1 删除重复值

zhaopin = df.drop\_duplicates(inplace = False)  
zhaopin.shape

4.1.2 缺失值处理

计算每个特征中缺失值个数(下图左)：

print(df.isnull().sum())

图(左)图(右)

部分填充代码：df.职位.fillna('Java开发',inplace=True)

职位缺失我们用"Java开发"填充，公司缺失我们用"其他"填充，公司信息缺失我们用"其他"填充，地区缺失我们用"其他"填充，人数缺失我们用"若干人"填充，时间缺失我们用"20119/3/25"填充，公司地址缺失我们用"其他"填充，

工作经验缺失很有可能是"不限经验", 我们就用"不限经验"填充，

学历缺失很有可能是"不限学历", 我们就用"不限学历"填充，

福利缺失很有可能包括"五险一金", 我们就用"五险一金"填充，

岗位职责与工作要求缺失皆用"面议"填充，(岗位职责与工作要求在下面的数据加工时由岗位信息分离)

我们再来处理后的每个特征中缺失值个数(上图右)

## 4.2 数据加工

包括使用excel去除详细招聘数据中[] " " 等无关信息；使用excel区分公司类型、公司规模、公司性质（即拆分 公司信息）；使用excel区分岗位职责、工作要求 (即拆分 岗位信息)；使用Python区分最高、最低薪资，统一薪资单位为元/月（即拆分 薪资）；使用Python统一地区格式。以获取到的广东区的详细招聘数据为例。

4.2.1 excel去除无关信息

CTRL+F 查询[] " " 等无关信息并删除。

4.2.2 excel拆分"公司信息"

由于公司信息中包含公司类型、公司规模、公司性质三类信息，且每条记录中均有‘ | ’分隔开，故可用excel的分列功能以 ‘ | ’拆分"公司信息"。

拆分前的公司信息如下图(左)所示，拆分后的公司类型、公司规模、公司性质如下图(右)所示。

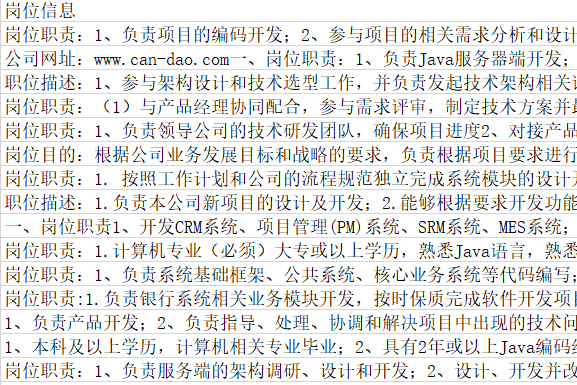
图(左)

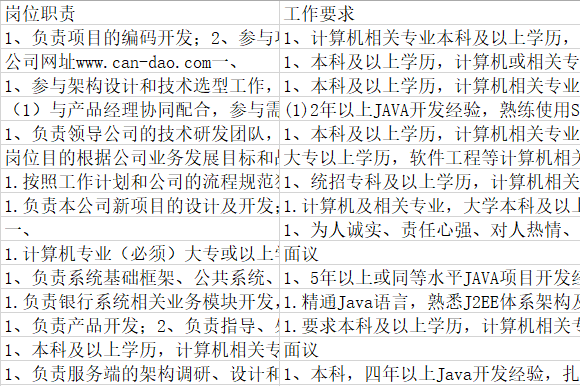
图(右)

4.2.3 excel拆分"岗位信息"

"岗位信息"中包含了岗位职责与工作要求，由于技术有限，无法在网络爬虫时通过准确的正则表达式或模糊匹配进行分离，故使用excel自带的功能区分岗位职责、工作要求 (即拆分 岗位信息)。具体过程略。

拆分前的岗位信息如下图(左)所示，拆分后的岗位职责与工作要求如下图(右)所示。

图(左)

图(右)

4.2.4 Python处理地区格式

地区取到城市，部分代码如下，处理前后对比图略：

for i in range(1,nrows):

area = table.row\_values(i)[5][:2]*#地区取到城市，把区域去掉*

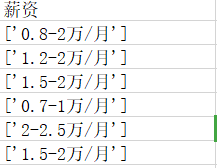
4.2.5 Python处理薪资格式

由于爬取的薪资数据是范围数据，不便于进行数据分析，我们将对薪资进行处理，区分最高、最低薪资，并用统一薪资单位为千/月(或者元/月)。

主要代码如下：

*#利用正则表达式提取月薪，把待遇规范成千/月的形式*def get\_salary(salary):  
 if '-'in salary: #针对1-2万/月或者10-20万/年的情况，包含-  
 low\_salary=re.findall(re.compile('(\d\*\.?\d+)'),salary)[0]  
 high\_salary=re.findall(re.compile('(\d?\.?\d+)'),salary)[1]  
 if u'万' in salary and u'年' in salary:#单位统一成千/月的形式  
 low\_salary = float(low\_salary) / 12 \* 10  
 high\_salary = float(high\_salary) / 12 \* 10  
 elif u'万' in salary and u'月' in salary:  
 low\_salary = float(low\_salary) \* 10  
 high\_salary = float(high\_salary) \* 10  
 else:#针对20万以上/年和100元/天这种情况，不包含-，取最高工资，没有最低工资  
 #low\_salary= re.findall(re.compile('(\d\*\.?\d+)'), salary)[0]  
 low\_salary = ""  
 high\_salary= re.findall(re.compile('(\d\*\.?\d+)'), salary)[0]  
 if u'万' in salary and u'年' in salary:#单位统一成千/月的形式  
 #low\_salary = float(low\_salary) / 12 \* 10  
 high\_salary = float(high\_salary) / 12 \* 10  
 elif u'万' in salary and u'月' in salary:  
 #low\_salary = float(low\_salary) \* 10  
 high\_salary = float(high\_salary) \* 10  
 elif u'元'in salary and u'天'in salary:  
 #low\_salary = float(low\_salary)/1000\*21  
 high\_salary = float(high\_salary)/1000\*21#每月工作日21天  
 return low\_salary,high\_salary

处理前的薪资如下图(左)所示，处理后的最低薪资与最高薪资如下图(右)所示。

(图左)(图右)

# 第五章 可视化分析

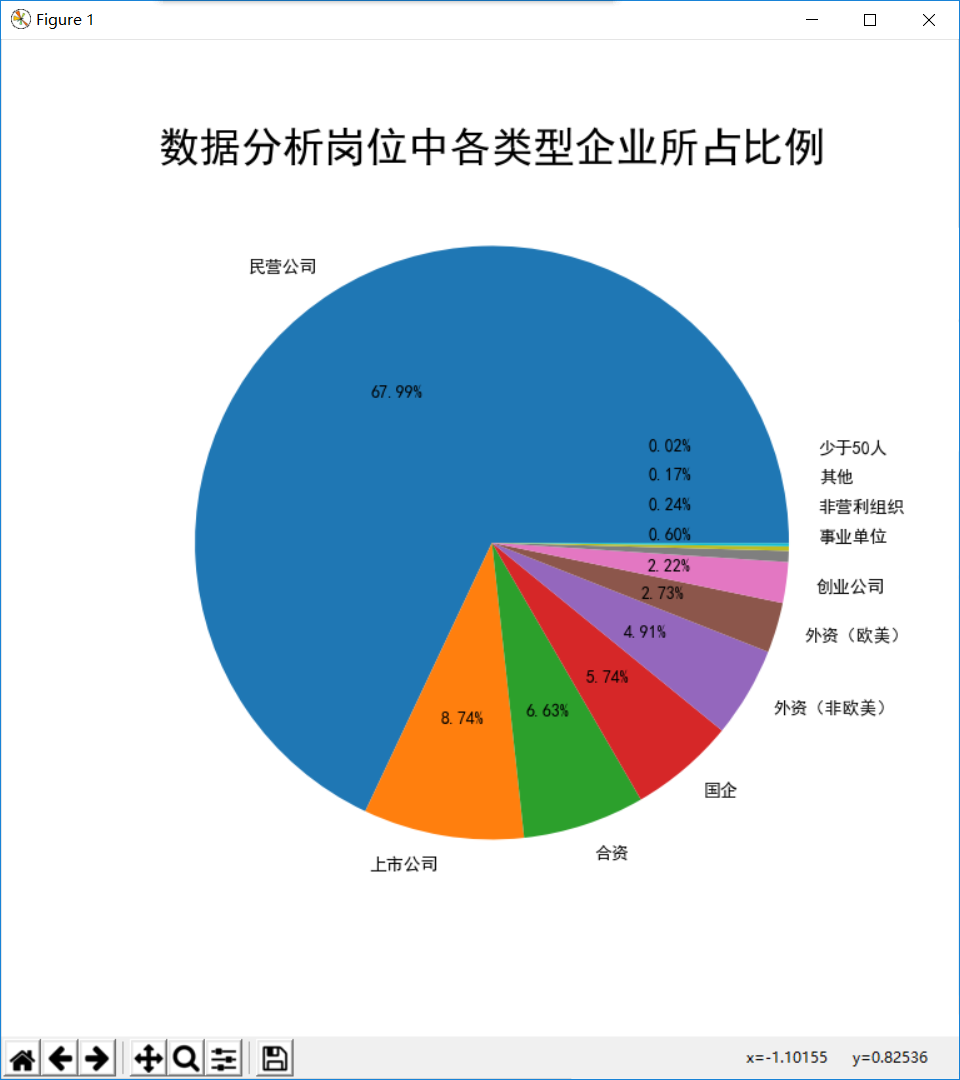
主要对处理加工后的招聘数据进行可视化分析，包括公司类型；公司规模；学历和工作经验；薪资水平；公司性质词云；福利词云；岗位职责词云；工作要求词云。以处理后的广东区的详细招聘数据为例，共计11448条招聘数据。

## 5.1 公司类型可视化分析

5.1.1 Python部分代码



5.1.2 可视化结果



5.1.3 小结

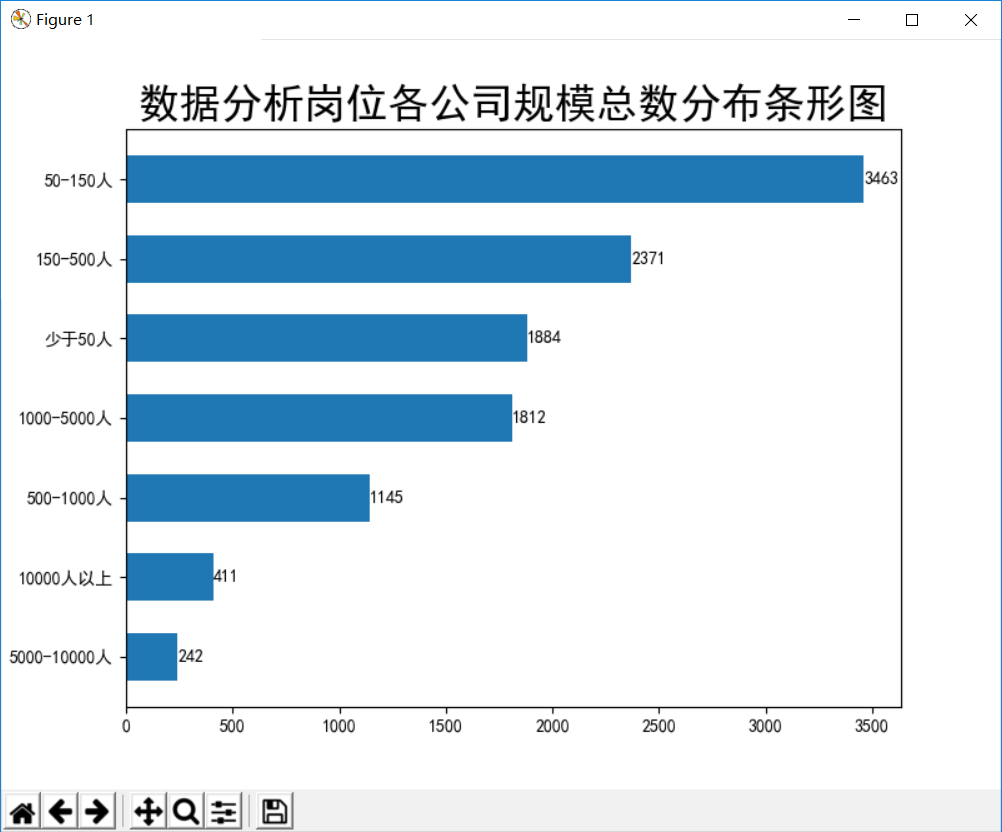
可以看出在(广东)招聘数据分析中，公司类型主要以民营公司为主，其次是上市公司、合资、国企等。

## 5.2 公司规模可视化分析

5.2.1 Python部分代码



5.2.2 可视化结果

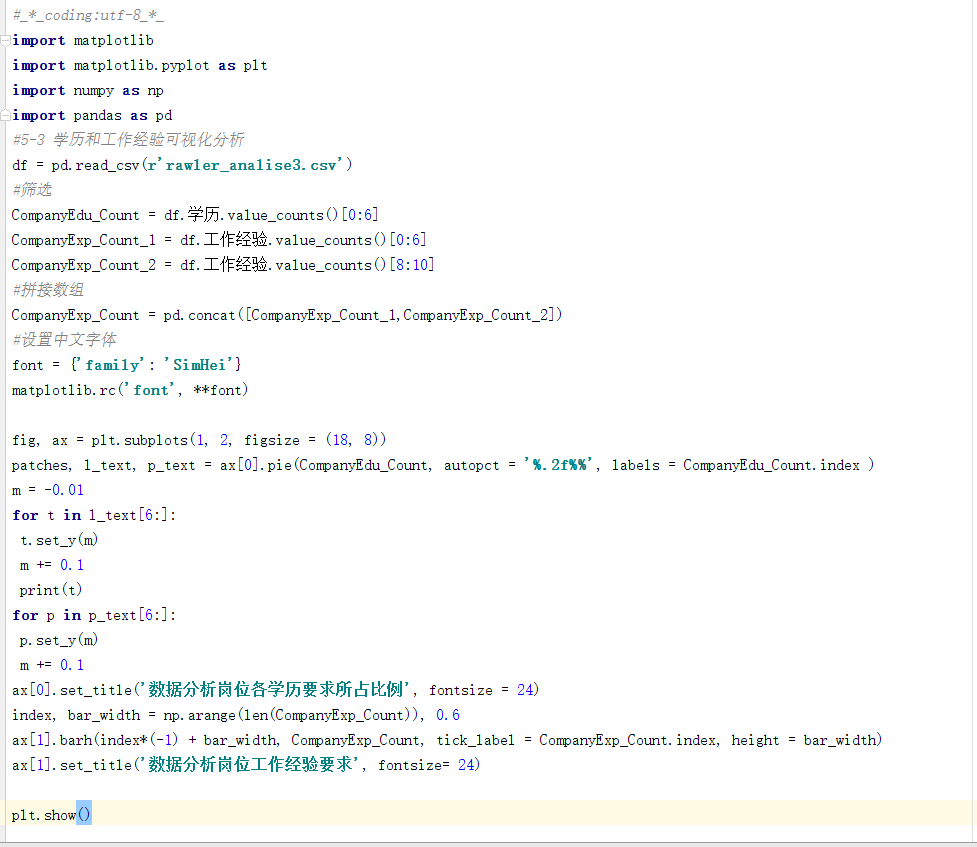


5.2.3 小结

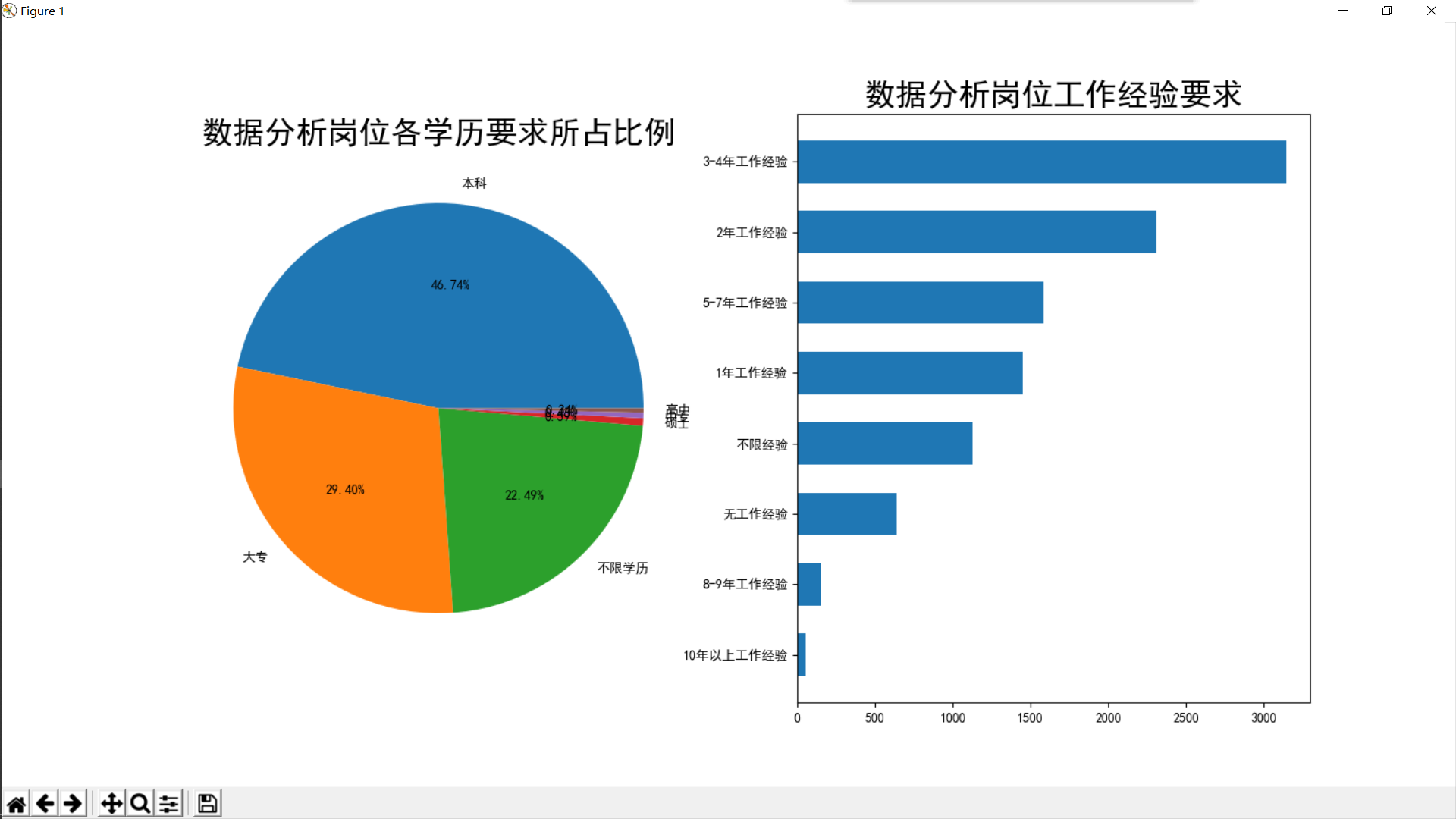
可以看出在(广东)招聘数据分析中，公司规模主要为150-500人为主，其次是50-150人，1000-5000人等。

## 5.3 学历和工作经验可视化分析

5.3.1 Python部分代码



5.3.2 可视化结果



5.3.3 小结

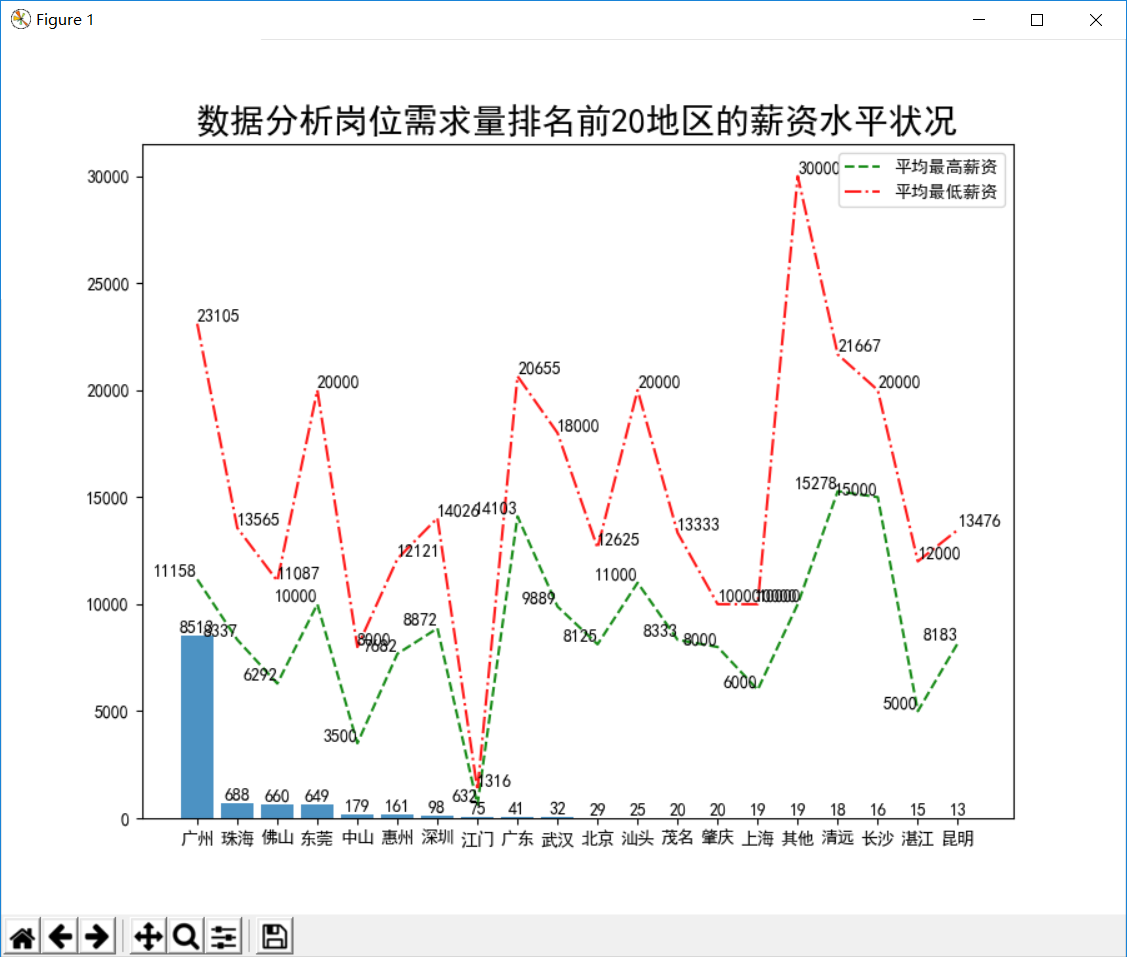
学历要求主要为本科以上、大专以上、不限学历为主。工作经验多要求为3-4年，其次为2年、5-7年、一年、不限经验等。

## 5.4 薪资与地区的关系分析

5.4.1 Python部分代码



5.4.2 可视化结果



5.4.3 小结

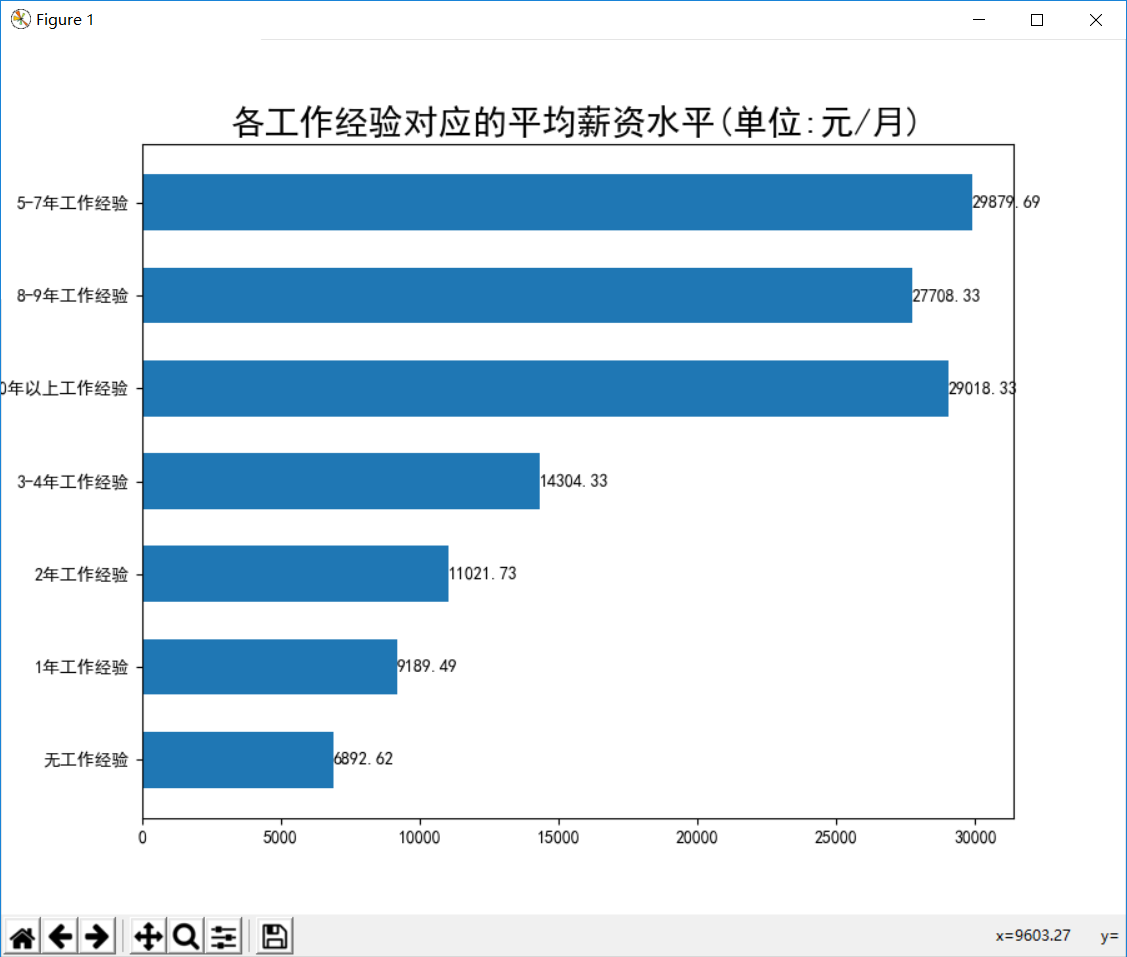
可以看出广州需求量最高，平均工资也是最高的，相对有发展空间，所以很多需求分析求职者都会选择去广州发展，其次是广东的各个地区占一定比重，工资相对广州低一点，在大城市尤其像北京，需求量很低，所以工资也不是很高，也没什么发展空间。

## 5.5 薪资与工作经验关系分析

5.5.1 Python部分代码



5.5.2 可视化结果



5.5.3 小结

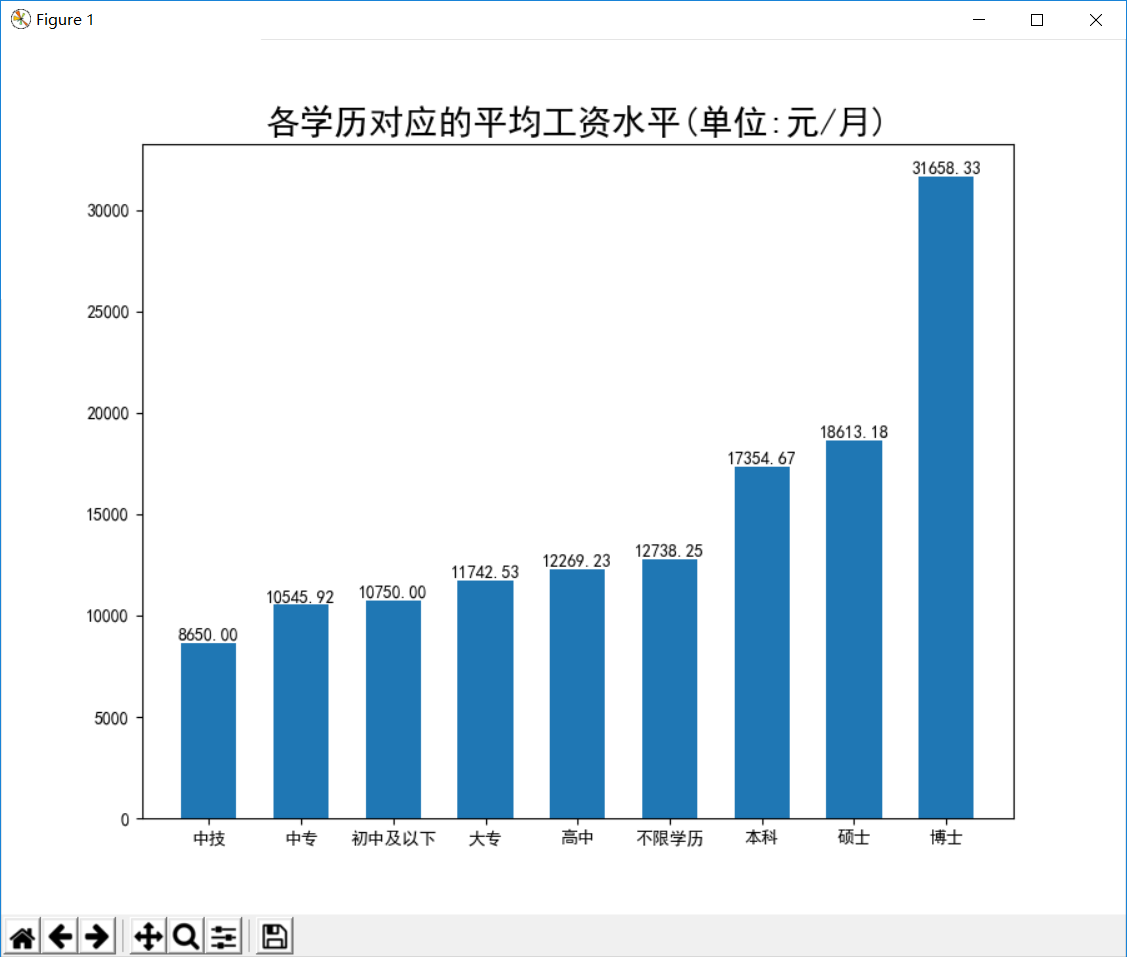
很明显，工作经验越多，工资会越高，在在5-7年达到高峰，随着年龄的增长，8-9年开始慢慢降，所以工作巅峰在5-7年。

## 5.6 薪资与学历的关系分析

5.6.1 Python部分代码



5.6.2 可视化结果

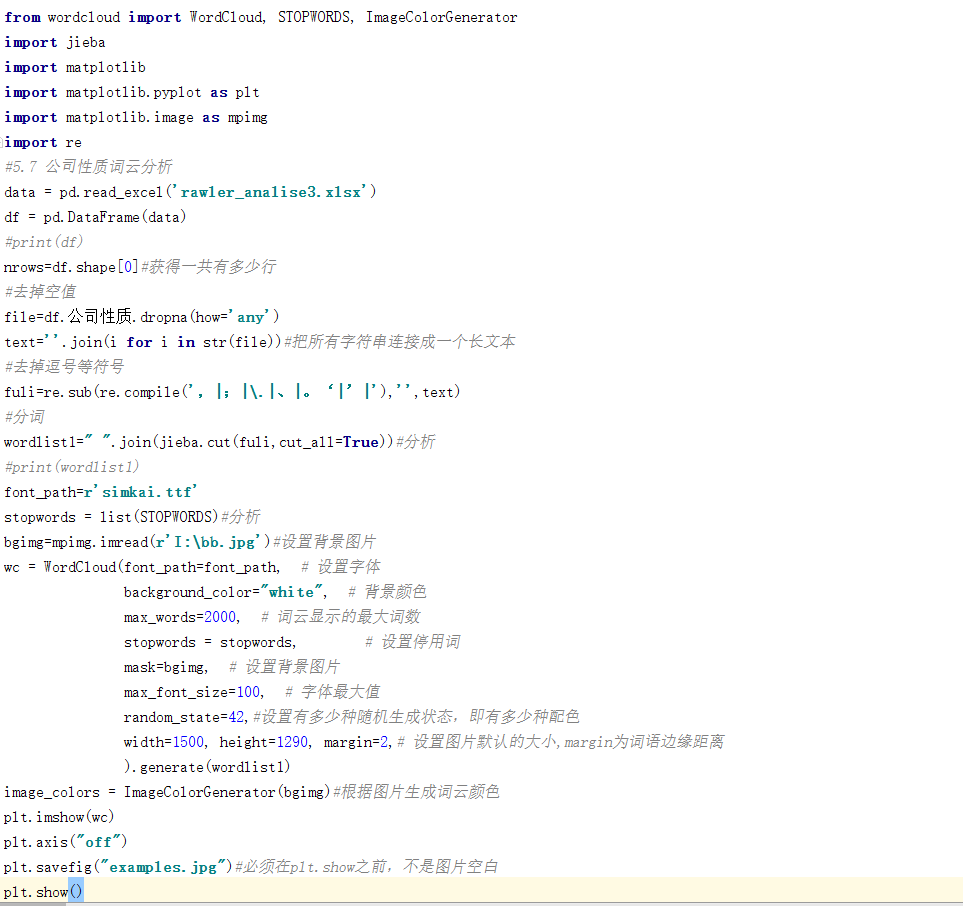


5.6.3 小结

从图可以看出，学历越高，工资越高，所以读书很重要。博士毕业平均达到三万多，其次是硕士，本科等。

## 5.7 公司性质词云分析

5.7.1 Python部分代码



5.7.2 词云可视化结果

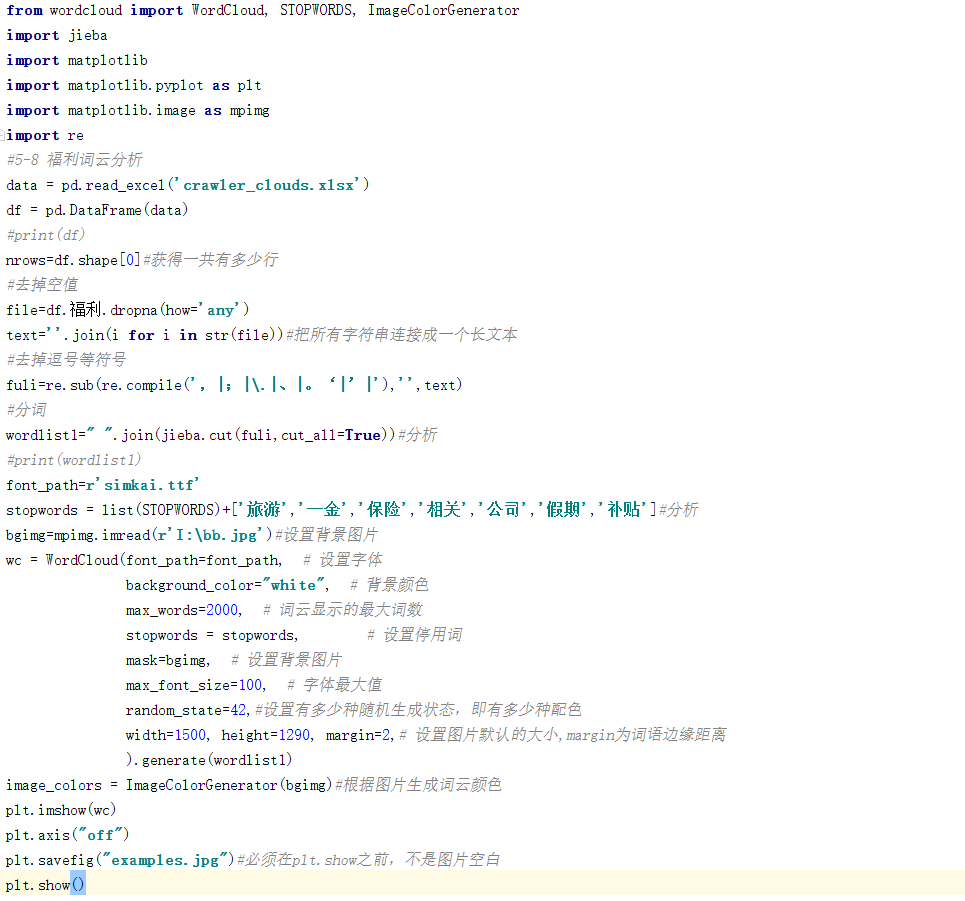


5.7.3 小结

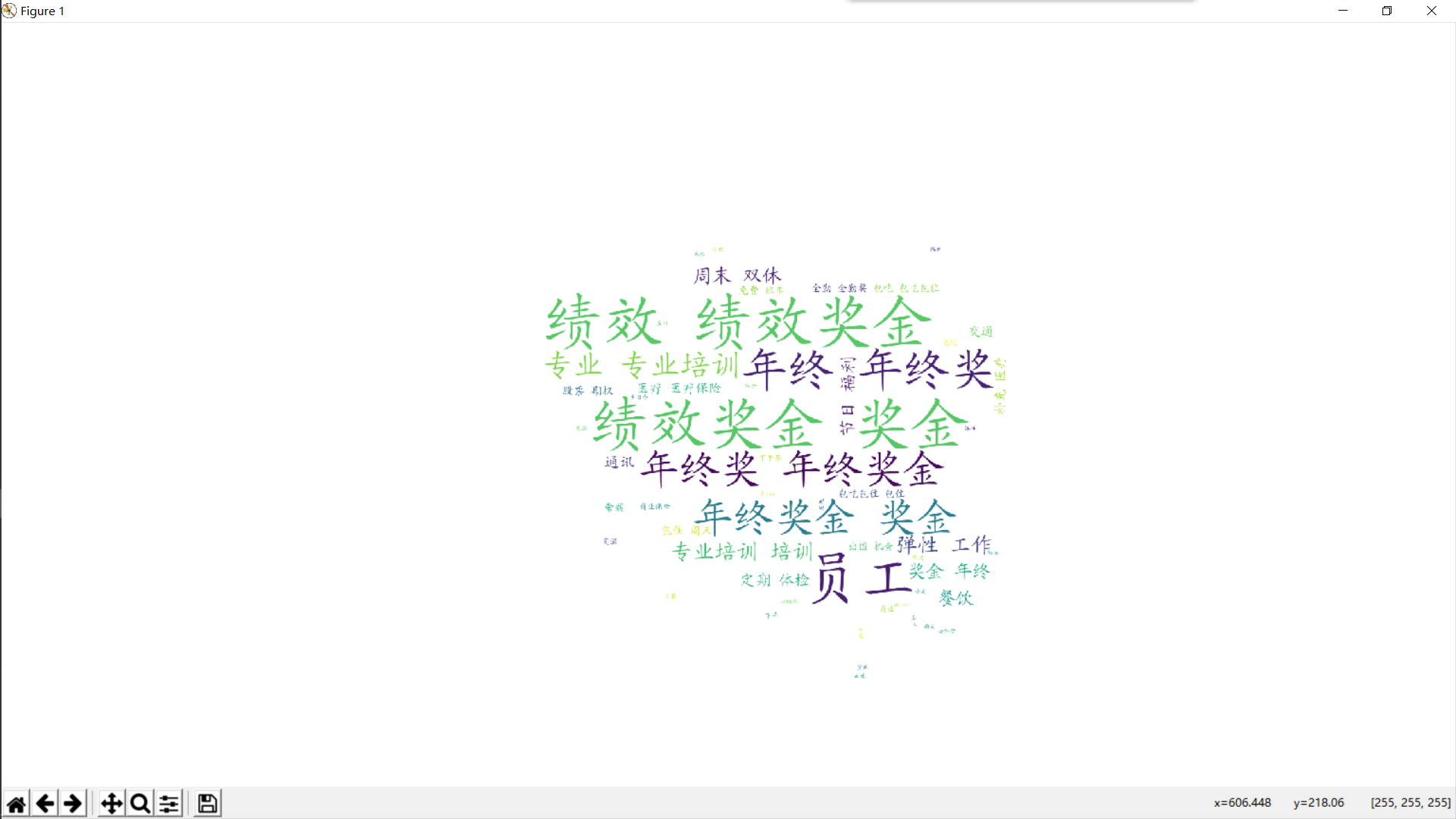
可以看出跟计算机有关的公司才需要java工程师，也跟互联网有关。

## 5.8 福利词云分析

5.8.1 Python部分代码



5.8.2 词云可视化结果

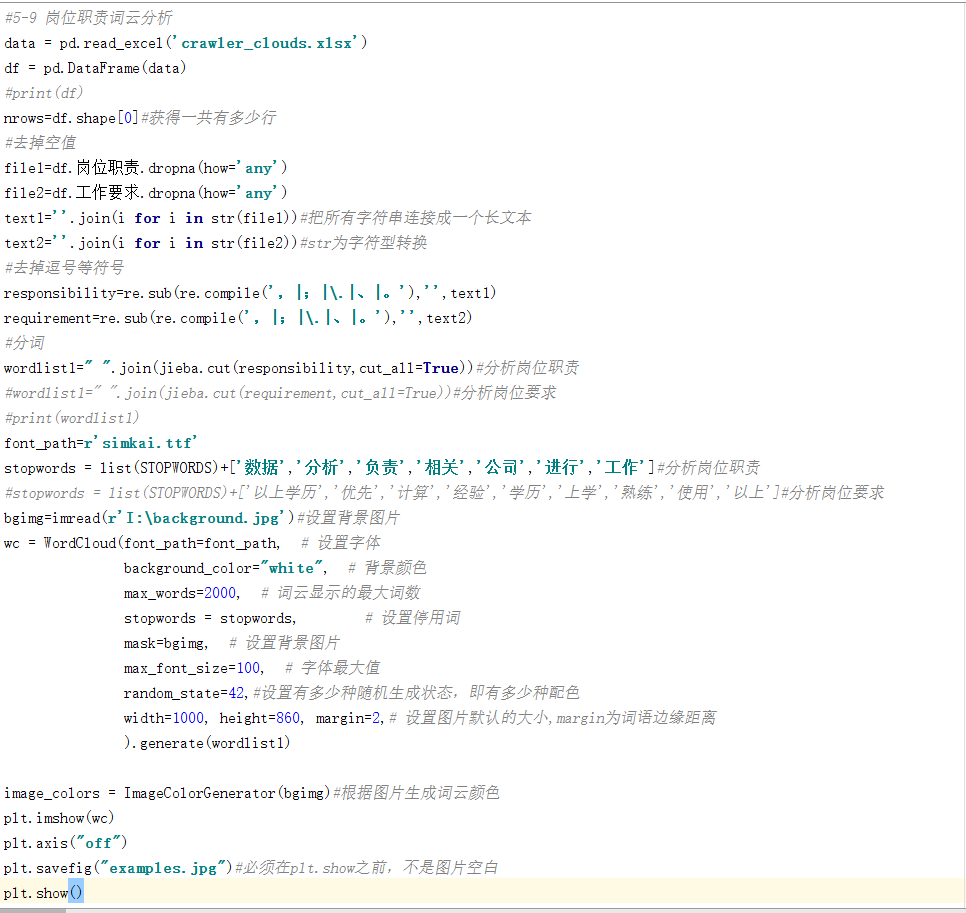


5.8.3 小结

对java工程师的福利跟绩效挂钩，福利有绩效奖金，像年终奖、年终奖金等。

## 5.9 岗位职责词云分析

5.9.1 Python部分代码



5.9.2 词云可视化结果

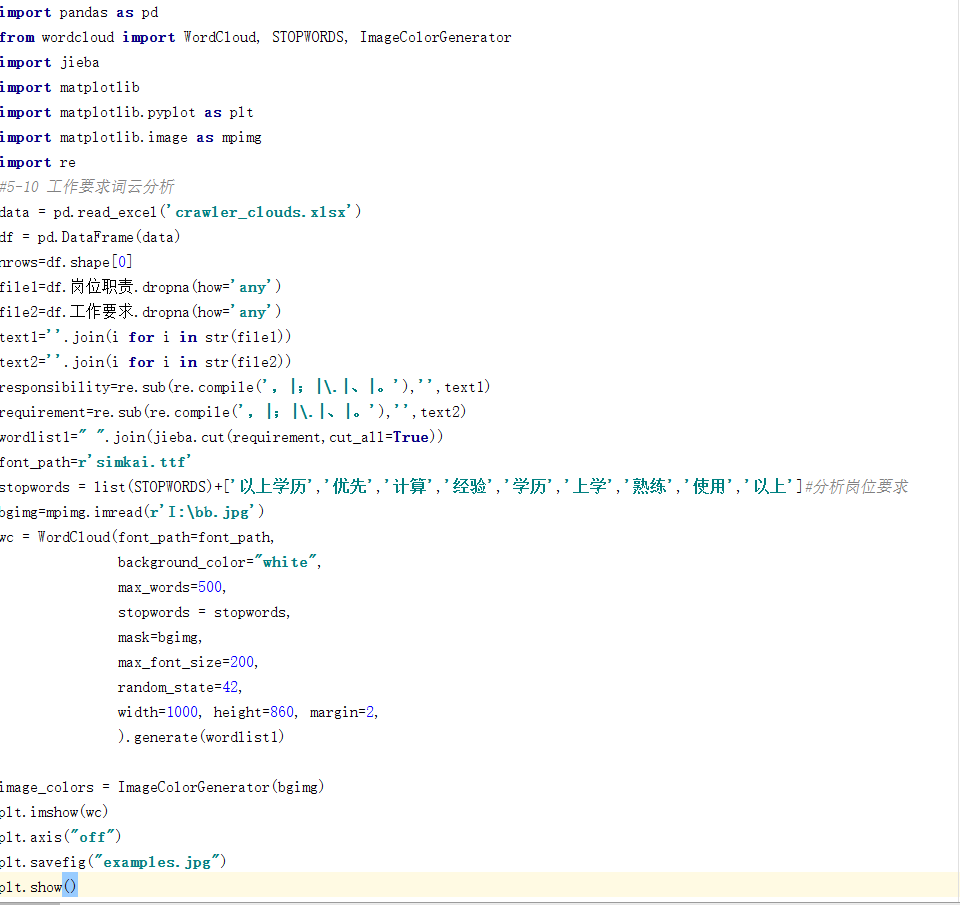


5.9.3 小结

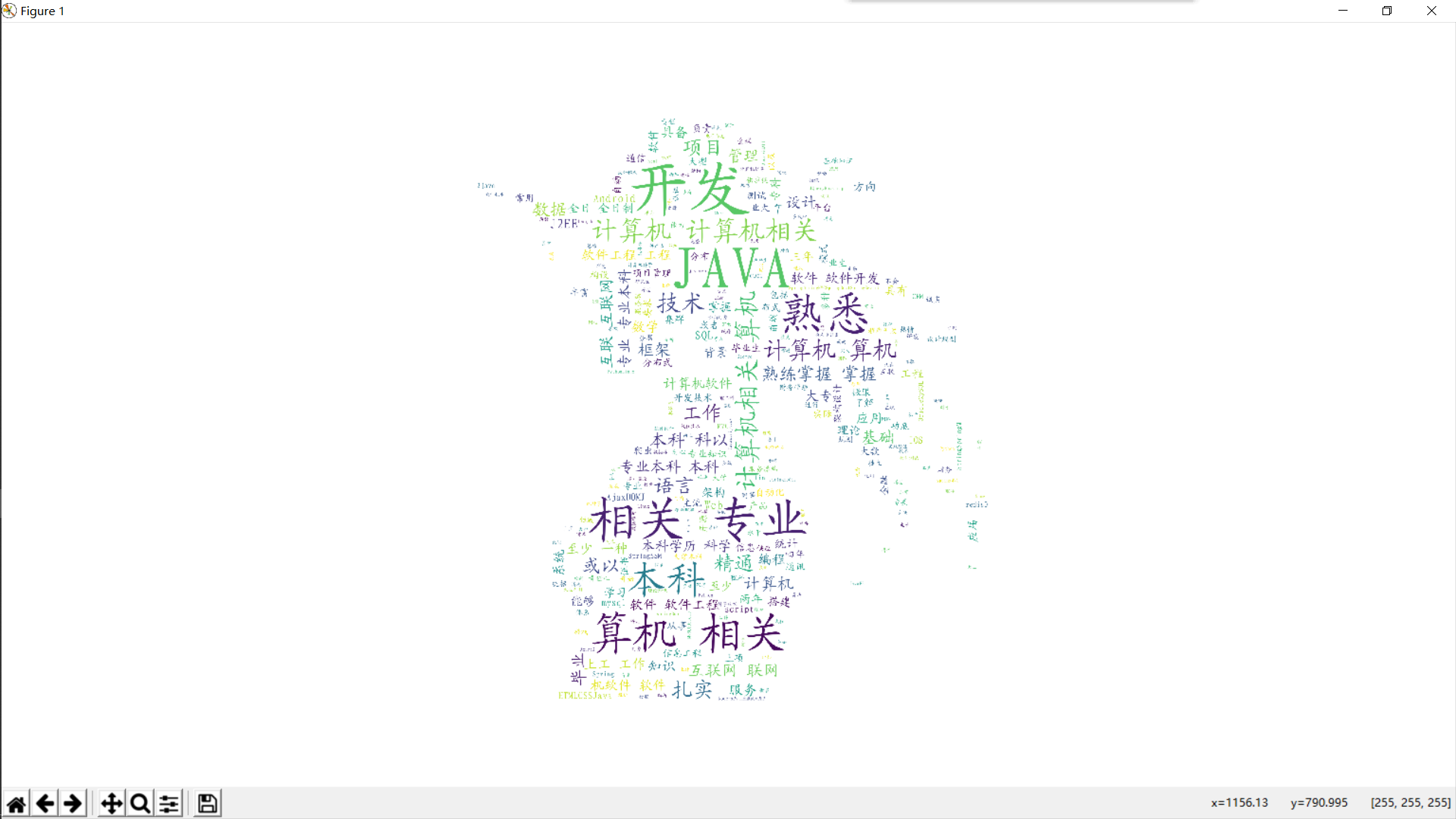
Java相关工作人员负责开发、设计系统架构、研发产品、管理项目要求、管理团队。

## 5.10 工作要求词云分析

5.10.1 Python部分代码



5.10.2 词云可视化结果



5.10.3 小结

从事java开发工作要求熟悉java开发技术、计算机技术、要求是计算机相关专业、大多数要求本科以上。

# 

# 第六章 总结

## 6.1 招聘信息总结

通过爬取了数据分析岗位的地区需要、工作经验与该岗位的关系、学历上与该岗位的关系、以及该岗位的工作职责、工作应聘要求、还有各企业对该岗位的需求、各个城市的平均薪资情况等，可以看出在广州和广东的一些其他地区待遇比较好、工资比较高、发展空间较大，其他地区工资普遍在一万左右，这些城市基本达到饱和状态，需求比较低，所以发展空间较小，所以才会有更多的人选择在广州发展 。

## 6.2 数据分析结论的可靠性(不足)

因为只是爬取了一个求职网站的一部分信息，因此不能保证一定的准确性，具有一定的片面性。