**Python商业数据可视化**

目录

[一、 分析目标 2](#_Toc44761653)

[二、 操作流程 2](#_Toc44761654)

[1 引用库 2](#_Toc44761655)

[2 使用爬虫进行数据爬取 2](#_Toc44761656)

[ Requests爬虫获取源码并解析 4](#_Toc44761657)

[ 文本储存 4](#_Toc44761658)

[三、 数据文本 5](#_Toc44761659)

[3 处理分割误差 7](#_Toc44761660)

[4 生成词云并统计词频 8](#_Toc44761661)

[5 可视化 9](#_Toc44761662)

[四、 生成的图表和词云 11](#_Toc44761663)

[1 地区景点数占比饼图（非重点） 11](#_Toc44761664)

[2 景区类型热度条形图（非重点） 12](#_Toc44761665)

[3 词云图 13](#_Toc44761666)

[4 欢乐谷类景区定价、销量和成交量（重点） 13](#_Toc44761667)

[四、 数据分析结论 16](#_Toc44761668)

[五、 缺点和问题 17](#_Toc44761669)

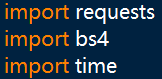
[六、 程序源码 18](#_Toc44761670)

# 分析目标

对“去哪旅行网”中的热门景点进行爬取，获取的内容包括：景区名称、景区价格、景区地点和景点门票月销量，并且检查各景区定价是否合理，给出优化方案。

# 操作流程

## 引用库



requests为爬虫本体，bs4为解析网页标签库，time为实现时间间隔功能的库。

## 使用爬虫进行数据爬取

经过检查网站，热门景点页面中只有前14页为有效数据，故爬取前14页景点标题，14个网址的唯一区别是网址末尾的接口：“”，故用循环语句即可遍历所有网页，在程序中我们遍历14页：

景区名称标签如下：

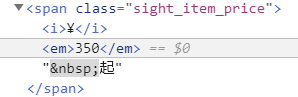


故使用bs4中的find\_all方法：

(解析后的源码).find\_all('a',attrs={'class':'name'})

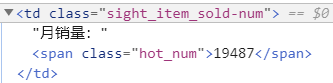
爬取景点位置、景点价格和景点门票月销量同理。

价格标签为：



bs4的方法为：.find\_all('span',attrs={'class':'sight\_item\_price'})

月销量标签：



bs4的方法为：.find\_all('span',attrs={'class':'hot\_num'})

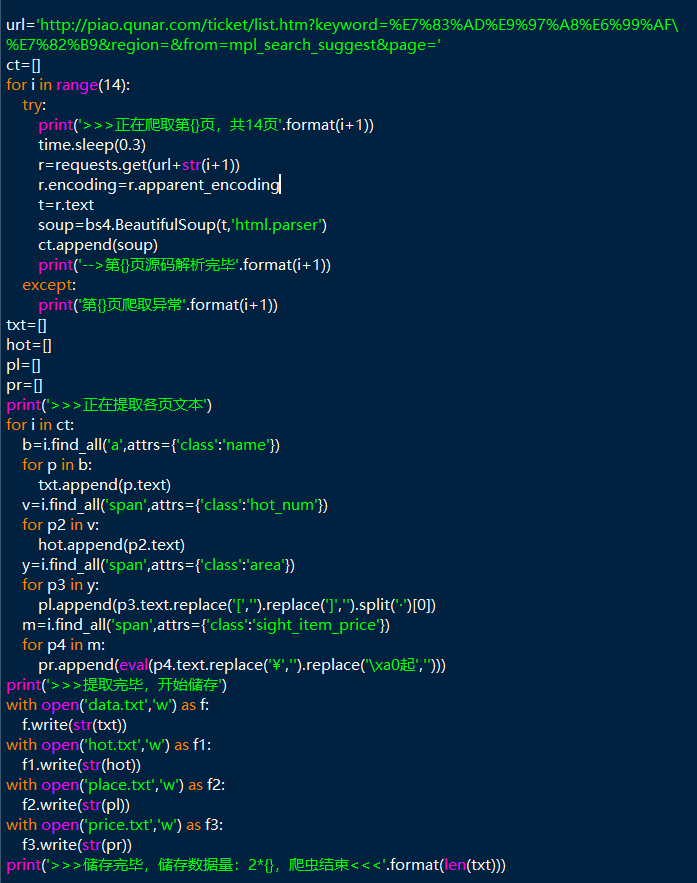
地点标签为：



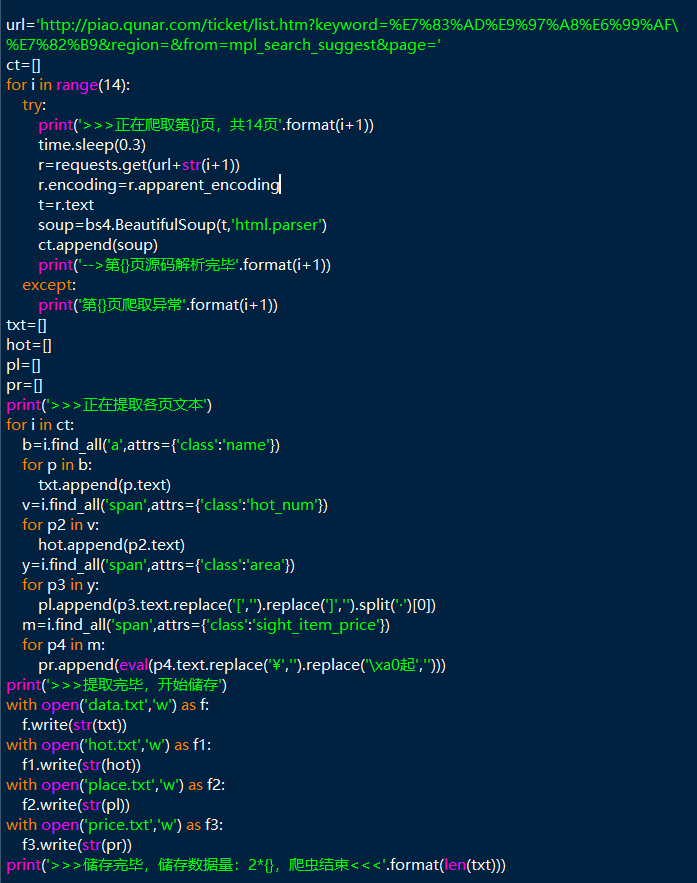
bs4的方法为：.find\_all('span',attrs={'class':'area'})

爬虫部分代码如下：

### Requests爬虫获取源码并解析



### 文本储存

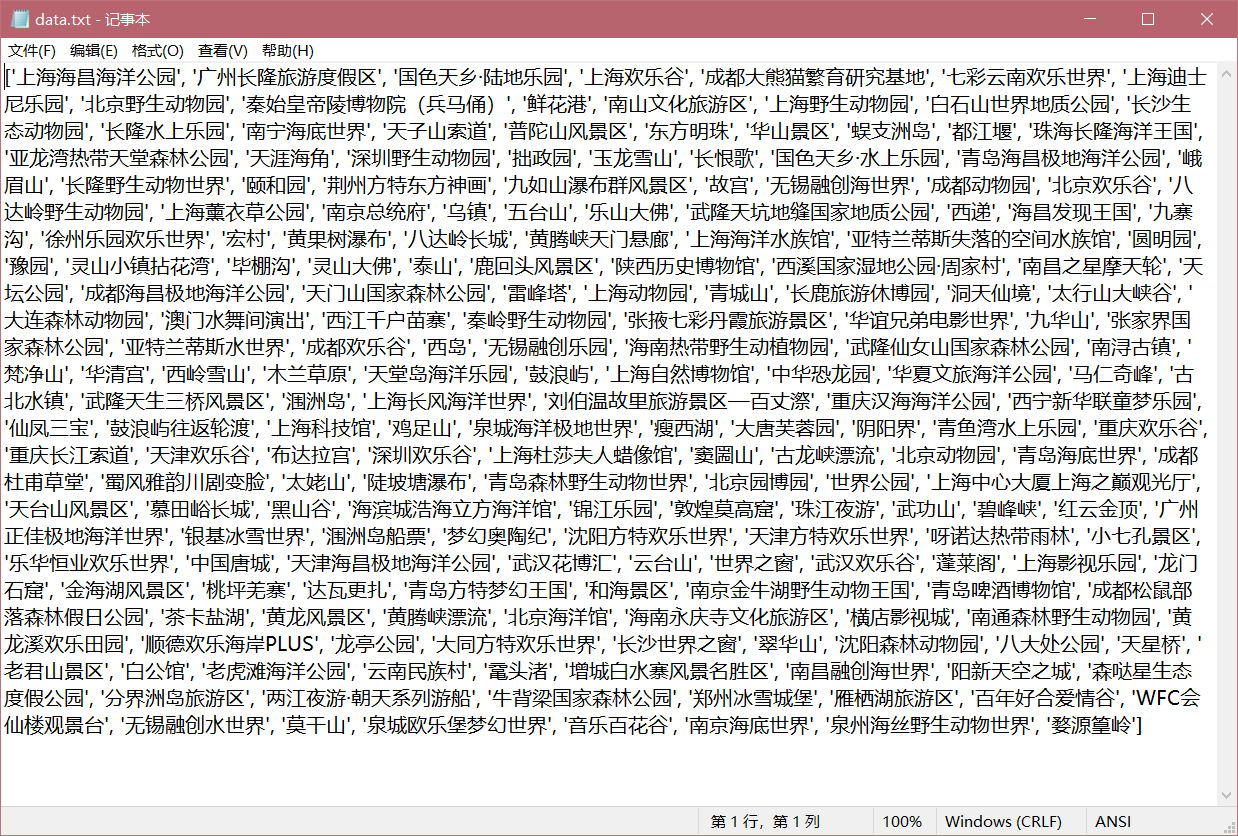


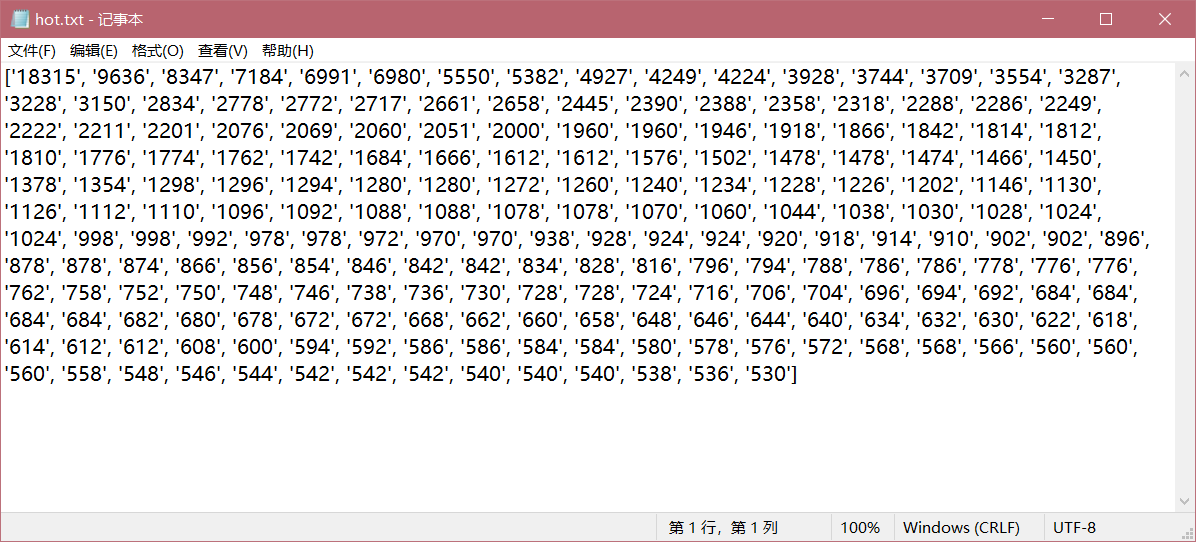
获取到的文本内容不够标准，比如获取价格的文本为“售价156￥起”，要把前后不相关的文字内容去掉，用replace方法替换，而地点为诸如“[成都·四川]”的，我们只取城市名称，用replace切割掉中括号split方法后再用列表的索引方法即可。

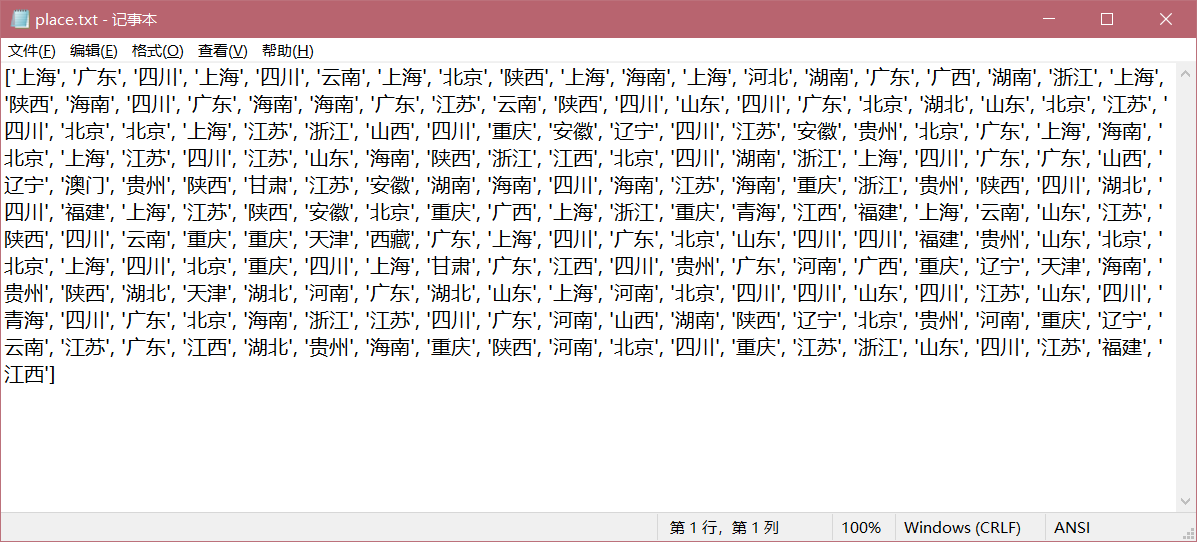
储存了各景区名称、月销量和所在地区，其中景区名称保存在变量txt中，月销量储存在hot中，地区储存在pl中，并分别储存在程序根目录的“data”、“hot”和“place”文本文件中。

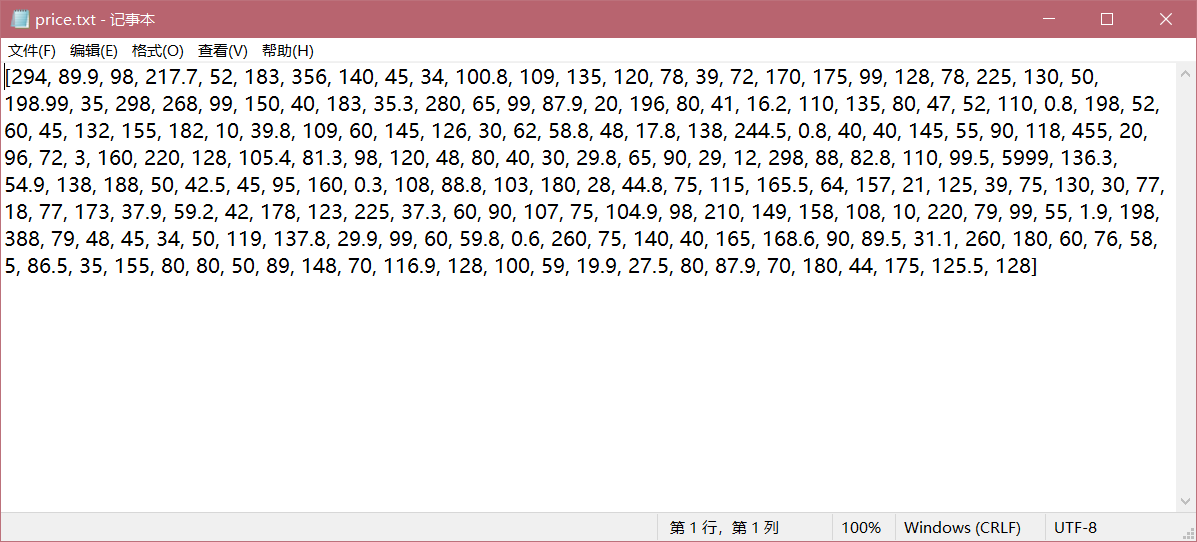
# 数据文本

上述储存完毕的文本文件的内容展示：

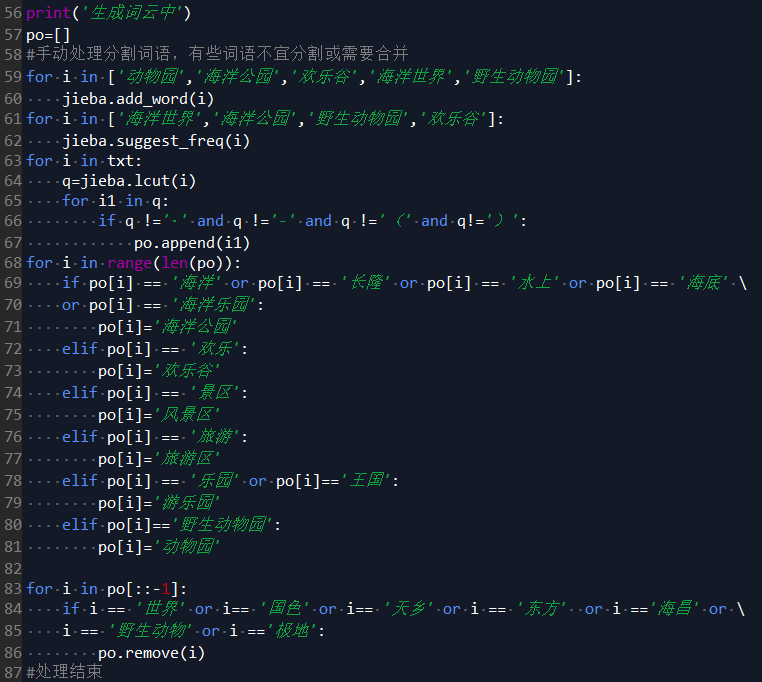






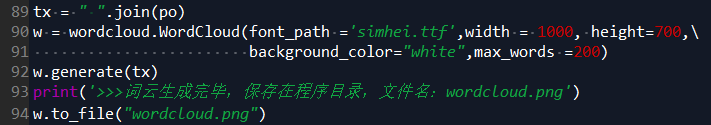


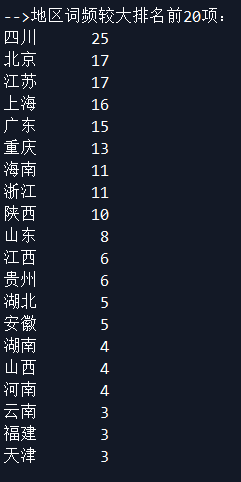
## 处理分割误差

由于切割后的词语存在一些误差，故手工合并了一些近义词，删除了一些干扰词，调整方法如下：

其中添加词语'动物园','海洋公园','欢乐谷','海洋世界','野生动物园'，不可分割词语'海洋世界','海洋公园','野生动物园','欢乐谷'，词语替换直接参考代码68行至81行。

## 生成词云并统计词频

生成词云部分：

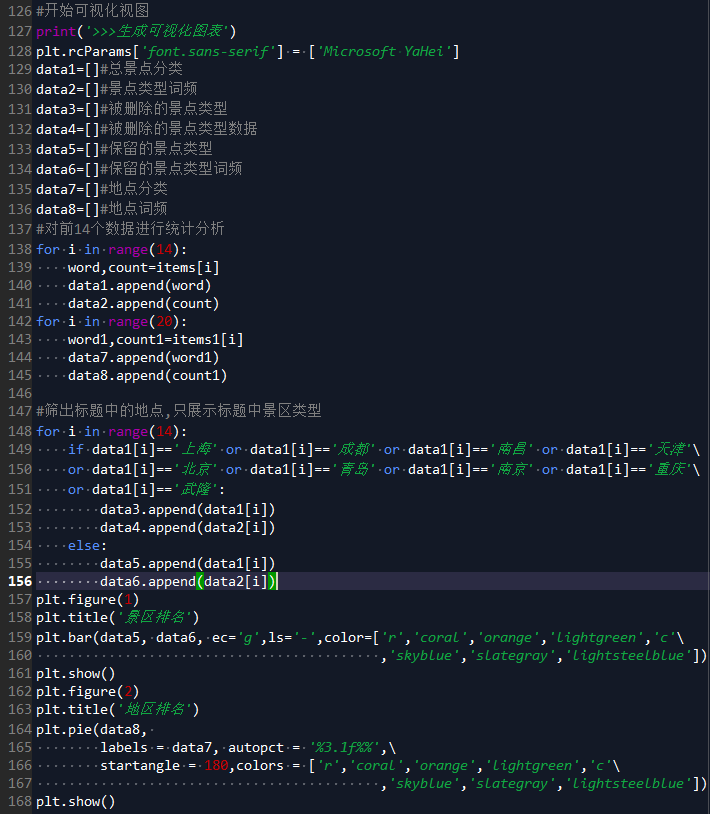
用处理之后的文本生成词云并且统计词频，取出前20项输出展示：  

其完整代码部分如下：



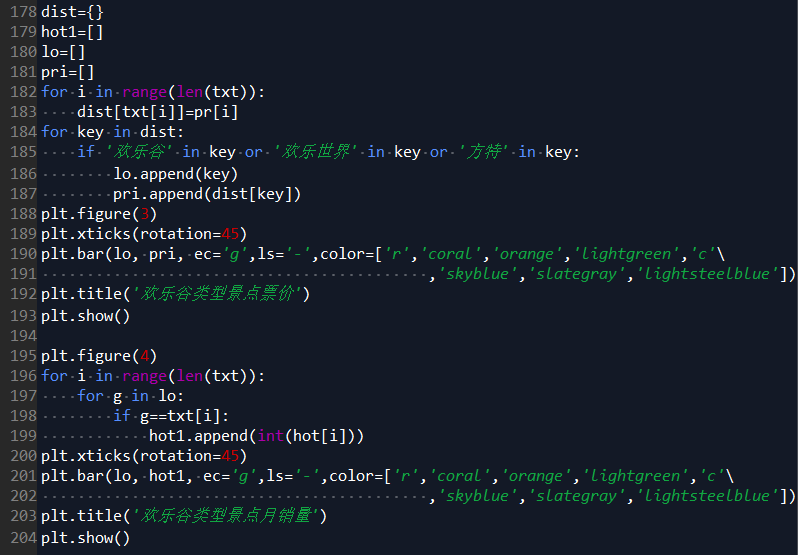
## 可视化

将地名手动输入判断条件中，取出所有城市和对应的词频，剩下的即为景点类型的名称和对应词频，只展示景点类型的名称和对应词频即可。但在此没有删除掉标题中的地点词语，而是存到了另外的列表中。



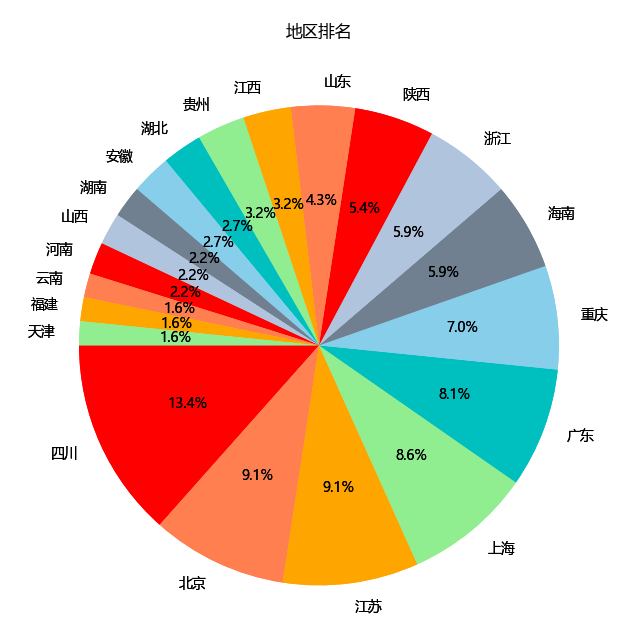
利用饼图表示每个城市中热门景点的占比，条形图表示不同类型景点的热门程度。

最后对欢乐谷类型景点进行销量-价格分析。

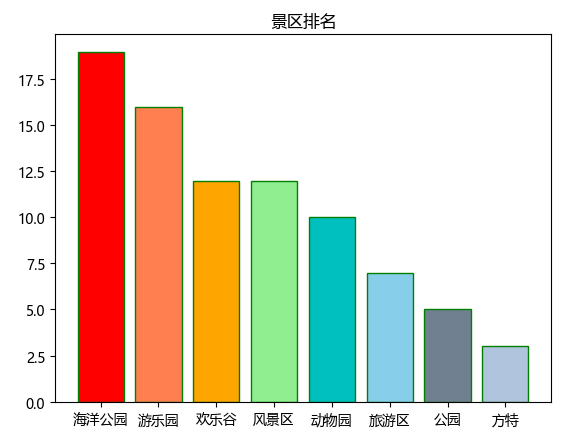


# 生成的图表和词云

## 地区景点数占比饼图（非重点）



## 景区类型热度条形图（非重点）



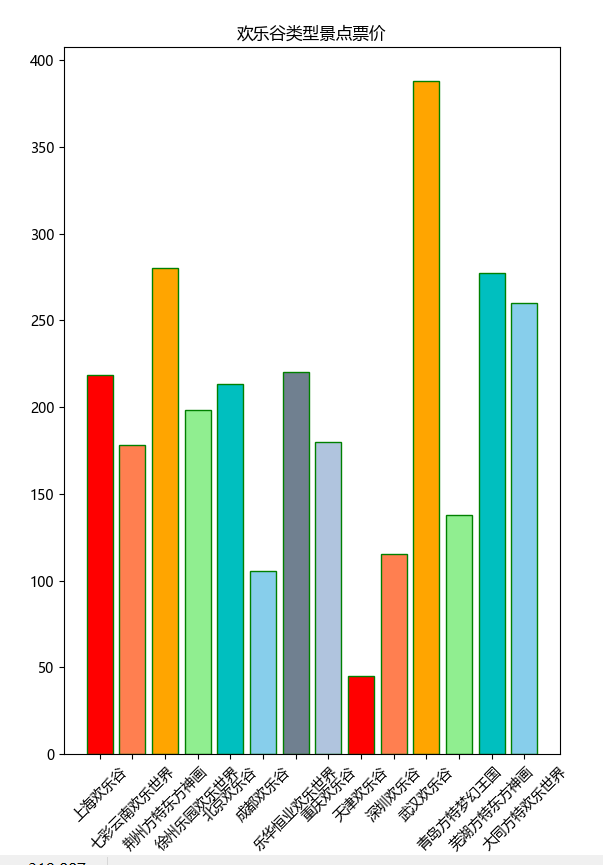
其中方特、欢乐谷、游乐园性质较为近似，但这不是分析重点，在这不做进一步处理。后续将细分欢乐谷类型景点的销量和价格。

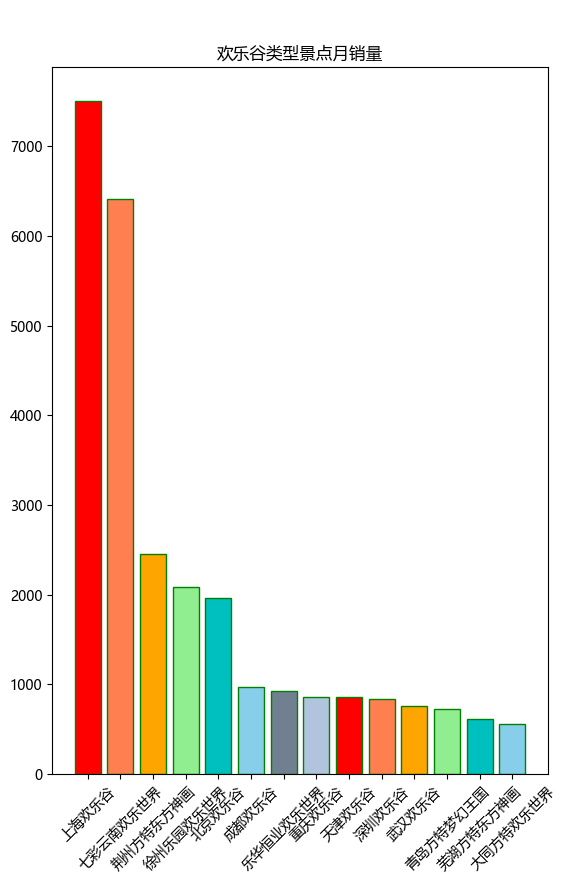
## 词云图



## 欢乐谷类景区定价、销量和成交量（重点）

注意：此处欢乐谷类景区包括 欢乐谷、欢乐世界、方特这几个游乐园，票价为成人票。





成交量图

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 月销量 | 价格 | 成交量 | 成交量排名 |
| 上海欢乐谷 | 7508 | 218.3 | 1638996 | 1 |
| 七彩云南欢乐世界 | 6418 | 178 | 1142404 | 2 |
| 荆州方特东方神画 | 2451 | 280 | 686280 | 3 |
| 徐州乐园欢乐世界 | 2082 | 198 | 412236 | 5 |
| 北京欢乐谷 | 1960 | 213 | 417480 | 4 |
| 成都欢乐谷 | 972 | 105.4 | 102448.8 | 11 |
| 乐华恒业欢乐世界 | 926 | 220 | 203720 | 7 |
| 重庆欢乐谷 | 862 | 180 | 155160 | 9 |
| 天津欢乐谷 | 854 | 44.8 | 38259.2 | 14 |
| 深圳欢乐谷 | 840 | 115 | 96600 | 13 |
| 武汉欢乐谷 | 754 | 388 | 292552 | 6 |
| 青岛方特梦幻王国 | 722 | 138 | 99636 | 12 |
| 芜湖方特东方神画 | 612 | 277 | 169524 | 8 |
| 大同方特欢乐世界 | 558 | 260 | 145080 | 10 |

# 数据分析结论

北京、上海、重庆的热门景点数量多，并且比较热门，可能是因为这些直辖市的基础设施完善、公共服务水平高，并且人口密集，故具有最多的旅游景点和最高的旅游热度，四川省的热门景点数量最多，四川地区有盆地和高原、山地等，景点多样化，且高原地区适合夏天避暑。

在景点类型方面，海洋世界和游乐园的热度最高，一方面是天气逐渐炎热，人们倾向于去海洋世界追求清凉的感觉，二是疫情过后，出游人数不多，游乐园比平时的游玩效率更高，所以游乐园也很受追捧。

随着天气逐渐炎热，建议适当调低海洋世界的价格，吸引更多的游客前来游玩；游乐场可以创新销售方案，例如出售季卡或年卡。各个旅游城市可以加强基础设施建设，提高景区热度，如果辖区内有合适的避暑景点，可以加大宣传力度，并且投入资金进行完善，可以有效在夏天增加客流量。

从供给需求角度看，一般来说价格越高，需求越少。从销量和价格的图表中可以明显发现，武汉欢乐谷的价格最高，销量很低，武汉疫情刚结束不久，票价仍然保持高位，人们的储蓄难以接受这样么高的娱乐成本，从成交量上看，武汉欢乐谷的成交量排名第六，仅次于北京欢乐谷，所以成交量上看并不低，猜测的一个解释是由于目前武汉人前往欢乐谷的意愿不高，不如维持高价保持成交量，待市民娱乐意愿高时再适当降价。

与武汉欢乐谷类似的是荆州方特东方神话，一个是地理位置相近，受疫情影响情况类似，二是价格也在较高的位置，但是荆州方特东方神话的月销量大于武汉欢乐谷，收入也更高，因此，武汉欢乐谷的定价还有待调整。

# 缺点和问题

对景区类型的分类还存在问题，很难从景区名字上采用一个有效方案来区分景点的类型。

代码部分没有模块化编程，而是从头写到尾，难以维护和修改。

# 程序源码

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

"""

Created on Mon Jun 8 22:59:54 2020

@author: LiSunBowen

"""

import requests

import bs4

import wordcloud

import jieba

import time

import matplotlib.pyplot as plt

url1='http://piao.qunar.com/ticket/list.htm?keyword=%E7%83%AD%E9%97%A\

8%E6%99%AF%E7%82%B9&region=&from=mpl\_search\_suggest&sort=pp&page='

#爬虫开始

ct=[]

for i in range(14):

try:

print('>>>正在爬取第{}页,共14页'.format(i+1))

time.sleep(0.3)

r=requests.get(url1+str(i+1))

r.encoding=r.apparent\_encoding

t=r.text

soup=bs4.BeautifulSoup(t,'html.parser')

ct.append(soup)

print('-->第{}页源码解析完毕'.format(i+1))

except:

print('第{}页爬取异常'.format(i+1))

txt=[]

hot=[]

pl=[]

pr=[]

print('>>>正在提取各页文本')

for i in ct:

b=i.find\_all('a',attrs={'class':'name'})

for p in b:

txt.append(p.text)

v=i.find\_all('span',attrs={'class':'hot\_num'})

for p2 in v:

hot.append(p2.text)

y=i.find\_all('span',attrs={'class':'area'})

for p3 in y:

pl.append(p3.text)

m=i.find\_all('span',attrs={'class':'sight\_item\_price'})

for p4 in m:

pr.append(eval(p4.text.replace('¥','').replace('\xa0起','')))

print('>>>提取完毕，开始储存')

with open('C:/Users/LiSunBowen/Desktop/商业数据可视化/data.txt','w') as f:

f.write(str(txt))

with open('C:/Users/LiSunBowen/Desktop/商业数据可视化/hot.txt','w') as f1:

f1.write(str(hot))

with open('C:/Users/LiSunBowen/Desktop/商业数据可视化/place.txt','w') as f2:

f2.write(str(pl))

with open('C:/Users/LiSunBowen/Desktop/商业数据可视化/price.txt','w') as f3:

f3.write(str(pr))

print('>>>储存完毕,储存位置：C:/Users/LiSunBowen/Desktop/商业数据可视化，储存数据量:\

2\*{},爬虫结束<<<'.format(len(txt)))

#爬虫部分结束

print('\n>>>生成词云中')

po=[]

#手动处理分割词语，有些词语不宜分割或需要合并

for i in ['动物园','海洋公园','欢乐谷','海洋世界','野生动物园']:

jieba.add\_word(i)

for i in ['海洋世界','海洋公园','野生动物园','欢乐谷']:

jieba.suggest\_freq(i)

for i in txt:

q=jieba.lcut(i)

for i1 in q:

if q !='·' and q !='-' and q !='（' and q!='）':

po.append(i1)

for i in range(len(po)):

if po[i] == '海洋' or po[i] == '长隆' or po[i] == '水上' or po[i] == '海底' \

or po[i] == '海洋乐园':

po[i]='海洋公园'

elif po[i] == '欢乐':

po[i]='欢乐谷'

elif po[i] == '景区':

po[i]='风景区'

elif po[i] == '旅游':

po[i]='旅游区'

elif po[i] == '乐园' or po[i]=='王国':

po[i]='欢乐谷'

elif po[i]=='野生动物园':

po[i]='动物园'

for i in po[::-1]:

if i == '世界' or i== '国色' or i== '天乡'or i=='东方' or i=='海昌' or\

i=='野生动物' or i == '极地':

po.remove(i)

#处理结束

tx = " ".join(po)

w = wordcloud.WordCloud(font\_path ='simhei.ttf',width = 1000, height=700,\

background\_color="white",max\_words =200)

w.generate(tx)

print('>>>词云生成完毕，保存在程序目录，文件名：wordcloud.png')

w.to\_file("wordcloud.png")

print('>>>开始统计词频')

counts={}

for word in po:

if len(word)==1:

continue

else:

counts[word] = counts.get(word,0)+1

items = list(counts.items())

items.sort(key=lambda x:x[1],reverse=True)

print('-->景点词频较大排名前{:2}项：'.format(20))

for i in range(20):

word,count = items[i]

print('{0:<5}{1:>5}'.format(word,count))

print()

counts1={}

npl=[]

for i in pl:

npl.append(i.replace('[','').replace(']','').split('·')[0])

for word in npl:

counts1[word] = counts1.get(word,0)+1

items1 = list(counts1.items())

items1.sort(key=lambda x:x[1],reverse=True)

print('-->地区词频较大排名前{:2}项：'.format(20))

for i in range(20):

word1,count1 = items1[i]

print('{0:<5}{1:>5}'.format(word1,count1,))

print()

#词频统计结束

#开始可视化视图

print('>>>生成可视化图表')

plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']

data1=[]#总景点分类

data2=[]#景点类型词频

data3=[]#被删除的景点类型

data4=[]#被删除的景点类型数据

data5=[]#保留的景点类型

data6=[]#保留的景点类型词频

data7=[]#地点分类

data8=[]#地点词频

#对前14个数据进行统计分析

for i in range(14):

word,count=items[i]

data1.append(word)

data2.append(count)

for i in range(20):

word1,count1=items1[i]

data7.append(word1)

data8.append(count1)

#筛出标题中的地点,只展示标题中景区类型

for i in range(14):

if data1[i]=='上海' or data1[i]=='成都' or data1[i]=='南昌' or data1[i]=='天津'\

or data1[i]=='北京' or data1[i]=='青岛' or data1[i]=='南京' or data1[i]=='重庆'\

or data1[i]=='武隆':

data3.append(data1[i])

data4.append(data2[i])

else:

data5.append(data1[i])

data6.append(data2[i])

plt.figure(1)

plt.title('景区排名')

plt.bar(data5, data6, ec='g',ls='-',color=['r','coral','orange','lightgreen','c'\

,'skyblue','slategray','lightsteelblue'])

plt.show()

plt.figure(2)

plt.title('地区排名')

plt.pie(data8,

labels = data7, autopct = '%3.1f%%',\

startangle = 180,colors = ['r','coral','orange','lightgreen','c'\

,'skyblue','slategray','lightsteelblue'])

plt.show()

dist={}

hot1=[]

lo=[]

pri=[]

for i in range(len(txt)):

dist[txt[i]]=pr[i]

for key in dist:

if '欢乐谷' in key or '欢乐世界' in key or '方特' in key:

lo.append(key)

pri.append(dist[key])

plt.figure(3)

plt.xticks(rotation=45)

plt.bar(lo, pri, ec='g',ls='-',color=['r','coral','orange','lightgreen','c'\

,'skyblue','slategray','lightsteelblue'])

plt.title('欢乐谷类型景点票价')

plt.show()

plt.figure(4)

for i in range(len(txt)):

for g in lo:

if g==txt[i]:

hot1.append(int(hot[i]))

plt.xticks(rotation=45)

plt.bar(lo, hot1, ec='g',ls='-',color=['r','coral','orange','lightgreen','c'\

,'skyblue','slategray','lightsteelblue'])

plt.title('欢乐谷类型景点月销量')

plt.show()