Python 商业数据可视化

目录

一、 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。

二、 操作流程

1 引用库

```
import requests
import bs4
import time
```

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。

2 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：“=pp&page=1”，故用循环语句即可遍历所有网页，在程序中我们遍历 14 页：

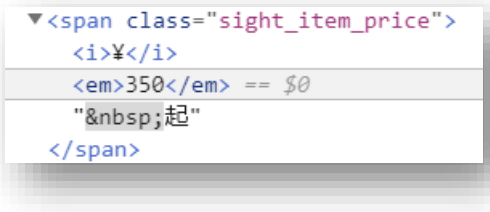
景区名称标签如下：

故使用 bs4 中的 find_all 方法：

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

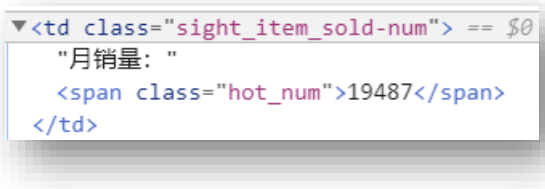
爬取景点位置、景点价格和景点门票月销量同理。

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

```
url='http://piao.qunar.com/ticket/list.htm?keyword=%E7%83%AD%E9%97%A8%E6%99%AF\
%E7%82%B9&region=&from=mpl_search_suggest&page='
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(url+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))
```

- 文本储存

```
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.replace('[','').replace(']','').split('·')[0])
    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('data.txt','w') as f:
    f.write(str(txt))
with open('hot.txt','w') as f1:
    f1.write(str(hot))
with open('place.txt','w') as f2:
    f2.write(str(pl))
with open('price.txt','w') as f3:
    f3.write(str(pr))
print('>>>储存完毕，储存数据量：2*{}，爬虫结束<<<'.format(len(txt)))
```

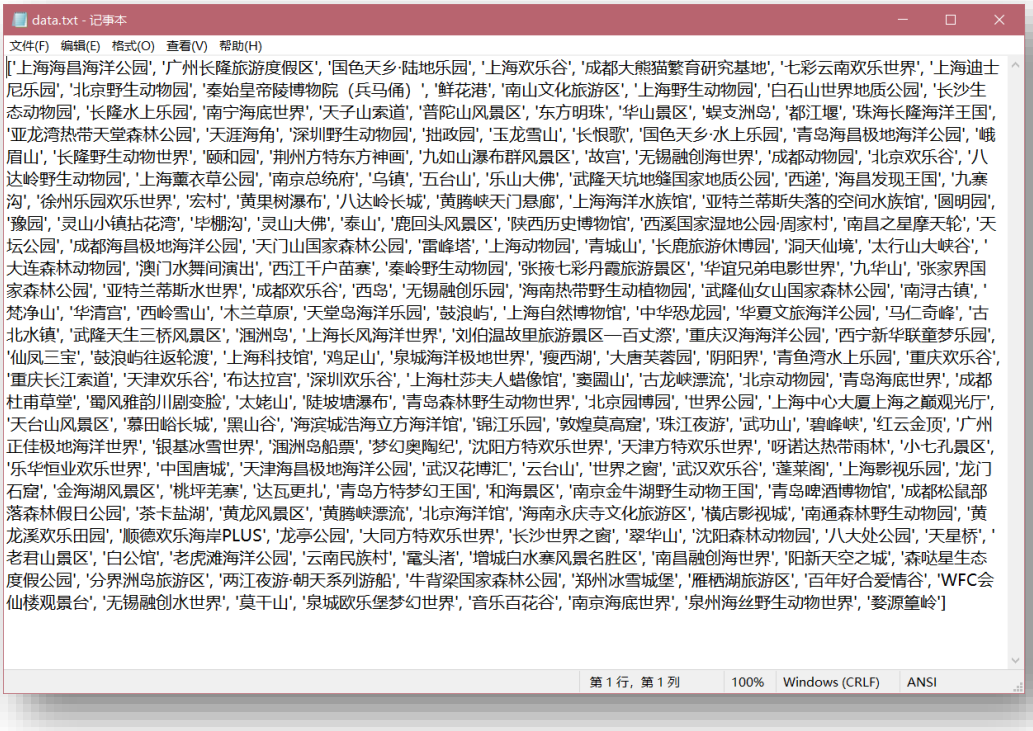
获取到的文本内容不够标准，比如获取价格的文本为“售价 156￥起”，要把

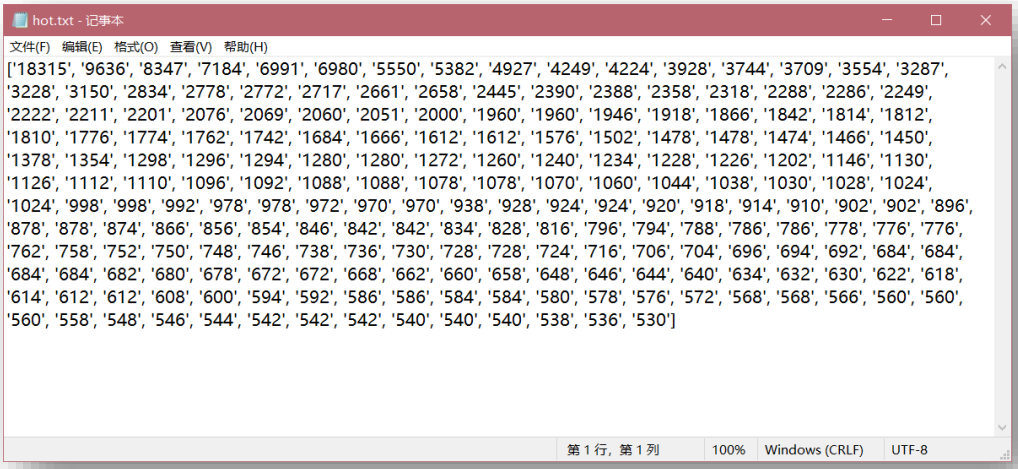
前后不相关的文字内容去掉，用 replace 方法替换，而地点为诸如“[成都·四川]”的，我们只取城市名称，用 replace 切割掉中括号 split 方法后再用列表的索引方法即可。

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。

三、 数据文本

上述储存完毕的文本文件的内容展示：

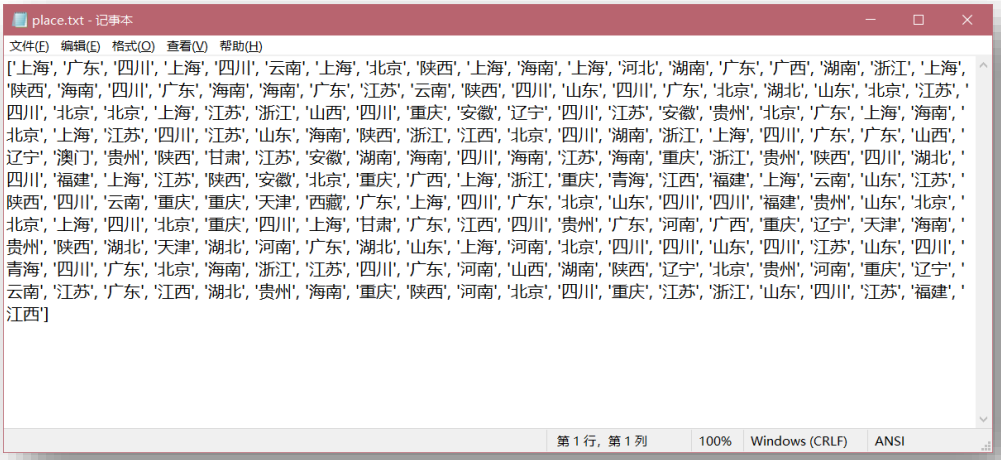



hot.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

['18315', '9636', '8347', '7184', '6991', '6980', '5550', '5382', '4927', '4249', '4224', '3928', '3744', '3709', '3554', '3287',
'3228', '3150', '2834', '2778', '2772', '2717', '2661', '2658', '2445', '2390', '2388', '2358', '2318', '2288', '2286', '2249',
'2222', '2211', '2201', '2076', '2069', '2060', '2051', '2000', '1960', '1960', '1946', '1918', '1866', '1842', '1814', '1812',
'1810', '1776', '1774', '1762', '1742', '1684', '1666', '1612', '1612', '1576', '1502', '1478', '1478', '1474', '1466', '1450',
'1378', '1354', '1298', '1296', '1294', '1280', '1280', '1272', '1260', '1240', '1234', '1228', '1226', '1202', '1146', '1130',
'1126', '1112', '1110', '1096', '1092', '1088', '1088', '1078', '1078', '1070', '1060', '1044', '1038', '1030', '1028', '1024',
'1024', '998', '998', '992', '978', '978', '972', '970', '970', '938', '928', '924', '924', '920', '918', '914', '910', '902', '902', '896',
'878', '878', '874', '866', '856', '854', '846', '842', '842', '834', '828', '816', '796', '794', '788', '786', '786', '778', '776', '776',
'762', '758', '752', '750', '748', '746', '738', '736', '730', '728', '728', '724', '716', '706', '704', '696', '694', '692', '684', '684',
'684', '684', '682', '680', '678', '672', '672', '668', '662', '660', '658', '648', '646', '644', '640', '634', '632', '630', '622', '618',
'614', '612', '612', '608', '600', '594', '592', '586', '586', '584', '584', '580', '578', '576', '572', '568', '568', '566', '560', '560',
'560', '558', '548', '546', '544', '542', '542', '542', '540', '540', '540', '538', '536', '530']

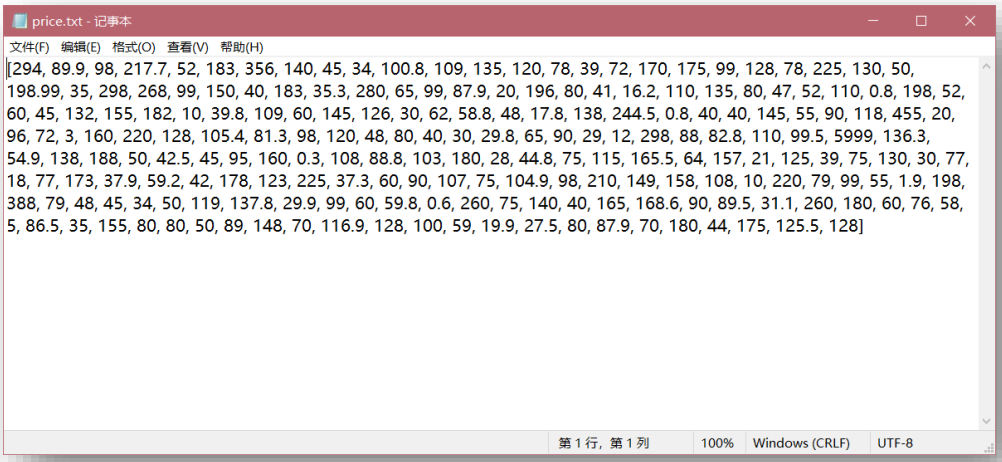
第 1 行，第 1 列 100% Windows (CRLF) UTF-8


place.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

['上海', '广东', '四川', '上海', '四川', '云南', '上海', '北京', '陕西', '上海', '海南', '上海', '河北', '湖南', '广东', '广西', '湖南', '浙江', '上海',
'陕西', '海南', '四川', '广东', '海南', '海南', '广东', '江苏', '云南', '陕西', '四川', '山东', '四川', '广东', '北京', '湖北', '山东', '北京', '江苏', '
四川', '北京', '北京', '上海', '江苏', '浙江', '山西', '四川', '重庆', '安徽', '辽宁', '四川', '江苏', '安徽', '贵州', '北京', '广东', '上海', '海南', '
北京', '上海', '江苏', '四川', '江苏', '山东', '海南', '陕西', '浙江', '江西', '北京', '四川', '湖南', '浙江', '上海', '四川', '广东', '广东', '山西', '
辽宁', '澳门', '贵州', '陕西', '甘肃', '江苏', '安徽', '湖南', '海南', '四川', '海南', '江苏', '海南', '重庆', '浙江', '贵州', '陕西', '四川', '湖北', '
四川', '福建', '上海', '江苏', '陕西', '安徽', '北京', '重庆', '广西', '上海', '浙江', '重庆', '青海', '江西', '福建', '上海', '云南', '山东', '江苏', '
陕西', '四川', '云南', '重庆', '重庆', '天津', '西藏', '广东', '上海', '四川', '广东', '北京', '山东', '四川', '四川', '福建', '贵州', '山东', '北京', '
北京', '上海', '四川', '北京', '重庆', '四川', '上海', '甘肃', '广东', '江西', '四川', '贵州', '广东', '河南', '广西', '重庆', '辽宁', '天津', '海南', '
贵州', '陕西', '湖北', '天津', '湖北', '河南', '广东', '湖北', '山东', '上海', '河南', '北京', '四川', '四川', '山东', '四川', '江苏', '山东', '四川', '
青海', '四川', '广东', '北京', '海南', '浙江', '江苏', '四川', '广东', '河南', '山西', '湖南', '陕西', '辽宁', '北京', '贵州', '河南', '重庆', '辽宁', '
云南', '江苏', '广东', '江西', '湖北', '贵州', '海南', '重庆', '陕西', '河南', '北京', '四川', '重庆', '江苏', '浙江', '山东', '四川', '江苏', '福建', '
江西']

第 1 行，第 1 列 100% Windows (CRLF) ANSI


price.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

[294, 89.9, 98, 217.7, 52, 183, 356, 140, 45, 34, 100.8, 109, 135, 120, 78, 39, 72, 170, 175, 99, 128, 78, 225, 130, 50,
198.99, 35, 298, 268, 99, 150, 40, 183, 35.3, 280, 65, 99, 87.9, 20, 196, 80, 41, 16.2, 110, 135, 80, 47, 52, 110, 0.8, 198, 52,
60, 45, 132, 155, 182, 10, 39.8, 109, 60, 145, 126, 30, 62, 58.8, 48, 17.8, 138, 244.5, 0.8, 40, 40, 145, 55, 90, 118, 455, 20,
96, 72, 3, 160, 220, 128, 105.4, 81.3, 98, 120, 48, 80, 40, 30, 29.8, 65, 90, 29, 12, 298, 88, 82.8, 110, 99.5, 5999, 136.3,
54.9, 138, 188, 50, 42.5, 45, 95, 160, 0.3, 108, 88.8, 103, 180, 28, 44.8, 75, 115, 165.5, 64, 157, 21, 125, 39, 75, 130, 30, 77,
18, 77, 173, 37.9, 59.2, 42, 178, 123, 225, 37.3, 60, 90, 107, 75, 104.9, 98, 210, 149, 158, 108, 10, 220, 79, 99, 55, 1.9, 198,
388, 79, 48, 45, 34, 50, 119, 137.8, 29.9, 99, 60, 59.8, 0.6, 260, 75, 140, 40, 165, 168.6, 90, 89.5, 31.1, 260, 180, 60, 76, 58,
5, 86.5, 35, 155, 80, 80, 50, 89, 148, 70, 116.9, 128, 100, 59, 19.9, 27.5, 80, 87.9, 70, 180, 44, 175, 125.5, 128]

第 1 行，第 1 列 100% Windows (CRLF) UTF-8

3 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：

```
print('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]='游乐园'
    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
```

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。

4 生成词云并统计词频

生成词云部分：

```
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")
```

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：

```
>>>开始统计词频
-->景点词频较大排名前20项：
海洋公园     19
游乐园       16
欢乐谷       12
风景区       12
上海         10
动物园       10
旅游区        7
成都         6
公园         5
南昌         4
北京         3
方特         3
南京         3
青岛         3
武隆         3
国家森林公园     3
天津         3
重庆         3
七彩         2
广州         2
```

```
-->地区词频较大排名前20项：
四川         25
北京         17
江苏         17
上海         16
广东         15
重庆         13
海南         11
浙江         11
陕西         10
山东          8
江西          6
贵州          6
湖北          5
安徽          5
湖南          4
山西          4
河南          4
云南          3
福建          3
天津          3
```

其完整代码部分如下：

```
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
```

5 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。

```
#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()
```

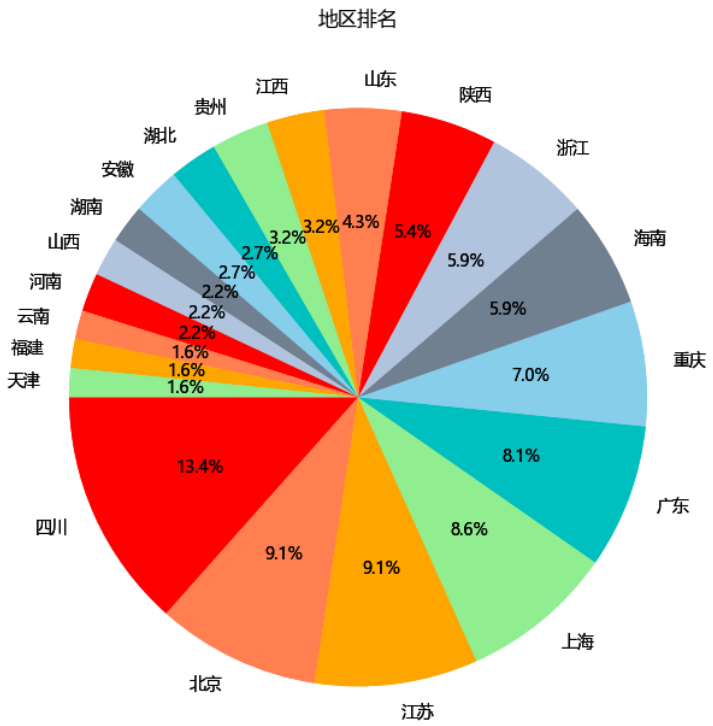
利用饼图表示每个城市中热门景点的占比，条形图表示不同类型景点的热门程度。

最后对欢乐谷类型景点进行销量-价格分析。

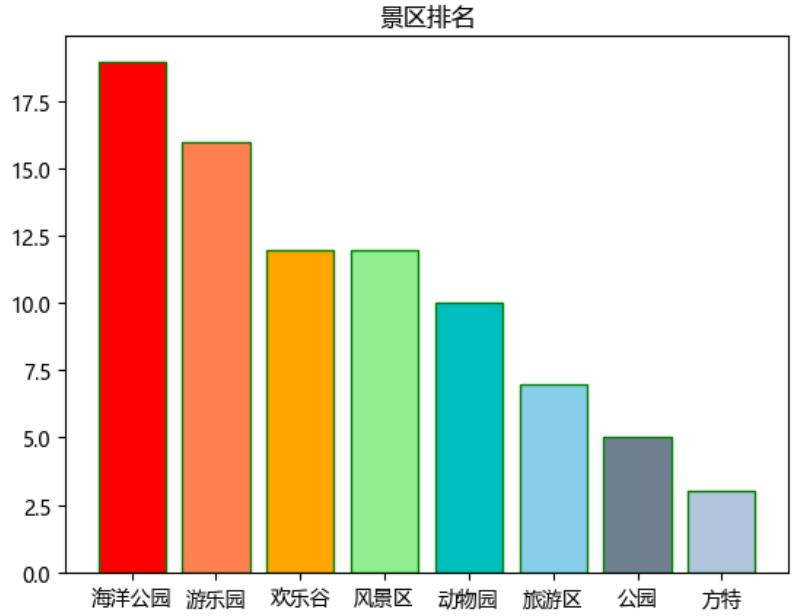
```
dist={}
hot1=[]
lo=[]
pri=[]
for i in range(len(txt)):
    dist[txt[i]]=pr[i]
for key in dist:
    if '欢乐谷' in key or '欢乐世界' in key or '方特' in key:
        lo.append(key)
        pri.append(dist[key])
plt.figure(3)
plt.xticks(rotation=45)
plt.bar(lo, pri, ec='g',ls='-',color=['r','coral','orange','lightgreen','c'\
                                      ,'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()
```

四、 生成的图表和词云

1 地区景点数占比饼图（非重点）



2 景区类型热度条形图（非重点）

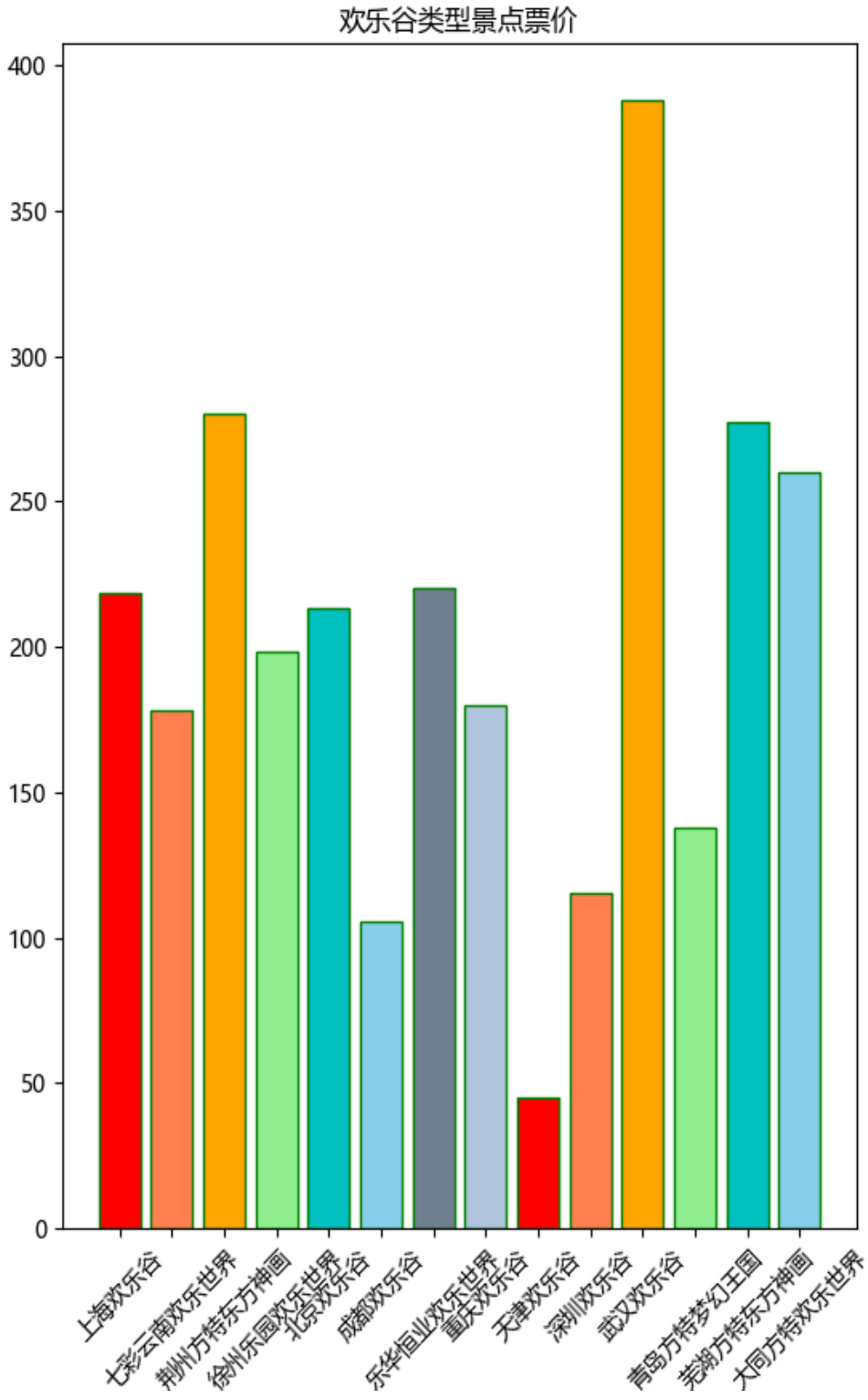


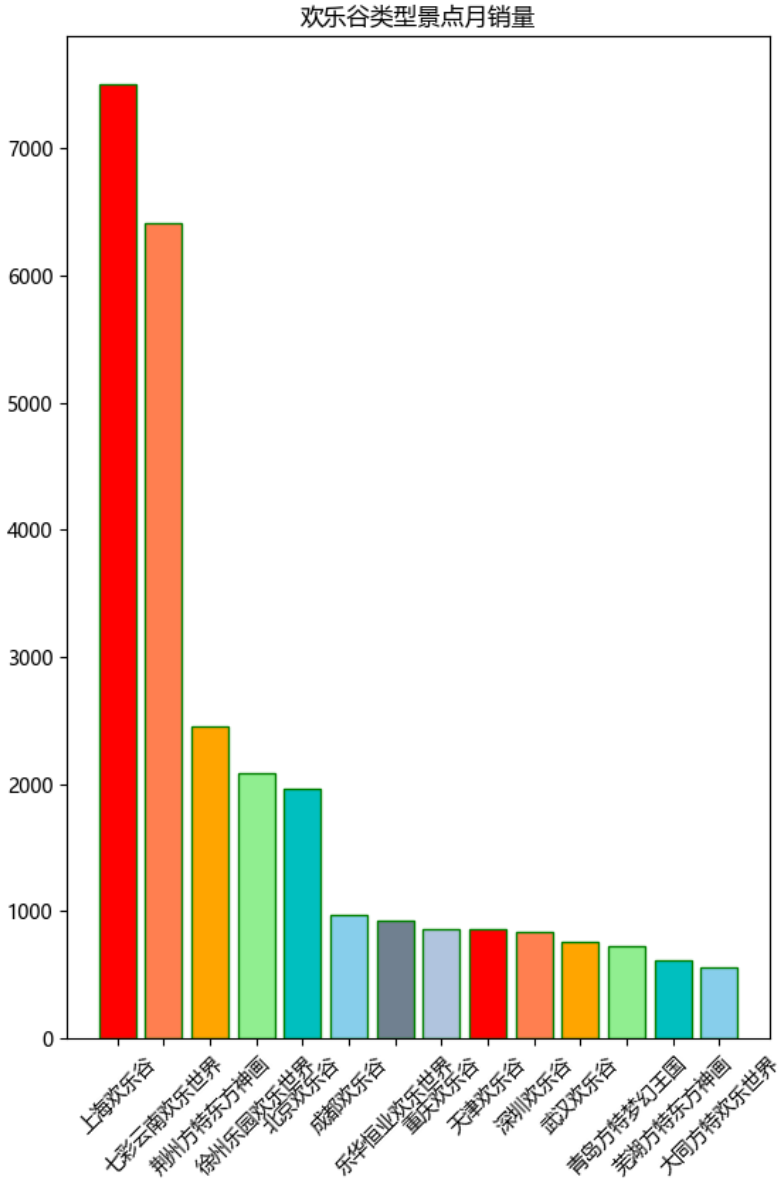
其中方特、欢乐谷、游乐园性质较为近似，但这不是分析重点，在这不做进一步处理。后续将细分欢乐谷类型景点的销量和价格。

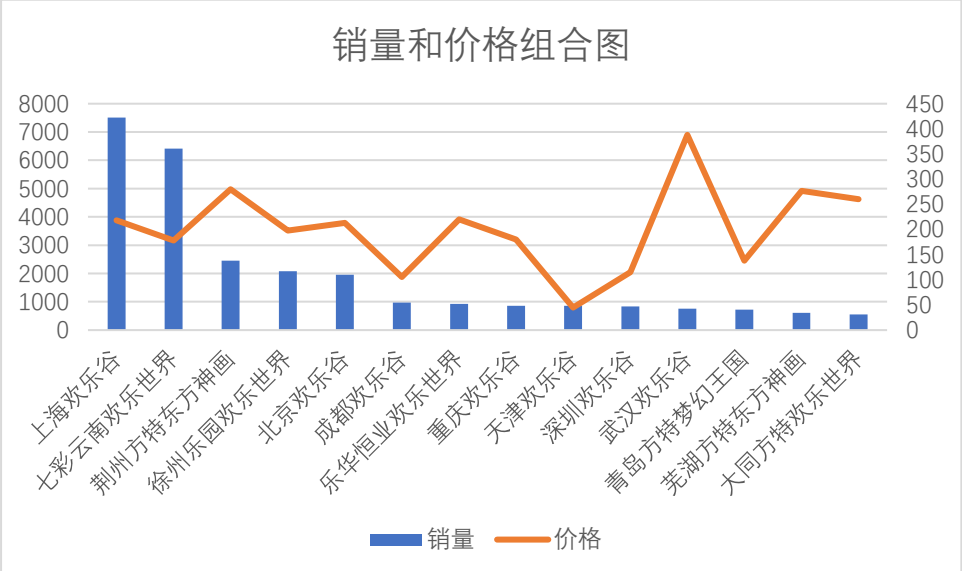
3 词云图

4 欢乐谷类景区定价、销量和成交量（重点）

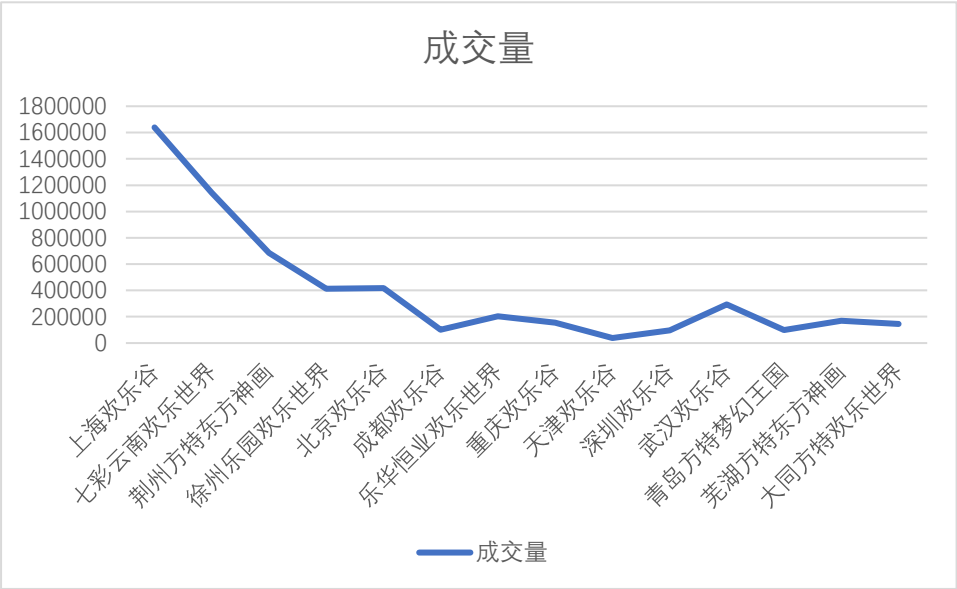
注意：此处欢乐谷类景区包括 欢乐谷、欢乐世界、方特这几个游乐园，票价为成人票。


欢乐谷类型景点票价
400
350
300
250
200
150
100
50
0
上海欢乐谷
七彩云南欢乐世界
荆州方特东方神画
徐州乐园欢乐世界
北京欢乐谷
成都欢乐谷
乐华恒业欢乐世界
重庆欢乐谷
天津欢乐谷
深圳欢乐谷
武汉欢乐谷
青岛方特梦幻王国
芜湖方特东方神画
大同方特欢乐世界


欢乐谷类型景点月销量
7000
6000
5000
4000
3000
2000
1000
0
上海欢乐谷
七彩云南欢乐世界
荆州方特东方神画
徐州乐园欢乐世界
北京欢乐谷
成都欢乐谷
乐华恒业欢乐世界
重庆欢乐谷
天津欢乐谷
深圳欢乐谷
武汉欢乐谷
青岛方特梦幻王国
芜湖方特东方神画
大同方特欢乐世界


销量和价格组合图
8000
7000
6000
5000
4000
3000
2000
1000
0
450
400
350
300
250
200
150
100
50
0
上海欢乐谷
七彩云南欢乐世界
荆州方特东方神画
徐州乐园欢乐世界
北京欢乐谷
成都欢乐谷
乐华恒业欢乐世界
重庆欢乐谷
天津欢乐谷
深圳欢乐谷
武汉欢乐谷
青岛方特梦幻王国
芜湖方特东方神画
大同方特欢乐世界
销量
价格

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

四、 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。

五、 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。

六、 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()
```