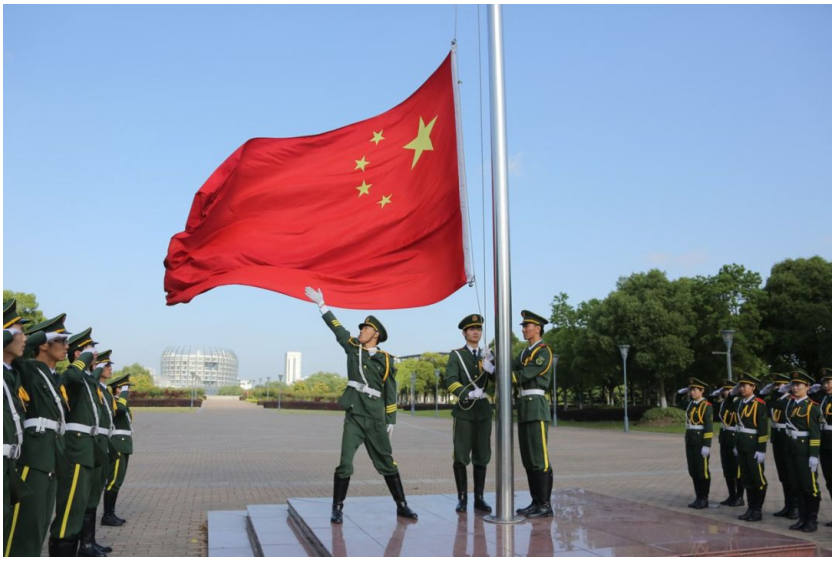
100 行 python 代码告诉你国庆哪些景点爆满

zone7

本文转载自公众号【zone7】，欢迎关注

前言

举国欢庆的国庆节马上就要到来了，你想好去哪里看人山人海了吗？还是窝在家里充电学习呢？说起国庆，塞车与爆满这两个词必不可少，去年国庆我在想要是我能提前知道哪些景点爆满就好了，就不用去凑热闹了。于是我开始折腾，想用 python 抓取有关出行方面的数据，便有了这篇文章。**如果我的文章对你有帮助，欢迎关注、点赞、转发，这样我会更有动力做原创分享。**



弘扬一下社会主义核心价值观

思考

（此段可跳过）要抓取出行方面的数据还不简单，直接去看看携程旅游、马蜂窝这类网站看看有没有数据抓取。但是实际上这些网站并没有比较好的格式化的数据供我们抓取，或许是我没找到吧。我在想，有没有什么折中的办法。然而，就这样半天过去了，突然想到，要出行肯定会查找相关的出行攻略吧，那么关键词就是一个突破口，可以查询百度指数来看看哪些景点被查询的次数最多，那么就可以大概知道哪些景点会爆满了。

统计结果

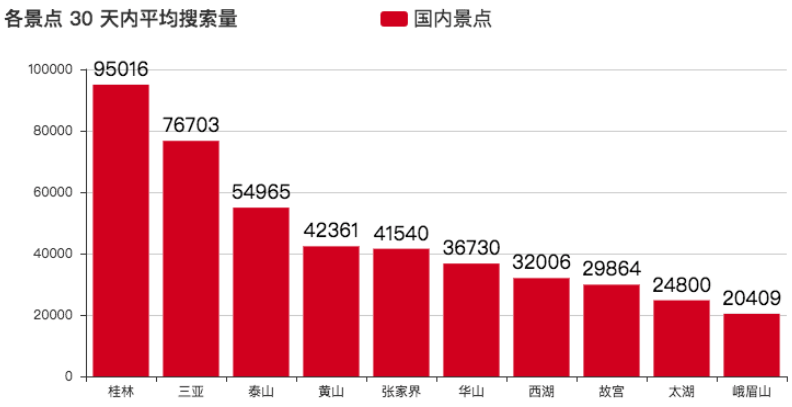
此次的统计结果只是从侧面反映景点爆满的问题，未必是完全准确的，仅供参考。此次统计的景点共有 100 个：

景点	地址	景点	地址
1 布达拉宫	拉萨	51 夫子庙	南京
2 稻城亚丁	四川甘孜藏族自治州	52 龙虎山	江西鹰潭
3 故宫	北京	53 宫三陵	沈阳
4 张家界	湖南	54 恒山	山西大同
5 九寨沟	四川阿坝	55 衡山	湖南衡阳
6 丽江古城	云南	56 黄帝陵	陕西延安
7 雅鲁藏布江大峡谷	西藏	57 闽西土楼	福建永定
8 乐山大佛	四川	58 黄龙景区	四川阿坝
9 万里长城	北京	59 晋祠	山西太原
10 宏村	安徽	60 井冈山	江西吉安
11 鼓浪屿	厦门	61 喀纳斯	新疆阿勒泰
12 婺源	江西上饶	62 海口	海南
13 纳木错	西藏	63 楼兰古城	新疆
14 外滩	上海	64 景德镇	江西
15 三清山	江西	65 庐山	江西九江
16 三亚	海南	66 罗平	云南
17 乌镇	浙江嘉兴	67 莫高窟	甘肃敦煌
18 凤凰古城	湖南	68 帕米尔高原	新疆
19 峨眉山	四川乐山	69 平遥古城	山西晋中
20 青海湖	青海	70 普陀山	浙江舟山
20 青海湖	青海	70 普陀山	浙江舟山
21 黄山	安徽	71 千户苗寨	贵州
22 洱海	云南大理	72 青岛	山东
23 元阳梯田	云南	73 曲阜三孔	山东济宁
24 长白山天池	吉林	74 日月潭	台湾
25 周庄	苏州昆山	75 三峡大坝	湖北宜昌
26 桂林	广西	76 三星堆遗址	四川德阳
27 长江三峡	重庆-湖北	77 沙坡头	宁夏中卫
28 呼伦贝尔	内蒙古	78 少林寺	河南登封
29 月牙泉	甘肃敦煌	79 神农架	湖北
30 颐和园	北京	80 瘦西湖	江苏扬州
31 黄果树瀑布	贵州	81 苏州园林	江苏
32 华山	陕西华阴	82 泰山	山东泰安
33 阿坝	四川	83 避暑山庄	河北承德
34 壶口瀑布	山西临汾-陕西延安	84 太湖	苏州
35 龙脊梯田	广西	85 滕王阁	江西南昌
36 维多利亚港	香港	86 五大连池	黑龙江
37 香格里拉	云南	87 武当山	湖北十堰
38 泸沽湖	四川-云南	88 西湖	浙江杭州
39 鸟巢	北京	89 阳朔西街	广西桂林
40 可可西里	青海玉树	90 西塘	浙江嘉兴

40 可可西里	青海玉树	90 西塘	浙江嘉兴
41 秦始皇兵马俑	西安	91 西夏王陵	宁夏银川
42 西双版纳	云南	92 雁荡山	浙江温州
43 趵突泉	山东济南	93 殷墟	河南安阳
44 大连	辽宁	94 玉龙雪山	云南丽江
45 中山陵	南京	95 云冈石窟	山西大同
46 大兴安岭	黑龙江	96 千岛湖	浙江杭州
47 大雁塔	西安	97 朱家角	上
48 丹霞山	广东韶关	98 珠穆朗玛峰	西藏
49 都江堰	四川成都	99 北戴河	河北秦岛
50 贺兰山	宁夏	100 自贡恐龙博物馆	四川

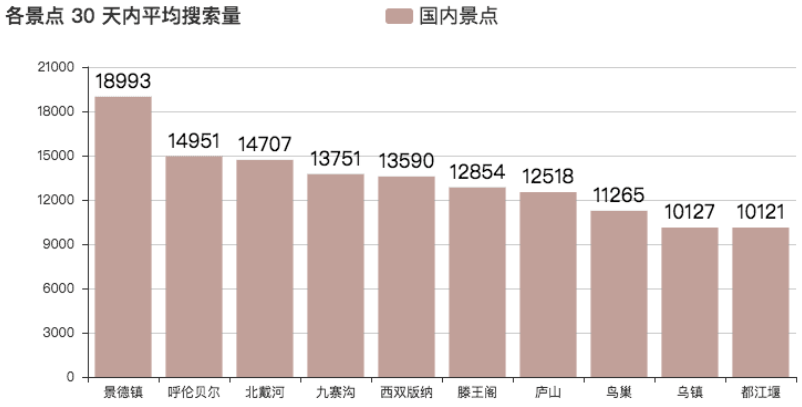
桂林、三亚、泰山的搜索量都是杠杠的，这第一梯队的地方能不去就别去了，去了也是人山人海的，爆满是无疑的了。

捂脸. jpg



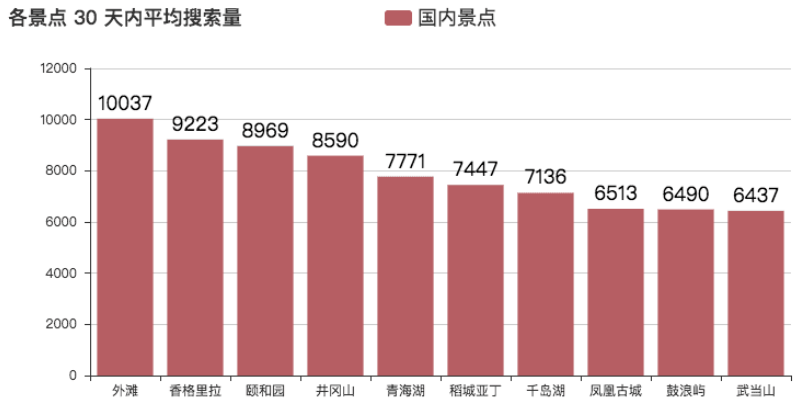
top0-10

第二梯队的搜索量也不差，日均搜索量还是上万的，谨慎行动。



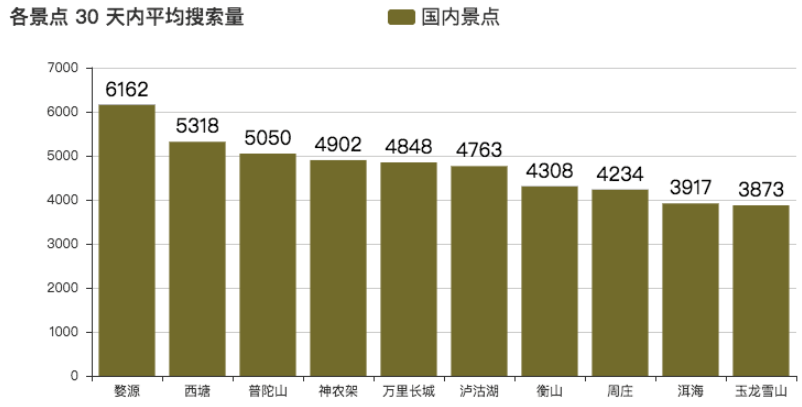
top10-20

第三梯队下来就可以考虑考虑，为了避免不必要的塞车与等待，建议大家还是呆在家里吧！！！



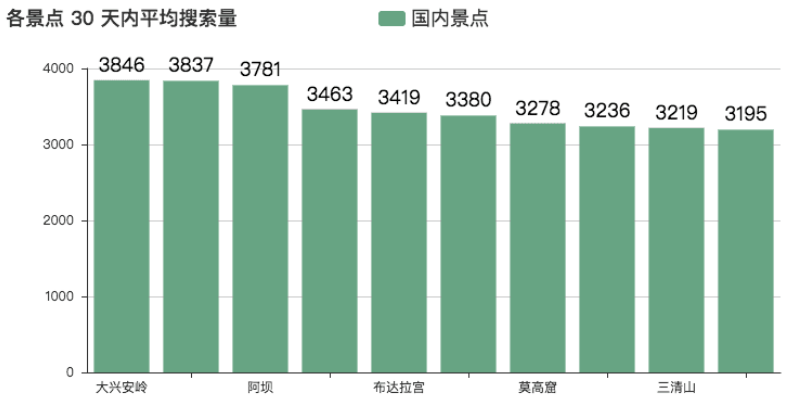
top20-30

第四梯队应该没太大的问题，建议出去溜达溜达。



top30-40

都到第五梯队了，就可以放心地玩耍了。经历了那么多的烦心事，是该好好放飞一下自己了。



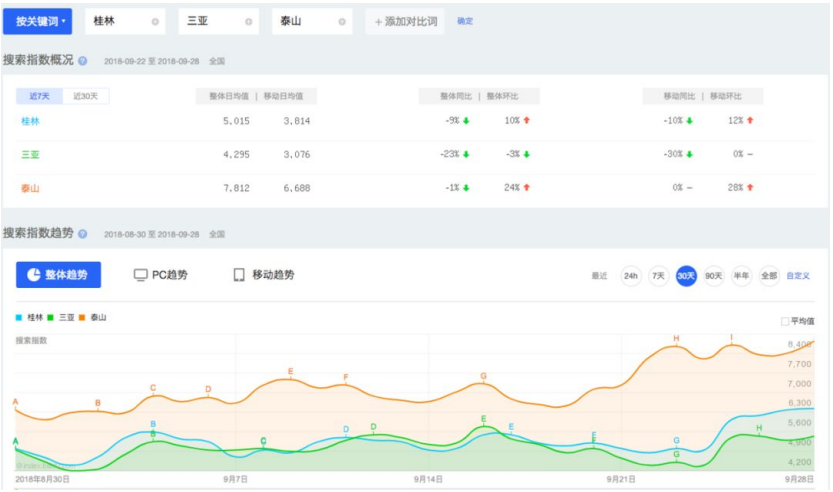
top40-50

爬虫技术分析

- 请求库：selenium
- HTML 解析：使用正则匹配
- 数据可视化：pyecharts
- 数据库：MongoDB
- 数据库连接：pymongo

爬虫分析实现

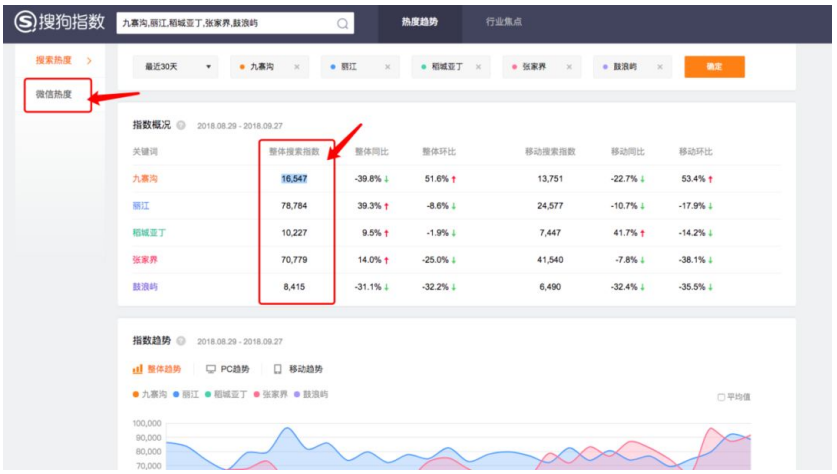
此次文章能够实现参考效果，完全是因为抖机灵。首先是选取爬虫来源，携程与马蜂窝没有结构化的数据，我们就换一种思路。首先是想到百度指数，如图：



百度指数

但是，分析源代码之后，你就会发现坑爹之处了，它的数据都是以图片展示的，你不能直接获取到源码，考虑到国庆马上就要到来，我换了一个指数平台，转战搜狗指数，这个平台可以直接获取到源数据，关键是，还有微信热度可以爬取。当然，你执意要使用百度指数，这里也是有方法的，抓取到数据之后，使用图像识别来识别文中的数据，提供一篇有思路的文章 [爬虫实战——四大指数之百度指数（三）链接：https://zhuanlan.zhihu.com/p/28973232]。

关于数据清洗方面，这里筛选了数据量过小，和数据量异常大的景点，详情在源码中查看。



搜狗指数

数据展示的代码片段

```
def show_data(self):
    for index in range(5):
        queryArgs = {"day_avg_pv": {"$lt": 100000}}
        rets = self.zfdb.national_month_index.find(queryArgs).sort("day_avg_pv", pymongo.DESCENDING).limit(10)
        atts = []
        values = []
        file_name = "top" + str(index * 10) + "-" + str((index + 1) * 10) + ".html"
        for ret in rets:
            print(ret)
            atts.append(ret["address"])
            values.append(ret["day_avg_pv"])
        self.show_line("各景点 30 天内平均搜索量", atts, values)
        os.rename("render.html", file_name)
```

爬虫代码实现

由于篇幅原因，这就只展示主要代码，详情请查看源码，可以查看GitHub地址：https://github.com/zonezoen/nationalDayIndex

```
# 这是数据爬取的代码片段
def get_index_data(self):
```

```
        try:
                for url in self.get_url():
                        print("当前地址为: " + url)
                        self.browser.get(url)
                        self.browser.implicitly_wait(10)
                        ret = re.findall(r'root.SG.data = (.*)}]};', self.browser.page_source)
                        totalJson = json.loads(ret[0] + "}]}")
                        topPvDataList = totalJson["topPvDataList"]
                        infoList = totalJson["infoList"]
                        pvList = totalJson["pvList"]
                        for index, info in enumerate(infoList):
                                for pvDate in pvList[index]:
                                        print("index => "+str(index)+"地址 => "+info["kwdName"] + "日
期 => " + str(pvDate["date"]) + " => " + str(pvDate["pv"]) + " => " + str(
                                                info["avgWapPv"]) + " => " + str(info["kwdSumPv"]
["sumPv"]) + " => ")
                                        self.zfdb.national_day_index.insert({
                                                "address": info["kwdName"],    # 地名
                                                "date": pvDate["date"],    # 日期
                                                "day_pv": pvDate["pv"],    # 日访问量
                                        })
                                self.zfdb.national_month_index.insert({
                                        "address": info["kwdName"],    # 地名
                                        "day_avg_pv": info["avgWapPv"],    # 平均访问量
                                        "sum_pv": info["kwdSumPv"]["sumPv"],    # 总访问量
                                })
        except :
```

```
            print("exception")
```

后记

整篇爬虫文章分析到这里就结束，不过还是对百度指数很有执念，想找个时间写一篇相关的文章才行，不搞定它感觉心里有块疙瘩，或许这就是程序员最后的倔强，最后祝大家国庆假期愉快，不用写代码。

往期推荐：

Python | 拥有选择权 ，才拥有概率 。

深度学习入门笔记系列 8 篇（ 集合 ）

碎碎念 | 钱不是那样赚的 。

zone

zone7

zone7

阅读原文