Crawler and data visualization by python

Yun Zhang
Bartow Hanvos Kent School Ningbo Campus
Email: sophiazhangbarsow@outlook

***Abstract* – World Wide Web has incorporated to people's daily lives and people are able to acquire the information they need online by simply typing few key words in the search box. However, for those users who require to access an general area with a specific criteria, such work is impossible because data is not in a managed and structured architecture across the web. Therefore, in terms of dealing with tremendous database, there are several tools we can employ for data mining, including crawler which is a preferable and effortless way to salvage data from webs. Additionally, for those people who deal with data everyday, data visualization can benefit them a lot. This paper presents two systematic project: one for the crawler in website *www.lianjia.com*, a Chinese housing website like *Redfin.* The other one is the visualization of the friends information in WeChat which is the Chinese social network like Facebook.**

I. Introduction

Web crawler is a tool to fetch the web pages, automatically download the content that the user focuses on, and use certain logic to index the downloaded data. *HTML* is a document containing both tags and text. The function of the web browser is to read the *HTML* and display them through the web page. Therefore, in other words, *HTML* is the web page. The *HTML* contains various kinds of tags, like *<div>*, defining a area of the web page, and *<a>*, defining a interlink. Additionally, these tags appear in pairs, representing the begin and the end. The main library involved in the first project is called *Beautiful Soup*. As for the function of *Beautiful Soup*, it converts the HTML document to a complex tree structure where all the tags, like *<div>*, can be targeted by *Beautiful Soup*. Therefore, when the users can initialize *Beautiful Soup* with the attributes and the names of the tags, then the *Beautiful Soup* could locate the selected tag in order to fetch the relevant data. By this way, users are able to access to any specific data with predefined requirement.

The next project is about data visualization, using WeChat Application Programming Interface. API is a set of established programming instructions and standards released by a software company to the public so that other software developers can either access a Web-based software or, Web tool, or design products that are charged by its service. Therefore, by incorporating python, the most popular programming language in this world, users are able to conduct their own products powered by *API*. The second project is directly working with the WeChat *API*, applying it to load all the information of the friends, such as the name, personal signature, gender, and geographic location. Then utilize charts or map to visualize the sex ratio and the geographic locations of friends. Also, I set up an Auto-responding chatbot using WeChat *API*.

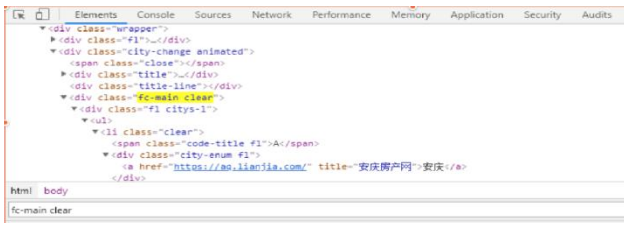
II. Materials and Methodology

There are two projects, one is crawler on web *www.lianjia.com* and *www.douban.com*, and the other one is WeChat friend information visualization and auto-responding chatbot .

A. Coding For Crawler

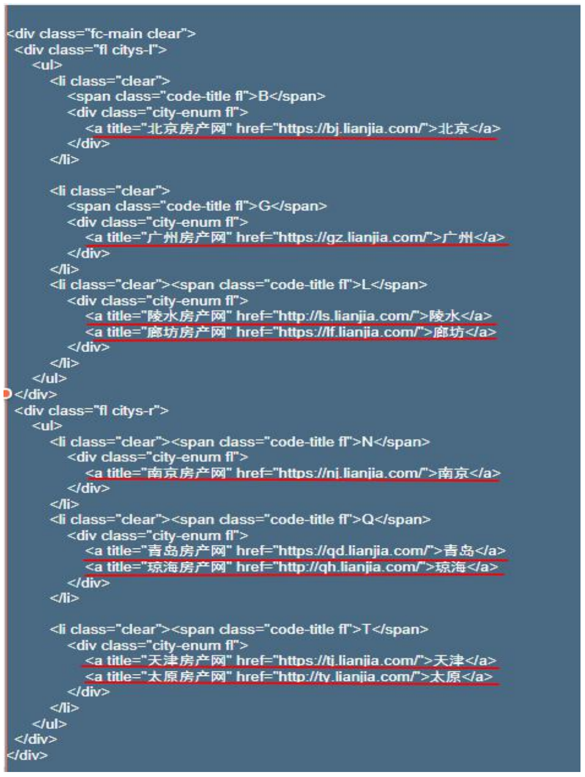
To run the crawler successfully, the first step is to locate the accurate tags inside the HTML document from the website.

[1] All source code is available at the following URL: *https://github.com/YUNZHANG123/python_life_fun*



```
▼<div class="wrapper">
  ▶<div class="fl">…</div>
  ▼<div class="city-change animated">
     <span class="close"></span>
    ▶<div class="title">…</div>
     <div class="title-line"></div>
    ▼<div class="fc-main clear">
      ▼<div class="fl citys-l">
        ▼<ul>
          ▼<li class="clear">
             <span class="code-title fl">A</span>
            ▼<div class="city-enum fl">
               <a href="https://aq.lianjia.com/" title="安庆房产网">安庆</a>
             </div>
fc-main clear
```

This image is part of the HTML document of the web *www.lianjia.com*, and the goal is to find the *url*, a abbreviation of the address of a web page, of the city names.



```
<div class="fc-main clear">
 <div class="fl citys-l">
   <ul>
     <li class="clear">
       <span class="code-title fl">B</span>
       <div class="city-enum fl">
         <a title="北京房产网" href="https://bj.lianjia.com/">北京</a>
       </div>
     </li>

     <li class="clear">
       <span class="code-title fl">G</span>
       <div class="city-enum fl">
         <a title="广州房产网" href="https://gz.lianjia.com/">广州</a>
       </div>
     </li>
     <li class="clear"><span class="code-title fl">L</span>
       <div class="city-enum fl">
         <a title="陵水房产网" href="http://ls.lianjia.com/">陵水</a>
         <a title="廊坊房产网" href="https://lf.lianjia.com/">廊坊</a>
       </div>
     </li>
   </ul>
 </div>
 <div class="fl citys-r">
   <ul>
     <li class="clear"><span class="code-title fl">N</span>
       <div class="city-enum fl">
         <a title="南京房产网" href="https://nj.lianjia.com/">南京</a>
       </div>
     </li>
     <li class="clear"><span class="code-title fl">Q</span>
       <div class="city-enum fl">
         <a title="青岛房产网" href="https://qd.lianjia.com/">青岛</a>
         <a title="琼海房产网" href="http://qh.lianjia.com/">琼海</a>
       </div>
     </li>

     <li class="clear"><span class="code-title fl">T</span>
       <div class="city-enum fl">
         <a title="天津房产网" href="https://tj.lianjia.com/">天津</a>
         <a title="太原房产网" href="http://ty.lianjia.com/">太原</a>
       </div>
     </li>
   </ul>
 </div>
</div>
```

After digging into *<div class="fc-main clear">,* the cities and their *urls* show up. So *<div class="fc-main clear">* is the parent node, and the children node is the city name and its url. With BeautifulSoup we can get all the information we need.

```
Crawler of city
city_tags=bsobj.find("div",{"class":"fc-main
clear"}).findChildren("a")

with open("./cityslian.csv", "w") as f:
    writ = csv.writer(f)
    for city_tag in city_tags:
        # get the href link of <a> tag
        city_url = city_tag.get("href").encode("utf-8")
        # get the names of <a> tag
city_name = city_tag.get_text()
        print(city_name)
        print(city_url)
```

Above is the first part of the code to acquire all the city names in *www.lianjia.com.*
1. Employing BeautifulSoup(*import bs4)*
2. Displaying the urls of the web *www.lianjia.com*
3. Reading the *HTML* page of web *www.lianjia.com*
4. Get *BeautifulSoup* object.

Following is the second part of the code.

1. Get all <a> tags under <div class="fc-main clear">.
2. save the data in the file "citys.csv".
3. get the href link of <a> tag.
4. get the names of <a> tag.
(The names inside *<a>* tag is the names of the cities)

```
Crawler for Douban
from urllib.request  import urlopen
From bs4 import
BeautifulSoup="https://movie.douban.com/top250?start=0&filt
er="
html = urlopen(url).read()
soup = BeautifulSoup(html,"lxml")
grid_view=soup.find("ol",attrs={'class':'grid_view'})
lis = grid_view.find_all("li")
for li in lis:
    span= li.find("span",class_="title")
```

The acquisition of the movie names in *www.douban.com* is based on the similar principle.

1. Fist we import BeautifulSoup.
2. Go to the HTML page.
3. Find the <div> that contains the names of the movies .
4. Call "Find_all" to get all the information.
5. The string of the <li> tag is the movie name.

B. Coding For Visualization

To achieve the friend information visualization, we need to get all friends information with the help of WeChat API and dump the data into a file named "friends_info.txt". For example, "梅川,袁旭老师,1,辽宁,大连,,0,3" is the information of one friend acquired from WeChat. The attributes are listed in the order of "Nickname, Remarkname, sex, Province, city, signature, Starfriends"

Here the project does two different visualizations:

1. Initialize a list named **"**sex**"**
2. Open the file named **"**friends_info.txt**"**.
3. Fetch the gender for each friend which is in the third position of the attributes row in the file and add the result to the list .
4. Male represents as 1, female as 2, and unknown as 0.

The above code is used to construct the pie chart, such as the title and some notations. After the chart construction, we apply the list of sex into the chart.

The visualization of the geographic locations of friends begins with retrieving the locations of each friend from the *friends_info.txt*, adding them into a list and then employing the list of locations to the Chinese.

```
Applying BeautifulSoup
import sys

import csv

from urllib.request  import urlopen

from bs4 import BeautifulSoup

url = "https://www.lianjia.com"

html = urlopen(url).read()

bsobj = BeautifulSoup(html, "lxml")
```

```
Visualization of geographic Locations
from pyecharts import Map
import webbrowser

province_dict = {'北京': 0, '上海': 0, '天津': 0, '重庆': 0,
    '河北': 0, '山西': 0}
provinces = []
with open('ma.txt', mode='r', encoding='utf-8') as f:
    rows = f.readlines()
    for row in rows:
        provinces.append(row.split(',')[3])
for province in provinces:
    if province in province_dict.keys():
        province_dict[province] += 1
    else:
        province_dict['国外及未知'] += 1
map = Map("地域分布")
map.add("地域分布
",province_dict.keys(),province_dict.values(),is_visualmap=
True)
map.render("geo.html")
```

```
Visualizing Gender Ratio
pie  =  Pie('sex  ratio  of  friends',  'total  number  of
friends：%d' % len(sex), title_pos='center')
pie.add('', attr, value, radius=[30, 75], rosetype='area',
is_label_show=True,
        is_legend_show=True, legend_top='bottom')
pie.render('ratio.html')
webbrowser.open('ratio.html', new=2)
Return Image
```

C. Issues & Debugging

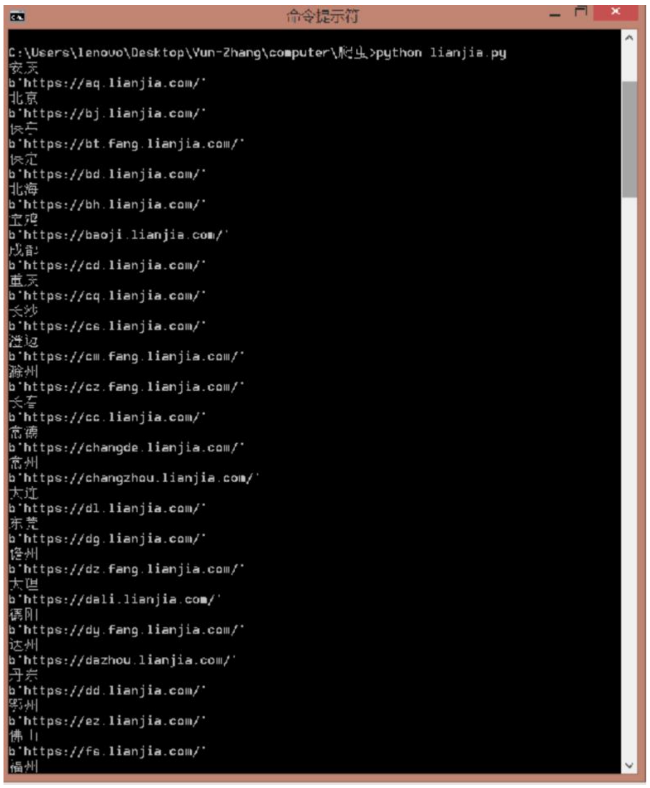
A major issue in pre-processing is the grammar. As a computer science language, python requires strict structures and grammar in order to successfully run the codes. As completing the code, several grammar mistakes exist.

One of common mistakes is the tab&space. When a sentence is moved to the next line, the four spaces have to be retyped otherwise the computer is not able to recognize it. Additionally, each subordinate or secondary code should leave four more spaces ahead than the former one. Coding people often forget to leave four spaces or less than four, and these structures are neither acceptable.

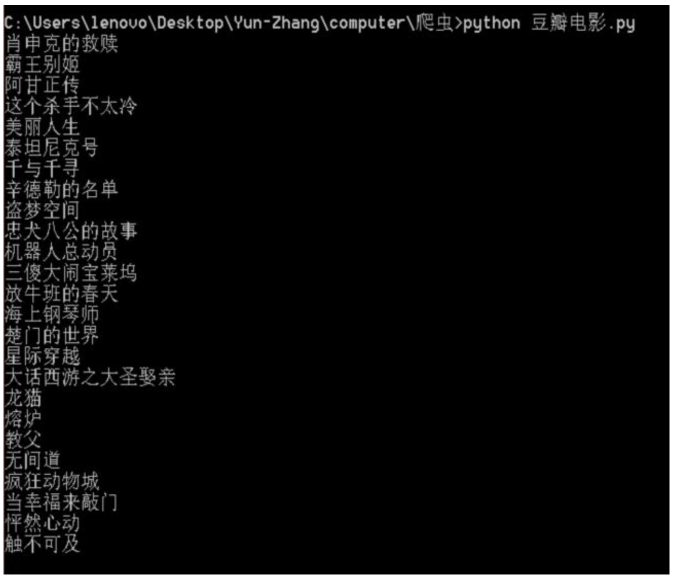
In addition, for the code of crawler, dealing with list can be tricky. If the goal is to get all the city names under a <div>, the first step is to employ Find_all function to get all the children tags (of which the string is the city name) of that <div>, and add them into a list. Then comes the trick. If users directly ask for fetching all the strings simultaneously, computer would be confused how to implement. Therefore, to achieve this goal, utilizing the circulation formula “for .. in .. :”, which require the string of each tag one by one in the list, is necessary.

III. Results

For the crawler project, the command lines display all the city names and movie names of the chosen targets in the html document of the web.

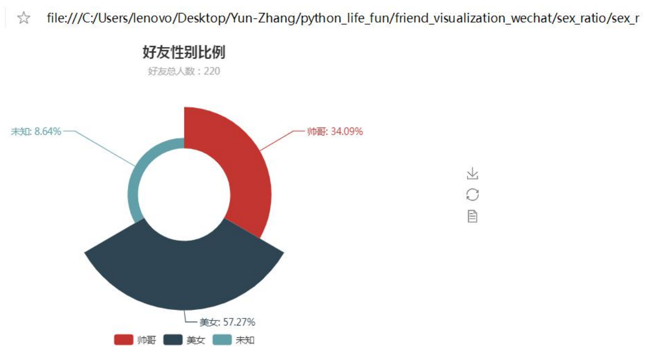


```
命令提示行
C:\Users\lenovo\Desktop\Yun-Zhang\computer\爬虫>python lianjia.py
安庆
b'https://aq.lianjia.com/'
北京
b'https://bj.lianjia.com/'
包头
b'https://bt.fang.lianjia.com/'
保定
b'https://bd.lianjia.com/'
北海
b'https://bh.lianjia.com/'
宝鸡
b'https://baoji.lianjia.com/'
成都
b'https://cd.lianjia.com/'
重庆
b'https://cq.lianjia.com/'
长沙
b'https://cs.lianjia.com/'
澄迈
b'https://cm.fang.lianjia.com/'
滁州
b'https://cz.fang.lianjia.com/'
长春
b'https://cc.lianjia.com/'
常德
b'https://changde.lianjia.com/'
常州
b'https://changzhou.lianjia.com/'
大连
b'https://dl.lianjia.com/'
东莞
b'https://dg.lianjia.com/'
德州
b'https://dz.fang.lianjia.com/'
大理
b'https://dali.lianjia.com/'
德阳
b'https://dy.fang.lianjia.com/'
达州
b'https://dazhou.lianjia.com/'
丹东
b'https://dd.lianjia.com/'
鄂州
b'https://ez.lianjia.com/'
佛山
b'https://fs.lianjia.com/'
福州
```

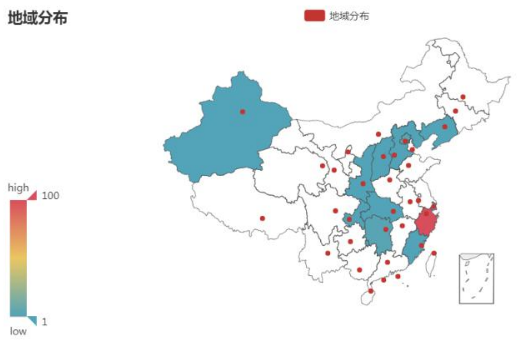


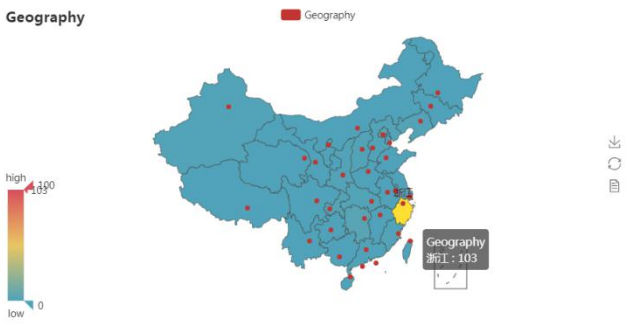
```
C:\Users\lenovo\Desktop\Yun-Zhang\computer\爬虫>python 豆瓣电影.py
肖申克的救赎
霸王别姬
阿甘正传
这个杀手不太冷
美丽人生
泰坦尼克号
千与千寻
辛德勒的名单
盗梦空间
忠犬八公的故事
机器人总动员
三傻大闹宝莱坞
放牛班的春天
海上钢琴师
楚门的世界
星际穿越
大话西游之大圣娶亲
龙猫
熔炉
教父
无间道
疯狂动物城
当幸福来敲门
怦然心动
触不可及
```

The results for WeChat visualization for gender ratio and friends geographic locations are shown below.



A website of pie chart above plainly displays the ratio of female, male, and unknown -- 57.27%, 34.09%, and 8.64% respectively -- in WeChat.





Besides, a website of Chinese map above pinpoints several locations where a figure of how many friends coming from this region is displayed.

On the left side of the graph, users can adjust the color to indicate the population in each region. For example, region coloring in blue has zero to ten people, and region coloring in red has more than 90 people.

IV. Discussion & Future Work

Web database can be easily accessed and its massive potential of information is a preferred choice for scientists, researchers, and users. To take a full advantage of the web usage, future study can focus on new ideas and innovative ways for loading information from web database more productively and effectively in order to implement a progressive growth to keep up the pace with the constantly renewing information. Hence, crawler can be developed as a hybrid to conduct complicated tasks for human.

For the use of Python, further study can explore other functions of WeChat API, such as responding friends according to their message, so that the language can diversify and specify to different people. Additionally, API of other computer products can be employed to accomplish a more comprehensive project.

V. Acknowledgement

I really thank Xu for teaching me crawler and python, who also offers reference, insightful ideas, and corrections on this paper. I would also like to thank Ye for further suggestions to this paper.

VI. References

[1] “ Application programming interface” , Web. 12 Oct 2019. <https://en.wikipedia.org/wiki/Application_programming_interface>