实验 2-2 爬取微博上各类红酒的关注人信息

建议课时：30 分钟

一、 实验目的

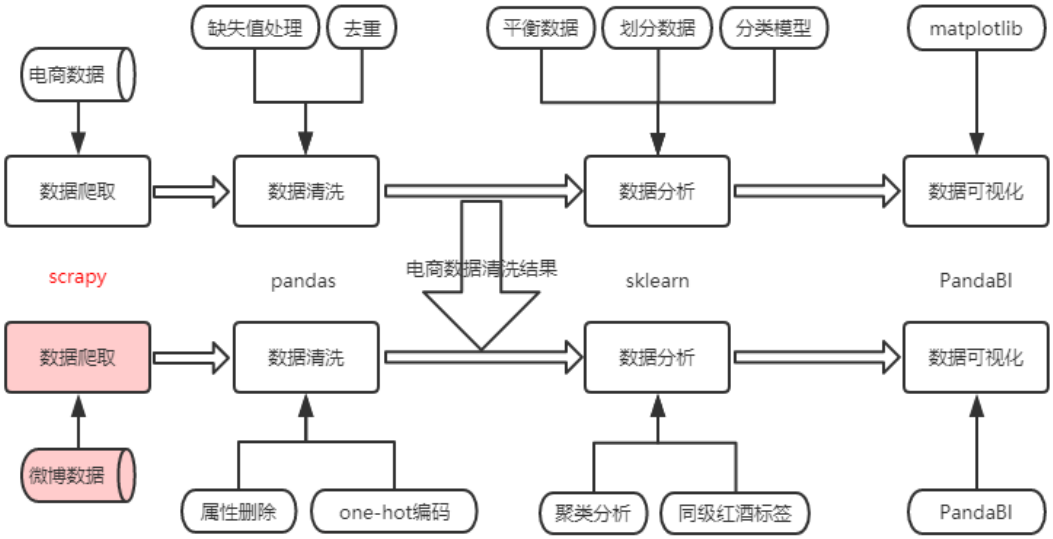
- 了解 scrapy 的常用函数
- 了解爬虫编写的流程

二、 实验环境

Python3 开发环境，第三方包有 scrapy，re

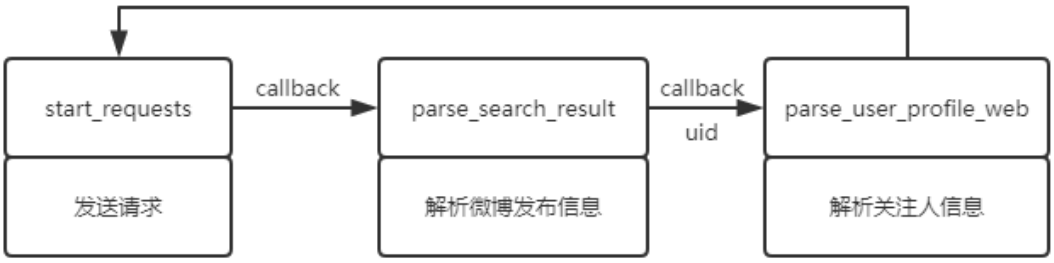
三、 实验步骤

本节处理的内容有：



微博上搜索红酒的相关话题，可以得到微博的相关发布信息，发布人，发布时间等，再由发布人获取其身份信息，所在地信息，内容如下图所示：

我们需要通过编写 python 脚本进行爬取，下面选取核心代码讲解爬取京东红酒属性数据的逻辑流程。



1. 搜索关键词

根据红酒品牌列表加红酒字样作为搜索关键词，实现代码如下：

```
with open('./wine_project/data/红酒品牌.csv', 'r', encoding='utf-8') as f:
    csv_reader = csv.reader(f)
    for zh_name, en_name in csv_reader:
        print('++++++++++crawling:{}-{}'.format(zh_name, en_name))
        if zh_name.strip():
            wine_name = zh_name + ' 红酒'
            q_string = '100103type=61&q={kw}&t=0'.format(kw=wine_name)
            yield Request(real_time_search_url.format(containerid=quote(q_string), page=page),
                          meta={'kw': wine_name, 'page': 1}, callback=self.parse_search_result)
        if en_name.strip():
            wine_name = en_name + ' 红酒'
            q_string = '100103type=61&q={kw}&t=0'.format(kw=wine_name)
            yield Request(real_time_search_url.format(containerid=quote(q_string), page=page),
                          meta={'kw': wine_name, 'page': 1}, callback=self.parse_search_result)
```

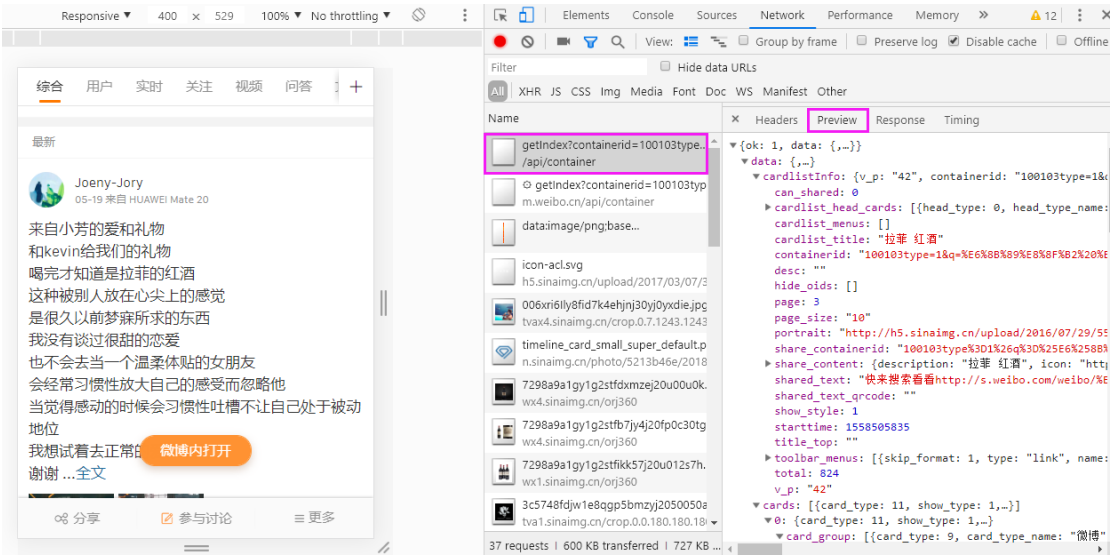
一个红酒品牌有中文和英文两种表达方式，代码中两种名称都要作为搜索关键词进行检索。

通过 scrapy 的 Request 创建京东数据获取对象，并且通过 yield 构造一个生成器函数，将获取的数据按需返回，防止因数据太多造成内存溢出等问题。

爬取的数据会通过 callback 函数回调，我们这里指定 parse_search_result 函数，该函数的任务是解析爬取的数据。

2. 实时相关话题中的微博基础信息

根据关键词搜索可以得到实时话题相关的微博信息，如下图显示：



可见我们可以从 api 的返回值中解析出该微博的基础信息，代码如下：

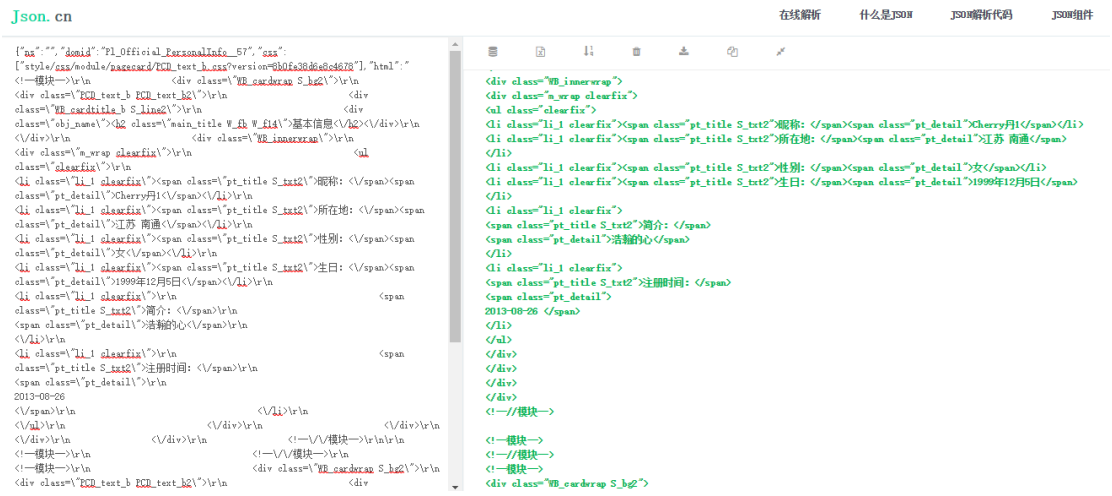
```
json_data = json.loads(response.text)
has_next_page = False
if 'data' in json_data and 'cards' in json_data['data']:
    for card in json_data['data']['cards']:
        if card['card_type'] != 11:
            continue
        for item in card['card_group']:
            if item['card_type'] == 9 and item['mblog'].get('user'):
                has_next_page = True
                blog = item['mblog']
                u = blog['user']
                uid = u.get('id')
                ret = {}
                ret['id'] = blog['id']
                ret['url'] = 'https://weibo.com/{uid}/{bid}'.format(uid=uid, bid=blog['bid'])
                ret['keyword'] = kw

                ret['uid'] = uid
                ret['post_time'] = parse_create_at(blog['created_at'])
                ret['post_text'] = blog.get('text') or blog.get('raw_text')
                ret['post_source'] = blog['source']
                ret['description'] = u['description']
                ret['follow_count'] = u['follow_count']
                ret['followers_count'] = u['followers_count']
                ret['gender'] = u['gender']
                ret['name'] = u['screen_name']
                yield ret
```

3. 微博用户的属性信息

由上一步中我们可以获取到用户的 uid，即可以转到某用户的微博下，比如 https://weibo.com/5990920544，由于手机端访问会缺失用户信息，所以采用电脑端查看网页源代码的方式来提取信息。定位到待提取文本的位置，如下图所示：

<script>FM.view({"ns":"","domid":"Pl_Official_PersonalInfo__57","css":["style/css/module/pagecard/PCD_text_b.css?version=8b0fe38d6e8c4678"],"html":"<!--模块-->\r\n<div class=\"WB_cardwrap S_bg2\">\r\n<div class=\"PCD_text_b PCD_text_b2\">\r\n<div class=\"WB_cardtitle_b S_line2\">\r\n<div class=\"obj_name\"><h2 class=\"main_title W_fb W_f14\">基本信息</h2> \r\n \r\n<div class=\"WB_innerwrap\">\r\n<div class=\"m_wrap clearfix\">\r\n<ul class=\"clearfix\">\r\n<li class=\"li_1 clearfix\"><span class=\"pt_title S_txt2\">昵称：<span class=\"pt_detail\">Cherry丹1
\r\n<li class=\"li_1 clearfix\"><span class=\"pt_title S_txt2\">所在地：<span class=\"pt_detail\">江苏 南通
\r\n<li class=\"li_1 clearfix\"><span class=\"pt_title S_txt2\">性别：<span class=\"pt_detail\">女
\r\n<li class=\"li_1 clearfix\"><span class=\"pt_title S_txt2\">生日：<span class=\"pt_detail\">1999年12月5日
\r\n<li class=\"li_1 clearfix\">\r\n<span class=\"pt_title S_txt2\">简介：\r\n<span class=\"pt_detail\">浩翰的心\r\n
\r\n<li class=\"li_1 clearfix\">\r\n<span class=\"pt_title S_txt2\">注册时间：\r\n2013-08-26\r\n\r\n
\r\n
\r\n \r\n \r\n \r\n \r\n<!--\/\/模块-->\r\n\r\n<!--模块-->\r\n<div class=\"PCD_text_b PCD_text_b2\">\r\n<div class=\"WB_cardtitle_b S_line2\">\r\n<div class=\"obj_name\"><h2 class=\"main_title W_fb W_f14\">教育信息</h2> \r\n \r\n<div class=\"WB_innerwrap\">\r\n<div class=\"m_wrap clearfix\">\r\n<ul class=\"clearfix\">\r\n<li class=\"li_1 clearfix\"><span class=\"pt_title S_txt2\">大学：\r\n<span class=\"pt_detail\">\r\n华东理工大学 (2017年)\r\n<br\/>\r\n\r\n
\r\n
\r\n \r\n \r\n \r\n \r\n<!--\/\/模块-->\r\n<!--模块-->\r\n<!--\/\/模块-->\r\n"})</script>

这源代码的网页中看比较乱，我们可以把内容粘贴到 json.cn 的网页上查看，如下所示：



解析的代码如下：

```
def parse_user_profile_web(self, response):
    ret = response.meta
    matcher = user_profile_ptn.findall(response.text)
    profile_dict = {}
    if not matcher:
        print('============not matched, url:{}'.format(response.url))
        return
    matcher = matcher[0].strip().replace("\\r", "").replace("\\n", "").replace("\\", "")
    selector = etree.HTML(matcher)
    for i in selector.xpath('//li'):
        key = i.xpath('./span[1]/text()')
        if i.xpath('./span[2]/a'):  # 主动发现带有链接的标签值
            value = i.xpath('string(./span[2])')
            value = '|'.join([i.strip() for i in value.split()]) if value else None
        else:
            value = i.xpath('./span[2]/text()')
            value = '|'.join([i.strip() for i in value if i.strip()]) if value else ''
        key = key[0].strip().replace(': ', '') if key else None
        profile_dict[key] = value
    ret['profile'] = profile_dict
    yield ret
```

四、 实验结果

示例如下：

{

"uid": 5101075910,

"post_time": "2019-05-14 11:45:15",

"post_text": "回复@岁月偷走了他的酒:有时候觉得爱国情怀是天生长在骨血里的//@岁月偷走了他的酒:起来 不愿做奴隶的人们 把我们的血肉筑成我们新的长城 每次听我都热泪盈眶",

"post_source": "微博 weibo.com",

"post_url": "https://weibo.com/5101075910/HpKZ3l0h2",

"follow_count": 102,

"followers_count": 334,

"keyword": "长城 酒",

"profile": {

"昵称": "霸天小王子每天都要开心呀",

"所在地": "广东 深圳",

"性别": "女",

"生日": "1995 年 10 月 19 日",

"简介": "把你的全部，奉献给你热爱的一切",

"注册时间": "2014-05-01",

"大学": "华南理工大学|(2012 年)|其他",

"标签": "美图摄影|美食|星座运势"

}

}

最终的爬取结果在 微博红酒.csv 文件中。