R爬虫及进行文本挖掘

周世祥

2020/3/22

数据获取方式

大数据时代，最不缺的是数据，数据就是黄金，就是石油，可是作为个人来说，获取数据并不容易，特别是有价值的数据。这个时候，爬虫就开始行动了，所谓的爬虫就是我们用编程语言写的程序，能够不知疲倦地替我们去广阔的互联网上替我们搜寻信息。你到一个陌生的地方，想找一个便宜的房子，从网上一个一个页面去搜索，太慢了，效率低。你想研究新冠病毒的发病模型，数据哪儿来，写个爬虫就替你做了。

如果你学过Python，一定听说过大名鼎鼎的爬虫框架–scrapy [https://baike.baidu.com/item/scrapy/7914913?fr=aladdin (https://baike.baidu.com/item/scrapy/7914913?fr=aladdin)].

框架的好处是方便，安装好了就可以用，代码量少，效率高，不好的地方就是灵活性不够，有些地方对用户来说不透明。对一些项目来说，我们用R的几行代码就可以自动化地采集数据。

当然学习爬虫需要先明确一些概念，比如，Http协议，静态网页和动态网页，json格式，selenium自动化测试。

静态页面和动态页面

静态页面并不是指没有动态效果的网页，现在的H5中JavaScript已经能做出漂亮的动画效果，静态网页指的是HTML网页在我们客户端请求时候已经客观存在于网页服务器上了。

动态网页是指在收到请求的时候，根据请求用服务器程序(PHP,JSP ,ASPX)“动态”地生成HTML网页。比如，你上教务系统上查看自己的成绩，你只能看到自己的信息，你看到的网页和别人不一样。你用百度地图导航时，随着位置不同，地图需要不断更新。动态页面说到底，需要后台数据库服务器支持，数据必须不断更新。

尽管H5前端编程工资待遇不错的，然而只会前端，知识面太窄，很容易被淘汰的，所以现在有些机构美其名曰，全栈工程师，就是加上一些后端的编程技术进行补充。

H5的流行是有道理的，在这个云时代，我们要转变思想了，不需要买强劲的服务器，阿里云，腾讯云，华为云都提供云服务，我们个人只需有一个终端就可以，这个终端可以是笔记本，手机等轻终端，我们可以把软件或应用部署在云上，终端上只需安装一个web容器就可以，这个容器就是浏览器，想想微软为什么要把ie集成到操作系统，就知道浏览器是互联网的入口。web发展到现在，你可以感觉到，单机版的软件没有出路，PC端的软件越来越少，连一个驱动精灵，替我们安装电脑驱动的软件都有web版了。现在我们上网课，数不清的在线直播平台，功能越来越强大。这里说马化腾引以为傲的微信，腾讯的核心产品，是一种不需要下载安装即可使用的应用，它实现了应用“触手可及”的梦想，用户扫一扫或搜一下即可打开应用。

web的流行可见是有历史原因的。

web页面的构成

web其实就是HTML文件，HTML文件由三部分组成：内容是什么，HTML脚本，描述怎么样，即CSS样式，动作行为，即JavaScript。 JavaScript对HTML，CSS进行操纵(增、删、改、查)。

如果程序能解析HTML结构就能控制页面，从而爬取相关的信息。

DOM的结构

DOM文档对象模型[https://baike.baidu.com/item/DOM%E5%AF%B9%E8%B1%A1/6621083?fr=aladdin (https://baike.baidu.com/item/DOM%E5%AF%B9%E8%B1%A1/6621083?fr=aladdin)]，是W3C组织推荐的处理可扩展标记语言的标准编程接口。前面讲到web页面由各种层次的标签元素构成的，随便找来一个页面源代码，你会看到最上层有一个html，里面会有head,title等等标签，从数据结构上看，总体上看是一个树形结构，实际上，见过markdown，latex，你了解到他们都是标记语言，结构都是类似的。这些结构不想我们的矩阵或excel表格那么工整，它们都是非结构化的数据，所以想提取信息，需要费点功夫的。

推荐一本好书《细说DOM编程》，兄弟连出品的，兄弟连在线机构，可惜在这次病毒流行中没能坚持住，倒闭了。

JSON

JSON是什么，我们从网上收集的数据大多是JSON格式，特别是通过API方式，你可以把JSON理解为一个格式化好的数据。 R语言中先安装JSON包。

```
install.packages("J:/R课件/rjson_0.2.20.zip", repos = NULL, type = "win.binary")
```

```
setwd('J:/R课件')
library(rjson) #加载rjson包
result<- fromJSON(file="input.json") #这个文件提前下载好
print(result)
```

```
## $ID
## [1] "1" "2" "3" "4" "5" "6" "7" "8"
##
## $Name
## [1] "Rick"     "Dan"      "Michelle" "Ryan"     "Gary"     "Nina"     "Simon"
## [8] "Guru"
##
## $Salary
## [1] "623.3"  "515.2"  "611"    "729"    "843.25" "578"    "632.8"  "722.5"
##
## $startDate
## [1] "1/1/2012"   "9/23/2013"  "11/15/2014" "5/11/2014"  "3/27/2015"
## [6] "5/21/2013"  "7/30/2013"  "6/17/2014"
##
## $Dept
## [1] "IT"         "Operations" "IT"         "HR"         "Finance"
## [6] "IT"         "Operations" "Finance"
```

```
json_data_frame<- as.data.frame(result)
# R语言的数据框是它的创新
print(json_data_frame)
```

```
##   ID     Name Salary  startDate       Dept
## 1  1     Rick  623.3   1/1/2012         IT
## 2  2      Dan  515.2  9/23/2013 Operations
## 3  3 Michelle    611 11/15/2014         IT
## 4  4     Ryan    729  5/11/2014         HR
## 5  5     Gary 843.25  3/27/2015    Finance
## 6  6     Nina    578  5/21/2013         IT
## 7  7    Simon  632.8  7/30/2013 Operations
## 8  8     Guru  722.5  6/17/2014    Finance
```

我们看到json格式有点像Python中的字典，可以参考网站https://www.runoob.com/json/json-tutorial.html。

Xpath和正则表达式

Xpath即XML路径语言，是一种用来确定XML文档中的某部分位置的语言，XML文档是前面讲的HTML等超集。Xpath基于XML的树状结构，提供在数据结构树中找寻节点的能力。可以当作小型的查询语言。R语言的XML包基于Xpath提供许多功能函数,https://www.runoob.com/xpath/xpath-tutorial.html (https://www.runoob.com/xpath/xpath-tutorial.html)。

正则表达式用来检索某个模式的文本，R语言的XML包基于正则表示式提供了grep(),sub(),regexpr()等功能函数，进行字符串的模式匹配和索引工作。每一种语言都有正则表达式操作语法。

获取静态web内容主要使用RCurl，XML包。RCurl包封装了HTTP协议接口，实现了HTTP的功能。本质上理解成一个命令行形式的浏览器。

下面我们用R的包RCurl不打开浏览器，从网上下载信息。

```
library("RCurl")
url.exists(url="www.baidu.com")  #判断URL是否存在
```

```
## [1] TRUE
```

```
h<- basicHeaderGatherer()


txt<-getURL(url="http://www.baidu.com",headerfunction=h$update)
names(h$value)
```

```
## NULL
```

```
h$value()
```

```
##
Accept-Ranges
##
"bytes"
##
Cache-Control
##
"no-cache"
##
Connection
##
"keep-alive"
##
Content-Length
##
"14615"
##
Content-Type
##
"text/html"
##
Date
##                                                                                                                                    "Su
n, 22 Mar 2020 02:49:03 GMT"
##
P3p
##                                                                                                                            "CP=\" OTI
DSP COR IVA OUR IND COM \""
##
P3p
##                                                                                                                            "CP=\" OTI
DSP COR IVA OUR IND COM \""
##
Pragma
##
"no-cache"
##
Server
##
"BWS/1.1"
##
Set-Cookie
##                        "BAIDUID=223A7425150BF923A989B6CB9063933C:FG=1; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=214748364
7; path=/; domain=.baidu.com"
##
Set-Cookie
##                            "BIDUPSID=223A7425150BF923A989B6CB9063933C; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=214748364
7; path=/; domain=.baidu.com"
##
Set-Cookie
##                                                      "PSTM=1584845343; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=214748364
7; path=/; domain=.baidu.com"
##
Set-Cookie
## "BAIDUID=223A7425150BF923060B6FCAFADEBEDF:FG=1; max-age=31536000; expires=Mon, 22-Mar-21 02:49:03 GMT; domain=.baidu.com; pa
th=/; version=1; comment=bd"
##
Traceid
##                                                                                                                      "158484534328
383567467540575160188933123"
##
Vary
##
"Accept-Encoding"
##
X-Ua-Compatible
##
"IE=Edge,chrome=1"
##
status
##
"200"
##
statusMessage
##
"OK"
```

上面的代码功能很简单，实现了查看服务器返回的头信息。

实现单页爬虫

xpath查看

按浏览器的F12功能键进行调试。

#实现单页爬虫功能

```
library("RCurl")
library ("XML")

url.exists(url<-"http://search.dangdang.com/?key=统计&act=input&page_index=1")
```

```
## [1] TRUE
```

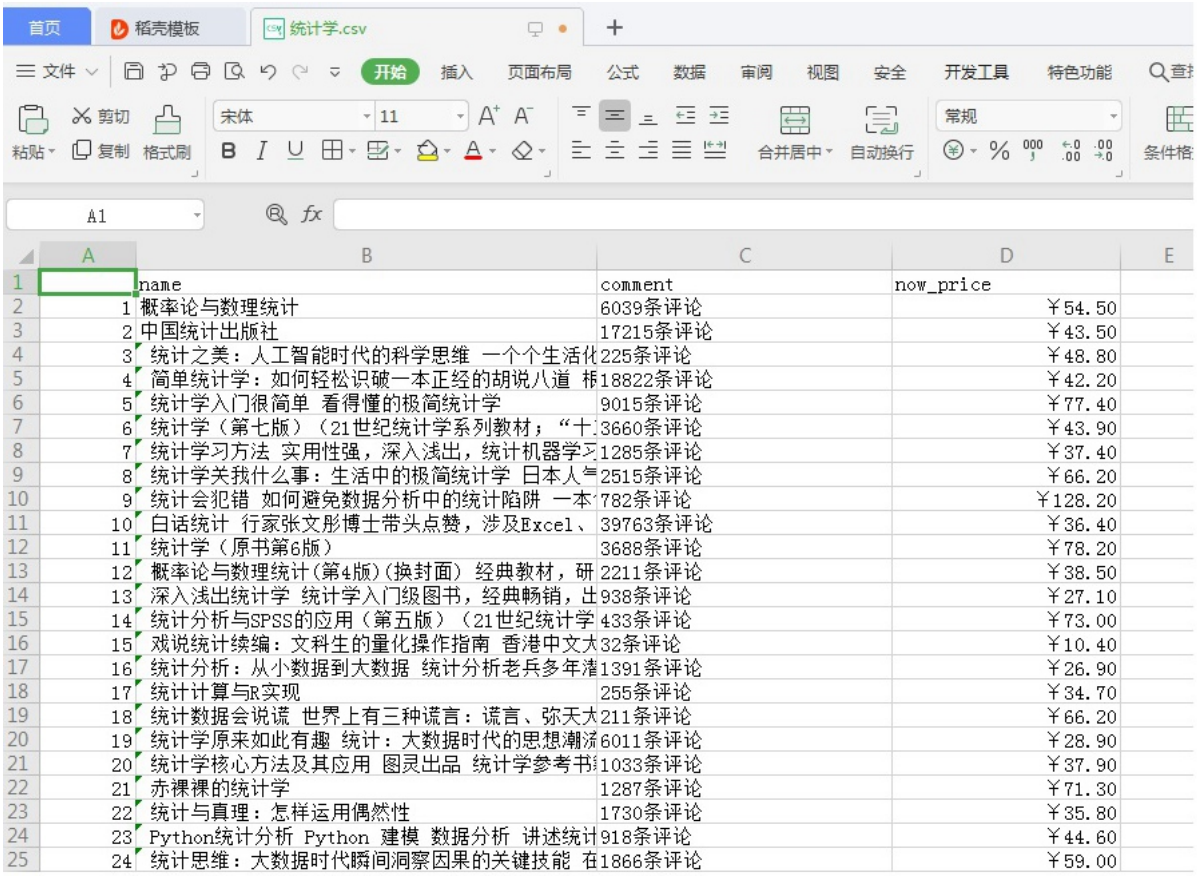
```
myheader<-c("User-Agent"="Mozilla/5.0 (iPhone; U; CPU iPhone OS 4_0_1 like MacOS X; ja-jp)AppleWebKit/532.9 (KHTML, like Gecko) Version/4.0.5 Mobile/8A306 Safari/6531.22.7",
"Accept"="text/html,application/xhtml+xml,appication/xml;q=0.9,*/*;q=0.8",
"Accept-Language"="en-us",
"Connection"="keep-alive",
"Accept-Charset"="GB2312,utf-8;q=0.7, *;q=0.7")

#这个地方是假装是有钱人，用苹果手机查看当当网信息，不容易被当当封网

webpage=getURL( url,httpheader=myheader, .encoding="GB2312") #RCurl包的getURL()函数读取URL对应的文件为字符串

mode(webpage) #webpage 是个字符串
```

```
## [1] "character"
```

```
temp=iconv(webpage,"GB2312", "UTF-8")#将编码转换为UTF-8

write.table(temp,"temp.txt")# 输出一下temp,中文没有乱码，可以进行下一步工作

pagetree=htmlTreeParse(temp,encoding="UTF-8",error=function(...){},useInternalNodes=TRUE,trim=TRUE)

#XML包的htrePar方法对HTML内容进行解析

mode(pagetree) #pagetree 是R内部使用的externalptr类型
```

```
## [1] "externalptr"
```

```
name0 <- xpathSApply(pagetree,"//*/a[@title]",xmlValue) #XML包的getNodeSet()函数以DOM对HTML文档进行检索

name<- name0[grep("统计",name0)]#挖掘图书名

name<-name[1:60]#发现有两个书没有价格

#对HTML文档进行检索

comment<-xpathSApply(pagetree,"//*/a[@name='itemlist-review']",xmlValue)#挖掘点评数量

now_price<- xpathSApply(pagetree,"//*/span[@class='search_now_price']",xmlValue)#挖掘图书现价

statistics <-data.frame(name, comment, now_price)
write.csv(statistics, "J:\\R课件\\统计学.csv")
```



	A	B	C	D	E
1		name	comment	now_price	
2	1	概率论与数理统计	6039条评论	￥54.50	
3	2	中国统计出版社	17215条评论	￥43.50	
4	3	统计之美：人工智能时代的科学思维 一个个生活化	225条评论	￥48.80	
5	4	简单统计学：如何轻松识破一本正经的胡说八道	18822条评论	￥42.20	
6	5	统计学入门很简单 看得懂的极简统计学	9015条评论	￥77.40	
7	6	统计学（第七版）（21世纪统计学系列教材；“十	3660条评论	￥43.90	
8	7	统计学习方法 实用性强，深入浅出，统计机器学习	1285条评论	￥37.40	
9	8	统计学关我什么事：生活中的极简统计学 日本人气	2515条评论	￥66.20	
10	9	统计会犯错 如何避免数据分析中的统计陷阱 一本	782条评论	￥128.20	
11	10	白话统计 行家张文彤博士带头点赞，涉及Excel、	39763条评论	￥36.40	
12	11	统计学（原书第6版）	3688条评论	￥78.20	
13	12	概率论与数理统计(第4版)(换封面) 经典教材，研	2211条评论	￥38.50	
14	13	深入浅出统计学 统计学入门级图书，经典畅销，出	938条评论	￥27.10	
15	14	统计分析与SPSS的应用（第五版）（21世纪统计学	433条评论	￥73.00	
16	15	戏说统计续编：文科生的量化操作指南 香港中文大	32条评论	￥10.40	
17	16	统计分析：从小数据到大数据 统计分析老兵多年潜	1391条评论	￥26.90	
18	17	统计计算与R实现	255条评论	￥34.70	
19	18	统计数据会说谎 世界上有三种谎言：谎言、弥天大	211条评论	￥66.20	
20	19	统计学原来如此有趣 统计：大数据时代的思想潮流	6011条评论	￥28.90	
21	20	统计学核心方法及其应用 图灵出品 统计学参考书	1033条评论	￥37.90	
22	21	赤裸裸的统计学	1287条评论	￥71.30	
23	22	统计与真理：怎样运用偶然性	1730条评论	￥35.80	
24	23	Python统计分析 Python 建模 数据分析 讲述统计	918条评论	￥44.60	
25	24	统计思维：大数据时代瞬间洞察因果的关键技能 在	1866条评论	￥59.00	

xpath查看

下一步可以用正则表达式去掉评论中文本，只留下数值，价格中的人民币符号，只留下价格。然后做数据分析。

这只是从当当单页中收取60个商品的信息。研究当当网页变化规律，就可以修改程序连续爬取多页信息.

网络数据的应用级API采集(以豆瓣为例)

API是Application Programming Interface的缩写。具体而言，就是某个网站，有不断积累和变化的数据。这些数据如果整理出来，不仅耗时，而且占地方，况且刚刚整理好就有过期的危险。大部分人需要的数据，其实都只是其中的一小部分，时效性的要求却可能很强。因此整理储存，并且提供给大众下载，是并不经济划算的。

可是如果不能以某种方式把数据开放出来，又会面对无数爬虫的骚扰。这会给网站的正常运行带来很多烦恼。折中的办法，就是网站主动提供一个通道。当你需要某一部分数据的时候，虽然没有现成的数据集，却只需要利用这个通道，描述你自己想要的数据，然后网站审核（一般是自动化的，瞬间完成）之后，认为可以给你，就立刻把你明确索要的数据发送过来。双方皆大欢喜。

今后你找数据的时候，也不妨先看看目标网站是否提供了API，以避免做无用功。

应用级(非数据库级)API是软件或网站平台的开发方提供的数据查询通道，为了使用API首先要查阅API的帮助文档(通常还需要注册开发者账号)。以豆瓣为例，其API帮助文档的官方网址为: https://developers.douban.com/wiki/?title=guide (https://developers.douban.com/wiki/?title=guide).

简单浏览API帮助，发现即使不注册开发者账号，也可以借助豆瓣API采集到想要的数据，例如在浏览器中输入 https://api.douban.com/v2/book/1220562.

即可返回编号为1220562的图书信息(JSON格式)。显然通过RCurl包可以以程序方式实现这个步骤，然后借助rjson包解析JSON格式的数据，即可获得我们想要的豆瓣网信息。这就是解决问题的关键思路。

还比如说从中国天气网api上：www.weather.com.cn 的获取天气信息。

```
Sys.setlocale(locale="Chinese")
```

```
## [1] "LC_COLLATE=Chinese (Simplified)_People's Republic of China.936;LC_CTYPE=Chinese (Simplified)_People's Republic of Chin
a.936;LC_MONETARY=Chinese (Simplified)_People's Republic of China.936;LC_NUMERIC=C;LC_TIME=Chinese (Simplified)_People's Republ
ic of China.936"
```

```
library("RCurl")
library("rjson")
url="https://api.douban.com/v2/book/20429677?apikey=0df993c66c0c636e29ecbb5344252a4a"

# 此处需要加apikey，豆瓣疑下线所有公开 API，所有请求都会报 msg:"invalid_apikey"，通过 imdb 号查豆瓣信息，这个需要研究研究

library(httr)#它类似于Python中的request软件包，类似于Web浏览器，可以完成和远端服务器的沟通。
response <-GET(url, user_agent="my@email.com this is a test")

#注意其中的status一项。我们看到它的返回值为200。以2开头的状态编码是最好的结果，意味着一切顺利；如果状态值的开头是数字4或者5，那就有问题了，你需要排查错误。
library(jsonlite)
```

```
##
## Attaching package: 'jsonlite'
```

```
## The following objects are masked from 'package:rjson':
##
##     fromJSON, toJSON
```

```
toJSON(fromJSON(content(response, as="text")), pretty = TRUE)
```

```
## {
##   "rating": {
##     "max": [10],
##     "numRaters": [16190],
##     "average": ["7.5"],
##     "min": [0]
##   },
##   "subtitle": ["生活、工作与思维的大变革"],
##   "author": ["[英] 维克托<U+2022>迈尔<U+2022>舍恩伯格（Viktor Mayer-Sch<U+00F6>nberger）"],
##   "pubdate": ["2012-12"],
##   "tags": [
##     {
##       "count": 9800,
##       "name": "大数据",
##       "title": "大数据"
##     },
##     {
##       "count": 6151,
##       "name": "互联网",
##       "title": "互联网"
##     },
##     {
##       "count": 3549,
##       "name": "数据挖掘",
##       "title": "数据挖掘"
##     },
##     {
##       "count": 3352,
##       "name": "大数据时代",
##       "title": "大数据时代"
##     },
##     {
##       "count": 2297,
##       "name": "互联网趋势",
##       "title": "互联网趋势"
##     },
##     {
##       "count": 1696,
##       "name": "计算机",
##       "title": "计算机"
##     },
##     {
##       "count": 1596,
##       "name": "数据",
##       "title": "数据"
##     },
##     {
##       "count": 1562,
##       "name": "社会学",
##       "title": "社会学"
##     }
##   ],
##   "origin_title": ["Big Data:A Revolution That Will Transform How We Live, Work, and Think"],
##   "image": ["https://img3.doubanio.com/view/subject/m/public/s24574862.jpg"],
##   "binding": ["平装"],
##   "translator": ["周涛"],
##   "catalog": ["引言　正在发生的生活、工作与思维的大变革\n第一部分　大数据时代的思维变革\n第1章　更多：不是随机样本，而是所有数据\n第2章　更杂：不是精确性，而是混杂性\n第3章　更好：不是因果关系，而是相关关系\n第二部分　大数据时代的商业变革\n第4章　数据化：一切皆可“量化”\n第5章　价值：“取之不尽，用之不竭”的数据创新\n第6章　角色定位：　数据、技术与思维的三足鼎立\n第三部分　大数据时代的管理变革\n第7章　风险：让数据主宰一切的隐忧\n第8章　掌控：自由与责任并举的数据管理\n结语　已经发生的未来"],
##   "ebook_url": ["https://read.douban.com/ebook/29224686/"],
##   "pages": ["261"],
##   "images": {
##     "small": ["https://img3.doubanio.com/view/subject/s/public/s24574862.jpg"],
##     "large": ["https://img3.doubanio.com/view/subject/l/public/s24574862.jpg"],
##     "medium": ["https://img3.doubanio.com/view/subject/m/public/s24574862.jpg"]
##   },
##   "alt": ["https://book.douban.com/subject/20429677/"],
##   "id": ["20429677"],
##   "publisher": ["浙江人民出版社"],
##   "isbn10": ["7213052543"],
##   "isbn13": ["9787213052545"],
##   "title": ["大数据时代"],
##   "url": ["https://api.douban.com/v2/book/20429677"],
##   "alt_title": ["Big Data:A Revolution That Will Transform How We Live, Work, and Think"],
##   "author_intro": ["他是十余年潜心研究数据科学的技术权威，他是最早洞见大数据时代发展趋势的数据科学家之一，也是最受人尊敬的权威发言人之一。他曾先后任教于世界最著名的几大互联网研究学府。现任牛津大学网络学院互联网治理与监管专业教授，曾任哈佛大学肯尼迪学院信息监管科研项目负责人，哈佛国家电子商务研究中网络监管项目负责人；曾任新加坡国立大学李光耀学院信息与创新策略研究中心主任。并担任耶鲁大学、芝加哥大学、弗吉尼亚大学、圣地亚哥大学、维也纳大学的客座教授。\n他的学术成果斐然，有一百多篇论文公开发表在《科学》《自然》等著名学术期刊上，他同时也是哈佛大学出版社、麻省理工出版社、通信政策期刊、美国社会学期刊等多家出版机构的特约评论员。\n他是备受众多世界知名企业信赖的信息权威与顾问。他的咨询客户包括微软、惠普和IBM等全球顶级企业；而他自己早在1986年与1995年就担任两家软件公司的总裁兼CEO，由他的公司开发的病毒通用程序，成为当时奥地利最畅销的软件产品。1991年跻身奥地利软件企业家前5名之列，2000年 被评为奥地利萨尔斯堡州的年度人物。\n他也是众多机构和国家政府高层的信息政策智囊。他一直专注于信息安全与信息政策与战略的研究，是欧盟专家之一，也是世界经济论坛、马歇尔计划基金会等重要机构的咨询顾问，同时他以大数据的全球视野，熟悉亚洲信息产业的发展与战略布局，先后担任新加坡商务部高层、文莱国防部高层、科威特商务部高层、迪拜及中东政府高层的咨询顾问。\n所著《大数据》一书是开国外大数据系统研究的先河之作，而在这之前，他已经在《经济学人》上和数据编辑肯尼斯.尼尔-库克耶一起，发表了长达14页的大数据专题
```

```
文章，成为最早洞见大数据时代趋势的数据科学家之一。而他的《删除》一书，同样被认为是关于数据的开创性作品，并且创造了“被遗忘的权利”的概念而在媒体圈和法律圈得到广泛运用。该书获得美国政治科学协会颁发的唐<U+2022>K<U+2022>普赖斯奖，以及媒介环境学会颁发的马歇尔<U+2022>麦克卢汉奖。同时受到《连线》、《自然》《华尔街日报》《纽约时报》等各大权威媒体广泛好评。"],
##   "summary": ["《大数据时代》是国外大数据研究的先河之作，本书作者维克托<U+2022>迈尔<U+2022>舍恩伯格被誉为“大数据商业应用第一人”，拥有在哈佛大学、牛津大学、耶鲁大学和新加坡国立大学等多个互联网研究重镇任教的经历，早在2010年就在《经济学人》上发布了长达14页对大数据应用的前瞻性研究。\n维克托<U+2022>迈尔<U+2022>舍恩伯格在书中前瞻性地指出，大数据带来的信息风暴正在变革我们的生活、工作和思维，大数据开启了一次重大的时代转型，并用三个部分讲述了大数据时代的思维变革、商业变革和管理变革。\n维克托最具洞见之处在于，他明确指出，大数据时代最大的转变就是，放弃对因果关系的渴求，而取而代之关注相关关系。也就是说只要知道“是什么”，而不需要知道“为什么”。这就颠覆了千百年来人类的思维惯例，对人类的认知和与世界交流的方式提出了全新的挑战。\n本书认为大数据的核心就是预测。大数据将为人类的生活创造前所未有的可量化的维度。大数据已经成为了新发明和新服务的源泉，而更多的改变正蓄势待发。书中展示了谷歌、微软、亚马逊、IBM、苹果、facebook、twitter、VISA等大数据先锋们最具价值的应用案例。"],
##   "ebook_price": ["39.99"],
##   "price": ["49.90元"]
## }
```

```
#因为我们知道返回的内容是JSON格式，所以我们加载jsonlite软件包，以便用清晰的格式把内容打印出来。
#我们把这个JSON内容存储起来。

result <- fromJSON(content(response, as="text"))

# Sys.setlocale('LC_ALL','Chinese')
# wp <- readLines(url,warn="F") #下载JSON页面,这一行有错误啊！！！
# ps <- fromJSON(wp)
# title<-ps$title
# publisher<-ps$publisher
# isbn10<-ps$isbn10
# price<-ps$price
# catalog<-ps$catalog
# avgRate<-ps[["rating"]]$average
# result<-c(title, publisher, isbn10, price, avgRate, catalog)
# names(result)<-c("title", "publisher", "isbn10", "price", "avgRate","catalog")
# result

# fileConn<-file("output.txt")
# writeLines(c("Hello","World"), fileConn)
# close(fileConn)
#write(result, file="图书信息.txt")
#-------write functions --------
##book information extraction
# dbook<- function(bookid){
# url=paste0("https://api.douban.com/v2/book/", paste(bookid))
# wp <- readLines(url, warn="F")
# Ps <- fromJSON(wp)
# title<-ps$title
# publisher<-ps$publisher
# isbn10<-ps$isbn10
# price<-ps$price
# catalog<-ps$catalog
# avgRate<-ps[["rating"]]$average
# result<-c(title, publisher, isbn10, price, avgRate, catalog)
# names(result)<-c("title", "publisher", "isbn10", "price", "avgRate", "catalog")
# write(result, file="图书信息.txt")
# return (result)
# }
# dbook(20429677)
```

我们借助R语言程序正确地收集到了豆瓣网上我们感兴趣的信息。进步阅读 API手册发现，更高级的采集功能必须注册开发者账号(甚至还有专用的SDK软件包)，如感兴趣，可以课外自行进行更加深入的学习。另外，rvest、httr等软件包也是R语言抓取网页数据的常用选择，rvest包的帮助文档介绍是“容易地收割(抓取)网页”，可见其功能之强大，强烈建议关注与学习!

以天龙八部作为离线文本数据

使用jiebaR分词包进行中文分词，去停用词。

构建词频统计表，最后利用wordcloud进行词云图可视化展示。

```
library(jiebaR)
```

```
## Loading required package: jiebaRD
```

```
#if(!require("wordcloud")){install.packages("wordcloud")};
library(RColorBrewer)
library(wordcloud)
engine<-worker()
# setwd("D:/R/test")
xajh<-read.table("天龙八部.txt",header=F,sep="\t",colClasses="character")
words<-engine<=xajh$V1
words1<-unlist(words)
words1<-words[words!=""]
words2<-words1[nchar(words1)>1 & nchar(words1)<7]
wordFreq25=sort(table(words2),decreasing=T)[1:25];wordFreq25
```

```
## words2
##    说道    什么    自己    虚竹    一个    段誉    武功    不是    萧峰    一声 王语嫣
##    2114    2008    1628    1516    1395    1267    1067     993     965     909     857
## 慕容复    咱们    师父    不知    心中 段正淳    知道 木婉清    出来    叫道    姑娘
##     841     813     782     776     775     741     738     735     732     682     658
##    如何    突然    便是
##     656     654     646
```

```
pal2<-brewer.pal(8,"Dark2")
wordcloud(names(wordFreq25),wordFreq25,min.freq=2,random.order=F,colors=pal2)
```

参考文献

1.新媒体数据挖掘：基于R语言》，深圳大学，王小峰，清华大学出版社，2018年2月。

2.https://www.jianshu.com/p/c2e030187495

3.https://www.jianshu.com/p/8091f86fe1f0

4. https://blog.csdn.net/jytlcl/article/details/88654544 (https://blog.csdn.net/jytlcl/article/details/88654544)
5. https://www.cnblogs.com/xihehe/p/8309023.html (https://www.cnblogs.com/xihehe/p/8309023.html)
6. https://blog.csdn.net/LEEBELOVED/article/details/83790006?ops_request_misc=%257B%2522request%255Fid%2522%253A%252215848452821972484644457 6%2522%252C%2522scm%2522%253A%25222014 (https://blog.csdn.net/LEEBELOVED/article/details/83790006?ops_request_misc=%257B%2522request%255Fid%2522%253A%252215848452821972484644457 6%2522%252C%2522scm%2522%253A%25222014task