Python与金融数据挖掘(7)

文欣秀

wenxinxiu@ecust.edu.cn

案例分析


people.com.cn
入空间
创新实践育人计划
Welcome to Pyth...
PyTorch（二）—...
Python Program...
New Tab
首页
党政
要闻
观点
互动
可视化
地方
民文
English
合作网站
举报专区
任平：中国特色社会主义是实现中华民族伟大复兴的必由之路
坚持党的全面领导是坚持和发展中国特色社会主义的必由之路 沿着必由之路夺取新的更大胜利 牢记“五个必由之路”，引领新时代中国特色社会主义事业行稳致远
5/9
走进中国羌族博物馆 了解“云朵上的民族”
总书记的人民情怀|总书记的"三农"情怀
在构建新时代中非命运共同体之路上携手同行
李强在海南调研：加快建设中国特色自由贸易港
韩正会见来华的部分跨国公司负责人
新时代新征程新伟业|上海建设科技创新中心
西藏走在乡村振兴大道 纪念百万农奴解放 综述
学习宣传贯彻党的二十大精神
今日谈|不断以中国新发展为世界提供新机遇
河北在对接京津、服务京津中加快发展 专访
亚洲发展融资面临挑战 呼吁政府社会共行动
上海聚焦高水平人才建设 激发创新创造活力
上海横沙新洲预计2035年建成 总产值超300亿

简单爬虫程序基本步骤

- 获取网页源码
- 根据源码中所在链接特点写出正则表达式
- 用正则对象匹配获取目标链接
- 用循环结构遍历目标链接并自动下载信息

Requests对象的属性

属性	说明
r. text	网页响应内容的字符串形式
r. encoding	猜测网页响应内容编码方式
r. apparent_encoding	从网页内容分析出编码方式
r. content	网页响应内容的二进制形式
r.status_code	200表示连接成功，404表示失败

爬取网页

```
import requests
url="http://www.taobao.com"
r=requests.get(url)
print(r.status_code)
r.encoding=r.apparent_encoding
data=r.text
print(data)
```

爬取单张图片

```
import requests
url="http://www.people.com.cn/NMediaFile/2023/0329/MAIN202303
291337317271860541036.jpg"
r=requests.get(url)
data=r.content
fobj=open("result.jpg","wb")
fobj.write(data)
fobj.close()
```

判断邮件地址是否合法

假定某E-mail地址由三部分构成：英文字母或数字（1～10个字符）、“@”、英文字母或数字（1～10个字符）、“.”，最后以com或org结束，其正则表达式为：

’^[a-zA-Z0-9]{1,10}@[a-zA-Z0-9]{1,10}.(com|org)$’

输入E-mail地址测试串，忽略大小写，输出判断是否符合设定规则。

字符串匹配表

模式	描述
\d	匹配一个数字字符
\w	匹配一个字母、数字及下划线字符
\s	匹配一个空白字符
.	匹配一个任意字符，换行符除外
\n	匹配一个换行符
\t	匹配一个制表符

字符串匹配表

模式	描述
*	匹配前面的字符0次或n次
+	匹配前面的字符1次或n次
?	匹配前面的字符0次或1次
()	匹配括号内表达式，也表示一个组
{n}	匹配n个字符
[]	表示字符范围，方括号中只能取一个

re库的内置函数

- match()：用于从起始位置匹配
- search()：搜索整个字符串，返回第一次出现位置
- findall()：以列表形式返回全部能匹配的字串
- compile()：创建一个正则表达式对象

内置函数示例

```
>>> import re
>>> string="I love ECUST. ECUST loves me."
>>> re. match("I", string)
>>> re. match("ECUST", string)
>>> re. search("I", string)
>>> re. search("ECUST", string)
>>> re. findall("ECUST", string)
```

正则表达式修饰符含义

修饰符	描述
re.I	使匹配对大小写不敏感
re.L	做本地化识别（locale-aware）匹配
re.M	多行匹配，影响 ^ 和 $
re.S	使 . 匹配包括换行在内的所有字符
re.U	根据Unicode字符集解析字符。这个标志影响 \w, \W, \b, \B.
re.X	该标志通过给予你更灵活的格式以便你将正则表达式写得更易于理解。

内置函数示例

```
>>> import re
>>> string="I love it. It loves me."
>>> re. findall("it", string)

>>> test=re. compile("it", re.I)
>>> re. findall(test, string)
>>> test. findall(string)
```

判断邮件地址是否合法

```
import re
p=re. compile('^[a-zA-Z0-9]{1,10}@[a-zA-Z0-9]{1,10}.(com|org)$',re.I)
while True:
    s=input("请输入测试E-mail地址(输入 '0' 退出程序):\n")
    if s=='0':
        break
```

判断邮件地址是否合法

```
m=p. match(s)    #m=re.match(p,s)
print(m)
if m:
    print('%s符合规则' %s)
else:
    print('%s不符合规则' %s)
```

字符串匹配模式

字符串匹配通常分为贪婪匹配和非贪婪匹配。

贪婪匹配：是一种尽可能多地匹配字符的匹配模式，是正则表达式默认匹配方式

例如：在匹配字符串"aaaaaa"时，'**a{2,4}**'默认匹配4个'a'

非贪婪匹配：一种尽可能少地匹配字符的模式

例如：在匹配字符串"aaaaaa"时，'**a{2,4}?**'匹配2个'a'

贪婪匹配模式示例

```
import re
res = '文本A百度新闻文本B，新闻标题\
文本A新浪财经文本B，文本A搜狐新闻文本B新闻网址'
p_source = '文本A(.*)文本B'
source = re.findall(p_source, res)
print(source)
```

['百度新闻文本B，新闻标题 文本A新浪财经文本B，文本A搜狐新闻']

非贪婪匹配模式示例

```
import re
res = '文本A百度新闻文本B，新闻标题 \
文本A新浪财经文本B，文本A搜狐新闻文本B新闻网址'
p_source = '文本A(.*?)文本B'
source = re.findall(p_source, res)
print(source)
```

['百度新闻', '新浪财经', '搜狐新闻']

思考题一

已知result= re. search(r'b.*a', 'banana')，则匹配结果是（　　）

A 'banana'

B 'ba'

C 'ban'

D 'bana'

思考题二

已知result= re. search(r'b.*?i', 'www.blibli.com')，则匹配结果是（　）

A 'blibli'

B 'blibli.com'

C 'www.blibli.com'

D 'bli'

正则表达式示例

```
import re
s='''<img src="C:\\XH.jpg" width="300"/>
      <img src="C:\\FX.jpg" width="300"/>'''
result=re.findall('<img src="(.*?)" ', s)
print(result)
```

['C:\\XH.jpg', 'C:\\FX.jpg']

正则表达式示例

```
import re
s=''<img src="C:\\XH.jpg" width="300"/>
      <img src="C:\\FX.jpg" width="300"/>''
result=re.findall('<img src="(.*?)" width="(.*?)"', s)
print(result)
```

[('C:\\XH.jpg', '300'), ('C:\\FX.jpg', '300')]

豌豆荚正则示例

```
page=u'''
<li class="parent-cate">
<a class="cate-link" c>旅游出行</a><ul>
<li class="child-cate"><a href="http://www.wandoujia.com/category/596"
title="综合旅游服务">综合旅游服务</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/598"
title="攻略">攻略</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/600"
title="酒店·住宿">酒店·住宿</a></li>
</ul></li>
'''
```

豌豆荚正则示例

```
#coding=utf-8
import re
page=u'''<li class="parent-cate">
<a class="cate-link" c>旅游出行</a>
<ul><li class="child-cate"><a href="http://www.wandoujia.com/category/596" title="综合旅游服务">
综合旅游服务</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/598" title="攻略">攻略</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/600" title="酒店·住宿">酒店·住
宿</a></li>
</ul></li>
'''
data=re.findall('href="(.*?)"\s+title="(.*?)"', page)
print(data)
```

豌豆荚正则存入文件

```
#coding=utf-8
import re
page=u''' <li class="parent-cate">
<a class="cate-link" c>旅游出行</a>
<ul><li class="child-cate"><a href="http://www.wandoujia.com/category/596" title="综合旅游服务">综合旅游服务</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/598" title="攻略">攻略</a></li>
<li class="child-cate"><a href="http://www.wandoujia.com/category/600" title="酒店·住宿">酒店·住宿</a></li>
</ul></li>
'''
data=re.findall('href="(.*?)"\s+title="(.*?)"', page)
fobj=open("test.txt", "w")
for line in data:
    fobj. write(line[0]+"  "+line[1]+"\n")
fobj. close()
```

爬取人民网链接和标题

```
import requests
import re
url="http://www.people.com.cn"
html=requests. get(url)
html. encoding=html. apparent_encoding
data=html. text
#print(data)
```

爬取人民网链接和标题

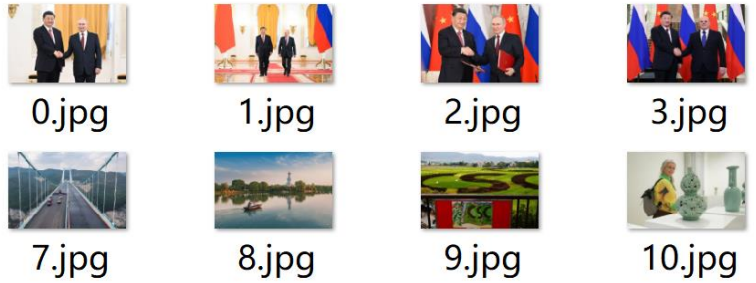
```
reg=r'<a href="http://.*?.html" target="_blank">.*?</a>'
urls=re.findall(reg, data)
print(urls)
fobj=open("result.csv",'w', encoding="gb2312")
for titu in urls:
    result=re.split('" target="_blank">',titu)
    fobj.write(result[0][9:]+','+result[1][:-4]+'\n')
fobj.close()
```

爬取新闻图片素材

```
import requests
import re
url='http://www.people.com.cn'
r=requests.get(url)
r.encoding=r.apparent_encoding
paper=r.text
regrule=r' target=_blank><img src="(.*\.jpg)"'
urls=re.findall(regrule, paper)
```

爬取新闻图片素材

```
x=0
for imgu in urls:
    flo=requests.get(url+imgu)
    filename='C:\\Users\\jluwe\\Desktop\\result\\%d.jpg' % x
    f=open(filename,'wb')
    filesize=f. write(flo. content)
    x+=1
```



各个网站的Robots协议

网站名称	Robtos协议
百度	https://www.baidu.com/robots.txt
京东	https://www.jd.com/robots.txt
QQ	https://www.qq.com/robots.txt
..	..

思　考

```
import re
res = '<h3 class="c-title"><a href="网址" data-click="{一堆英文}"><em>阿里巴巴</em>代码竞赛现全球首位AI评委 能为代码质量打分</a>'
p_title = '<h3 class="c-title">.*?>(.*?)</a>'
title = re.findall(p_title, res)
print(title)
```

['阿里巴巴代码竞赛现全球首位AI评委 能为代码质量打分']

re.sub()方法

```
import re
title  =  ['<em>阿里巴巴</em>代码竞赛现全球首位AI评委能为代码质量打分']
result = re.sub('<.*?>', '', title[0])
print(result)
```

阿里巴巴代码竞赛现全球首位AI评委 能为代码质量打分

淘宝爬虫案例

大数据定义

大数据（big data）： 一种数据规模大到在获取、存储、管理、分析方面大大超出了传统数据库软件工具能力范围的数据集合。它包括**结构化、半结构化和非结构化数据**，非结构化数据越来越成为数据的主要部分。

结构化数据

	A	B
1	学号	姓名
2	20002370	权泽睿
3	20002512	张宸煜
4	20002513	费诚成
5	20002514	李凌瑶
6	20002515	刘宇晨
7	20002516	邓庚麒
8	20002517	王飞扬
9	20002519	黄志鹏
10	20002520	周学勤
11	20002521	诸建飞

半结构化数据

```
<person>
    <name>A</name>
    <age>13</age>
    <gender>female</gender>
</person>
```

非结构化数据

大数据的特点

Volume(大量)： 存储单位至TB、PB、EB级别

Velocity(高速)： 处理速度**快**、时效性要求**高**

Variety(多样)： **结构化**、**半结构化**及**非结构化**

Value(价值)： 数据价值密度**低**、需要**算法**挖掘

Python支持的数据库

- SQLite
- MySQL
- MongoDB
- Redis
- Microsoft SQL Server 2000
- ….

SQLite数据库连接

- 和数据库建立连接
- 执行sql语句，接收返回值
- 关闭数据库连接

谢　谢