# -\*- coding: utf-8 -\*-

"""

Created on Tue Jun 26 09:43:02 2018

@author: maxmaxwfl

"""

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

import time

import urllib

import re

import os

import urllib2

import pandas as pd

from bs4 import BeautifulSoup,Comment

import datetime

import hashlib

#标题过滤，返回标题中所包含的标签

def head\_filter(headline):

#待搜索关键词列表

keyword\_list = [u"社区",u"坊会",u"坊众福利会",u"街坊会",u"街坊福利会",u"街坊福利事务促进会",u"街坊福利委员会",

u"社会企业",u"业主立案法团",u"互助委员会",u"业主委员会"]

labels=[]

for i in range(len(keyword\_list)):

if keyword\_list[i] in headline:

labels.append(keyword\_list[i])

return labels

#时间过滤，筛选出2018,2017,2016年的新闻，其中日期以'/'分割,例如对于2018/07/05,split函数作用就是变为["2018","07","05"]的列表

def time\_filter(times):

year = times.split('/')[0]

if(year==u'2018' or year==u'2017' or year==u'2016'):

return True

else:

return False

#提取某一具体新闻网页中的各个信息

#提取某一具体新闻网页中的各个信息

def get\_link\_content\_text(url):

html=urllib.urlopen(url)

#使用beautifulsoup解析网页内容

sub\_sp=BeautifulSoup(html,"html.parser",from\_encoding="gb18030")

#设置唯一文件名

#获取今天的日期

today = datetime.date.today().strftime('%Y%m%d')

#加上CN前缀

a\_id = "CN"+today

#使用全局变量

global page\_nums

#需根据实际内容修改类型

#作者

author = ""

#图片

all\_img=[]

#附件MD5

attach\_md5 = []

#附件源文件

attach\_ori = []

"""

正文部分:需根据标签修改find\_all中的标签名

html\_context 保存的是带html原代码的原文

all\_context 是通过正则表达式将html\_context中所有<>去掉得到内容

"""

all\_context = ""

#找到网页中 所有td标签，class为'article-content'的内容

context = sub\_sp.find('td',attrs={'class':'article-content'})

#去除style标签

[s.extract() for s in context('style')]

if(context):

"""

html\_context =""

#因为context结果是个数组，因此需用for循环读取context中所有元素

for line in context:

html\_context +=str(line)

#去除html中的所有<>内容

html\_label = re.compile('<.\*?>')

all\_context = html\_label.sub("",html\_context)

#去除空格

all\_context = all\_context.strip()

#得到摘要为正文的第一段

abstract = all\_context.split("。")[0]

"""

#由刘畅学弟改进，使用了更简洁的语句

html\_context = context.prettify()

all\_context = context.text.strip()

abstract = all\_context.encode("utf-8").split("。")[0]

else:

all\_context=""

abstract=""

html\_context=""

#设置储存路径

home\_dir = "e:/"+today

page\_str = str(page\_nums)

a\_id += page\_str.zfill(6)

#设置每个网页的路径

dir\_path = os.path.join(home\_dir,a\_id)

#如果路径没有该文件夹，则创建新的

if not os.path.exists(home\_dir):

os.mkdir(home\_dir)

#如果子文件夹路径不存在，则创建文件夹

if not os.path.exists(dir\_path):

os.mkdir(dir\_path)

#附件部分

#找出所有包含doc,xls,docx,pdf,txt,xlsx,csv,xml的路径

attach\_link = sub\_sp.find(href=re.compile("doc|xls|docx|pdf|txt|xlsx|csv|xml"))

#如果有附件

if(attach\_link):

#获取附件路径

file\_path = attach\_link["href"]

#对于路径，如果没有http，则将原网站homepage+路径作为附件路径

if 'http' not in file\_path:

file\_path = homepage+ '/' + file\_path

#获取附件名字

file\_name = file\_path.split('/')[-1]

#附件名字加""后加入附件原名称数组

attach\_ori.append("\"%s\""%file\_name)

file\_before = file\_name.split('.')[0]

file\_after = file\_name.split('.')[1]

#计算文件名MD5值

m2 = hashlib.md5()

m2.update(file\_before)

md5\_file\_before = m2.hexdigest()[8:-8]

md5\_filename = md5\_file\_before + '.'+ file\_after

#md5附件名字加""后加入附件原名称数组

attach\_md5.append("\"%s\""%md5\_filename)

resource = requests.get(file\_path)

#得到附件存储路径

file\_save\_path = dir\_path + '/' +md5\_filename

#将路径编码为utf-8

file\_save\_path = file\_save\_path.encode('utf-8')

file\_path = file\_path.encode("utf-8")

with open(file\_save\_path, mode="wb") as fh:

fh.write(resource.content)

#图片部分

#用beautifulsoup解析html\_context

sub\_c = BeautifulSoup(html\_context,"html.parser",from\_encoding="gb18030")

#找出正文中所有的img标签

imgs = sub\_c.select("p img")

for i in range(len(imgs)):

#读取img标签中的src标签，获取图片路径

imgpath=imgs[i]['src']

#获取src路径中'/'后的最后一截，如对于/edit/uploadfile/20180709161440651.jpg，提取的是20180709161440651.jpg

imgname = imgpath.split('/')[-1]

#去除imgname中的'?'

imgname= re.sub(r'\?',"",imgname)

#如果字符串长度大于4,且倒数第四位不是'.'，则需手动加.jpg

if (len(imgname)>=4):

if(imgname[-4] != '.'):

imgname += '.jpg'

else:

imgname += '.jpg'

#对图片进行MD5编码

#将图片名按照'.'进行分割

imgname\_before = imgname.split('.')[0]

imgname\_after = imgname.split('.')[1]

m2 = hashlib.md5()

m2.update(imgname\_before)

md5\_before = m2.hexdigest()[8:-8]

md5\_imgname = md5\_before + '.'+ imgname\_after

all\_img.append("\"%s\""%md5\_imgname)

#将html\_context中的img标签替换为 image:(MD5加密后图片名)

img\_str = "image:(%s)"%md5\_imgname

#将img\_str编码为utf-8

img\_str = img\_str.encode('utf-8')

#使用正则表达式将html\_context内的img标签替换为img\_str字符串，re.sub中的count参数意思为一次for循环替换一个<img>标签

html\_context = re.sub("<img .\*?>",img\_str,html\_context,count=1)

#对于路径，如果没有http，则将原网站homepage+路径作为图片路径

if 'http' not in imgpath:

whole\_img = homepage+ '/' + imgpath

else:

whole\_img = imgpath

#得到图片存储路径

path = dir\_path + '/' +md5\_imgname

#将路径编码为utf-8

path = path.encode('utf-8')

#将图片url路径编码为utf-8

whole\_img = whole\_img.encode('utf-8')

#保存图片

try:

urllib2.urlopen(whole\_img,timeout=1)

urllib.urlretrieve(whole\_img,path)

except IOError,UnicodeDecodeError:

print whole\_img

return a\_id,abstract,author,html\_context,all\_context,attach\_md5,attach\_ori,all\_img

#找出要提取的新闻网页url

def get\_content(bs,save\_context):

global page\_nums

#需要更改

source= u"上海市浦东新区周家渡街道"

country = u"中国"

context\_city="上海"

city = u"上海"

constitute = u"上海市浦东新区周家渡街道"

article\_type ='html'

small\_img = ""

#找出所有新闻链接和对应时间

link\_a\_all = bs.find\_all('td',attrs={'width':'561'})

time\_all = bs.find\_all('td',attrs={'class':'sqdt2'})

for i in range(len(link\_a\_all)):

one\_context =[]

#找出新闻表格中的<a>标签

link\_a = link\_a\_all[i].a

#用.string提取时间标签中的时间，需更改string切片的位置

time\_str = time\_all[i].string[1:-2]

if(link\_a.string and time\_str):

#得到新闻标题中的标签

labels = head\_filter(link\_a.string)

#判断时间是否为2018,2017,2016年

time\_flag = time\_filter(time\_str)

#如果标签中有内容，而且时间符合条件

if (len(labels)>0 and time\_flag==True):

#提取符合条件新闻的url，需要修改

sub\_url = link\_a['href']

whole\_url = homepage + '/' + sub\_url

#print whole\_url

a\_id,abstract,author,html\_context,all\_context,attach\_md5,attach\_ori,all\_img=get\_link\_content\_text(whole\_url)

#print whole\_url

#唯一串行号 英文标题 中文标题 英文摘要 中文摘要

#作者 发布单位 发布时间 标签 英文正文含标签 英文正文 中文正文含标签 中文正文

#附件名称（md5存储） 附件原名称 文章内容图片 文章内容城市 信息源网址 信息源国家

#信息源城市 信息源机构 文章类型 搜索关键词 缩略图

one\_context.append(a\_id)

one\_context.append("")

one\_context.append(link\_a.string)

one\_context.append("")

one\_context.append(abstract)

one\_context.append(author)

one\_context.append(source)

one\_context.append(time\_str)

one\_context.append(labels)

one\_context.append("")

one\_context.append("")

one\_context.append(html\_context)

one\_context.append(all\_context)

one\_context.append(attach\_md5)

one\_context.append(attach\_ori)

one\_context.append(all\_img)

one\_context.append(context\_city)

one\_context.append(whole\_url)

one\_context.append(country)

one\_context.append(city)

one\_context.append(constitute)

one\_context.append(article\_type)

one\_context.append(labels)

one\_context.append(small\_img)

save\_context.append(one\_context)

page\_nums +=1

else:

print 'not added'

return save\_context

#需修改，根据不同网站来改homepage

homepage='http://www.zjd.gov.cn'

#需根据每个网站具体数目来设置下一个网站的page\_nums,如网站1有10个，则对于网站2则需设page\_nums=11,类似于offset的概念

page\_nums = 0

test=[]

#设置selenium浏览器为chrome浏览器

driver = webdriver.Chrome()

#进入主页

driver.get("http://www.zjd.gov.cn/index.asp")

#根据搜索框id = 'searchinput'，找到搜索框位置

elem =driver.find\_element\_by\_id("searchinput")

#清楚搜索框中内容

elem.clear()

#输入社区

elem.send\_keys(u"社区")

#找到搜索按钮，并点击

driver.find\_element\_by\_class\_name("input\_04").click()

#用beautifulsoup解析按钮点击后的网页

sub\_sp=BeautifulSoup(driver.page\_source,"html.parser",from\_encoding="gb18030")

test = get\_content(sub\_sp,test)

#人工看发现只有关键词'社区'有搜索结果，其他几个关键词都没有搜索结果，而且搜“社区”后从第十四页开始都是2015年的新闻了，因此只用再点12次下一页

for i in range(13):

#找到下一页的按钮，并点击

driver.find\_element\_by\_link\_text(u"下一页").click()

sub\_sp=BeautifulSoup(driver.page\_source,"html.parser",from\_encoding="gb18030")

#print sub\_sp

test = get\_content(sub\_sp,test)

today = datetime.date.today().strftime('%Y%m%d')

#设置csv文件保存路径，一个网站一个csv，以后网站的csv文件改名为pachong1.csv,pachong2.csv，。。。。以此类推

csv\_path = 'e:/'+today + '/'+'pachong.csv'

#将数组转为pandas dataframe

df1 = pd.DataFrame(test,columns=["唯一串行号","英文标题","中文标题","英文摘要","中文摘要",

"作者","发布单位","发布时间","标签","英文正文含标签","英文正文","中文正文含标签","中文正文",

"附件名称（md5存储）","附件原名称","文章内容图片","文章内容城市","信息源网址","信息源国家",

"信息源城市","信息源机构","文章类型","搜索关键词","缩略图"])

#去除重复的“中文标题”

df1 = df1.drop\_duplicates("中文标题")

#将dataframe写入文件

df1.to\_csv(csv\_path,encoding='utf-8')

蓝色的部分为需要修改的路径，我在windows下用的是e:/+爬虫日期，如果在linux和mac系统上请自行修改路径

绿色部分为需要根据网页具体标签或架构来修改的地方

for i in range(13):

#找到下一页的按钮，并点击

driver.find\_element\_by\_link\_text(u"下一页").click()

sub\_sp=BeautifulSoup(driver.page\_source,"html.parser",from\_encoding="gb18030")

#print sub\_sp

test = get\_content(sub\_sp,test)

NOTES 1.对于上面这段代码而言，因为我爬的这个网站“下一页”是用js脚本写的，点击下一页后网站的url不发生任何变化，因此需要每次点击下一页的按钮。但如果你们爬的那个网站点“下一页”后，url会发生变化，则可以直接解析url。如

for i in range(5,11):

new\_url = 'http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq\_dtyw/List/list\_%i.htm'%i

html=urllib.urlopen(new\_url)

sub\_sp = BeautifulSoup(html,"html.parser",from\_encoding="gb18030")

test = get\_content(sub\_sp,test)

这种方法就是直接爬取http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq\_dtyw/List/list\_5.htm

<http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq_dtyw/List/list_6.htm>

<http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq_dtyw/List/list_7.htm>

<http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq_dtyw/List/list_8.htm>

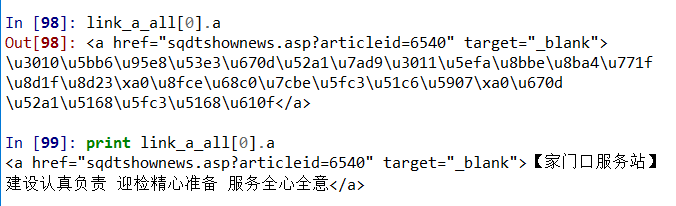
<http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq_dtyw/List/list_9.htm>

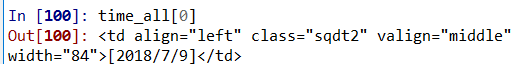
<http://www.jinqiao2010.gov.cn/website/html/pdjq/jrjq_dtyw/List/list_10.htm>

这几个网页的内容，而无需用到selenium，从而加快程序运行速度。不过有搜索框的网站程序运行时间不长，因此可改可不改

NOTES2 为了避免浪费时间，可使用get\_link\_content\_text（url）函数先进行测试，看看对于单个网站这个函数会不会出现问题，再测试一下get\_content函数中的link\_a\_all = bs.find\_all('td',attrs={'width':'561'})

time\_all = bs.find\_all('td',attrs={'class':'sqdt2'})

看看这两个有没有问题，然后可以运行一下，然后在spyder控制台直接输入变量名，看看是不是理想的结果，如



根据这些结果修改代码后再执行全部代码