目 录

[1、总体说明 1](#_Toc385422271)

[2、详细说明 1](#_Toc385422272)

[2.1 下载模块 1](#_Toc385422273)

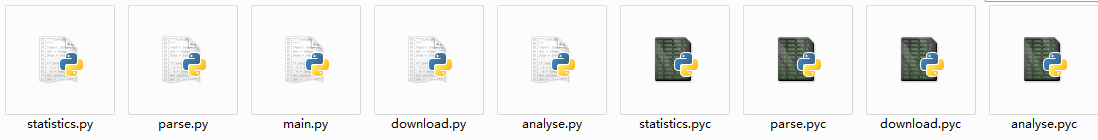
[2.2 解析模块 3](#_Toc385422274)

[2.3 分析模块 5](#_Toc385422275)

[2.4 统计模块 7](#_Toc385422276)

# 1、总体说明

程序代码主要包括如图所示的python脚本文件。其中main.py为主文件，download.py为下载模块，parse.py为解析模块，analyse.py为分析模块，statistics.py为统计模块。各个模块的功能分别为：下载模块从搜索引擎中爬取关键字的搜索结果html页面；解析模块解析结果页面，得到各个结果条目的标题和来源url信息，并根据条目的来源url下载对应的html页面；分析模块通过分析下载的html页面，判断该页面是否是跟关键词相关的（是否为碳信息披露情况）；最后的统计模块统计判断结果，按年份和季度汇总数据。



程序中主要使用urllib2模块进行http请求并返回html页面；使用BeautifulSoup解析html页面，或者结合chardet判断并转换html页面编码；使用正则表达式匹配html页面中的特定模式；使用readability对html页面进行去噪，排除广告评论等干扰，找到关键内容。

程序使用bing搜索引擎，元搜索url为：http://cn.bing.com/search?q=%s&qs=n&sc=8-3&

sp=-1&sk=&first=%d，每个搜索结果页面共10个条目，这里取前5页结果共50个条目作为测试。设置系统默认编码为utf-8。

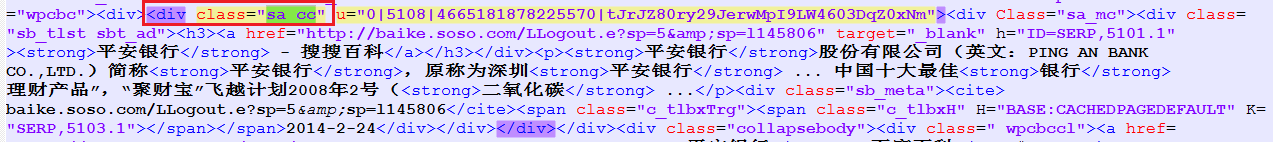
# 2、详细说明

下面对各个模块的功能实现和运行结果分别做说明：

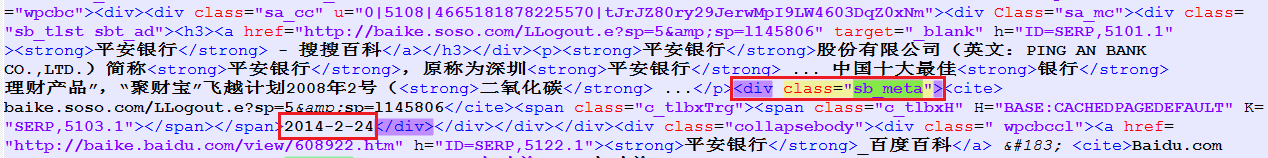
## 2.1 下载模块

**1、功能实现**

下载模块采用python的urllib2库进行http请求，返回请求内容并保存为html页面。为按时间对结果进行统计，这里在下载结果页面的同时统计每个条目的更新时间：通过BeautifulSoup查找条目所在的<div class="sa\_cc">标签，标签如图所示：



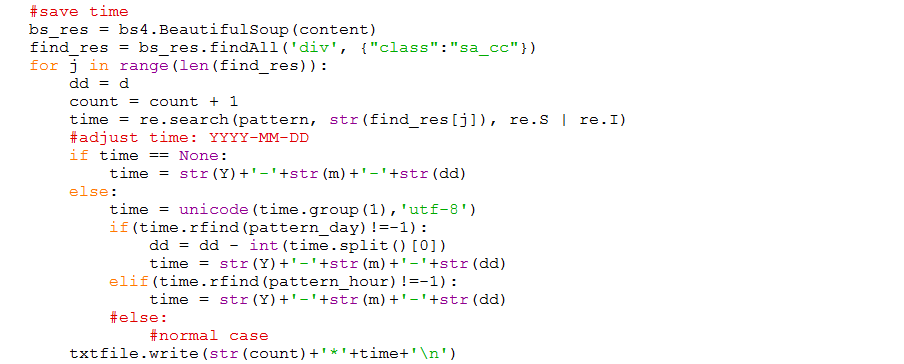
然后使用正则表达式匹配标签内容中<div class="sb\_meta">的标签，该标签内容包含条目的更新时间。标签如图所示：



正则表达式如图所示：



此外条目的更新时间有两种：一种是标准时间格式，例如“2013-12-1”；另一种是以当前时间为基准的相对时间，例如“2 天前”、“1 天前”，“12 小时前”的等形式。根据包含的“天前”和“小时前”这类关键词，将相对时间转换为标准时间格式：例如当前时间为2014-4-10，条目的更新时间为“2 天前”，则转换为标准时间格式为2014-4-8。相关代码如图所示：



**2、运行结果**

以关键词“平安银行+二氧化碳”为例子。下载模块得到的搜索结果html页面源码如下：



搜索结果html页面显示如下：

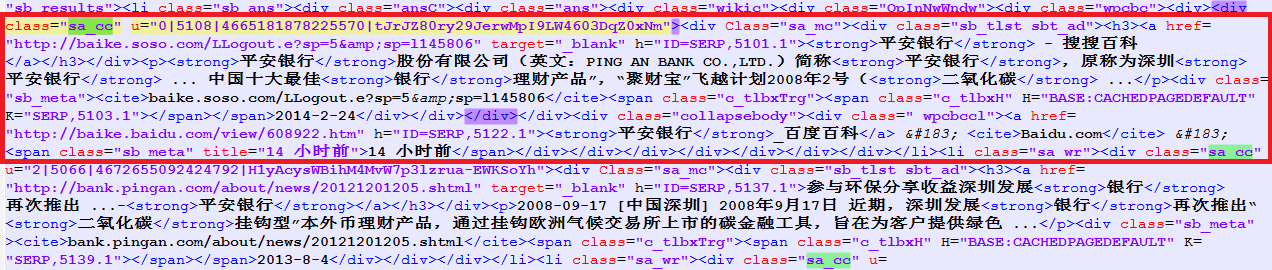




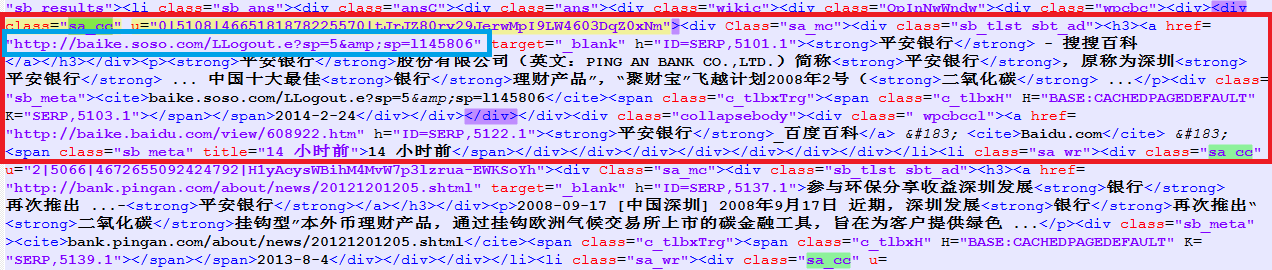
## 2.2 解析模块

**1、功能实现**

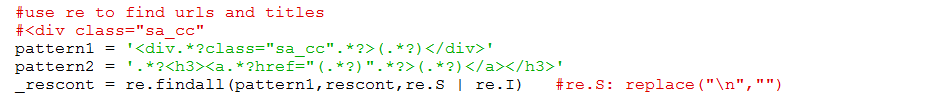
解析模块采用正则表达式匹配html页面中<div class="sa\_cc">标签来定位每个条目的内容所在，然后在该标签内容中匹配<h3><a href="…"></a></h3>标签，href的属性值就是条目的来源url。html页面中每个条目的内容即是前后2个<div class="sa\_cc">标签之间的内容，如图所示：



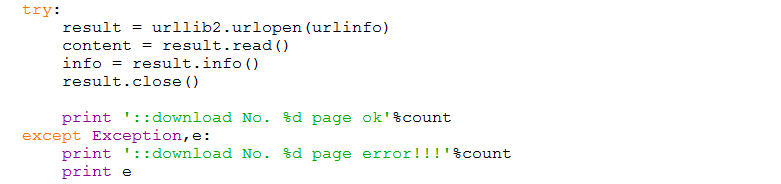
条目的来源url如图所示：



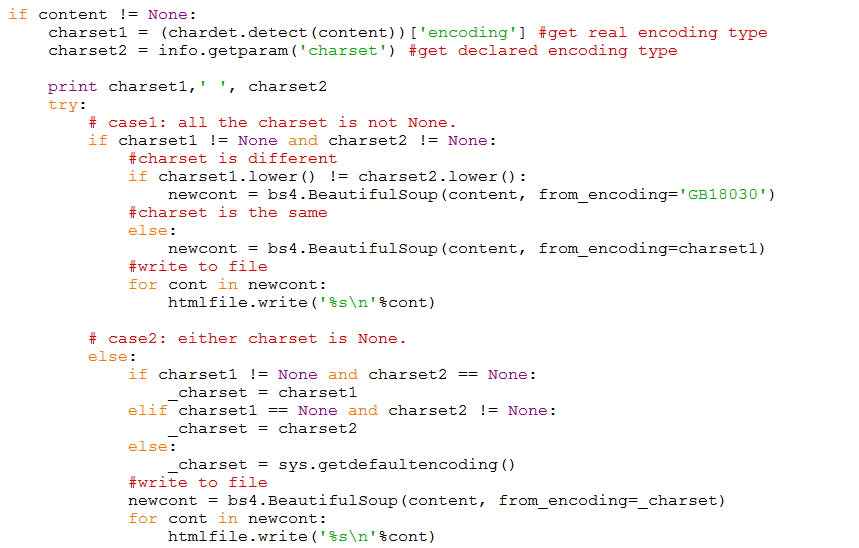
对应的正则表达式如图所示：



获取到条目的来源url后，采用urllib2发起请求，返回并保存条目的原始html页面。相关代码如图所示：

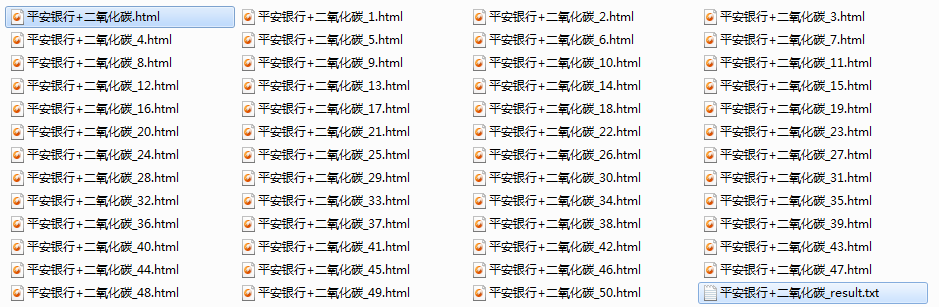


此处由于网页的实际编码有可能超出标示的编码范围，为了保证不出现乱码，做了如下处理：首先根据chardet判断html页面内容的实际编码，同时根据请求返回的info内容判断网页的标示编码。如果两个编码都不为空且不相同，则将页面编码扩展到最大范围的GB18030编码；如果两个编码有一个为空，则按照非空的编码来对html页面内容进行编码并保存；如果两个编码均为空，则按照系统默认编码来对html页面内容进行编码并保存（这里设置系统默认编码为utf-8）。相关代码如图所示：

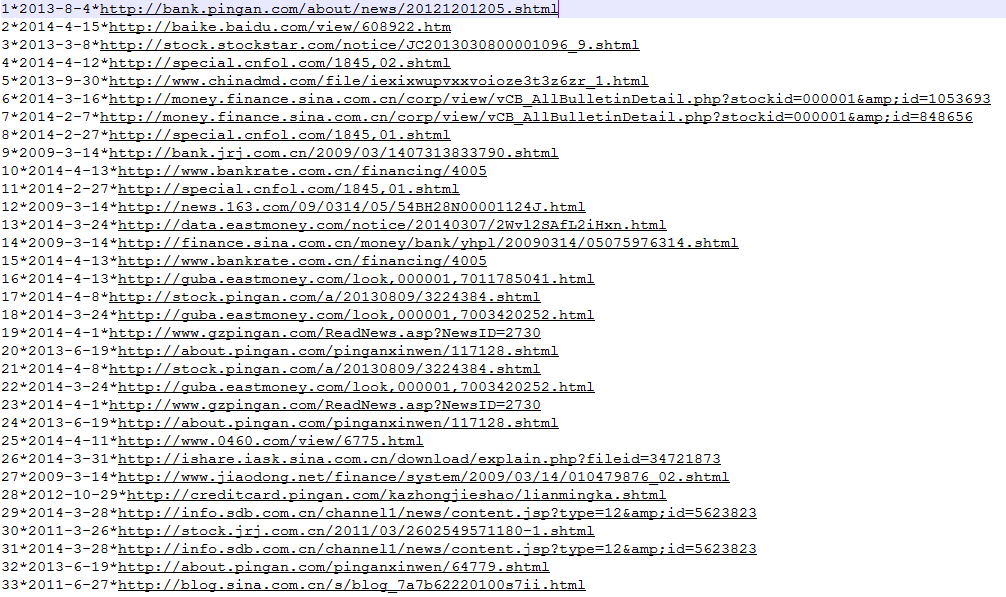


**2、运行结果**

以关键词“平安银行+二氧化碳”为例子。下载得到的原始html页面如图所示：

****

解析得到的每个来源url存储于文本中（这里是上图中对的“平安银行+二氧化碳\_result.txt”）。其内容如图所示：

****

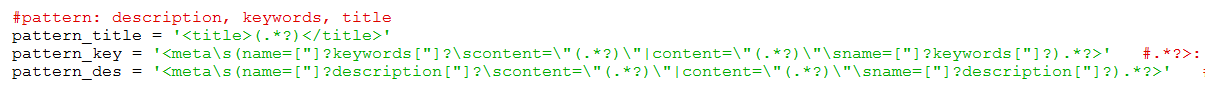
## 2.3 分析模块

**1、功能实现**

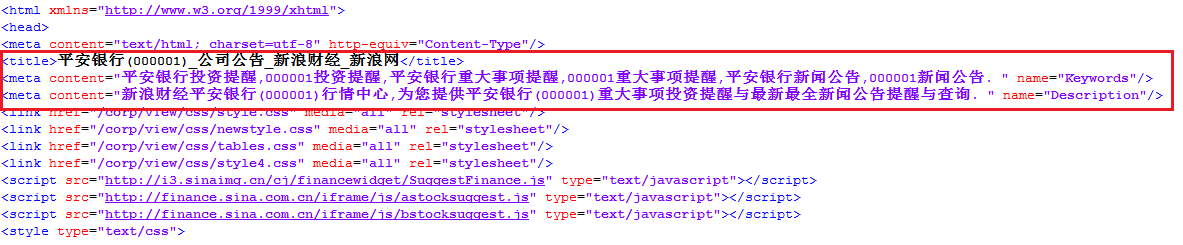
分析模块对每个搜索结果条目对应的原始html页面进行分析，判断是否与关键词相关。实现过程如下：

（1）首先把关键词分为“公司”和“名词”，例如“平安银行 二氧化碳”被拆分为“平安银行”与“二氧化碳”。

（2）使用正则表达式判断公司关键词“平安银行”是否在网页的title，keywords和description属性值中。正则表达式如图所示：

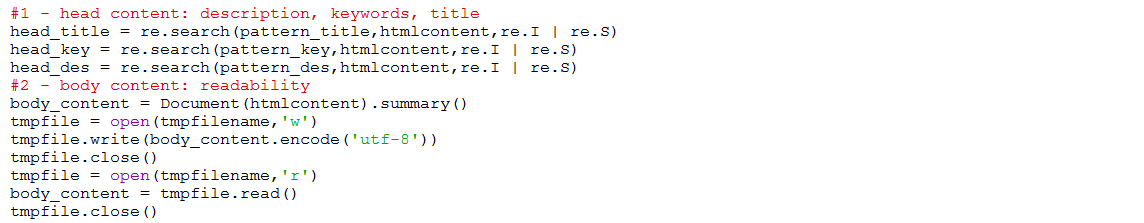


对应的标签分别为<title>…</title>，<meta name="keywords" content="…"/>，<meta name="description" content="…"/>，如图所示：



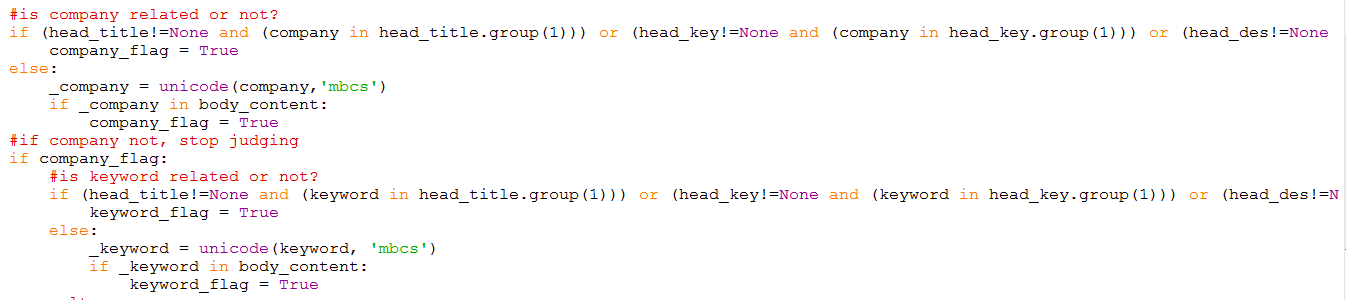
（3）如果公司关键词包含在（2）所示的网页属性值中，则继续判断名词关键词“二氧化碳”是否在网页的内容中：首先同步骤（2）使用正则表达式判断名词关键词是否也包含在（2）所示的网页属性值中。如果包含则判断网页与关键词有关并返回True；如果不包含则转到步骤（4）。

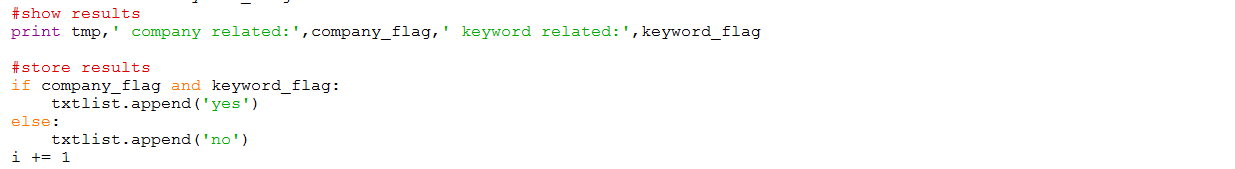
（4）结合readability判断名词关键词是否在网页正文中。如果名词关键词在网页正文中，则判断网页与关键词有关并返回True；如果不包含则返回False。这里为了排除html页面中广告评论等无关内容的干扰，这里采用readability对网页正文进行去噪得到关键内容，然后再对名词关键词进行判断，如图所示：



（5）如果公司关键词不包含在（2）所示的网页属性值中，则直接判断该网页与关键词无关并返回Fasle。

整个判断逻辑对应的代码如图所示：





**2、运行结果**

以关键词“平安银行+二氧化碳”为例子。分析得到的结果以“编号\*日期\*来源url\*是否相关”的格式存储于文本中（这里是平安银行+二氧化碳\_result.txt）。内容如图所示：



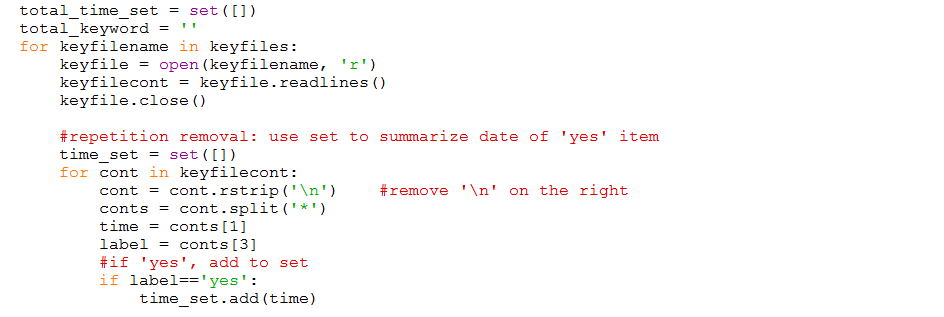
## 2.4 统计模块

**1、功能实现**

统计模块对每个关键词的分析结果按年份和季度进行汇总，主要包含按照更新时间进行去重操作。实现过程如下：

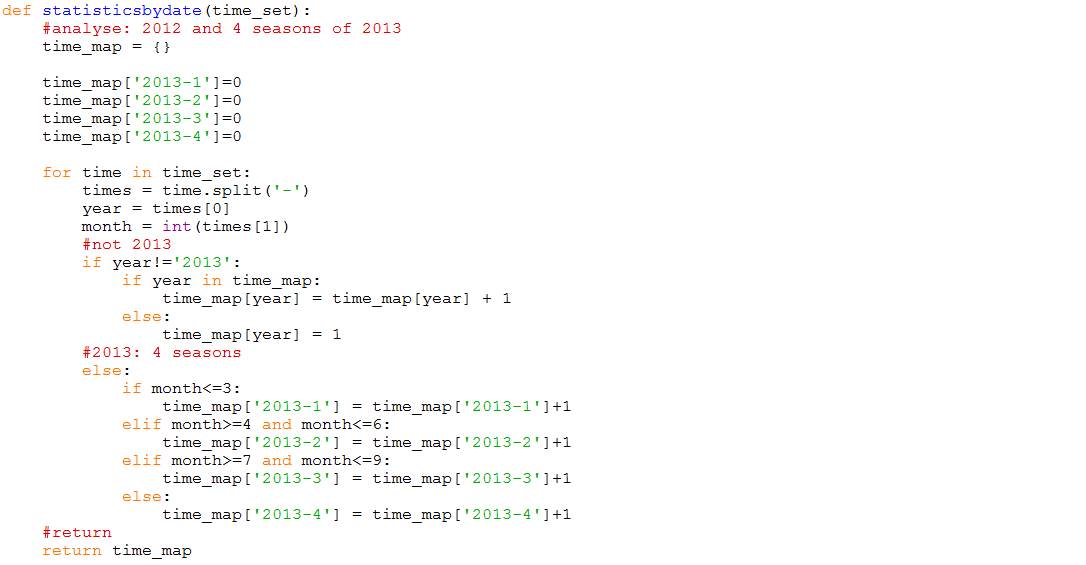
（1）对于每个名词关键词（例如这里的“二氧化碳”），根据分析模块给出的结果找出标记为“yes”的url对应的日期（例如“2011-6-27”）；

（2）将所有日期放入set中（等同于进行去重操作），则set中元素的个数就是去重之后标记为“yes”的url总条数。代码如图所示：



（3）对每个日期进行分析，找出同一年份的日期。特殊地，对于2013年的日期，找出每个季度的日期。

（4）对于步骤（3）中的所有日期，以年份作为键，以同一年（2013年的同一季度）的日期的条数作为值，建立dictionary，并输出到文本。代码如图所示：



（5）对于多个名词关键词，根据分析模块分别给的结果，按步骤（2）（3）（4）统计并输出结果。

**2、运行结果**

以关键词“平安银行+二氧化碳+碳排放”为例子。分别统计二氧化碳和碳排放相关的html页面个数，并按时间统计排序，得到如图的结果文本：

