In [1]:

```
import requests
import re
from bs4 import BeautifulSoup
import bs4
import pandas as pd
import datetime
```

In [2]:

```
def getHtmlUrl(ulist, htmls):    # 得到全部信件链接,并获得信件类型
    for html in htmls:
        soup = BeautifulSoup(html, 'html.parser')
        for link in soup.find_all('a'):
            links = link.get('href')
            if re.match('viewPublic.jsp\?id=.*?&cxm=',
                        str(links)):
                ulist.append('http://wlwz.changsha.gov.cn/webapp/cs/email/' +
                             links)
        # 利用beautifulsoup提取表格中指定列属性
        trs = soup.find('div', class_='information_table').find_all('tr')
        for tr in trs:
            for td in tr.find_all('td')[2:3]:
                Type.append(td.getText())
```

In [3]:

```
def getHtmlText(urls):           # 爬取页面内容
    texts = []
    i = 1
    for url in urls:
        try:
            headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108
            r = requests.get(url, headers=headers, timeout=100)
            r.raise_for_status()
            r.encoding = r.apparent_encoding
            texts.append(r.text)
            print(i)
            i += 1
        except:
            print('链接失败')
    return texts
```

In [4]:

```
def workdays(start, end):    # 计算两个日期间工作日/记得检查数据是否存在巨大差距，比方s：2015，e：1900
    # 记得需要删除标题行
    from datetime import datetime,timedelta
    from chinese_calendar import is_workday
    if start > end:
        start,end = end,start
    counts = 0
    while True:
        if start > end:
            break
        if is_workday(start):
            counts += 1
        start += timedelta(days=1)
    return counts
```

In [5]:

```
def get_field(article):                    # 领域识别功能
    import jieba
    import csv
    fields = {}
    fieldName = []
    # 读取领域库
    with open('C:\\Users\\23031\\Desktop\\信件领域.txt','r', encoding='utf-8' ) as f:
        for line in f.readlines():
            data = line.replace('\n','').split('；')      # 注意中英文
            fieldName.append(data[0])
            for keyword in data[1:]:
                fields[keyword] = data[0]
    frequency = {name: 0 for name in fieldName}

    # 文本分词
    sw = pd.read_csv(r'C:\Users\23031\Desktop\停用词.txt',
                     encoding='utf-8',sep='\n',quoting=csv.QUOTE_NONE,header=None)
    # 将文档分词并去除停用词
    stop_list = sw[0].tolist()
    word_cut = [i for i in jieba.lcut(article) if i not in stop_list]
    # 去除无关字符串
    while True:
        if '\n' in line:
            line.remove('\n')
        elif '\t' in line:
            line.remove('\t')
        elif ' ' in line:
            line.remove(' ')
        elif '\r' in line:
            line.remove('\r')
        elif '\r\n' in line:
            line.remove('\r\n')
        elif '\xa0' in line:
            line.remove('\xa0')
        else:
            break
    words=[]
    for content in word_cut:
        words.append(content)
    words = list(set(words))        # 去除重复元素
    # 统计、排序
    for word in words:
        try:
            frequency[fields[word]] += 1
        except Exception:
            pass
    result = sorted(frequency.items(), key=lambda x: x[1], reverse=True)

    if result[0][1] == 0:
        return '其他事件'
    else:
        return result[0][0]
```

In [6]:

```
def fillList(htmls):
    from datetime import datetime       # 用于提取信件归属年份
    i = 1
    for html in htmls:
        print(':', i)
        i += 1
        soup = BeautifulSoup(html, 'html.parser')
        for tag in soup.find_all('div', class_='incoming_letter'):
            title = tag.find('div', class_='mailbox_title').get_text()      # 标题
            try:
                appraise = ''
                appraise = tag.find('span', class_='dissatisfied').get_text()
                appraise = appraise.lstrip('满意度：')   # 删掉开头的 满意度： 字段
                appraise = appraise.strip()            # 删去'\n', '\r', '\t', ' '
            except:
                print(appraise)
            contents = tag.find('div', class_='mailbox_reader').get_text()  # 文字内容
            name = tag.findAll('span', class_='human')
            try:
                depname = ''
                depname = name[1].contents[0]                # 回复部门
            except:
                print(depname)
            time = tag.findAll('span', class_='time')
            try:
                begintime = str(time[0].contents[0])
                endtime = str(time[1].contents[0])
            except:
                begintime = '2000-01-01'
                endtime = '2000-01-02'
                print('匹配不到时间标签')

            try:                            # 将字符型转换成Date;预防爬取内容里面出现多种格式
                response = 0                                 # 初始化
                year = 0                                     # 初始化
                if  re.search(r'(\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{1,2}:\d{1,2})', begintime) != None:
                    begintime =datetime.strptime(begintime,'%Y-%m-%d %H:%M:%S')
                else:
                    begintime = datetime.strptime(begintime, '%Y-%m-%d').date()
                if  re.search(r'(\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{1,2}:\d{1,2})', endtime) != None:
                    endtime = datetime.strptime(endtime,'%Y-%m-%d %H:%M:%S')
                else:
                    endtime = datetime.strptime(endtime, '%Y-%m-%d').date()
                year = endtime.year                                 # 信件归属年份
                response = workdays(begintime, endtime)             # 政府回应时长
            except:
                print('时间问题')
                print(title)

            try:                                                   # 领域识别
                field = ''
                field = get_field(contents)
            except:
                print('无法识别领域')

            # 写入列表
            Title.append(title)
            DepName.append(depname)
            BeginTime.append(begintime)
            EndTime.append(endtime)
            Appraise.append(appraise)
            Year.append(year)
            Response.append(response)
            Field.append(field)
            Contents.append(contents)
```

In [13]:

```
urls = [
    "http://wlwz.changsha.gov.cn/webapp/cs/email/index.jsp?orgId=&cflag=1&type=&stype=1&emailList.offset={}&emailList.desc=false".form
    for i in range(5001,5401)
]

html_a = getHtmlText(urls)
catalog = []                            # 存储所有信件页面链接
Type = []                               # 信件类型
getHtmlUrl(catalog, html_a)
Title = []
DepName = []
BeginTime = []
EndTime = []
Appraise = []
Year = []
Response = []
Contents = []
Field = []
html_b = getHtmlText(catalog)
fillList(html_b)

# 主分析对象
dataframe = pd.DataFrame({'Title':Title,'DepName':DepName, 'Type':Type, 'BeginTime':BeginTime, 'EndTime':EndTime,
                          'Appraise': Appraise, 'Year':Year,'Response': Response, 'Field': Field})
dataframe.to_csv('C:\\Users\\23031\\Desktop\\长沙市_市长信箱_1.csv',mode='a', encoding='gb18030',index=False, sep=',')
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```
15
16
17
18
19
```

In [18]:

```
# 备份，添加了信件具体内容
dataframe1 = pd.DataFrame({'Title':Title,'DepName':DepName, 'Type':Type, 'Appraise': Appraise,
                           'Year':Year,'Response': Response, 'Field': Field, 'Contents':Contents})
dataframe1.to_csv('C:\\Users\\23031\\Desktop\\长沙市_备份_t.csv',mode='a', encoding='gb18030',index=False, sep=',')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-18-2a893deee975> in <module>
      1 # 备份，添加了信件具体内容
      2 dataframe1 = pd.DataFrame({'Title':Title,'DepName':DepName, 'Type':Type, 'Appraise': Appraise,
----> 3                            'Year':Year,'Response': Response, 'Field': Field, 'Contents':Contents})
      4 dataframe1.to_csv('C:\\Users\\23031\\Desktop\\长沙市_备份_t.csv',mode='a', encoding='gb18030',index=False, sep=
',')

E:\Anaconda3\lib\site-packages\pandas\core\frame.py in __init__(self, data, index, columns, dtype, copy)
    409             )
    410         elif isinstance(data, dict):
--> 411             mgr = init_dict(data, index, columns, dtype=dtype)
    412         elif isinstance(data, ma.MaskedArray):
    413             import numpy.ma.mrecords as mrecords

E:\Anaconda3\lib\site-packages\pandas\core\internals\construction.py in init_dict(data, index, columns, dtype)
    255             arr if not is_datetime64tz_dtype(arr) else arr.copy() for arr in arrays
    256         ]
--> 257     return arrays_to_mgr(arrays, data_names, index, columns, dtype=dtype)
    258 
    259 

E:\Anaconda3\lib\site-packages\pandas\core\internals\construction.py in arrays_to_mgr(arrays, arr_names, index, col
umns, dtype)
     75     # figure out the index, if necessary
     76     if index is None:
---> 77         index = extract_index(arrays)
     78     else:
     79         index = ensure_index(index)

E:\Anaconda3\lib\site-packages\pandas\core\internals\construction.py in extract_index(data)
    366             lengths = list(set(raw_lengths))
    367             if len(lengths) > 1:
--> 368                 raise ValueError("arrays must all be same length")
    369 
    370             if have_dicts:

ValueError: arrays must all be same length
```

In [30]:

```
len(Field)
```

Out[30]:

```
12011
```

In [ ]: