Software Design – Mini project1

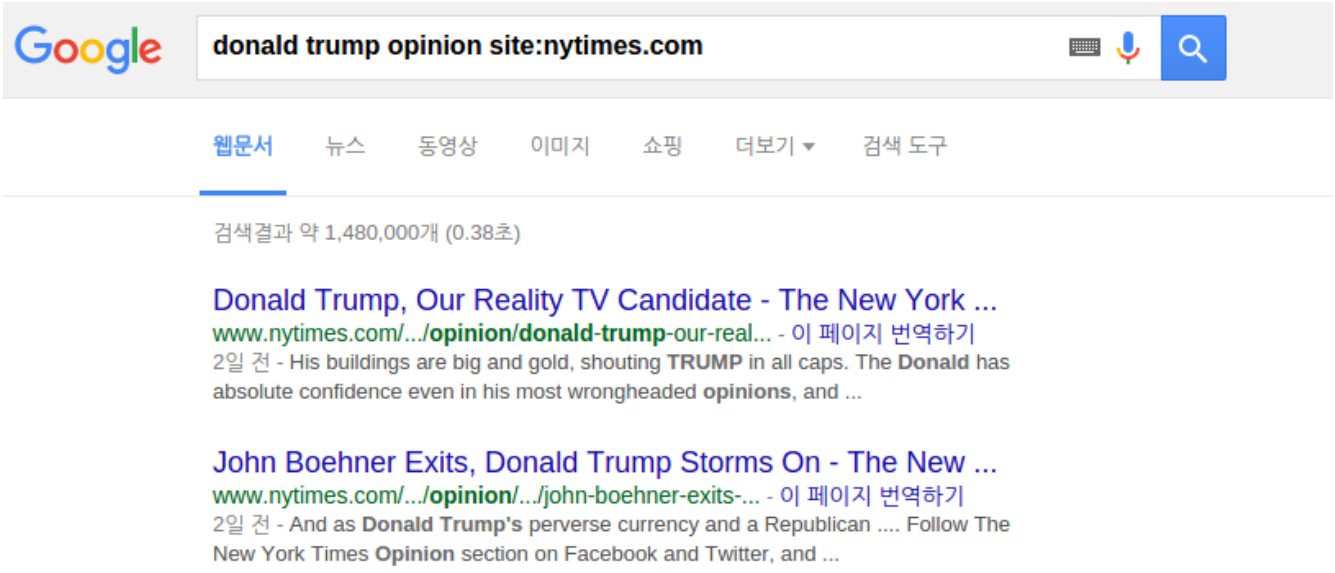
Jake Jung

1. Project overview

The mini project was about data mining and processing. We should get the data from websites and by using and analyzing the data, make some useful results.
In the mini project, I tried to analyze political bias of American media. To accomplish the goal I used google search engine from pattern, and typed same keyword related to the political issues. I chose 4 journals which are known for supporting democrats and republicans. For the pro-democrats media I picked New york times journal and CBS news and for pro- republican media wallstreet journal and fox news was chosen.
First, I tried to go to each media's site and get the articles related to the presidential candidates, but it was hard to get articles from the sites and some sites such as nytimes.com had an limited access on it. Therefore I used google search and added 'site: ' in the last of the keyword. By using this method I could get articles and data which were written by journals easily. I used “Donald Trump opinion site:nytimes.com”, “Donald Trump opinion site:cbsnews.com”, “Donald Trump opinion site:wsj.com”, “Donald Trump opinion site:foxnews.com” and harvested the urls.

After getting the urls I saved the urls in the local files because google had limited queries per day. Then I loaded the local files from other python cord and by using simple process turned the string to arrays of neat urls.

For the last step, I should analyz the contents of each urls which was the hardest. I tried with the simple basic steps which is making arrays or words representing each parties and get the number of the word in each sites. However it was really hard to make an word that can evaluate each parties and the analysis results showed no implications. Trying some more steps by myself I tried googling and could find good free political and sentiment analysis tool from indico. By using the machine learning based analysis tool I could manage to analyze the urls much more efficiently. The indico API provided results related to the preferred party based on its text and I sumed up all the results of each site and made a statistics.



2. Implementations

```python
#cord to mining the urls from google and save it to local .txt files
# using patter. to search from Google

from pattern.web import Google
from pattern.web import SEARCH

# using indicoio for polytical and sentiment analysis
import indicoio
indicoio.config.api_key = '8d05933c4c2ca769d1e064dfbea1fe8a'

# declare arrays which save raw url mined from pattern.search
# new york times urls, cbs new urls, wallstreet journal urls, foxnew urls
rawurl_nytimes=[]
rawurl_cbsnews=[]
rawurl_wsj=[]
rawurl_foxnews=[]
journal_names=['nytimes', 'cbsnews', 'wsj', 'foxnews']
rawurls=[rawurl_nytimes, rawurl_cbsnews, rawurl_wsj, rawurl_foxnews]

g=Google()


#get the New York Times url
for journal, raw_url_title in zip(journal_names, rawurls):

    #in order to get 30 urls with the keyword, used for-loop
    for i in range(1,4):

        # search google results correspoding to the following keyword
        for result in g.search('Donlad Trump opinion site:'+journal+'.com', start=i):

            print result.url
            # append the urls to the rawurl_ array
            raw_url_title.append(result.url)

    print raw_url_title
    print len(raw_url_title)

    # saves the keyword to the local file in order to reduce query
    # we will use this file for analyzing later on

    f=open('url_'+journal+'.txt', "w")
    print >>f, raw_url_title
    f.close()

```

Briefly speaking the cord is about getting web data by using patter google search engine and append the url string to rawurl_site array and then save each sites url to different files. By using

for i in range(1,4):
I could get 30 results from each keyword, and by using
for journal, raw_url_title in zip(journal_names, rawurls)
I could get total 120 results by the loop.

At first I made the cord messy because we should get articles from 4 sites I used 4 different for-loops which changes the word after site(ex donald trump opinion site:nytimes.com, donald trump opinion site:cbsnews.com). However by using for loop with arrays I could shortened the cord like above and

now if I want to harvest another site's result about donald trump I can just add the site's name in the jornal_names array and rawurls array.

```
#mini project open the url file which we saved from harvesting and execute political analysis*/
########################################
folder=["url_nytimes.txt", "url_cbsnews.txt", "url_wsj.txt", "url_foxnews.txt"]

#for each files split the string by comma from the array
for textfiles in folder:
    f= open(textfiles, 'r')

    line=f.readline()
    url_dummy=line.split(',')

    ## get all urls from the saved file
    i=0
    for i in range(len(url_dummy)-1):
        # get rid of useless html.
        url_dummy[i]=url_dummy[i][3:-1]
        print url_dummy[i]
        i=i+1

    ## because last url has on more ' , get rid of it
    url_dummy[-1]=url_dummy[-1][3:-2]
    print len(url_dummy)

    ## do political analysis using indicoio using the API and apped it to the array
    analysis=[]
    j=0
    for j in range(len(url_dummy)):
        analysis.append(indicoio.political(url_dummy[j]))
        j=j+1


```

I could get the raw urls from the MP1.py cord however the raw_urls include some html symbols and to make it to perfect urls we have to get rid of some strings around the urls. So by using the MP2 cord I could get perfect urls and put those to the indicoio API for political analysis like below.

```
    ## do political analysis using indicoio using the API and apped it to the array
    analysis=[]
    j=0
    for j in range(len(url_dummy)):
        analysis.append(indicoio.political(url_dummy[j]))
        j=j+1


    ## get the average of the analysis
    ## add all the results of the urls and divide with the number of urls

    sum_stats=[0,0,0,0]  #sum of all stats gained from indicoio
    for i in range(len(analysis)):
        sum_stats[0]=sum_stats[0]+analysis[i]["Libertarian"]
        sum_stats[1]=sum_stats[1]+analysis[i]["Green"]
        sum_stats[2]=sum_stats[2]+analysis[i]["Liberal"]
        sum_stats[3]=sum_stats[3]+analysis[i]["Conservative"]
        i=i+1
    print sum_stats
    aver_stats=[0,0,0,0]
    for i in range(4):
        aver_stats[i]=sum_stats[i]/float(len(analysis))


    print "[Libertarian   ,   Green   ,   Liberal   ,   Conservative]"
    print aver_stats

    ## move the stats to the text local text file
    ## write the political analysis results to stats file line by line
    f_stats.write(textfiles+" polytical analysis stat \n [ Libertarian:    ,   Green   ,   Liberal   ,   Conservative] \n")
    print >>f_stats, aver_stats
    f_stats.write("\n" )
    f.close()
```

The indicoio give the data in dictionary type and I got the average data of each 30 results and saved the average political analysis into stats.txt.

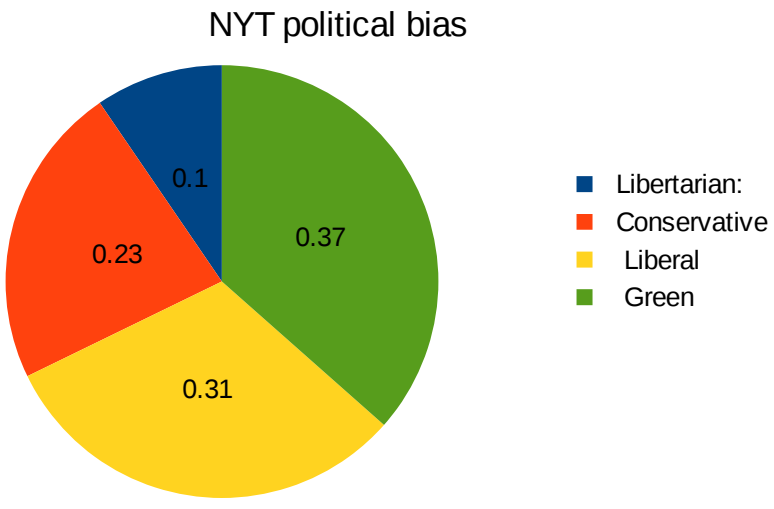
3. Output/ Results

I uploaded the results files with various keywords to the github
url_1 : Donald trump opinion
url_2: Donald trump poll
url_3: Presidential candidates

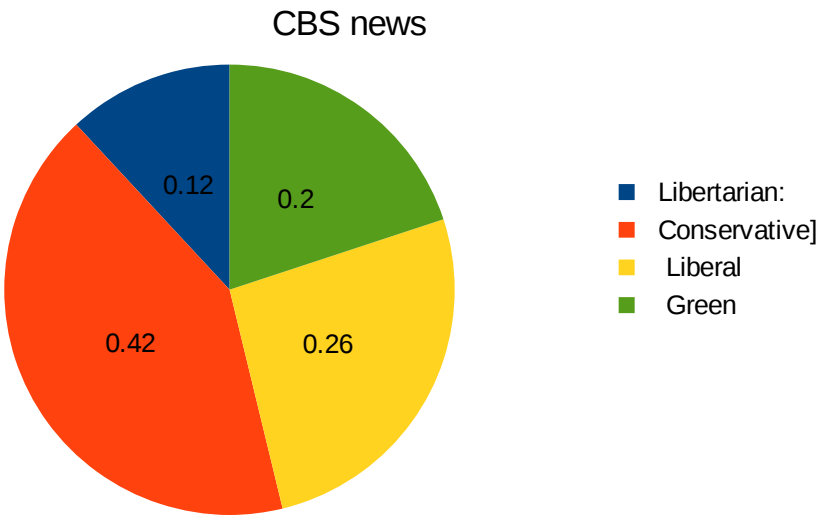
The url files are the result urls from each keywords and the stats files are the political analysis of each keyword. You can see the detail result files in the github.

Following graphs are the results of “Donald Trump opinion” in each sites.

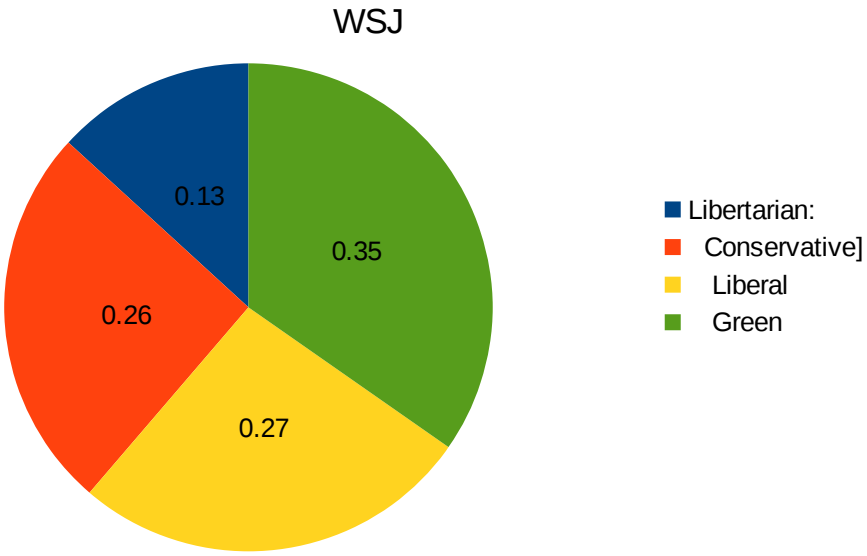
1. Newyork Times



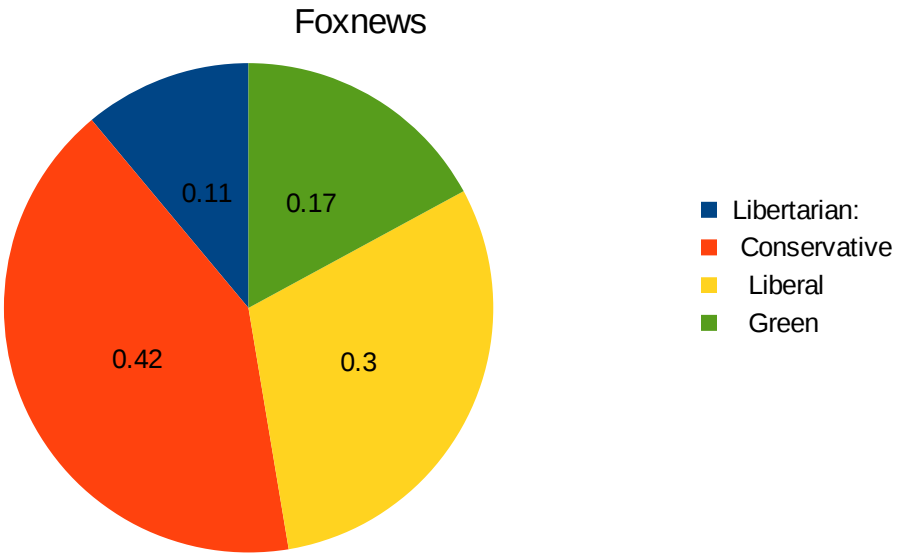
2. CBS news



3. Wallstreet journal



4. Foxnews



New york times showed high liberal bias compared to the conservative bias, however the cbs news was analyzed as highly conservatively biased which is different with our expectation. Wallstreet journal showed unbiased status which the liberal and conservative percentage was almost same. Lastly in case of Foxnews it was highly conservative.
The results of Newyork times and foxnews was similar with our anticipation but cbsnews and wallstreet journal was not.
We can get other results of political analysis if we just change the keyword.

4. Reflections

I could use what I learned from the think-python at the real coding. I totally understood how literate and loop works and little bit about data mining. It was really awesome to get data from web and do what I want!! Furthermore I learned how to use API and process string to use properly.

The mini project went pretty well and I have done what I wanted to do. However the result of the political analysis was not satisfying and had many things to improve. Indicoio political analysis was quite good however it also had a lot of wrong results with the real contents. Also because I picked “Donald Trump” as a keyword and that might made the result not accurate because he is someone with lots of gossip.

I might have improved little more if I had chosen to analyze the contents by myself not get help by other API. But for the first project I think it was good and I learned a lot in python and programming logic.