



뉴스를 재미있게 만드는 방법; 뉴스잼

부제 : 파이썬 라이브러리 여행

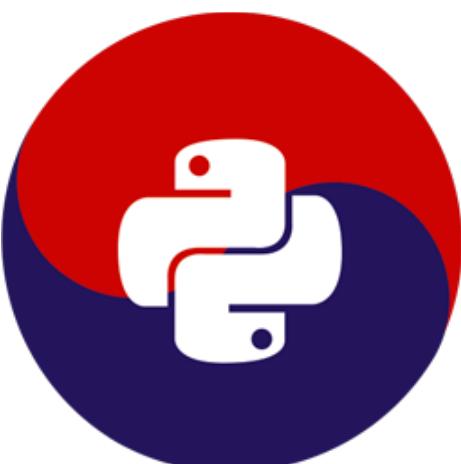
김경훈

유니스트 수리과학과

kyunghoon@unist.ac.kr

2016년 8월 13일

강남구 코엑스 전시장



Respect, Diversity at PyCon APAC 2016



미래창조과학부

뉴스를 재미있게
만드는 방법

||

뉴스를 잼있게
만드는 방법

||

News JAM
뉴스잼



뉴스를 재미있게 만든다.

아주 재미없게 뉴스를 만들어서 열 명이 보느니
재미있게 만들어서 천 명이 본다면
다소 팩트를 희생했다고 하더라도 전체적인 팩트는 성공한 것이 아닌가,
전체적인 팩트의 양은 큰 것이 아닌가

<이상호 GO발뉴스, 2012>

뉴스를 재미있게 만든다?

어떻게?

1. 뉴스를 재미있게 만들기 위한

수집

1.1 데이터 수집하기

- 1.1.1 데이터 소스
- 1.1.2 라이브러리 urllib2, request, mechanize
- 1.1.3 라이브러리 chardet, unidecode
- 1.1.4 라이브러리 pypspider, feedparser
- 1.1.5 라이브러리 robobrowser

1.2 데이터 파싱하기

- 1.2.1 라이브러리 beautifulsoup, lxml, pyparsing
- 1.2.2 라이브러리 python-goose
- 1.2.3 라이브러리 newspaper

1.3 데이터 저장하기

- 1.3.1 라이브러리 PyMongo
- 1.3.2 Cloud Service (mongolab, aws, azure)

2. 뉴스를 재미있게 만드는

분석

2.1 데이터의 전처리

- 2.1.1 형태소 분석기
- 2.1.2 라이브러리 Konlpy, umorpheme
- 2.1.3 docker를 활용한 휴대용 형태소 분석
- 2.1.4 라이브러리 NLTK, textblob

2.2 뉴스를 분류하기

- 2.2.1 LSI (Latent Semantic Indexing)
- 2.2.2 행렬의 계산
- 2.2.3 라이브러리 gensim

2.3 뉴스를 군집하기

- 2.3.1 NMF (Non-negative Matrix Factorization)
- 2.3.2 라이브러리 nimfa

2.4 토픽모델링

- 2.4.1 LDA (Latent Dirichlet Allocation)
- 2.4.2 라이브러리 gensim, lda

2.5 네트워크 만들기

- 2.5.1 뉴스 네트워크
- 2.5.2 라이브러리 networkx
- 2.5.3 라이브러리 pyneviz

3. 뉴스를 재미있게 이용하는

전달

3.1 웹으로 데이터 보여주기

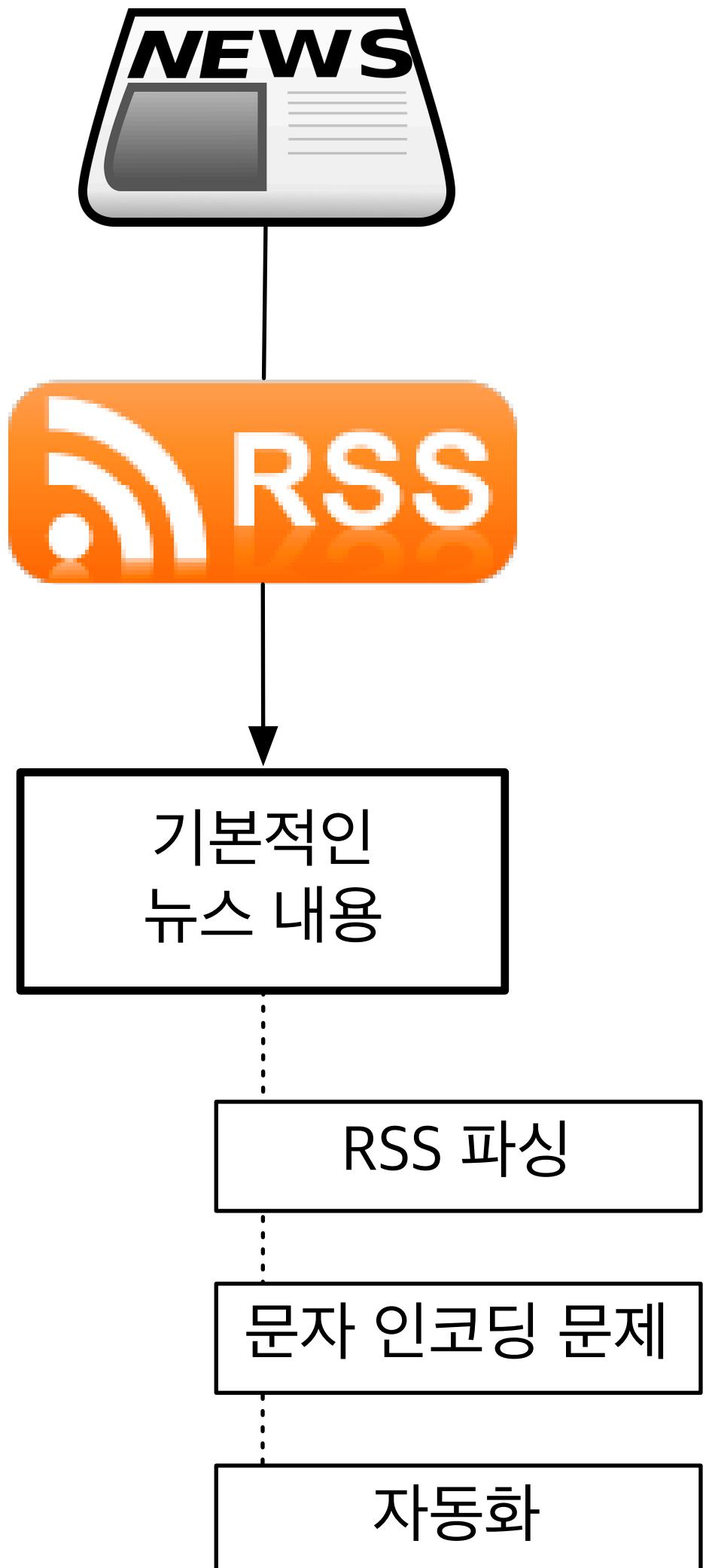
- 3.1.1 라이브러리 Flask

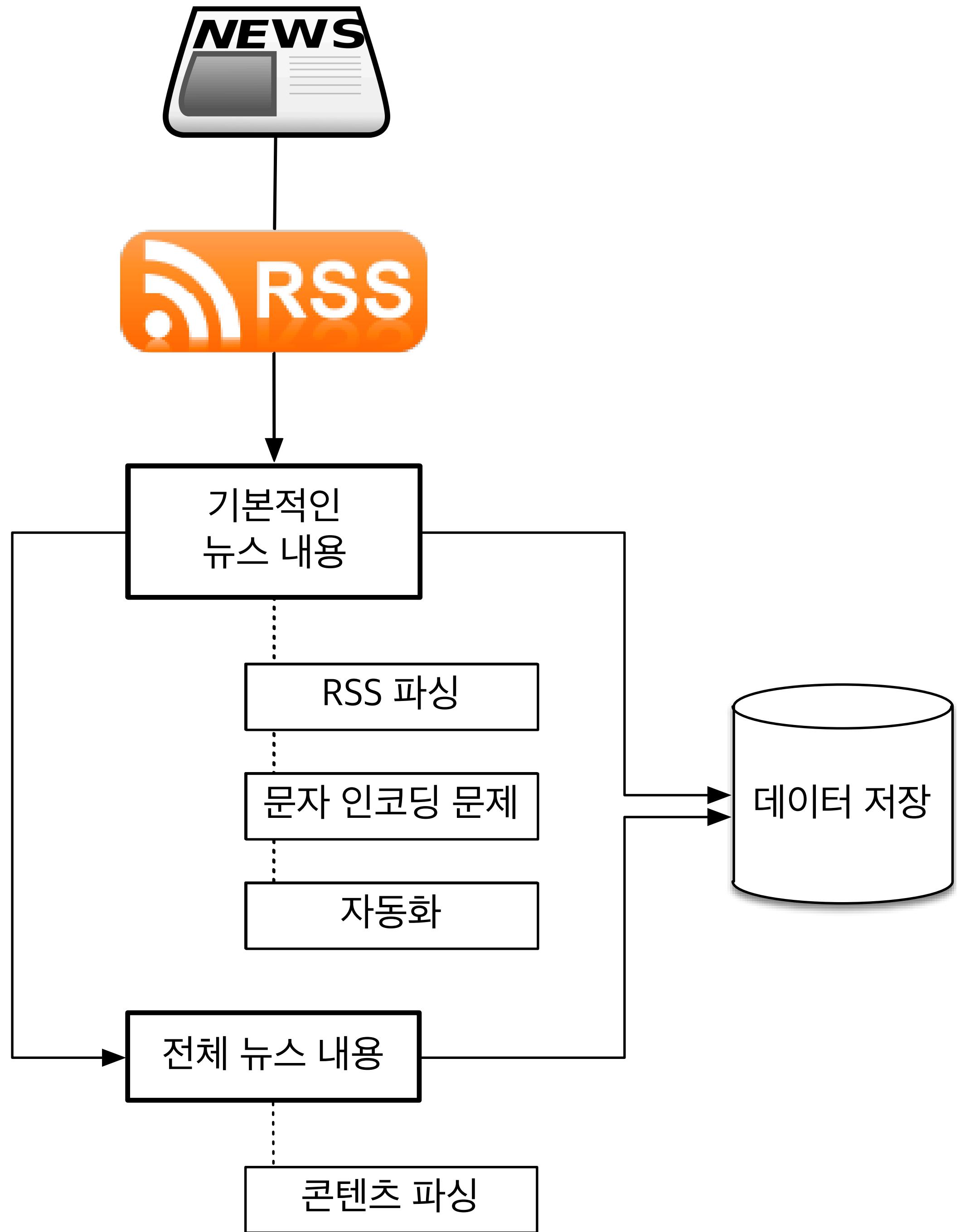
3.2 시각화하기

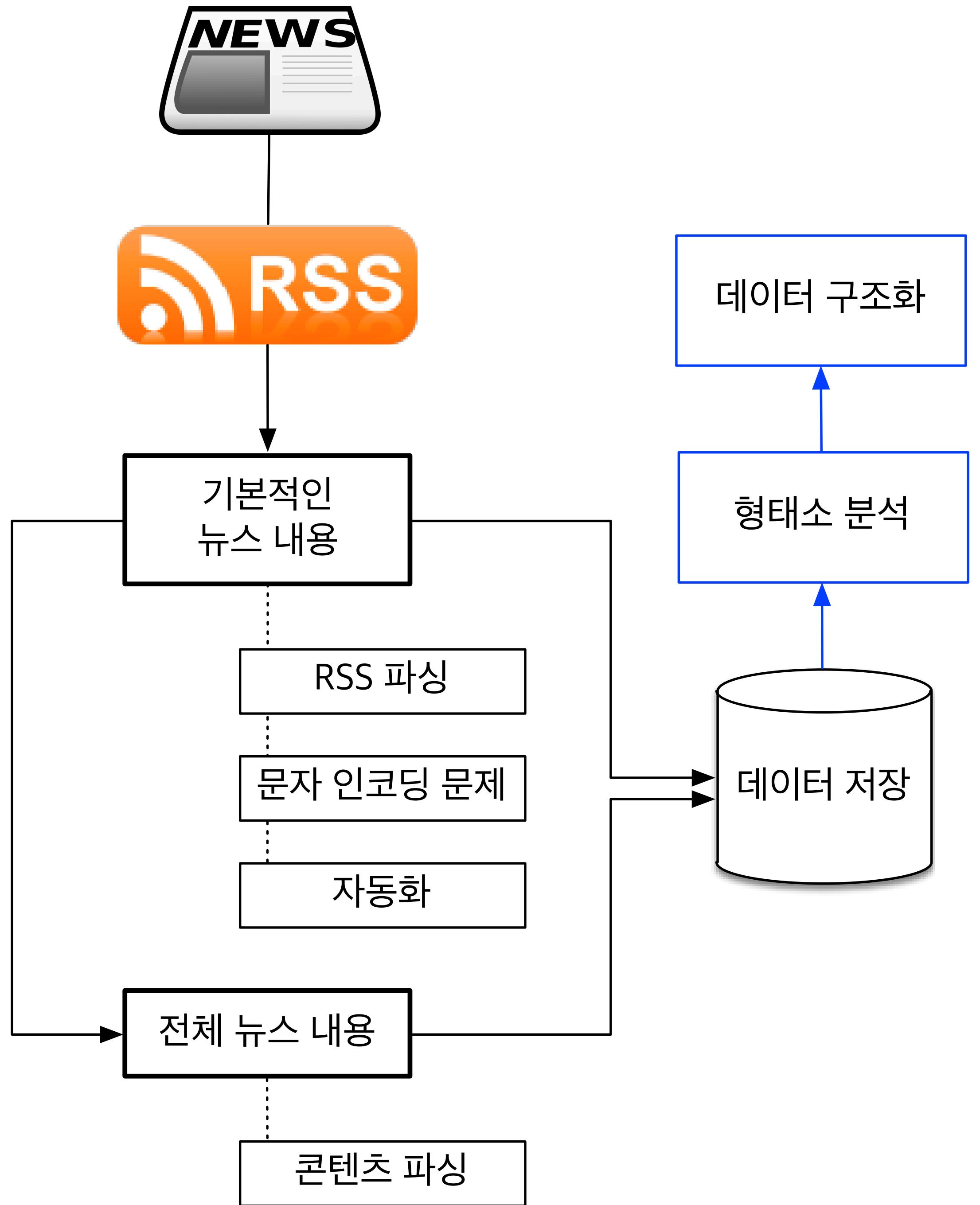
- 3.2.1 라이브러리 seaborn

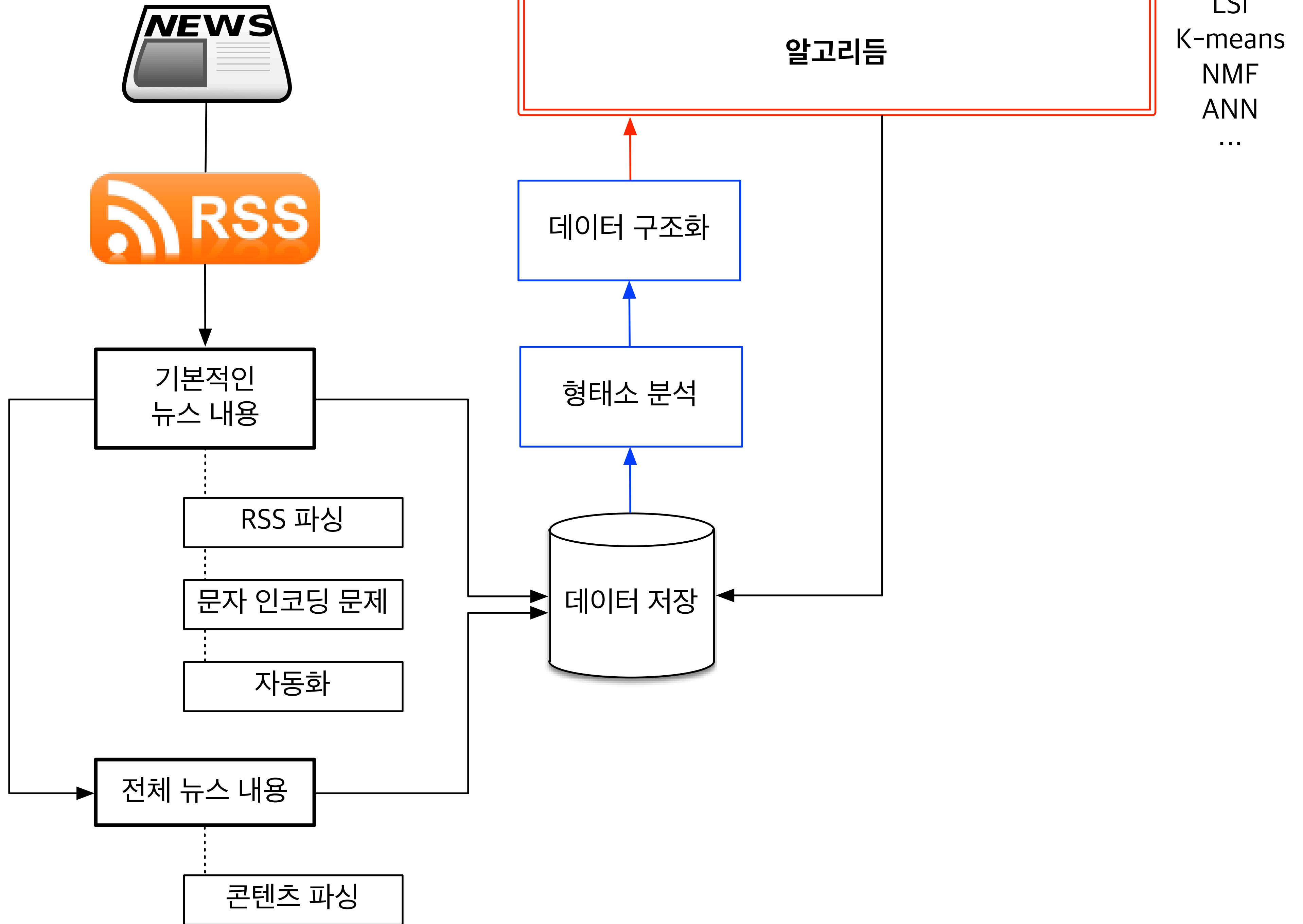
3.3 검색하기

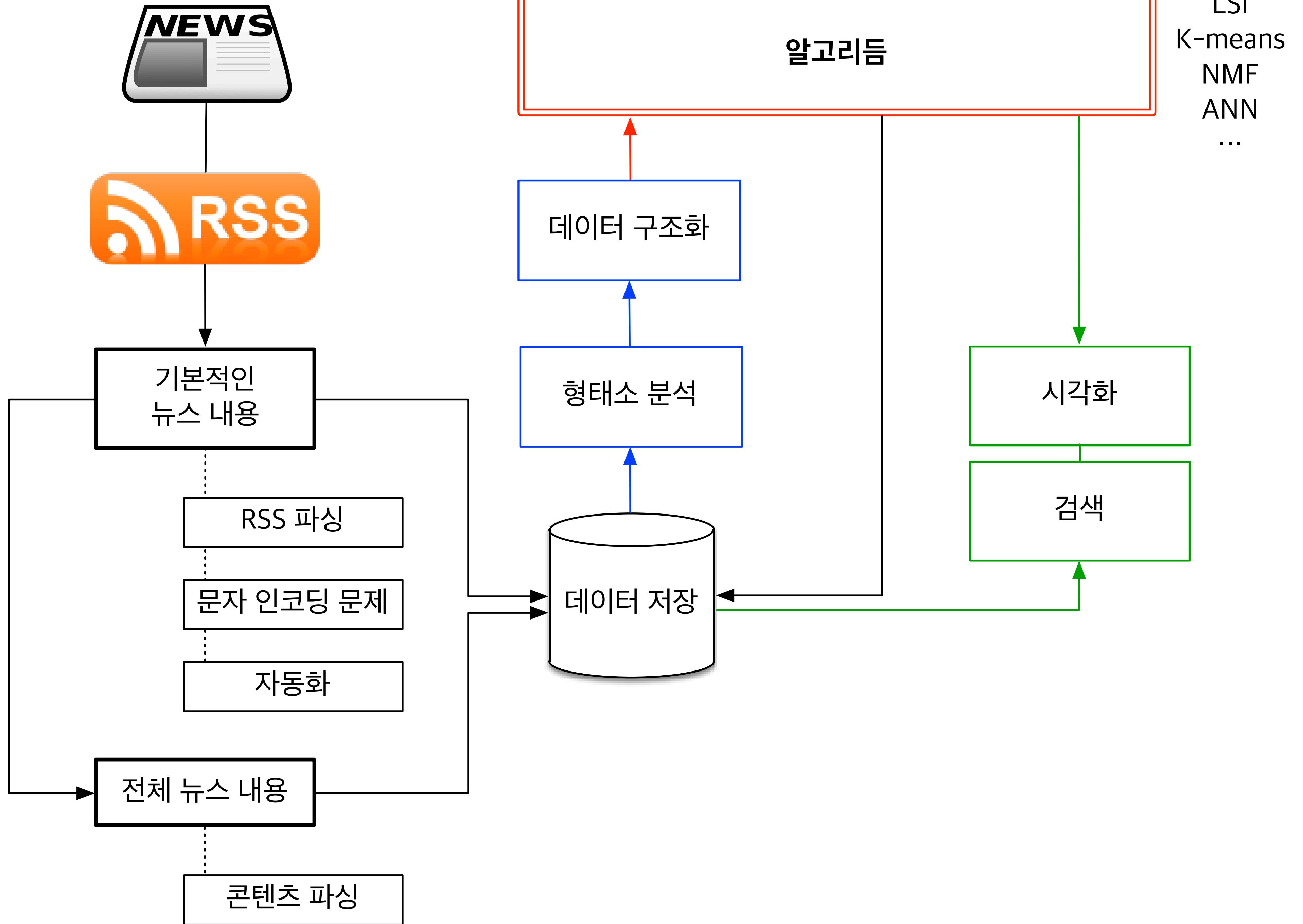
- 3.3.1 라이브러리 elasticsearch-py











1.1 데이터 수집하기

.1 데이터 소스

RSS (Rich Site Summary)

뉴스나 블로그 사이트에서 주로 사용하는 콘텐츠 표현 방식. XML 기반의 문서 포맷

이 페이지의 기사 선별 및 표시는 컴퓨터 프로그램에 의해 자동으로 결정됩니다.
표시된 시간 또는 날짜(시간대별 뉴스 기능에 포함)는 Google 뉴스에 기사가 추가되었거나 업데이트된
날짜입니다.

 [RSS](#) - 다른 뉴스 에디션 - [Google 뉴스 정보](#) - [피드 정보](#) - [도움말](#) - [의견 보내기](#) - [개인화 설정](#)

©2016 Google - [Google 홈](#) - [광고 프로그램](#) - [기업용 솔루션](#) - [Google 기술자료](#)

<https://news.google.co.kr/>

○ 뉴스

전체기사	 RSS	 HanRSS	 YAHOO!	 Google
주요기사	 RSS	 HanRSS	 YAHOO!	 Google
경제	 RSS	 HanRSS	 YAHOO!	 Google
사회	 RSS	 HanRSS	 YAHOO!	 Google
정치	 RSS	 HanRSS	 YAHOO!	 Google
라이프	 RSS	 HanRSS	 YAHOO!	 Google
지구촌	 RSS	 HanRSS	 YAHOO!	 Google
문화	 RSS	 HanRSS	 YAHOO!	 Google
IT과학	 RSS	 HanRSS	 YAHOO!	 Google

<http://rss.joins.com/>

1.1 데이터 수집하기

.1 데이터 소스

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0">
    <channel>
        <title>제목</title>
        <link>주소</link>
        <description>설명 (짤막하게)</description>

        <item>
            <title>제목</title>
            <link>주소/글 주소</link>
            <description>글 내용 전체 또는 일부</description>
            <pubDate>시간과 날짜</pubDate>
            <guid>주소/글 주소</guid>
        </item>

        <item>
            <title>제목</title>
            <link>주소/글 주소</link>
            <description>글 내용 전체 또는 일부</description>
            <pubDate>시간과 날짜</pubDate>
            <guid>주소/글 주소</guid>
        </item>
    </channel>
</rss>
```

1.1 데이터 수집하기

.1 데이터 소스

<https://gist.github.com/koorukuroo/330a644fcc3c9ffdc7b6d537efd939c3>

```
{'corp': '다음뉴스',
'name': '종합',
'url': 'http://media.daum.net/rss/today/primary/all/rss2.xml'},
```

553개의 RSS 주소

```
{'corp': '다음뉴스',
'name': '연예',
'url': 'http://media.daum.net/rss/today/primary/entertain/rss2.xml'}
```

...

1.1 데이터 수집하기

.1 데이터 소스

뉴스도 저작권법의 보호를 받습니다.

‘뉴스’는 저작권법의 보호를 받는 언론사의 ‘창작물’로 무단 복사·배포는 저작권법 위반입니다.

언론사의 뉴스기사와 보도사진은 일반적인 저작물과 마찬가지로 창작 노력이 깃든 저작물입니다.

뉴스 저작물도 음악·영화·게임 등 다른 저작물과 마찬가지로 저작권자의 허락을 받고 사용해야 합니다.

아직도 뉴스를 **몰래** 사용하고 있습니까?



1.1 데이터 수집하기

.2 라이브러리 urllib2, requests, mechanize

목적 : URL의 내용 가져오기

urllib2

```
import urllib2
req = urllib2.Request("http://pycon.kr")
r = urllib2.urlopen(req)
html = r.read()
```

*Python3에서는 urllib으로 통합

requests

```
import requests
r = requests.get('http://pycon.kr')
html = r.text
```

mechanize

```
import mechanize
browser = mechanize.Browser()
page = browser.open("http://pycon.kr")
html = page.read()
```

1.1 데이터 수집하기

.2 라이브러리 [urllib2](#), requests, mechanize

```
import urllib, urllib2

url = 'http://127.0.0.1:5000/?a=3'
values = { 'lang' : '파이썬', 'kind' : '컨퍼런스' }
data = urllib.urlencode(values)
req = urllib2.Request(url, data)
response = urllib2.urlopen(req)
html = response.read()
print html
```

Method: POST

GET: a=3

POST: lang=%ED%8C%8C%EC%9D%B4%EC%8D%AC&kind=%EC%BB%A8%ED%8D%BC%EB%9F%BC

목적 : POST METHOD

```
# -*- coding: utf8 -*-
from flask import Flask, request

app = Flask(__name__)

@app.route("/", methods=['GET', 'POST'])
def func():
    return 'Method: '+str(request.method)+'\nGET: '+str(request.query_string)+'\nPOST: '+str(request.stream.read())

if __name__ == "__main__":
    app.run(debug=True)
```

<https://gist.github.com/koorukuroo/28e0c01b2cd0ec83b3149adce30b21a5>

1.1 데이터 수집하기

.2 라이브러리 urllib2, [requests](#), mechanize

목적 : GET METHOD

```
import requests
```

```
url = 'http://127.0.0.1:5000/?a=3'  
values = {'lang': '파이썬', 'kind': '컨퍼런스'}  
r = requests.get(url, params=values)  
html = r.text  
print html
```

Method: GET

GET: a=3&lang=%ED%8C%8C%EC%9D%B4%EC%8D%AC&kind=%EC%BB%A8%ED%8D%BC%EB%9F%B0%EC%8A%A4

POST:

1.1 데이터 수집하기

.2 라이브러리 urllib2, [requests](#), mechanize

목적 : POST METHOD

```
import requests
```

```
url = 'http://127.0.0.1:5000/?a=3'
values = {'lang': '파이썬', 'kind': '컨퍼런스'}
r = requests.post(url, params=values)
html = r.text
print html
```

Method: POST

GET: a=3

POST: lang=%ED%8C%8C%EC%9D%B4%EC%8D%AC&kind=%EC%BB%A8%ED%8D%BC%EB%9F%
%B0%EC%8A%A4

1.1 데이터 수집하기

.2 라이브러리 urllib2, requests, mechanize

mechanize <https://pypi.python.org/pypi/mechanize/>

```
import mechanize  
browser = mechanize.Browser()  
page = browser.open(url)  
html = page.read()
```

인코딩에 문제가 있다면 mechanize!
but 2011..

1.1 데이터 수집하기

.3 라이브러리 robobrowser

robobrowser <https://github.com/jmcarp/robobrowser>

```
from robobrowser import RoboBrowser

browser = RoboBrowser(history=True)
browser.open(base_url)
form = browser.get_form(action='/login/')

form["username"] = 'username'
form["password"] = 'password'
browser.session.headers['Referer'] = base_url # for CSRF

browser.submit_form(form)
print(str(browser.select))
```

1.1 데이터 수집하기

.3 라이브러리 robobrowser

robobrowser <https://github.com/jmcarp/robobrowser>

```
# Look up the first song
songs = browser.select('.song_link')
browser.follow_link(songs[0])
lyrics = browser.select('.lyrics')
lyrics[0].text

# Back to results page
browser.back()

# Look up my favorite song
song_link = browser.get_link('trains')
browser.follow_link(song_link)
```

1.1 데이터 수집하기

.4 라이브러리 [pyspider](#), feedparser

목적 : 자동화

pyspider <http://docs.pyspider.org/en/latest/Quickstart/>

```
$ pip install pyspider  
$ pyspider
```

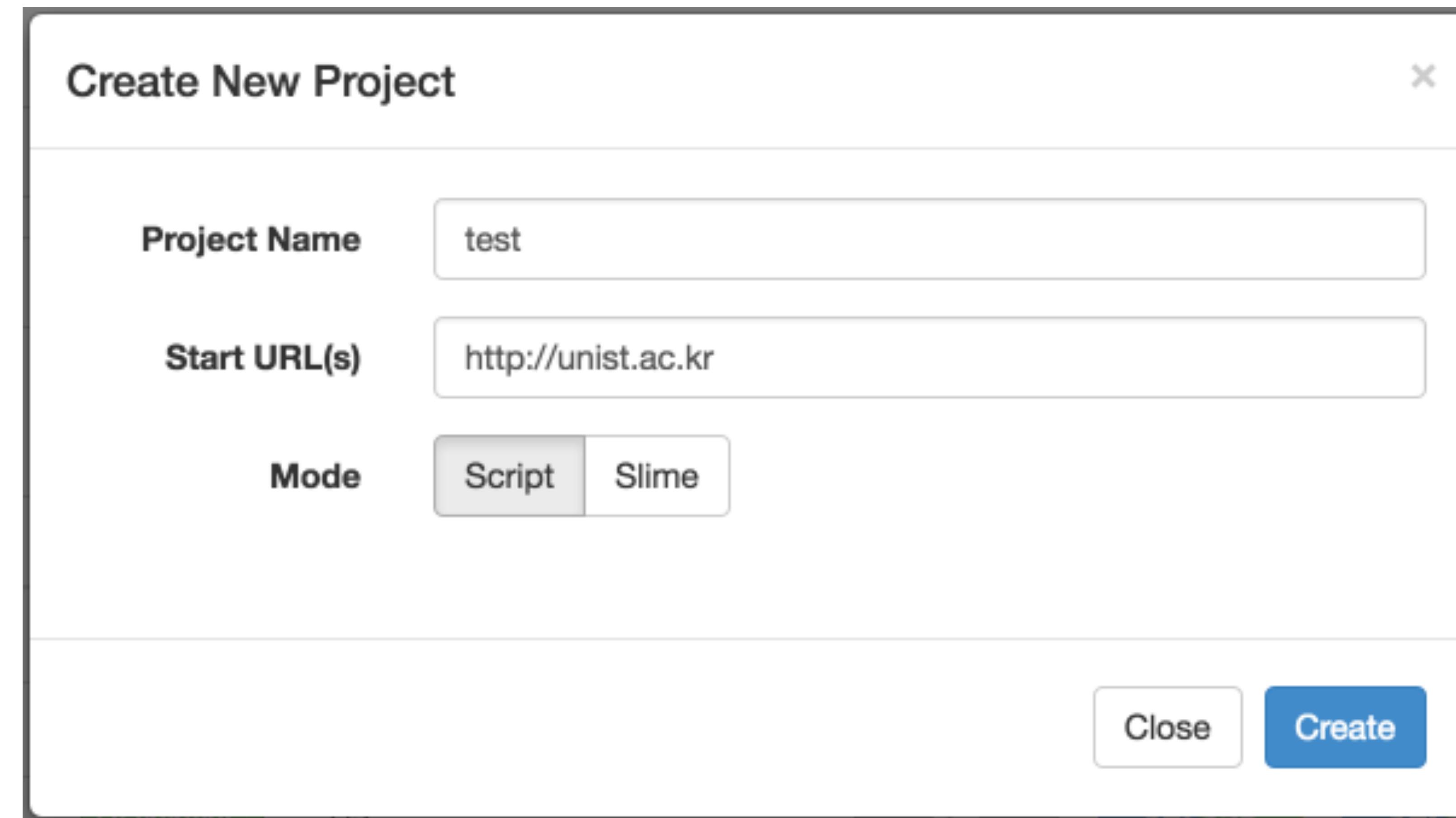
pyspider dashboard



1.1 데이터 수집하기

.4 라이브러리 [pyspider](#), feedparser

pyspider



1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

pyspider

pyspider > test1

Documentation WebDAV Mode

```
{ "process": { "callback": "on_start" }, "project": "test1", "taskid": "data:,on_start", "url": "data:,on_start" }
```

run save

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2016-08-10 17:20:39
# Project: test1

from pyspider.libs.base_handler import *

class Handler(BaseHandler):
    crawl_config = {}

    @every(minutes=24 * 60)
    def on_start(self):
        self.crawl('http://unist.ac.kr', callback=self.index_page)

    @config(age=10 * 24 * 60 * 60)
    def index_page(self, response):
        for each in response.doc('a[href^="http"]').items():
            self.crawl(each.attr.href, callback=self.detail_page)

    @config(priority=2)
    def detail_page(self, response):
        return {
            "url": response.url,
            "title": response.doc('title').text(),
        }
```

enable css selector helper web html follows messages

1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

pyspider

pyspider dashboard

scheduler	0	fetcher	0	processor	0	result_worker
0 + 0						
Recent Active Tasks						Create
group	project name	status	rate/burst	avg time	progress	actions
[group]	blue, blue, blue	TODO	1/3	5m	1h	1d
				all	Run	Active Tasks
					Results	

1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

pyspider

test1 - Results			
url	title	url	...
http://www.unist.ac.kr/faculty	🔗 "Searching faculty UNIST"	"http://www.unist.ac.kr/faculty/"	{}
http://www.unist.ac.kr/category/notifications/unist-notice-board/	🔗 "Notice UNIST"	"http://www.unist.ac.kr/category/notifications/unist-notice-board/"	{}
http://uee.unist.ac.kr/	🔗 "School of Urban and Environmental Engineering"	"http://uee.unist.ac.kr/"	{}
http://www.unist.ac.kr/	🔗 "UNIST"	"http://www.unist.ac.kr/"	{}
http://ece.unist.ac.kr/	🔗 "School of Electrical & Computer Engineering - School of Electrical & Computer E ngineering"	"http://ece.unist.ac.kr/"	{}
http://www.unist.ac.kr/about-unist/unist-identity/?tab_index=1	🔗 "UNIST Identity UNIST"	"http://www.unist.ac.kr/about-unist/unist-identity/?tab_index=1"	{}
http://www.unist.ac.kr/about-unist/directions/road/	🔗 "Directions UNIST"	"http://www.unist.ac.kr/about-unist/directions/road/"	{}
http://mne.unist.ac.kr/	🔗 "UNIST School of MECHANICAL AND NUCLEAR ENGINEERING"	"http://mne.unist.ac.kr/"	{}
http://www.unist.ac.kr/campus-life/academics/academic-administr ation/	🔗 "Academic Administration UNIST"	"http://www.unist.ac.kr/campus-life/academics/academic-administr ation/"	{}
http://lec.unist.ac.kr/index.sko	🔗 "Language Education Center"	"http://lec.unist.ac.kr/index.sko"	{}
http://www.unist.ac.kr/campus-life/	🔗 "Campus Life UNIST"	"http://www.unist.ac.kr/campus-life/"	{}
http://sls.unist.ac.kr/en/	🔗 "Home - School of Life Sciences"	"http://sls.unist.ac.kr/en/"	{}
http://www.unist.ac.kr/people	🔗 "Searching staff UNIST"	"http://www.unist.ac.kr/people/"	{}
http://news.unist.ac.kr/	🔗 "UNIST News Center"	"http://news.unist.ac.kr/"	{}
http://news.unist.ac.kr/unist-named-among-global-rising-stars-by- nature-index/	🔗 "UNIST Named among Global 'Rising Stars' by Nature IndexUNIST News Center UNIST News Center"	"http://news.unist.ac.kr/unist-named-among-global-rising-stars-by- nature-index/"	{}
http://www.unist.ac.kr/about-unist/overview/unist-at-a-glance-ne w/	🔗 "UNIST at a Glance UNIST"	"http://www.unist.ac.kr/about-unist/overview/unist-at-a-glance-ne w/"	{}

1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

pyspider

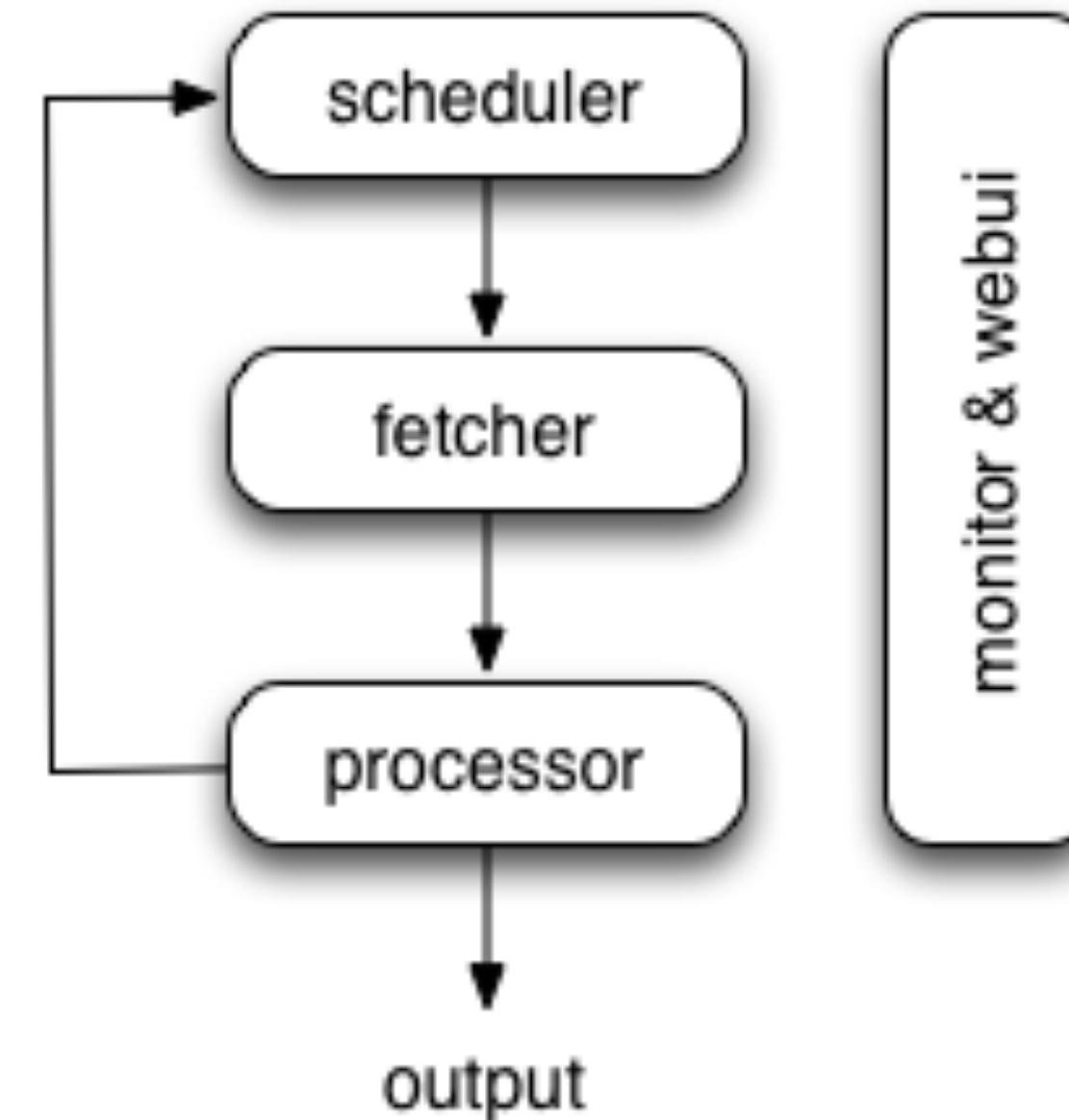
pyspider dashboard

scheduler	0	fetcher	0	processor	0	result_worker	
0 + 0							
Recent Active Tasks							
Create							
group	project name	status	rate/burst	avg time	progress	actions	
[group]	[REDACTED]	DEBUG	1/3	0.0+0.62ms	<div>5m: 2</div> <div>1h: 6</div> <div>1d: 14</div> <div>all: 2</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	0.0+0.41ms	<div>5m: 2</div> <div>1h: 6</div> <div>1d: 6</div> <div>all: 2</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	504.3+17.81ms	<div>5m: 135</div> <div>1h: 135</div> <div>1d: 135</div> <div>all: 67</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	29.2+2.84ms	<div>5m: 4</div> <div>1h: 4</div> <div>1d: 4</div> <div>all: 2</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	15.0+2.79ms	<div>5m: 4</div> <div>1h: 4</div> <div>1d: 4</div> <div>all: 2</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	0.0+0.39ms	<div>5m: 2</div> <div>1h: 6</div> <div>1d: 12</div> <div>all: 2</div>	Run Active Tasks Results	
[group]	[REDACTED]	RUNNING	1/3	0.0+0.40ms	<div>5m: 2</div> <div>1h: 183</div> <div>1d: 183</div> <div>all: 84</div>	Run Active Tasks Results	

1.1 데이터 수집하기

.4 라이브러리 [pyspider](#), feedparser

pyspider



PySpider : about [5700](#) pages/min

Scrapy : about [4800](#) pages/min

scheduler: 17000 pages/min

fetcher: 33000 pages/min (poolsize=100)

processor: 6000 pages/min

<https://gist.github.com/binux/67b276c51e988f8e2c31#comment-1339242>

PyCon 2014 Beijing - [pyspider 介绍\(개소\)](#)

<http://www.slideshare.net/roybinux/pyspider-pycon2014beijing>

1.1 데이터 수집하기

.4 라이브러리 pyspider, [feedparser](#)

목적 : Feed 파싱

feedparser <https://pythonhosted.org/feedparser/>

```
import feedparser
rawdata = """<rss version="2.0">
    <channel>
        <title>Sample Feed</title>
    </channel>
</rss>"""
d = feedparser.parse(rawdata)
print d[ 'feed' ][ 'title' ]
```

Sample Feed

1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

```
print d['feed']['title']
```

MBC뉴스 :: 뉴스(전체)

```
print d.entries[6].title
```

[코리아배드민턴] 이용대·유연성, 16강 진출

```
print d.entries[6].link
```

http://imnews.imbc.com/news/2015/sports/article/3772345_14737.html

```
print d.entries[6].description
```

배드민턴 남자복식 세계 최정상인 이용대(삼성전기)-유연성(수원시청)이 '2015 빅터 코리아 오픈 배드민턴 슈퍼시리즈'에서 쾌조의 출발을 했다. 능킹 1위 이용대-유연...

```
print d.entries[6].published
```

2015.09.16 21:23:59

```
print d.entries[6].published_parsed
```

time.struct_time(tm_year=2015, tm_mon=8, tm_mday=1, tm_hour=0, tm_min=0, tm_sec=0, tm_wday=5, tm_yday=213,

1.1 데이터 수집하기

.4 라이브러리 pyspider, feedparser

```
<item>
<title><! [CDATA[½É¾Æ¿í ¿ì»êÀ, •Í °i, ° Á¤ »Ó°iÅÐÀÌ 6, í °Ë
°Å] ]></title>
<link>http://imnews.imbc.com/news/2015/society/article/
3772348_14729.html</link>
<description><! [CDATA[ °Í»ê ±âÀå°æÂÛ¼'Â ¹ã'ÊÀº ½Ã°£
¿í »Ó½ÀÀÛÀ, •Í ½Ä'ç¿í ÄŞÀÔÇØ ±ÝÇ°À» ÅÐ¾Î¿Â ÇØÀÇ•Í 19»ì ÀÌ, Ð
±º µî 6, íÀ» °Ò±, ¼Ó ÀÔ°ÇÇß½À'ï'Ù. ÀÌ ±º µîÀº Áö³ 6¿ù
29Àï°ÎÅÍ ÇÑ'þ °£ °Í»ê ±âÀå... ]]></description>
<author><! [CDATA[ÀÌµÎ¿Ø] ]></author>
<category><! [CDATA[ »çÈ, ]]></category>
<pubDate>2015.09.16 21:34:47</pubDate>
</item>
```

1.1 데이터 수집하기

.5 라이브러리 [chardet](#), unidecode

목적 : 문자의 캐릭터셋

chardet: The Universal Character Encoding Detector

<https://github.com/chardet/chardet>

```
$ pip install chardet
```

Detects

- **ASCII, UTF-8, UTF-16 (2 variants), UTF-32 (4 variants)**
- Big5, GB2312, EUC-TW, HZ-GB-2312, ISO-2022-CN (Traditional and Simplified Chinese)
- EUC-JP, SHIFT_JIS, CP932, ISO-2022-JP (Japanese)
- **EUC-KR, ISO-2022-KR** (Korean)
- KOI8-R, MacCyrillic, IBM855, IBM866, ISO-8859-5, windows-1251 (Cyrillic)
- ISO-8859-5, windows-1251 (Bulgarian)
- ISO-8859-1, windows-1252 (Western European languages)
- ISO-8859-7, windows-1253 (Greek)
- ISO-8859-8, windows-1255 (Visual and Logical Hebrew)
- TIS-620 (Thai)

1.1 데이터 수집하기

.5 라이브러리 chardet, unidecode

목적 : 문자의 캐릭터셋

chardet

```
import chardet
print chardet.detect( '\xed\x95\x9c\xea\xb5\xad\xec\x96\xb4' )

{'confidence': 0.87625, 'encoding': 'utf-8'}

print '\xed\x95\x9c\xea\xb5\xad\xec\x96\xb4'.decode('utf-8')
```

한국어

1.1 데이터 수집하기

.5 라이브러리 chardet, unidecode

목적 : 문자의 캐릭터셋

chardet

```
import mechanize  
browser = mechanize.Browser()  
page = browser.open(url)  
html = page.read()  
  
print chardet.detect(html)  
{'confidence': 0.99, 'encoding': 'EUC-KR'}  
  
d = feedparser.parse(html.decode('EUC-KR'))
```

1.1 데이터 수집하기

.5 라이브러리 chardet, unidecode

목적 : 문자의 캐릭터셋

unidecode <https://pypi.python.org/pypi/Unidecode>

한국어

30 km/h

```
unidecode(u'한국어')
```

```
'hangugeo'
```

```
unidecode(u'30 \U0001d5c4\U0001d5c6/\U0001d5c1')
```

```
'30 km/h'
```

北京

kožušček

```
unidecode(u"\u5317\u4EB0")
```

```
'Bei Jing '
```

```
unidecode(u'ko\u017eu\u0161\u010dek')
```

```
'kozuscek'
```

1.1 데이터 수집하기

참고자료

PyCon US 2012 - Web scraping: Reliably and efficiently pull data from pages that don't expect it

<https://us.pycon.org/2012/schedule/presentation/317/>

<https://www.youtube.com/watch?v=52wxGESwQSA>

PyCon KOREA 2014 - 30분만에 따라하는 동시성 웹 스크래퍼

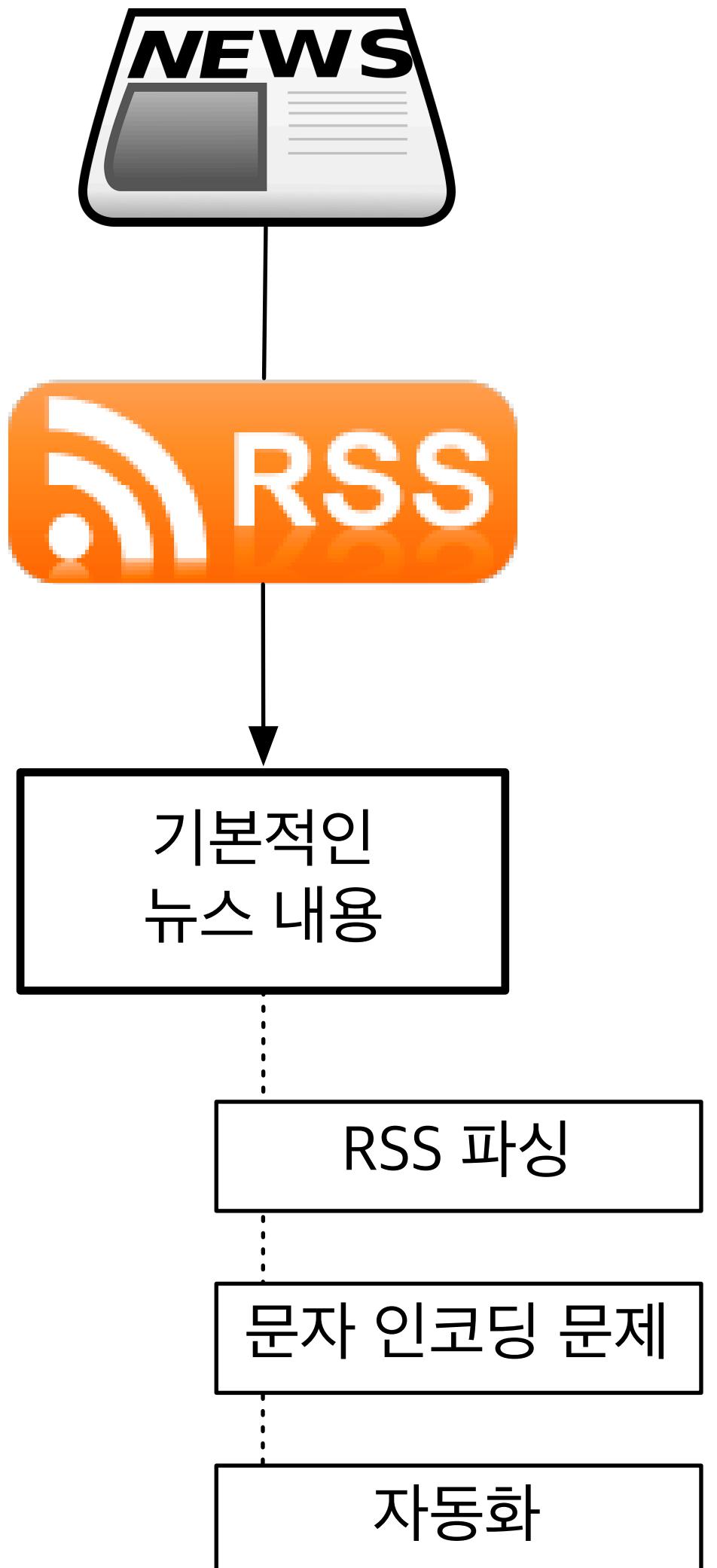
<https://www.pycon.kr/2014/program/15>

PyCon KOREA 2015 - Character Encoding in Python

<https://www.pycon.kr/2015/program/47>

PyCon KOREA 2016 - 문자열? 그런 달달한 것이 남아있긴 한가?

<https://www.pycon.kr/2016apac/program/5>





```
print d.entries[6].description
```

배드민턴 남자복식 세계 최정상인 이용대(삼성전기)-유연성(수원시청)이 '2015 빅터 코리아 오픈 배드민턴 슈퍼시리즈'에서 쾌조의 출발을 했다. 남자복식 세계랭킹 1위 이용대-유연...



```
print d.entries[6].description
```

배드민턴 남자복식 세계 최정상인 이용대(삼성전기)-유연성(수원시청)이 '2015 빅터 코리아 오픈 배드민턴 슈퍼시리즈'에서 쾌조의 출발을 했다. 남자복식 세계랭킹 1위 이용대-유 연...

```
print d.entries[6].link
```

http://imnews.imbc.com/news/2015/sports/article/3772345_14737.html

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

```
from bs4 import BeautifulSoup
```

```
# 데이터 가져오기
```

```
html = requests.get('https://www.pycon.kr/2016apac/program/list/').text
```

```
# 데이터 파싱
```

```
soup = BeautifulSoup(html)
```

```
print soup.title
```

```
<title>파이콘 APAC 2016</title>
```

The screenshot shows the PYCON APAC 2016 website. The header includes the PYCON APAC 2016 logo, navigation links for 파이콘 APAC, 프로그램, 장소, 발표안, 등록, and user account options (로그인, 언어). The main content area displays the '프로그램 목록' (Program Schedule) section, which includes a sidebar for navigating other conference sections like 선호도 조사, 키노트, 발표자, 스프린트와 튜토리얼, 라이팅 토크, and 열린 공간. The program schedule lists various sessions such as 'Profiling the Unprofilable', 'Introduction into aiohttp', and 'GPU Acceleration of a Global Atmospheric Model by Python combining with CUDA'. Below this is the '데이터와 학습' (Data and Learning) section with a list of topics including 'Decision making with Genetic Algorithms using DEAP' and 'Deep Learning with Python & TensorFlow'.

데이터와 학습

- Decision making with Genetic Algorithms using DEAP
- Creating AI chat bot with Python 3 and Tensorflow
- Introduction to deep learning for machine vision tasks using Keras.
- Python 으로 19대 국회 뽐내기
- Python + Spark, 머신러닝을 위한 완벽한 결혼
- 지적 대화를 위한 깊고 넓은 딥러닝 (Feat. TensorFlow) [슬라이드](#)
- 파이썬 데이터 분석 3종 세트 - statsmodels, scikit-learn, theano
- Deep Learning with Python & TensorFlow
- 나의 사진은 내가 지난 과거에 한 일을 알고 있다
- 기계학습을 활용한 게임 어뷰징 검출

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

```
for a in soup.find_all('a'):
    if 'program' in a['href']:
        print a
```

```
<a href="/2016apac/program/schedule">세부 일정</a>
<a href="/2016apac/program/list">프로그램 목록</a>
<a href="/2016apac/program/preference">선호도 조사</a>
<a href="/2016apac/program/keynote">키노트</a>
<a href="/2016apac/program/speaker">발표자</a>
<a href="/2016apac/program/tutorials">스프린트와 튜토리얼</a>
<a href="/2016apac/program/lightning_talk">라이트닝 토크</a>
<a href="/2016apac/program/ost">열린 공간</a>
<a href="/2016apac/program/27">Profiling the Unprofilable</a>
<a href="/2016apac/program/24">Introduction into aiohttp</a>
```

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

```
for a in soup.find_all('a'):
    if 'program' in a['href']:
        if a['href'][-1] in [str(n) for n in range(10)]:
            print a
```

Profiling the Unprofilable

Introduction into aiohttp

GPU Acceleration of a Global Atmospheric Model by Python combining with CUDA

High-performance Networking with Python

Linux Kernel instrumentation in Python

Django에서의 대용량 트래픽 처리 - 병목을 찾아라

You Might Not Want Async

Python Profiling and Performance Tuning

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

Parser	Python 2.7		Python 3.2	
	Speed (KB/s)	Success rate	Speed (KB/s)	Success rate
Beautiful Soup 3.2 (SGMLParser)	211	100%	-	-
html5lib (BS3 treebuilder)	253	99%	-	-
Beautiful Soup 4.0 + lxml	255	100%	2140	96%
html5lib (lxml treebuilder)	270	99%	-	-
Beautiful Soup 4.0 + html5lib	271	98%	-	-
Beautiful Soup 4.0 + HTMLParser	299	59%	1705	57%
html5lib (simpletree treebuilder)	332	100%	-	-
HTMLParser	5194	52%	3918	57%
lxml	17925	100%	14258	96%

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

파서	사용법	장점	단점
Python's html.parser	<code>BeautifulSoup(markup, "html.parser")</code>	내장형	Not very lenient
lxml's HTML parser	<code>BeautifulSoup(markup, "lxml")</code>	매우 빠름	External C dependency
lxml's XML parser	<code>BeautifulSoup(markup, "lxml-xml")</code> <code>BeautifulSoup(markup, "xml")</code>	매우 빠름	External C dependency
html5lib	<code>BeautifulSoup(markup, "html5lib")</code>	브라우저처럼 페이지를 처리 매우 관대함	매우 느림

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/#installing-a-parser>

<http://ecolemodev.wikinamu.com/파이썬#header-6>

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

pyparsing <http://pyparsing.wikispaces.com/>

```
text = u"""""Python 으로 19대 국회 뽕개기
Python + Spark, 머신러닝을 위한 완벽한 결혼
지적 대화를 위한 깊고 넓은 딥러닝 \(Feat. TensorFlow\)
파이썬 데이터 분석 3종 세트 - statsmodels, scikit-learn, theano
Deep Learning with Python & TensorFlow
나의 사진은 내가 지난 과거에 한 일을 알고 있다
기계학습을 활용한 게임 어뷰징 검출
검색 로그 시스템 with Python
PyLadies and PyGents
The stories about Django Girls Taipei
The PSF and our community
TOROS: Python Framework for Recommender System
10만 라인, 26280시간의 이야기""""
```

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

pyparsing

```
from pyparsing import strange, Word, nums, Combine  
koreanChars = strange(r"\xac00-\xd7a3")
```

```
pattern = Word(nums) + Word(koreanChars)
```

```
result = pattern.parseString(u"10만")
```

```
print result
```

```
[u'10', u'\ub9cc']
```

1.2 데이터 파싱하기

.1 라이브러리 beautifulsoup, lxml, pyparsing

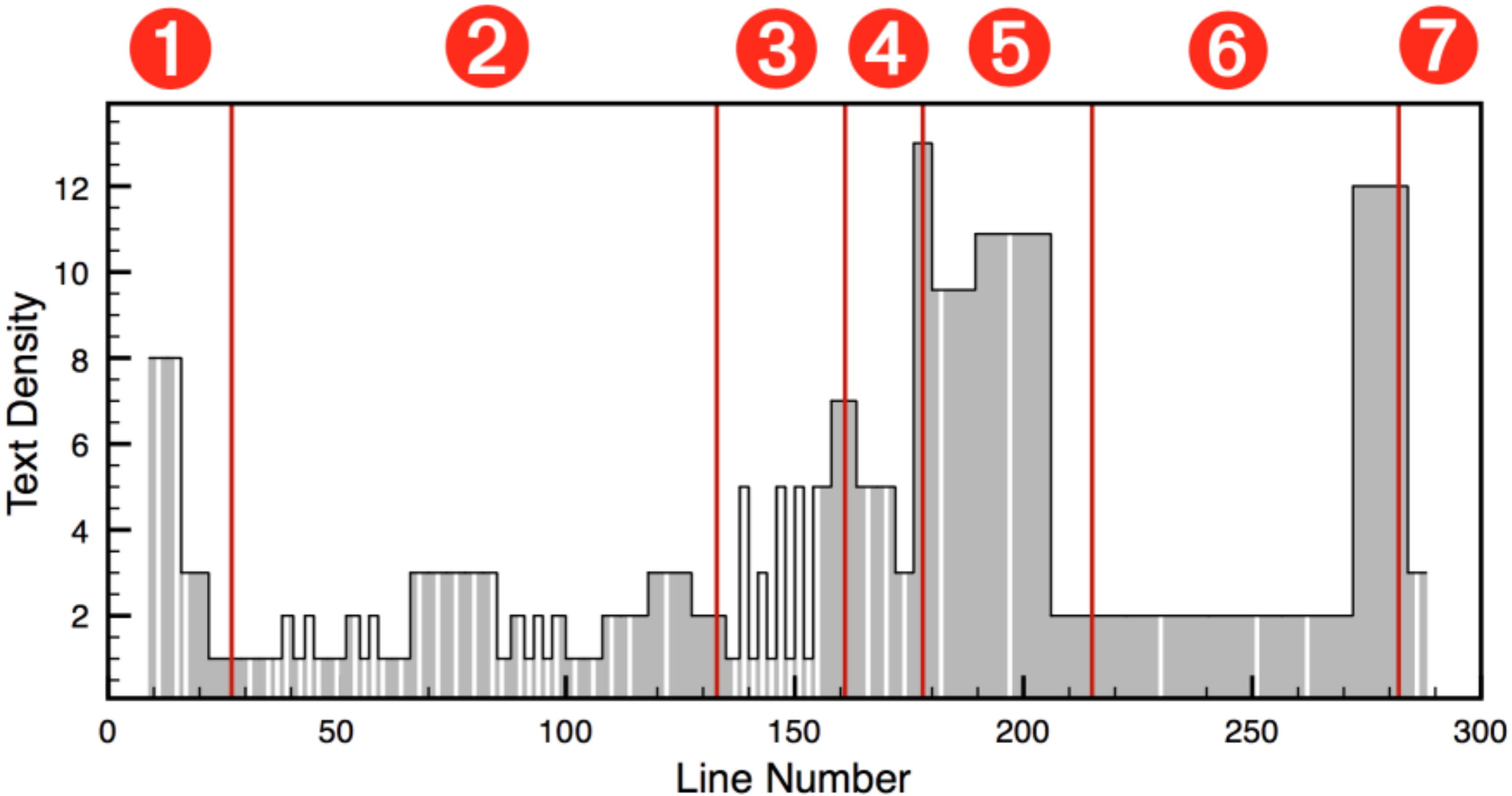
pyparsing

```
pat = Combine( Word(nums)+Word(koreanChars) )
for match, start, stop in pat.scanString(text):
    print match[0], '\t', start, '\t', stop
```

19대	41	44
3종	238	240
10만	838	841
26280시간의	846	854

1.2 데이터 파싱하기

.2 라이브러리 python-goose



Kohlschütter, Christian.

“Exploiting links and text structure on the Web:a quantitative approach to improving search quality”.

Diss. University of Hanover, 2011.

1.2 데이터 파싱하기

.2 라이브러리 python-goose

Goose - Article Extractor



```
$ git clone https://github.com/grangier/python-goose.git
$ cd python-goose
$ pip install -r requirements.txt
$ python setup.py install
```

1.2 데이터 파싱하기

.2 라이브러리 python-goose

```
from goose import Goose
url = 'https://www.washingtonpost.com/sports/olympics/at-rio-games-ryan-lochte-sports-silver-hair-and-an-familiar-title-'
g = Goose()
article = g.extract(url=url)
```

```
print article.title
```

Michael Phelps extends his legacy with two more gold medals at Rio Olympics

```
print article.meta_description
```

With victories in the 200-butterfly and 4x200 relay, swimmer has won 25 medals, 21 gold.

```
print article.cleaned_text
```

Years from now, it will be difficult to imagine that there were questions about what Michael Phelps might do here at age 31, in his fifth Olympics. The fact that he was out of his sport, then nearly out of his head, could seem utterly forgettable. What will remain are the nights like Tuesday, when he touched the wall first, gestured to the crowd that he wanted to hear more, then straddled the lane line and raised both arms.

Bronze the statue like that, perhaps? Yet those are the particulars from just one race on just one night, the 200-meter butterfly on Tuesday, gold again. We're getting to the point at the Olympic Aquatics Center where we could make a parlor game of picking Phelps's event and matching it with its celebration. Tuesday, he dragged his bones from the pool after one win, then dove in again — swimming the anchor leg on the Americans' 4x200-meter relay. Darn it if he didn't take gold again.

So flip two more beads over on Phelps's abacus. The gold in the 200 fly was the 20th of his career, the gold in the relay his 21st. He now has 25 total Olympic medals, and pointing out these are records doesn't fully convey the enormi-

1.2 데이터 파싱하기

.2 라이브러리 python-goose

```
from goose import Goose
from goose.text import StopWordsKorean
```

```
url = 'http://news.donga.com/3/all/20131023/58406128/1'
```

```
g = Goose({'stopwords_class':StopWordsKorean})
```

```
article = g.extract(url=url)
```

```
print article.title
```

[CEO&]창의적 열정과 소통, 초일류 시험인증 기업이 간다

```
print article.cleaned_text
```

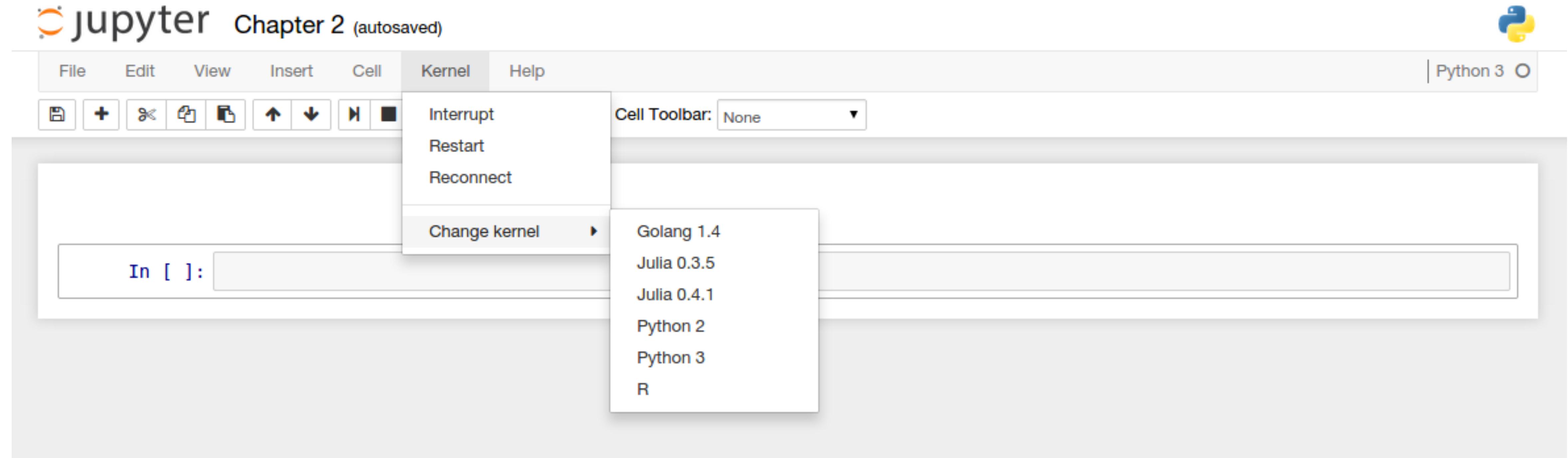
경기도 용인에 자리 잡은 민간 시험인증 전문기업 (주)디지털이엠씨 (www.digitalemc.com). 14년째 세계 각국의 통신·안전·전파 규격 시험과 인증 한 우물만 파고 있는 이 회사 박채규 대표가 만나기로 한 주인공이다. 그는 전기전자·무선통신·자동차 전장품 분야에서 국제적으로 인정받는 전문가다.“시험인증 분야는 새로운 기술에 대한 준비와 교육 없이는 아무것도 이루어 낼 수 없습니다. 따라서 임직원의 교육과 자질 향상을 위해 많은 시간과 비용을 투자하고 있지요.”(주)디지털이엠씨의 하루 일과는 항상 바쁘게 돌아간다. 이 회사에선 원어민이 진행하는 사내 영어강좌가 1년 내내 이어진다. 파트와 팀별로 전문기술교육, 세미나 등도 수시로 이뤄진다. 해외 인증기관과의 교류가 무엇보다 중요한 업무이기에 해당 국가의 규격, 시험, 인증에 대한 교육 강도는 상상을 초월한다. 회사의 막내에서부터 CEO까지 막힘없는 ‘소통경영’이 이루어지는 곳이 (주)디지털이엠씨다. 이 회사 이동훈 이사는 “소통을 수용하는 열린 기업문화, 창의적인 열정을 끌어내는 젊고 합리적인 리더십이 직원들이 몰입할 수 있는 일터를 만든 비결”이라고 귀띔했다. 1999년 설립된 (주)디지털이엠씨는 철저한 업무 분장을 통한 리스크관리 체계를 갖추고 있다. 기술시험 및 인증연구소와 영업팀, 관리팀, 해외지사(베이징·타슈켄트), 합작법인(브라질)으로 조직됐다. 분야별 시험시설 장비 기술전문인력 등은 세계 어느 곳과 경쟁해도 손색 없을 정도로 높은 수준을 자랑한다. 삼성, LG를 포함해 1300여 개 고객사를 대상으로 전 세계 150여 개 국가의 규격 인증 서비스를 제공하고 있다.

1.2 데이터 파싱하기

.3 라이브러리 newspaper

<https://github.com/codelucas/newspaper>

<https://github.com/codelucas/newspaper/tree/python-2-head>



Jupyter는 다중 커널 지원!

<https://github.com/jupyter/jupyter/issues/71>
<http://blog.nacyot.com/articles/2015-05-08-jupyter-multiple-pythons/>

1.2 데이터 파싱하기

.3 라이브러리 newspaper

```
import newspaper

url = 'http://www.nytimes.com/interactive/2016/08/10/sports/olympics/gymnastics-parents.html?hp&action=click&pgtype=Home'

article = newspaper.Article(url)

article.download()

article.parse()

article.title

'Watching Your Daughter Win Gymnastics Gold Looks Like This'

article.text

"Credit Leslye Davis/The New York Times\n\nRIO DE JANEIRO — The Olympics do not suffer from being photographed too little. Nearly every move by nearly every Olympian is photographed, filmed, recorded or even memed, consumed by billions of fans across the world.\n\nBut another drama happens off camera, as the athletes' parents — many of whom have spent years making all kinds of sacrifices for this moment to happen — watch their children perform on the biggest stage.\n\nDuring the women's team gymnastics competition on Tuesday, we turned some of our cameras around and focused on those parents.\n\nThe sequences below capture five American gymnasts — Simone Biles, Gabby Douglas, Laurie Hernandez, Madison Kocian and Aly Raisman — during their gold medal-winning performance alongside images of their parents, taken at about the same time.\n\nThomas Kocian Madison's father Madison Kocian Uneven bars Madison Kocian Uneven bars\n\nMadison Kocian, a bars specialist, performed one of the most difficult routines at the Games, and she nailed it. From start to finish, her father Thomas was a picture of intensity, usually moving on smiling. But when Kocian landed his final jump, Thomas was all smiles."
```

1.2 데이터 파싱하기

.3 라이브러리 newspaper

```
article = newspaper.Article(url)
```

```
article.download()
```

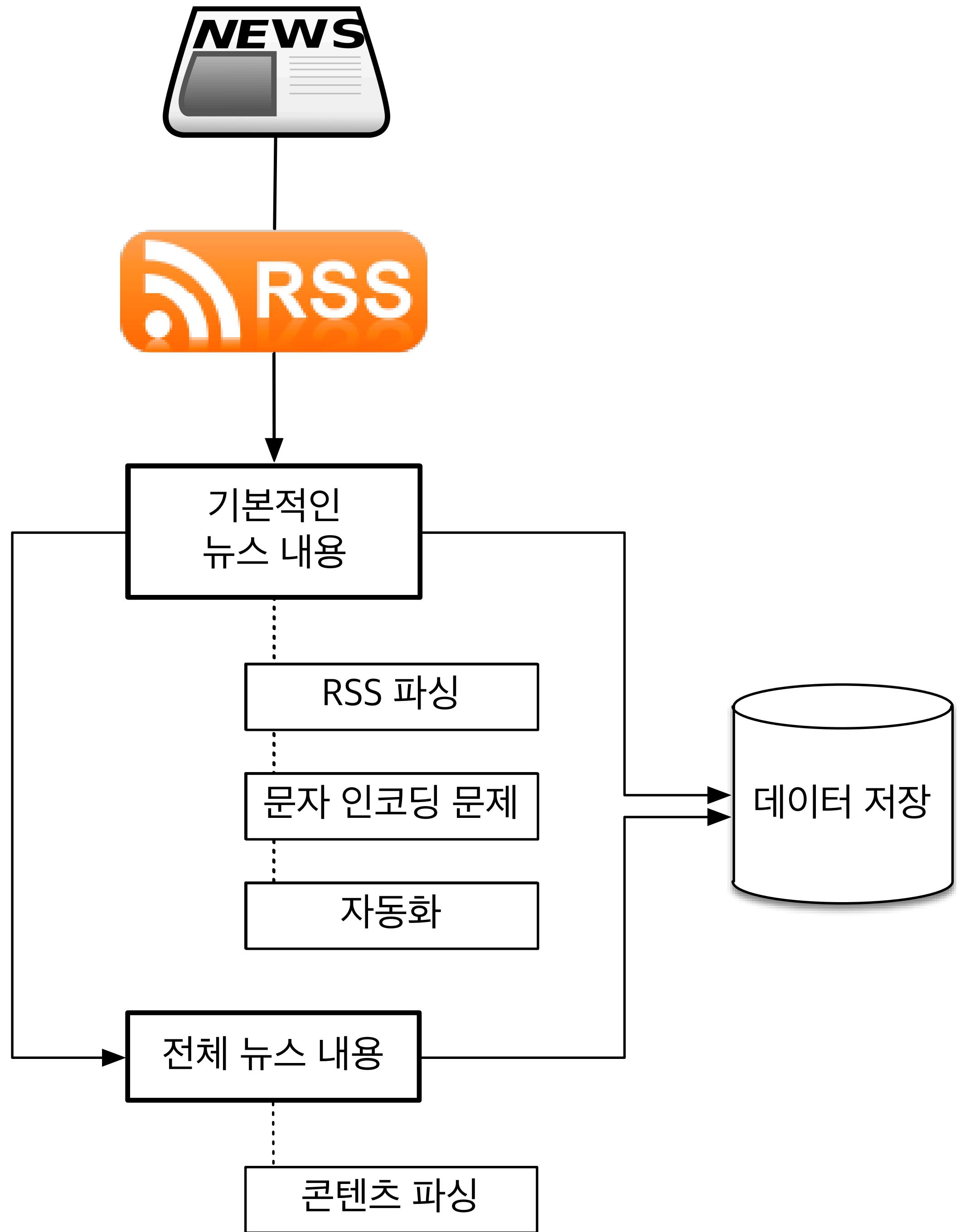
```
article.parse()
```

```
article.title
```

"UNIST, 고성능 '흑연·실리콘 복합체' 개발
"2차전지 우수 음극소재 원천기술 선점""

```
article.text
```

"UNIST, 고성능 '흑연·실리콘 복합체' 개발\n\n"2차전지 우수 음극소재 원천기술 선점" 8면 페이스북\n\n트위터\n\n카카오토리\n\n밴드\n\n네이버 블로그 2016년 08월 10일 (수) 김기곤 기자 nafol@iusm.co.kr ▲ 고출력 '흑연·실리콘 복합체' 연구에 참여한 UNIST 연구진.(왼쪽부터 채수종 연구원, 김남형 연구원, 조재필 교수, 마지막 연구원) ▲ '흑연·실리콘 복합체' 구조와 특징. 전기차 배터리에 적용땐\n\n주행거리 300km까지 연장\n\n가격·안전성 경쟁력 확보\n\n대량생산 장비도 개발\n\n\nUNIST(총장 정무영)는 에너지 및 화학공학부 조재필 교수팀이 기존 음극소재인 흑연보다 45% 용량을 늘린, 고출력 '흑연·실리콘 복합체'를 개발했다고 9일 밝혔다.\n\n특히 흑연·실리콘 복합체 음극소재를 대량으로 생산할 수 있는 장비도 개발했다. 이 장비를 이용하면 한 번에 300kg 이상의 음극소재를 6시간 정도에 제작 가능하다. 공정절차도 간단한 편이라 가격경쟁력도 확보했다는 평가를 받고 있다.\n\n조재필 교수는 "새로운 음극소재는 최근 연구개발 경쟁에 불붙은 전기차의 주행거리 연장에도 크게 기여할 것"이라며 "흑연계 물질만 사용하면 주행거리가 200km 안팎에 머물지만, 이번에 개발된 음극소재로 전기차 배터리를 만들면 300km까지 주행할 수 있을 것"이라고 기대했다.\n\n고에너지밀도·고성능·저가의 음극소재 개발이 지지부진한 상태에서 기존 음극소재로 쓰이는 흑연의 대체물질로 실리콘이 주목받고 있다. 실리콘 소재의 용량이 상용화된 흑연보다 10배 이상 커기 때문이다.\n\n하지만 실리콘 소재는 충전과 방전을 반복하는 동안 4배 정도 부피가 늘어나고, 전지 성능도 급격히 감소하는 문제점이 있다.\n\n조재필 교수팀이 이같은 문제점을 해결하기 위해 기존 흑연 음극소재에 실리콘 나노 코팅기술을 적용해 이종물질 간에 최적의 호환성을 갖는 흑연·실리콘 복합체를 구현한 것이다.\n\n이 물질은 충·방전 동안에도 부피가 크게 늘지 않았고, 전자와 리튬이온의 이동거리가 줄어들어 고속 충·방전이 가능해졌다.\n\n조재필 교수는 "이번에 개발한 음극 소재는 현재 개발된 일본과 중국의 경쟁사와 비교해 동일 용량을 가진 전지 평가에서 부피 팽창률이 15% 이상 감소됐다"며 "전 세계적으로 연구개발 경쟁이 불붙은 이차전지 개발에서 가격 및 안전성 측면에서 가장 우수한 음극소재 원천기술을 선점했다는 점에서 큰 의미가 있다"고 강조했다.\n\n그는 이어 "향후 전기 자동차나 중대형 에너지 저장장치에도 적용이 가능할 것으로 기대돼 국가 경쟁력 확보에도 핵심적인 역할을 할 것"이라고 밝혔다.\n\n한편 이차전지 연구 분



1.3 데이터 저장하기

.1 라이브러리 PyMongo



1. MongoDB 다운로드, 설치 및 데몬 실행
<https://www.mongodb.com/download-center>
2. \$ pip install pymongo

```
from pymongo import MongoClient
client = MongoClient('localhost', 27017)
db = client.testDatabase
```

1.3 데이터 저장하기

.1 라이브러리 PyMongo

Create

```
db.testCollection.insert_one({'a':1, 'b':'c', 'd':datetime.datetime.now()})
```

Read

```
db.testCollection.find_one()
```

```
{u'_id': ObjectId('57ab3445255b001dfca7bc50'),
 u'a': 1,
 u'b': u'c',
 u'd': datetime.datetime(2016, 8, 10, 23, 3, 49, 769000)}
```

Update

```
db.testCollection.update_one({'a':1}, {'$set': {'d': datetime.datetime.now()}})
db.testCollection.find_one()
```

```
{u'_id': ObjectId('57ab3445255b001dfca7bc50'),
 u'a': 1,
 u'b': u'c',
 u'd': datetime.datetime(2016, 8, 10, 23, 4, 49, 911000)}
```

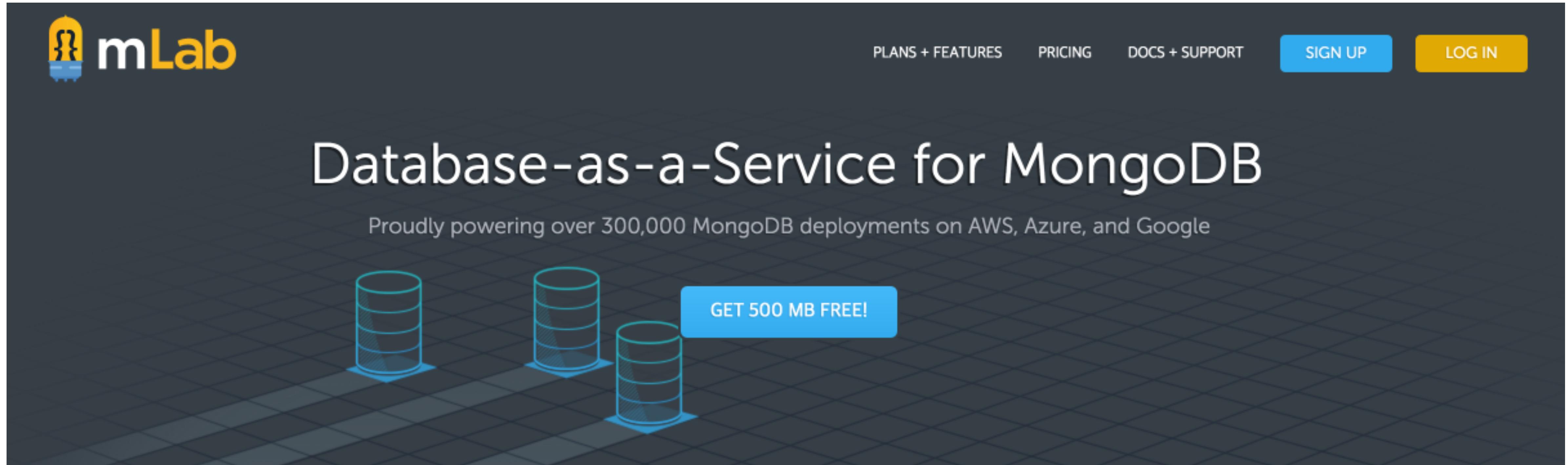
Delete

```
db.testCollection.delete_one({'a':1})
```

[https://api.mongodb.com/python/
current/tutorial.html](https://api.mongodb.com/python/current/tutorial.html)

1.3 데이터 저장하기

.2 Cloud Service (mongolab, aws, azure, etc.)



<https://mlab.com/>

1. 뉴스를 재미있게 만들기 위한

수집

1.1 데이터 수집하기

1.1.0 데이터 소스

1.1.1 라이브러리 urllib2, request, mechanize

1.1.2 라이브러리 chardet, unidecode

1.1.3 라이브러리 pyspider, feedparser

1.1.4 라이브러리 robobrowser

1.2 데이터 파싱하기

1.2.1 라이브러리 beautifulsoup, lxml, pyparsing

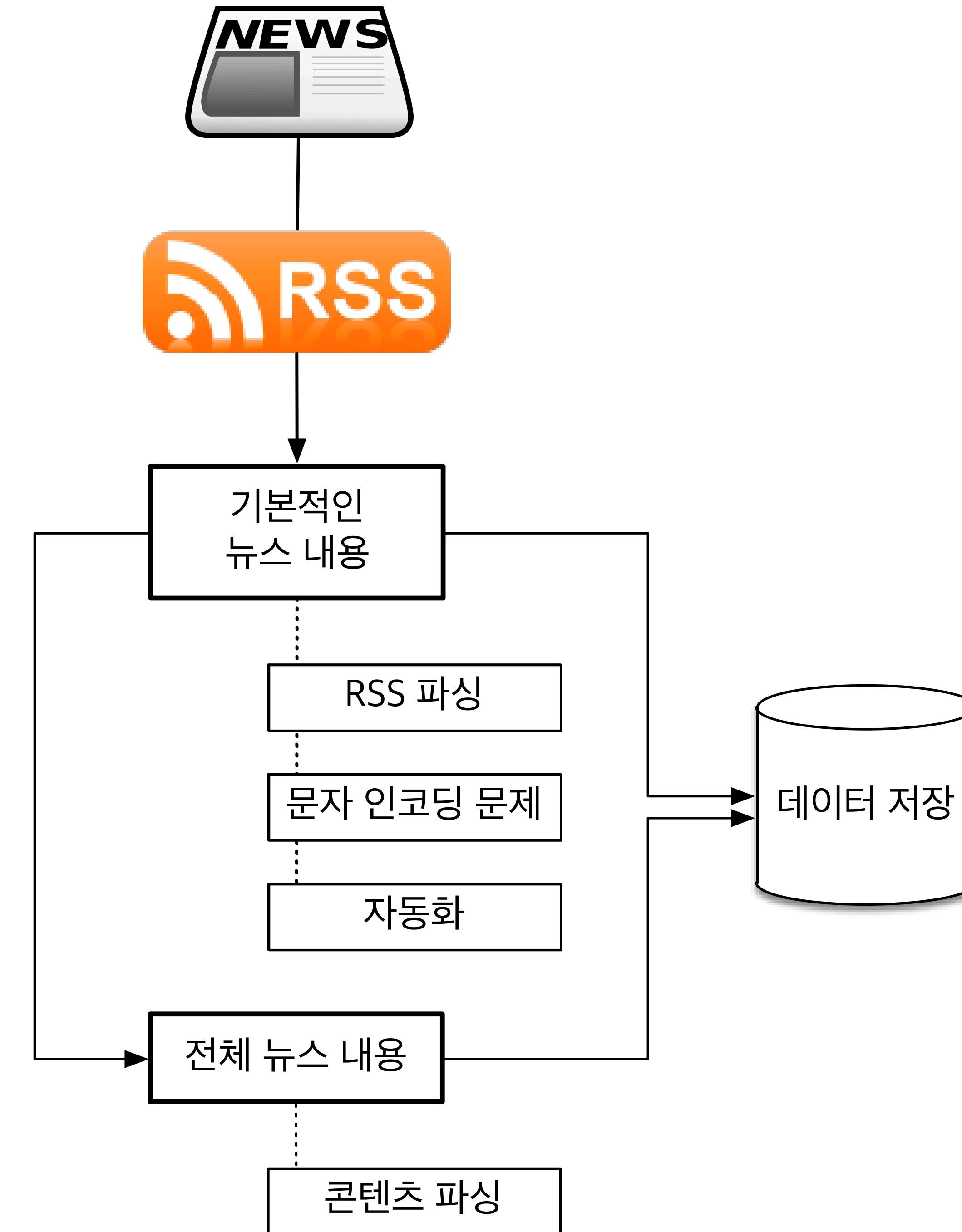
1.2.2 라이브러리 python-goose

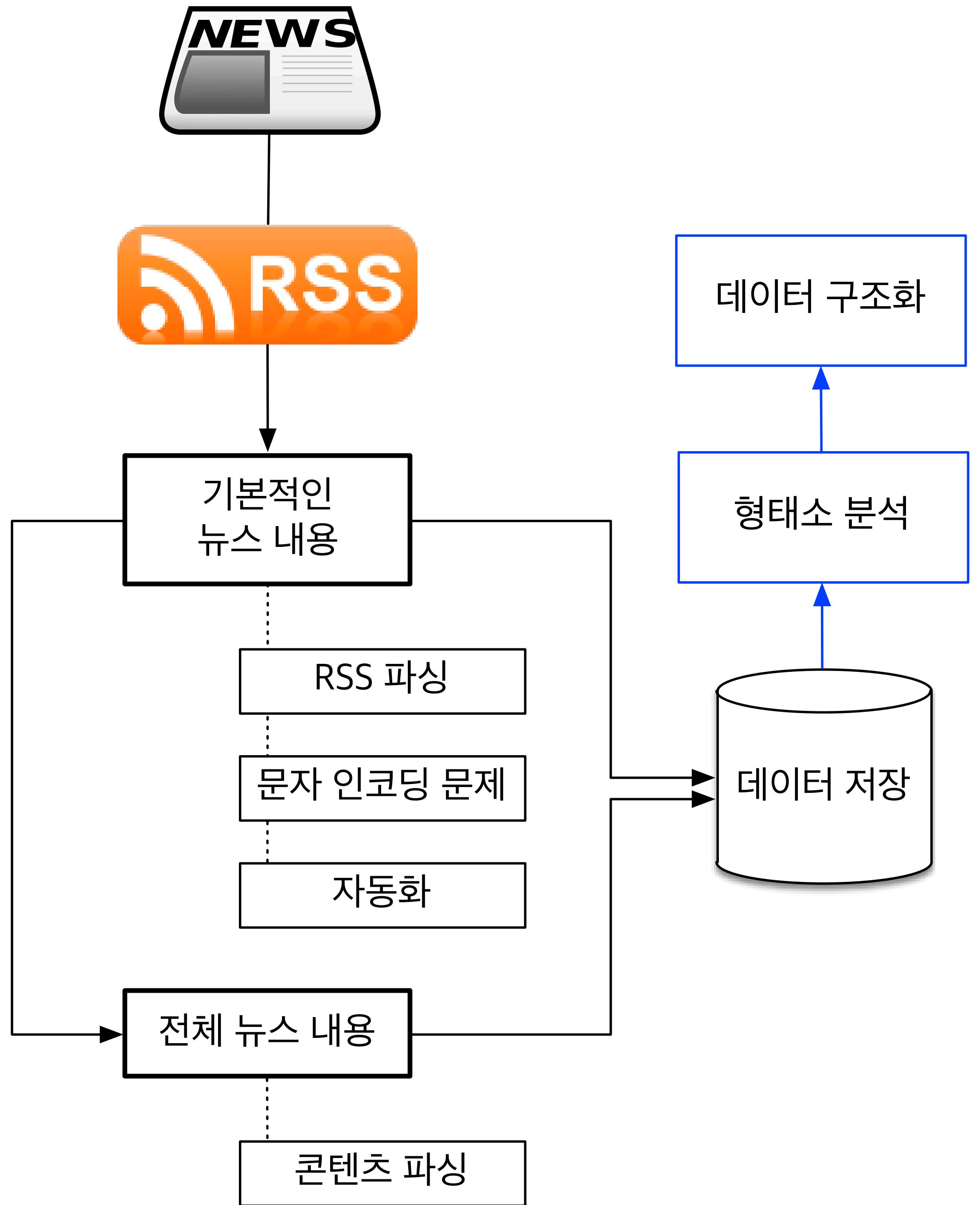
1.2.3 라이브러리 newspaper

1.3 데이터 저장하기

1.3.1 라이브러리 PyMongo

1.3.2 Cloud Service (mongolab, aws, azure)





2.1 데이터의 전처리

.1 형태소 분석기

형태소 : 일정한 뜻을 가진 가장 작은 말의 단위

Input

파이콘은 세계 각국의 파이썬 프로그래밍 언어 커뮤니티에서 주관하는 비영리 컨퍼런스입니다.

Output

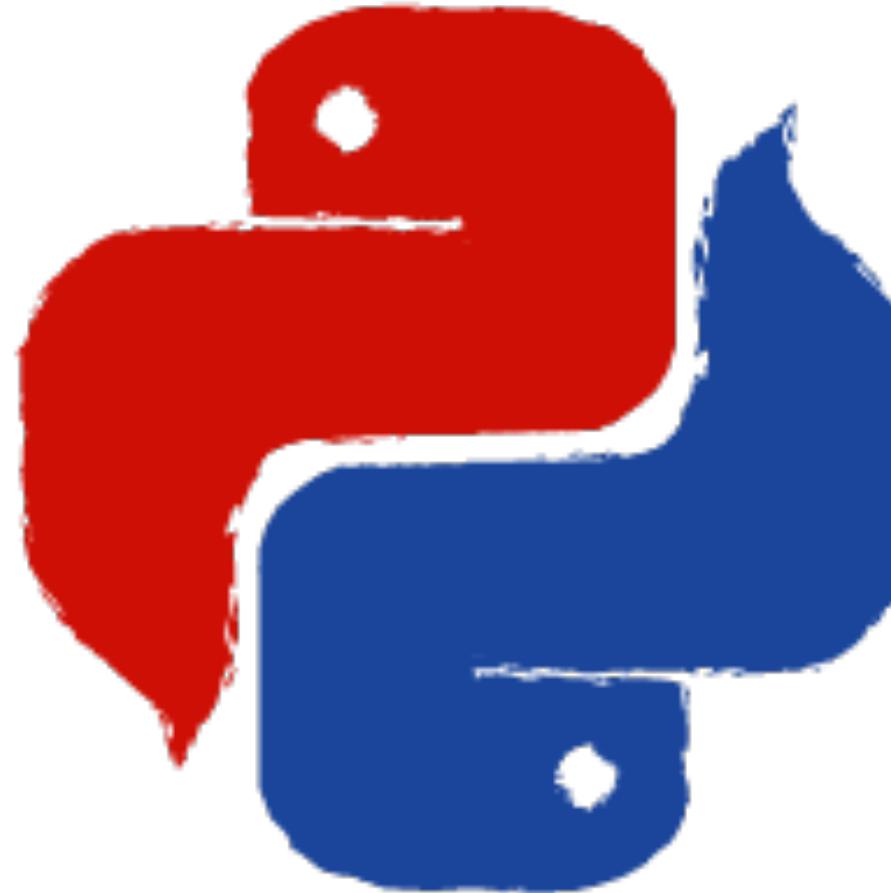
파이	NNG,* ,F,파이,*,*,*
콘	NNG,* ,T,콘,*,**
은	JX,* ,T,은,*,**
세계	NNG,* ,F,세계,*,**
각국	NNG,* ,T,각국,*,**
의	JKG,* ,F,의,*,**
파이썬	NNP,* ,T,파이썬,*,**
프로그래밍	NNG,* ,T,프로그래밍,*,**
언어	NNG,* ,F,언어,*,**
커뮤니티	NNG,* ,F,커뮤니티,*,**
에서	JKB,* ,F,에서,*,**
주관	NNG,* ,T,주관,*,**
하	XSV,* ,F,하,*,**
는	ETM,* ,T,는,*,**
비	XPN,* ,F,비,*,**
영리	NNG,* ,F,영리,*,**
컨퍼런스	NNG,* ,F,컨퍼런스,*,**
입니다	VCP+EF,* ,F,입니다,Inflect,VCP,EF,이/VCP/*+ㅂ니다/EF/* SF,*****

2.1 데이터의 전처리

.2 라이브러리 Konlpy, umorpheme

PyCon KOREA 2014 - 자바, 미안하다! 파이썬 한국어 NLP

<https://www.pycon.kr/2014/program/10>



KoNLPy

- **한나눔 프로젝트 Hannanum** (<http://klip.net/hannanum/>)
- **꼬꼬마 프로젝트 Kkma** (<http://kkma.snu.ac.kr/>)
- **코모란 Komoran** (http://www.shineware.co.kr/?page_id=835)
- **은전한닢 프로젝트 Mecab** (<https://bitbucket.org/eunjeon/mecab-ko/>)
- **트위터 Twitter** (<https://github.com/twitter/twitter-korean-text>)

2.1 데이터의 전처리

.2 라이브러리 Konlpy, umorpheme

PyCon KOREA 2015 - NetworkX를 이용한 네트워크 링크 예측

<https://www.pycon.kr/2015/program/35>

The screenshot shows a web application interface for the "Korean Morpheme Analyzer API". The top navigation bar includes "Information.Center" and a menu icon. On the left, a sidebar lists "Beta 0.1" (Updated 2015.08.07), "Logout", "Main", "Korean Morpheme API" (new), and "MyPage". The main content area displays the title "Korean Morpheme Analyzer API" and its subtitle "한국어 형태소 분석기 API". A breadcrumb navigation shows "Home > Morpheme > API". Below the title, there's an "Introduction" section with a bell icon. A message states: "We provide a Korean morpheme analyzer API for convenience. In order to get the api key, register your information." A blue "Service Registration" button is present. Under the "Registered Information" heading, there's a checked checkbox. Below it, four items are listed: "Working for" (데모용), "URL" (http://pycon.kr/2016apac), "Problem" (형태소 분석), and "API" (koorCC2YEOBDTQ). At the bottom right of this section is a "Delete" button.

Beta 0.1 Logout

Updated 2015.08.07.

Main

Korean Morpheme API new

MyPage

Korean Morpheme Analyzer API 한국어 형태소 분석기 API

Home > Morpheme > API

Introduction

We provide a Korean morpheme analyzer API for convenience. In order to get the api key, register your information.

Service Registration

Registered Information

Working for 데모용

URL http://pycon.kr/2016apac

Problem 형태소 분석

API koorCC2YEOBDTQ

Delete

2.1 데이터의 전처리

.2 라이브러리 Konlpy, umorpheme

<http://information.center/api/korean?sc=koorCC2YEOBDTQ&s=%ED%95%91%ED%8A%8D%EC%84%A4%ED%8A%A8>

```
{"0": {"data": "한글", "feature": "NNG"},  
"1": {"data": "입니다", "feature": "VCP+EC"}}
```

<http://information.center/api/korean?sc=koorCC2YEOBDTQ&s=%ED%95%91%ED%8A%8D%EC%84%A4%ED%8A%A8&nounlist=%ED%95%91%ED%8A%8D%EC%84%A4>

```
{"0": {"data": "한글입", "feature": "CUSTOM"},  
"1": {"data": "니다", "feature": "VCP+EC"}}
```

2.1 데이터의 전처리

.3 docker를 활용한 휴대용 형태소 분석

<https://github.com/koorukuroo/mecab-ko>

<https://github.com/koorukuroo/mecab-ko-web> # Docker 설치 방법 (Windows, Mac, Ubuntu)

Docker 실행

```
$ sudo docker pull koorukuroo/mecab-ko-web
$ sudo docker run -i -t koorukuroo/mecab-ko-web
172.17.0.43 (Docker Container IP)
127.0.0.1
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Python

```
>>> import urllib2
>>> response = urllib2.urlopen('http://172.17.0.43:5000/?text=안녕')
>>> text = response.read()
>>> print text
안녕 NNG,*,T,안녕,*,*,*,*
EOS
```

2.1 데이터의 전처리

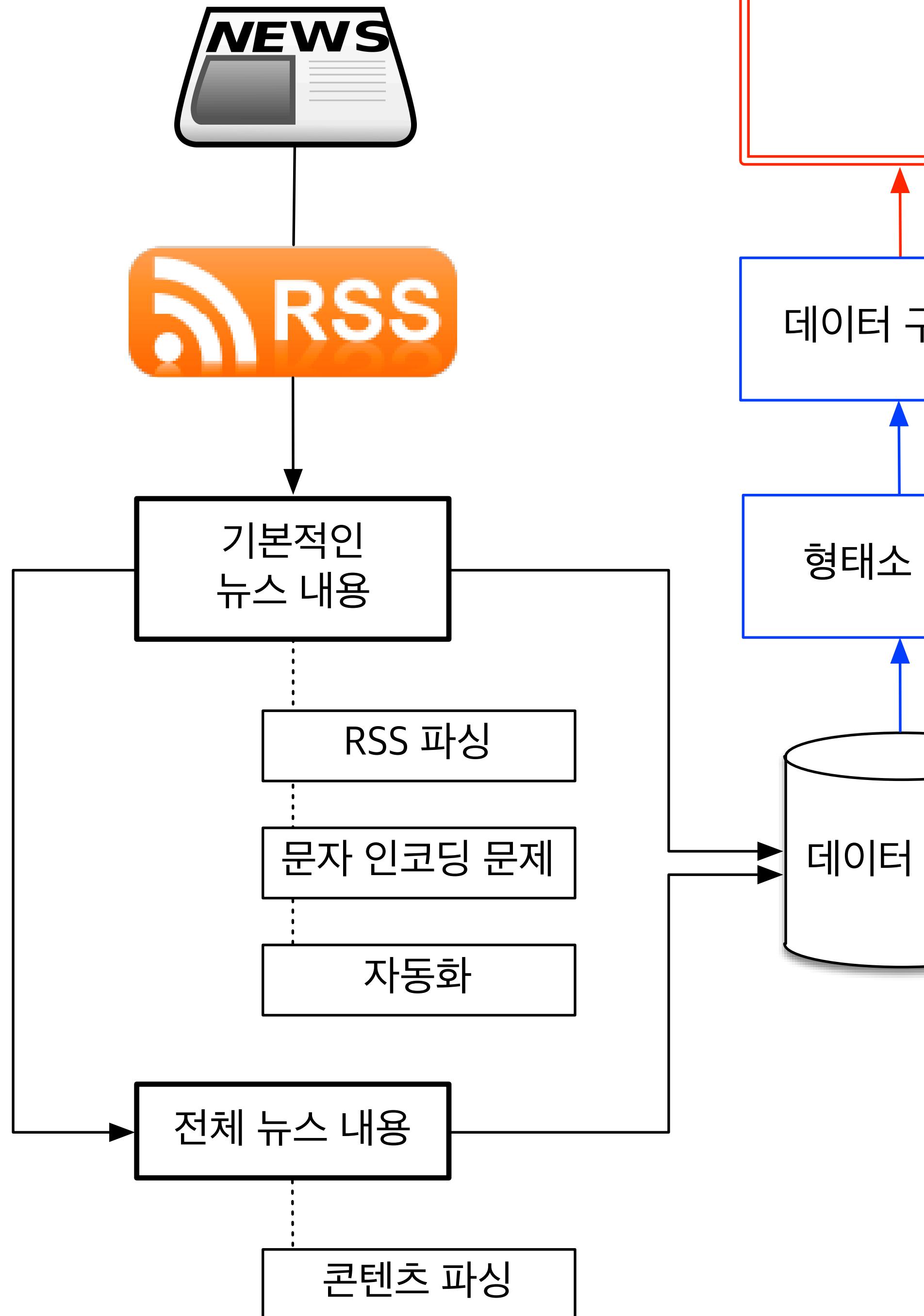
.4 라이브러리 NLTK, textblob

PyCon KOREA 2015 - 한국어와 NLTK, Gensim의 만남

<https://www.pycon.kr/2015/program/36>

튜토리얼 : <https://www.lucypark.kr/courses/2015-ba/text-mining.html>

- Tokenize
- Chunking
- TF-IDF (Term Frequency - Inverse Document Frequency)
- LSI (Latent Semantic Indexing)
- LDA (Latent Dirichlet Allocation)
- HDP (Hierarchical Dirichlet Processes)
- word2vec



- LSI (잠재 의미 색인)
- NMF (비음수 행렬 인수분해)
- LDA (잠재 디리끌레 할당)

- 비슷한 뉴스가 뭐가 있을까?
- 뉴스 주제가 뭐야?

2.2 뉴스를 분류하기

.1 LSI (Latent Semantic Indexing, 잠재 의미 색인)

	d1	d2	d3
w1	1	0	0
w2	0	1	0
w3	1	1	1
w4	1	1	0
w5	0	0	1

단어-문서 행렬

코사인 유사도

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

2.2 뉴스를 분류하기

.1 LSI (Latent Semantic Indexing, 잠재 의미 색인)

	d1	d2	d3
w1	1	0	0
w2	0	1	0
w3	1	1	1
w4	1	1	0
w5	0	0	1

단어-문서 행렬

```
import numpy as np
```

```
d1 = np.array([1,0,1,1,0])  
d2 = np.array([0,1,1,1,0])  
d3 = np.array([0,0,1,0,1])
```

```
np.dot(d1, d2) /\  
(np.sqrt(np.sum(d1**2))*np.sqrt(np.sum(d2**2)))
```

0.6666666666666674

```
np.dot(d1, d3) /\  
(np.sqrt(np.sum(d1**2))*np.sqrt(np.sum(d3**2)))
```

0.40824829046386296

2.2 뉴스를 분류하기

.1 LSI (Latent Semantic Indexing, 잠재 의미 색인)

$$\text{matrix} = \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^*$$

	d1	d2	d3
w1	1	0	0
w2	0	1	0
w3	1	1	1
w4	1	1	0
w5	0	0	1

=

-0.27	0.21	0.70	-0.53	0.30
-0.27	0.21	-0.70	-0.53	0.30
-0.71	-0.33	0	-0.10	-0.60
-0.55	0.43	0	0.64	0.29
-0.15	-0.77	0	0.10	0.60

특이값 분해 (SVD, Singular Value Decomposition)

2.35	0	0
0	1.19	0
0	0	1.00
0	0	0
0	0	0

-0.65	0.26	0.70
-0.65	0.26	-0.70
-0.36	-0.92	0

2.2 뉴스를 분류하기

.1 LSI (Latent Semantic Indexing, 잠재 의미 색인)

$$\text{matrix} = \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^*$$

	d1	d2	d3
w1	0.5	0.5	0
w2	0.5	0.5	0
w3	1	1	1
w4	1	1	0
w5	0	0	1

=

-0.27	0.21	0.70	-0.53	0.30
-0.27	0.21	-0.70	-0.53	0.30
-0.71	-0.33	0	-0.10	-0.60
-0.55	0.43	0	0.64	0.29
-0.15	-0.77	0	0.10	0.60

특이값 분해 (SVD, Singular Value Decomposition)

2.35	0	0
0	1.19	0
0	0	0
0	0	0
0	0	0

-0.65	0.26	0.70
-0.65	0.26	-0.70
-0.36	-0.92	0

2.2 뉴스를 분류하기

.2 행렬의 계산

```
matrix = np.matrix(np.transpose([d1, d2, d3]))
```

```
U, s, V = np.linalg.svd(matrix)
```

```
np.round(U, 2)
```

```
array([[ -0.28,   0.22,   0.71,  -0.53,   0.3 ],
       [-0.28,   0.22,  -0.71,  -0.53,   0.3 ],
       [-0.71,  -0.34,   -0. ,  -0.11,  -0.6 ],
       [-0.56,   0.44,   0. ,   0.64,   0.3 ],
       [-0.16,  -0.77,   -0. ,   0.11,   0.6 ]])
```

```
np.round(s, 2)
```

```
array([ 2.36,   1.2 ,   1.  ])
```

```
np.round(V, 2)
```

```
array([[ -0.66,  -0.66,  -0.37],
       [ 0.26,   0.26,  -0.93],
       [ 0.71,  -0.71,  -0. ]])
```

$$\text{matrix} = \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^*$$

2.2 뉴스를 분류하기

.3 라이브러리 gensim

<https://radimrehurek.com/gensim/tutorial.html>

RSS 읽기

```
data = []
for rss in news_rss:
    browser = mechanize.Browser()
    page = browser.open(rss['url'])
    html = page.read()
    data.append(html)
```



뉴스 제목, 내용, 링크로 분리

```
result = []
for idx, datum in enumerate(data):
    d = feedparser.parse(datum.decode(chardet.detect(datum)['encoding']))
    for i in d.entries:
        result.append([i['title'], i['summary'], i['link']])
```

2.2 뉴스를 분류하기

.3 라이브러리 gensim

형태소 분석해서 명사만 뽑기

```
documents = []
for idx, r in enumerate(result):
    text = r[0] +' '+r[1]
    values = { 's':text.encode( 'utf8' ) }
    r = requests.get( "http://information.center/api/korean?sc=koorJRK3XO1G0W" , params=values)

doc = []
body = json.loads(r.text)
for i in body:
    if body[i][ 'feature' ] in [ 'NNG' , 'NNP' ]:
        doc.append(body[i][ 'data' ])
documents.append(doc)
```

2.2 뉴스를 분류하기

.3 라이브러리 gensim

Gensim corpora

```
from gensim import corpora  
dictionary = corpora.Dictionary(documents)  
print dictionary
```

```
Dictionary(3517 unique tokens: [u'\uc81c\uc555', u'\uc785', u'\uc554\ucef7',  
u'\uc9c0\uc815', u'\ud55c\uad6d\uc740\ud589']...)
```

```
print dictionary.token2id
```

```
{u'\uc81c\uc555': 47,  
 u'\uc785': 1378,
```

2.2 뉴스를 분류하기

.3 라이브러리 gensim

Gensim corpora

```
dictionary.doc2bow(documents[200])
```

```
[(24, 1),  
 (64, 1),  
 (66, 1),  
 (75, 1),  
 (154, 1),  
 (163, 1),  
 (572, 1),  
 (605, 1),  
 ...]
```

2.2 뉴스를 분류하기

.3 라이브러리 gensim

Gensim TF-IDF (Term Frequency – Inverse Document Frequency)

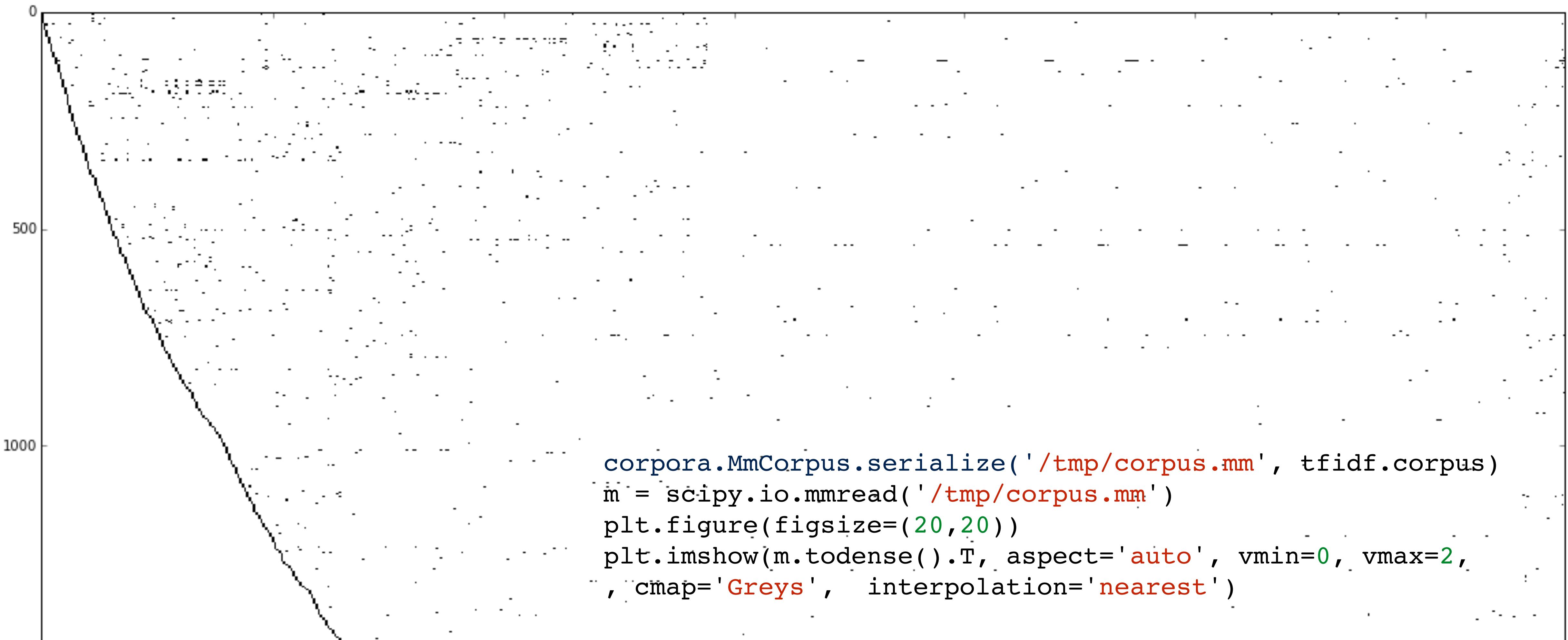
```
from gensim import models  
tf = [dictionary.doc2bow(doc) for doc in documents]  
tfidf_model = models.TfidfModel(tf)  
tfidf = tfidf_model[tf]
```

2.2 뉴스를 분류하기

.3 라이브러리 gensim

Gensim TF-IDF

matrix shape
(3517 , 661)



2.2 뉴스를 분류하기

.3 라이브러리 gensim

Gensim LSI

```
ntopics, nwords = 5, 5
lsi = models.lsimodel.LsiModel(tfidf, id2word=dictionary,
num_topics=ntopics)
for idx, i in enumerate(lsi.print_topics(num_topics=ntopics,
num_words=nwords)):
    print idx, ':', i
```

0 : 0.497*"철도" + 0.427*"파업" + 0.373*"노조" + 0.265*"운행" + 0.178*"보름"
1 : 0.548*"날씨" + 0.392*"주위" + 0.334*"눈" + 0.315*"성탄절" + 0.293*"목요일"
2 : -0.329*"시각" + -0.300*"뉴스" + 0.292*"운행" + -0.268*"정보" + -0.265*"교통"
3 : 0.420*"시각" + 0.338*"교통" + 0.336*"정보" + 0.302*"운행" + 0.217*"지연"
4 : -0.622*"뉴스" + -0.456*"주요" + -0.406*"오늘" + 0.223*"시각" + 0.180*"정보"

2.3 뉴스를 군집하기

.1 NMF (Non-negative Matrix Factorization, 비음수 행렬 인수분해)

$$V \approx W \cdot H$$

	d1	d2	d3
w1	1	0	0
w2	0	1	0
w3	1	1	1
w4	1	1	0
w5	0	0	1

=

	f1	f2
w1	0.56	0
w2	0.56	0
w3	1.13	3.35
w4	1.13	0
w5	0	3.39

	d1	d2	d3
f1	0.89	0.89	0.01
f2	0	0	0.29

2.3 뉴스를 군집하기

.2 라이브러리 nimfa

nimfa <https://github.com/marinkaz/nimfa>

```
import nimfa
nmf = nimfa.Nmf(matrix, seed="random_vcol", rank=2,
max_iter=2000)

fit = nmf()
W = fit.basis()
H = fit.coef()
```

2.4 토픽모델링

.1 LDA (Latent Dirichlet Allocation)

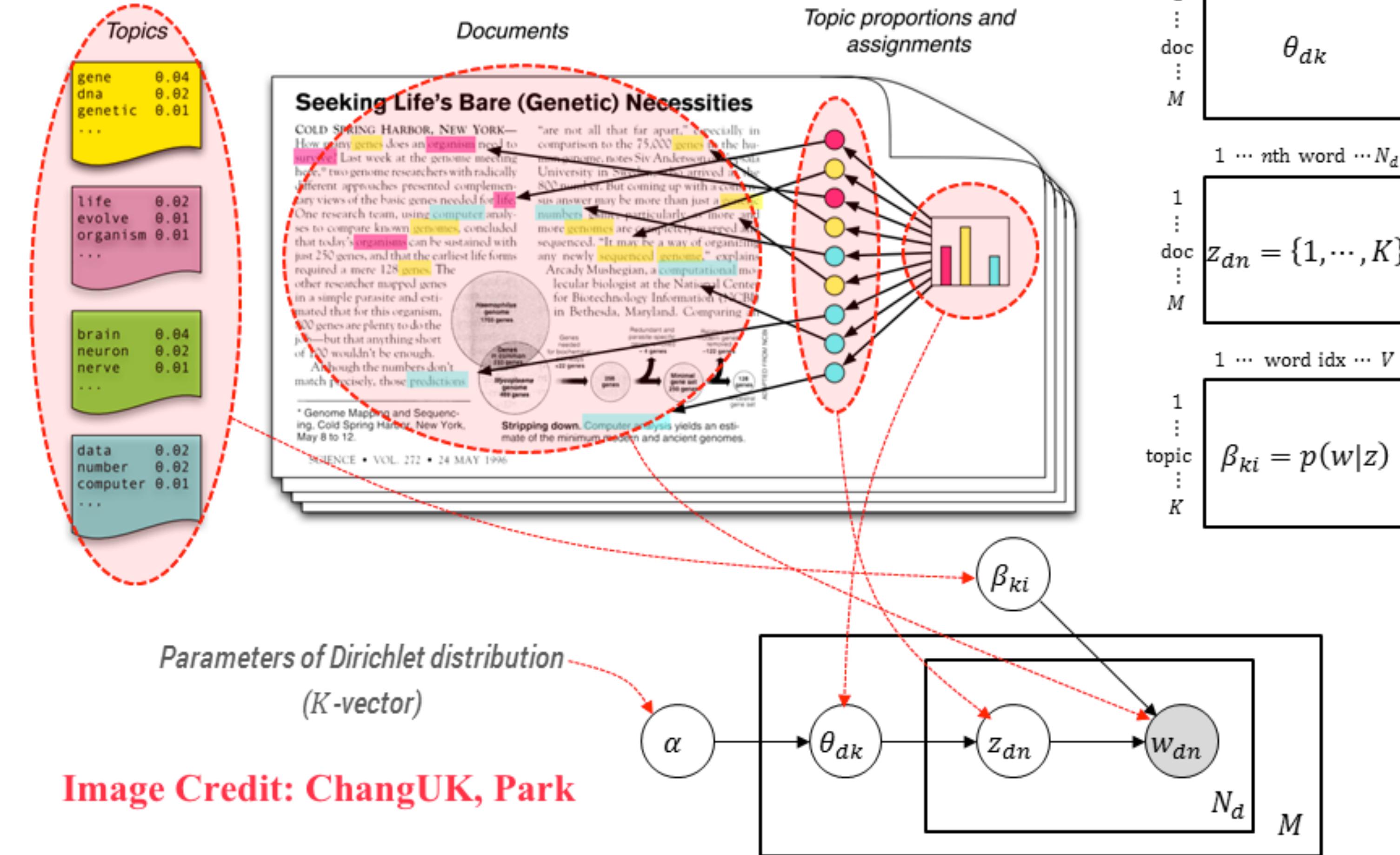


Image Credit: ChangUK, Park

<http://parkcu.com/>

<https://gist.github.com/ChangUk/a741e0ccf5737954956e>

2.4 토픽모델링

.2 라이브러리 gensim, pyLDAvis

gensim <https://radimrehurek.com/gensim/tutorial.html>

```
lda = models.ldamodel.LdaModel(corpus, id2word=dictionary,  
num_topics=ntopics, iterations=200000)  
  
for i in lda.print_topics(num_topics=ntopics, num_words=nwords):  
    print i[0], ':', i[1]
```

0 : 0.002*대통령 + 0.002*朴 + 0.002*원칙 + 0.002*작업 + 0.002*프로
1 : 0.002*운행 + 0.002*오늘 + 0.002*外 + 0.002*백신 + 0.002*파업
2 : 0.002*정치 + 0.002*적발 + 0.002*금융 + 0.002*날씨 + 0.002*추위
3 : 0.003*날씨 + 0.003*추위 + 0.002*크리스마스 + 0.002*철도 + 0.002*파업
4 : 0.003*철도 + 0.003*노조 + 0.002*경찰 + 0.002*오늘 + 0.002*뉴스

2.4 토픽모델링

.2 라이브러리 gensim, pyLDAvis

pyLDAvis <https://github.com/bmabey/pyLDAvis>

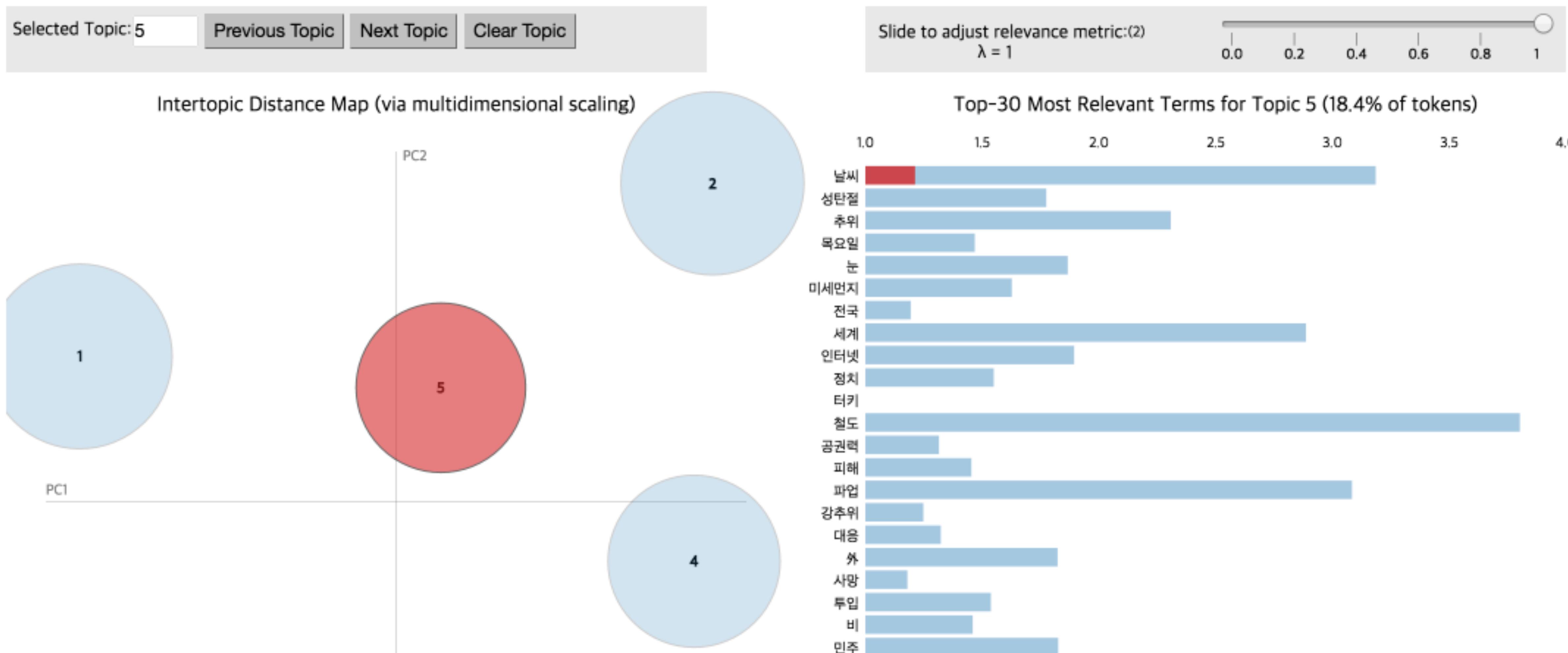
```
import pyLDAvis.gensim  
vis_data = pyLDAvis.gensim.prepare(lda, corpus, dictionary)  
  
pyLDAvis.display(vis_data)
```

jupyter에서 실행!

2.4 토픽모델링

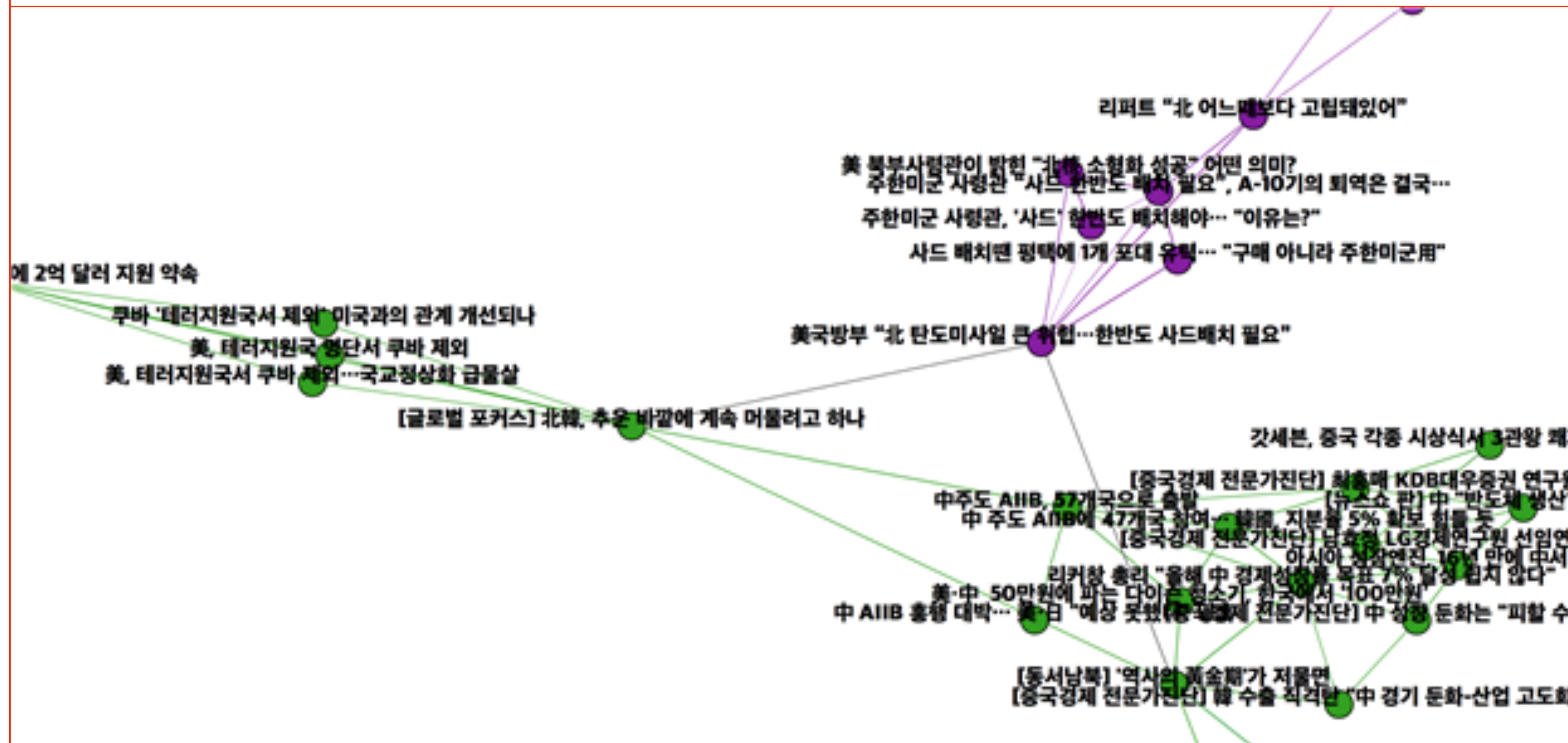
.2 라이브러리 gensim, pyLDAvis

pyLDAvis <https://github.com/bmabey/pyLDAvis>



2.5 네트워크 만들기

.1 뉴스 네트워크



2.5 네트워크 만들기

.2 라이브러리 networkx

PyCon KOREA 2014 - NetworkX를 이용한 네트워크 분석

<https://www.pycon.kr/2014/program/7>

PyCon KOREA 2015 - NetworkX를 이용한 네트워크 링크 예측

<https://www.pycon.kr/2015/program/35>

2.5 네트워크 만들기

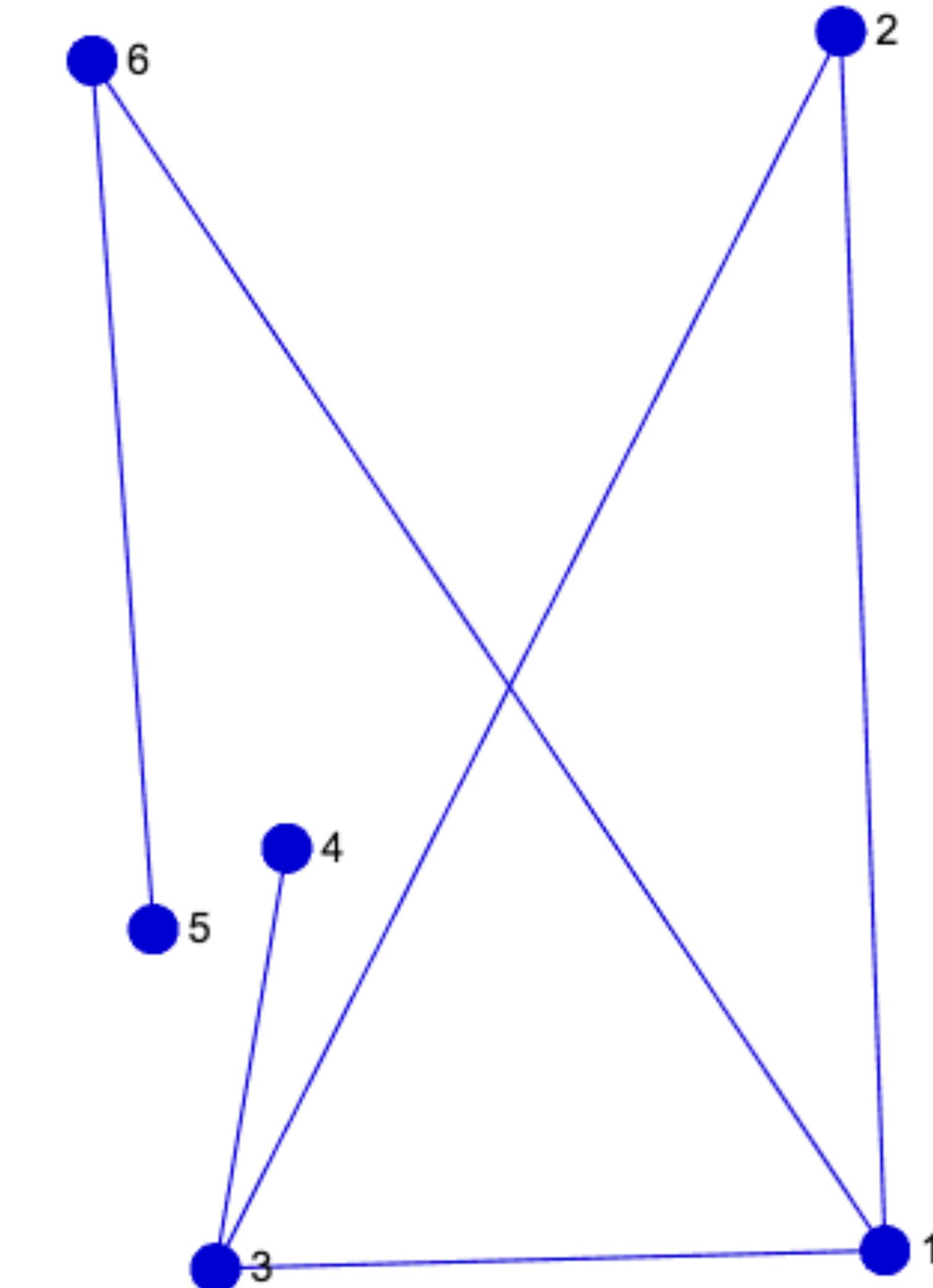
.3 라이브러리 pyneviz

pyneviz <https://github.com/koorukuroo/pyneviz>

데모 : <http://goo.gl/7U2lV>

```
import networkx as nx
import pyneviz.sigmaj as nvs

G = nx.Graph()
G.add_edges_from([(1,2),(2,3),(3,4),(5,6)])
G.add_edge(1, 3)
G.add_edge(1, 6)
nvs.make_gexf(G)
nvs.make_html(drawEdges='true')
nvs.view_html()
```



2.분석

참고자료

PyCon KOREA 2015 - 추천시스템이 word2vec을 만났을때

<https://www.pycon.kr/2015/program/58>

PyCon KOREA 2015 - 오늘 당장 딥러닝 실험하기

<https://www.pycon.kr/2015/program/48>

PyCon KOREA 2015 - 탐색적으로 큰 데이터 분석하기

<https://www.pycon.kr/2015/program/41>

PyCon KOREA 2014 - SociaLite: 빅 데이터 분석을 위한 파이썬 통합 쿼리 언어

<https://www.pycon.kr/2014/program/6>

PyCon KOREA 2014 - Big data with 0% java

<https://www.pycon.kr/2014/program/11>

2. 뉴스를 재미있게 만드는

분석

2.1 데이터의 전처리

- 2.1.1 형태소 분석기
- 2.1.2 라이브러리 Konlpy, umorpheme
- 2.1.3 docker를 활용한 휴대용 형태소 분석
- 2.1.4 라이브러리 NLTK, textblob

2.2 뉴스를 분류하기

- 2.2.1 LSI (Latent Semantic Indexing)
- 2.2.2 행렬의 계산
- 2.2.3 라이브러리 gensim

2.3 뉴스를 군집하기

- 2.3.1 NMF (Non-negative Matrix Factorization)
- 2.3.2 라이브러리 nimfa

2.4 토픽모델링

- 2.4.1 LDA (Latent Dirichlet Allocation)
- 2.4.2 라이브러리 gensim, lda

2.5 네트워크 만들기

- 2.5.1 뉴스 네트워크
- 2.5.2 라이브러리 networkx
- 2.5.3 라이브러리 pyneviz

3. 뉴스를 재미있게 이용하는

전달

3.1 웹으로 데이터 보여주기

3.1.1 라이브러리 Flask

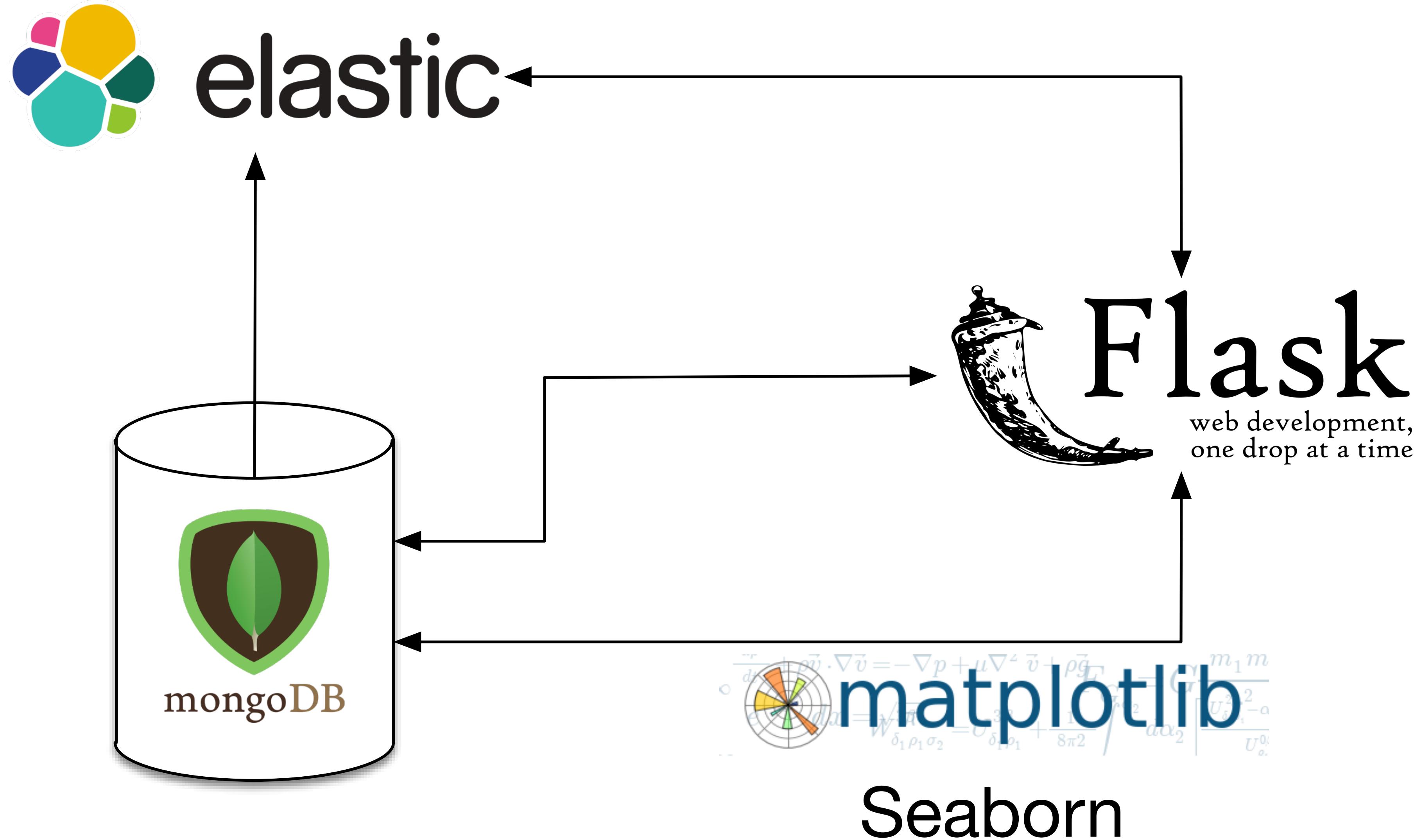
3.2 시각화하기

3.2.1 라이브러리 seaborn

3.3 검색하기

3.3.1 라이브러리 elasticsearch-py

3.1 웹으로 데이터 보여주기



1. 뉴스를 재미있게 만들기 위한

수집

1.1 데이터 수집하기

1.1.0 데이터 소스

1.1.1 라이브러리 urllib2, request, mechanize

1.1.2 라이브러리 chardet, unidecode

1.1.3 라이브러리 pyspider, feedparser

1.1.4 라이브러리 robobrowser

1.2 데이터 파싱하기

1.2.1 라이브러리 beautifulsoup, lxml, pyparsing

1.2.2 라이브러리 python-goose

1.2.3 라이브러리 newspaper

1.3 데이터 저장하기

1.3.1 라이브러리 PyMongo

1.3.2 Cloud Service (mongolab, aws, azure)

2. 뉴스를 재미있게 만드는

분석

2.1 데이터의 전처리

2.1.1 형태소 분석기

2.1.2 라이브러리 Konlpy, umorpheme

2.1.3 docker를 활용한 휴대용 형태소 분석

2.1.4 라이브러리 NLTK, textblob

2.2 뉴스를 분류하기

2.2.1 LSI (Latent Semantic Indexing)

2.2.2 행렬의 계산

2.2.3 라이브러리 gensim

2.3 뉴스를 군집하기

2.3.1 NMF (Non-negative Matrix Factorization)

2.3.2 라이브러리 nimfa

2.4 토픽모델링

2.4.1 LDA (Latent Dirichlet Allocation)

2.4.2 라이브러리 gensim, lda

2.5 네트워크 만들기

2.5.1 뉴스 네트워크

2.5.2 라이브러리 networkx

2.5.3 라이브러리 pyneviz

3. 뉴스를 재미있게 이용하는

전달

3.1 웹으로 데이터 보여주기

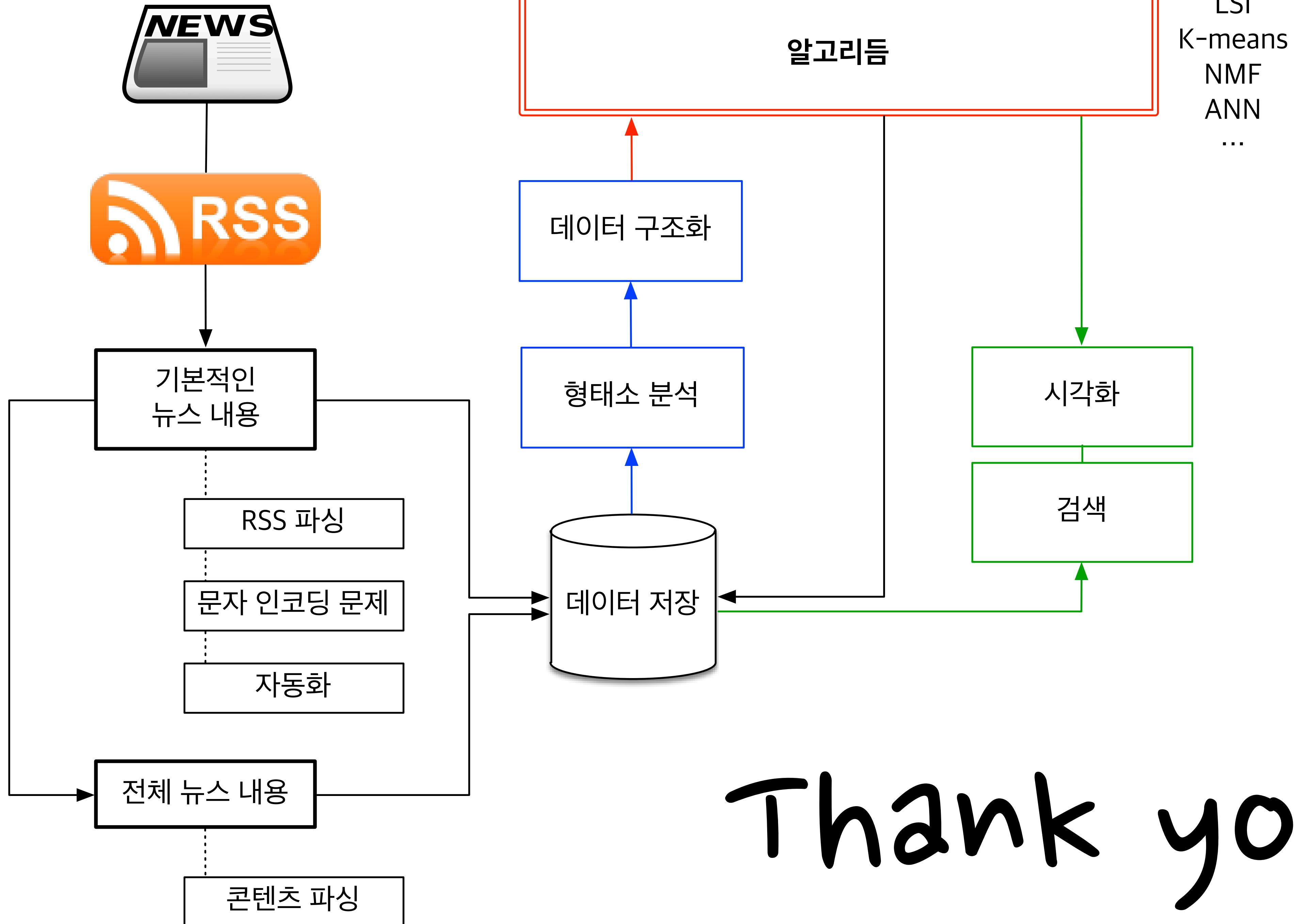
3.1.1 라이브러리 Flask

3.2 시각화하기

3.2.1 라이브러리 seaborn

3.3 검색하기

3.3.1 라이브러리 elasticsearch-py



Thank you!

LSI
K-means
NMF
ANN
...