예비프로젝트 #6 : Pubmed 논문지 저널별 분석

1829008 김민영

목차

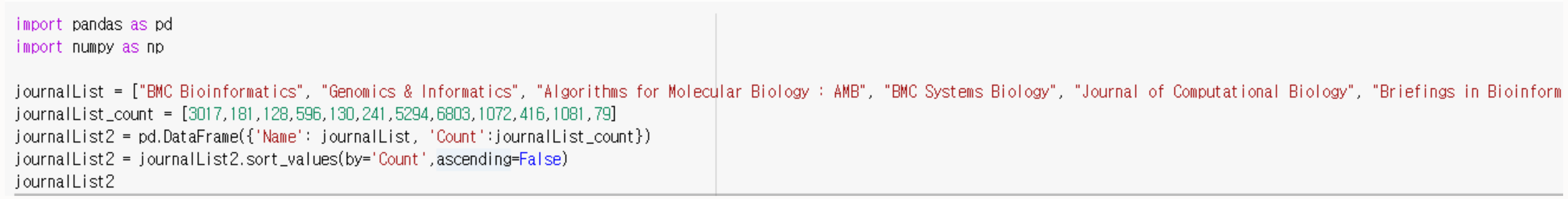
1. File Generator

- 2015~2019 Journal 별 논문 수 확인
- 2015~2019 5년간 Journal 별 논문 추출

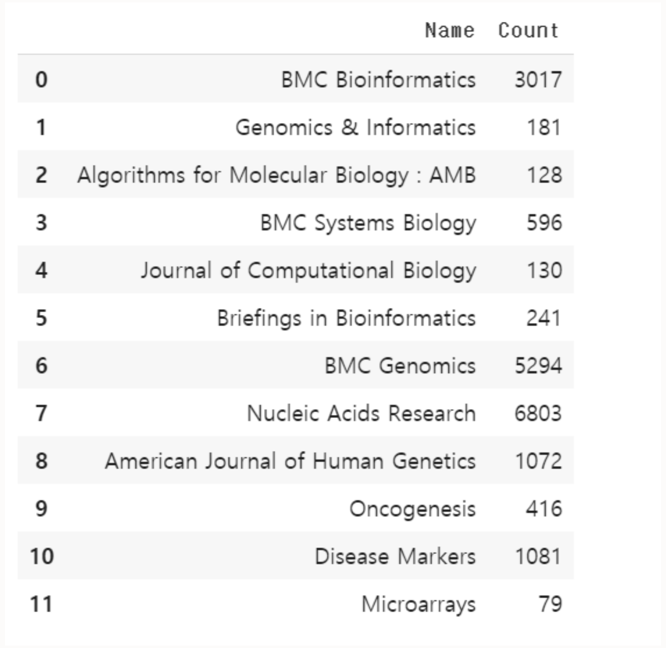
2015~2019 Journal 별 논문 수

[“BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefings in Bioinformatics", "BMC Genomics", "Nucleic Acids Research", "American Journal of Human Genetics", "Oncogenesis", "Disease Markers", "Microarrays“] 저널에 대해,

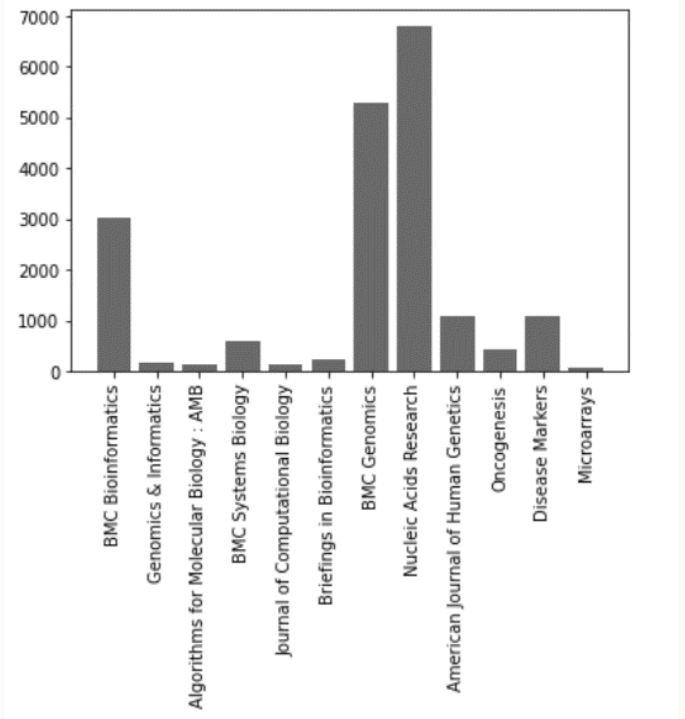
최근 5년 간의 12개의 Journal 별 발행 논문수를 살펴보면 다음과 같다.



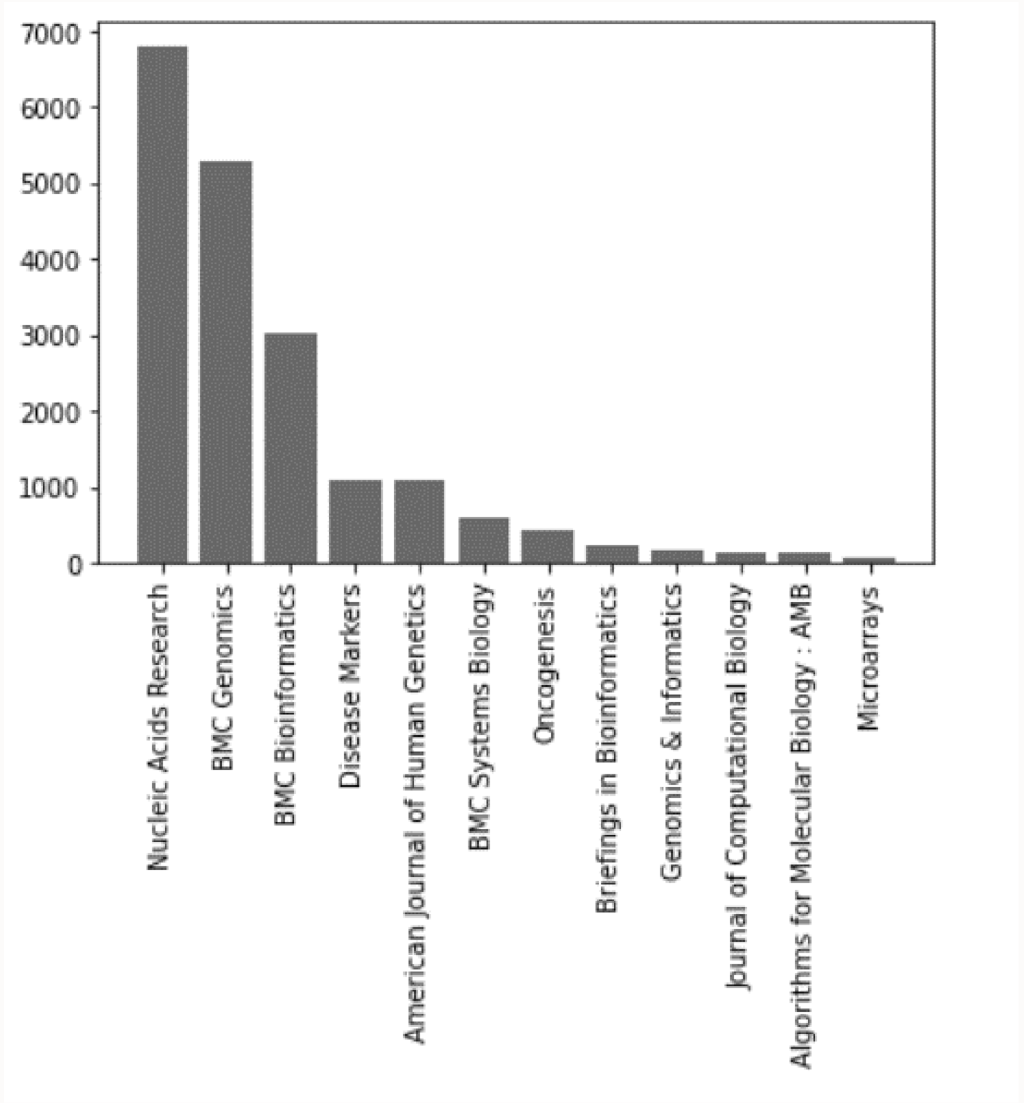
```
import pandas as pd
import numpy as np

journalList = ["BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefings in Bioinform
journalList_count = [3017,181,128,596,130,241,5294,6803,1072,416,1081,79]
journalList2 = pd.DataFrame({'Name': journalList, 'Count':journalList_count})
journalList2 = journalList2.sort_values(by='Count',ascending=False)
journalList2
```



	Name	Count
0	BMC Bioinformatics	3017
1	Genomics & Informatics	181
2	Algorithms for Molecular Biology : AMB	128
3	BMC Systems Biology	596
4	Journal of Computational Biology	130
5	Briefings in Bioinformatics	241
6	BMC Genomics	5294
7	Nucleic Acids Research	6803
8	American Journal of Human Genetics	1072
9	Oncogenesis	416
10	Disease Markers	1081
11	Microarrays	79



2015~2019 Journal 별 논문 수

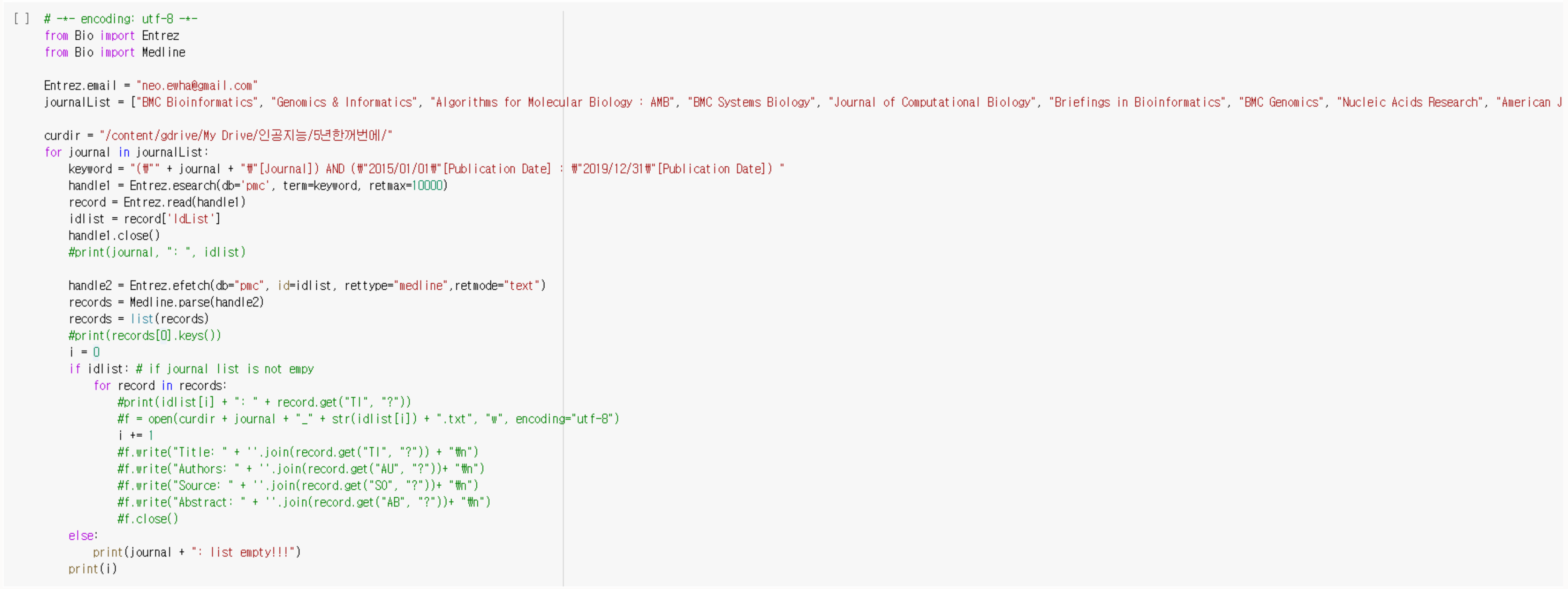


2015~2019년 5년간 저널별 논문 발행 수는 Nucleic Acids Research가 약 6000여개, BMC Genomics가 약 5000여개, BMC Bioinformatics 약 3000여개로 높았다.

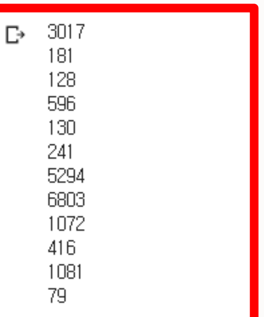
논문의 개수가 너무 많아 파일 관리하기가 어려우므로, 2015,2016,2017,2018,2019년 각각 abstract 만 따로 떼어내어 파일을 저장하였다.

2015~2019 Journal 별 논문 수 - Code

[2015~2019 5년간]
12개의 Journal List별로 파일을 생성하기 전, 5년간 발행된 논문의 개수를 확인하기위한 코드이다.



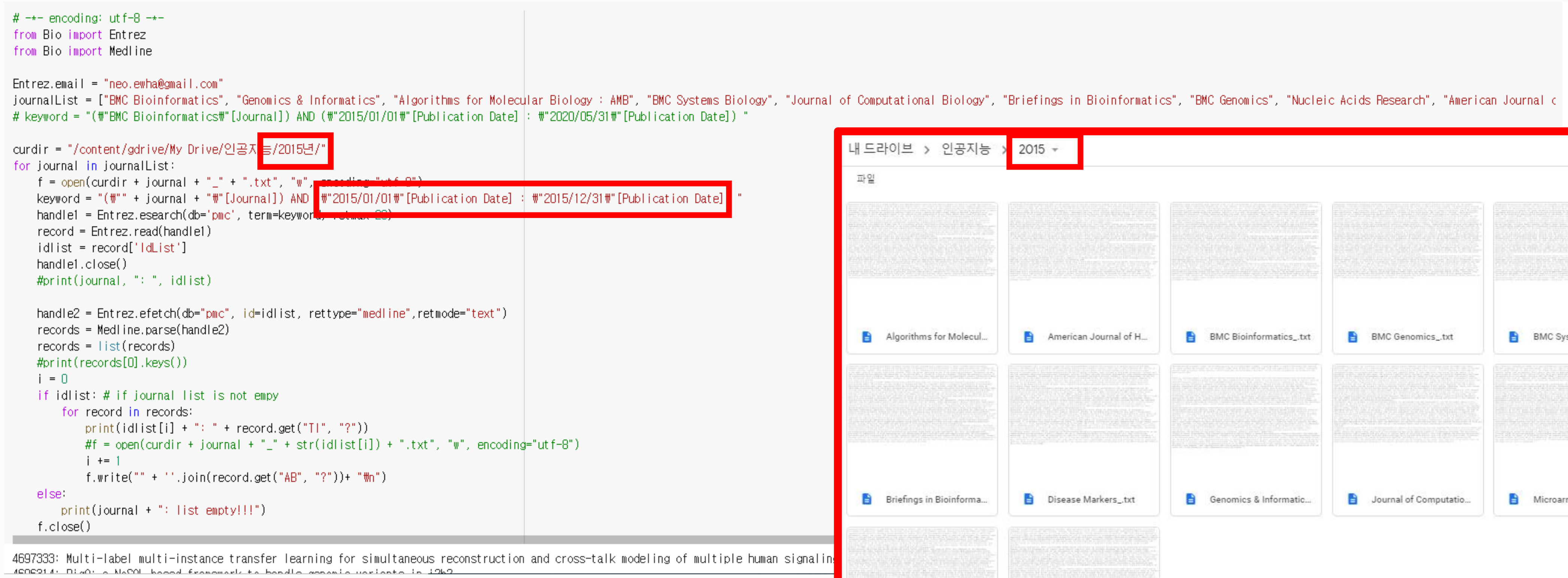
```
[ ] # -*- encoding: utf-8 -*-
    from Bio import Entrez
    from Bio import Medline

    Entrez.email = "neo.ewha@gmail.com"
    journalList = ["BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefings in Bioinformatics", "BMC Genomics", "Nucleic Acids Research", "American J

    curdir = "/content/gdrive/My Drive/인공지능/5년한꺼번에/"
    for journal in journalList:
        keyword = "(₩"" + journal + "₩"[Journal]) AND (₩"2015/01/01₩"[Publication Date] : ₩"2019/12/31₩"[Publication Date]) "
        handle1 = Entrez.esearch(db='pmc', term=keyword, retmax=10000)
        record = Entrez.read(handle1)
        idlist = record['IdList']
        handle1.close()
        #print(journal, ": ", idlist)

        handle2 = Entrez.efetch(db="pmc", id=idlist, rettype="medline",retmode="text")
        records = Medline.parse(handle2)
        records = list(records)
        #print(records[0].keys())
        i = 0
        if idlist: # if journal list is not empy
            for record in records:
                #print(idlist[i] + ": " + record.get("TI", "?"))
                #f = open(curdir + journal + "_" + str(idlist[i]) + ".txt", "w", encoding="utf-8")
                i += 1
                #f.write("Title: " + ''.join(record.get("TI", "?")) + "₩n")
                #f.write("Authors: " + ''.join(record.get("AU", "?"))+ "₩n")
                #f.write("Source: " + ''.join(record.get("SO", "?"))+ "₩n")
                #f.write("Abstract: " + ''.join(record.get("AB", "?"))+ "₩n")
                #f.close()
        else:
            print(journal + ": list empty!!!")
        print(i)
```



```
3017
181
128
596
130
241
5294
6803
1072
416
1081
79
```

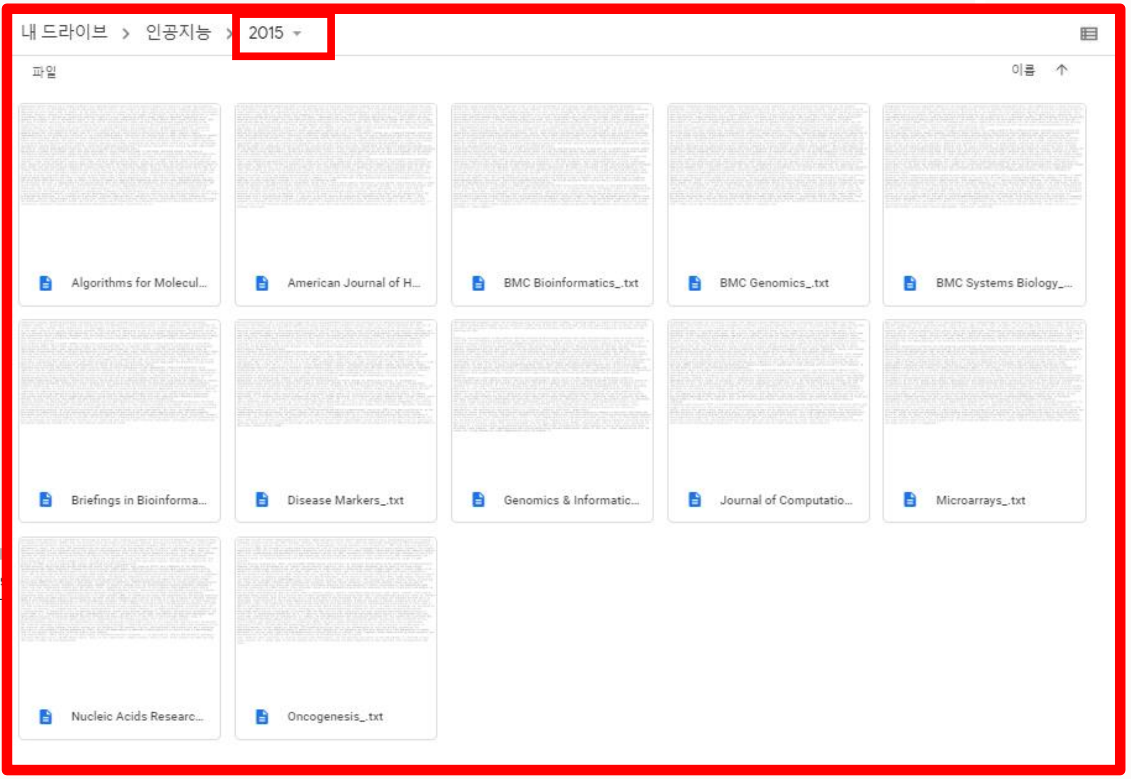
File Generator with Bio Python

[2015년]
12개의 Journal List별로 논문의 abstract만을 추출하여 파일 생성



```
# -*- encoding: utf-8 -*-
from Bio import Entrez
from Bio import Medline

Entrez.email = "neo.ewha@gmail.com"
journalList = ["BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefings in Bioinformatics", "BMC Genomics", "Nucleic Acids Research", "American Journal c
# keyword = "(₩"BMC Bioinformatics₩"[Journal]) AND (₩"2015/01/01₩"[Publication Date] : ₩"2020/05/31₩"[Publication Date]) "

curdir = "/content/gdrive/My Drive/인공지능/2015년/"
for journal in journalList:
    f = open(curdir + journal + "_" + ".txt", "w", encoding="utf-8")
    keyword = "(₩"" + journal + "₩"[Journal]) AND (₩"2015/01/01₩"[Publication Date] : ₩"2015/12/31₩"[Publication Date]) "
    handle1 = Entrez.esearch(db='pmc', term=keyword, retmax=20)
    record = Entrez.read(handle1)
    idlist = record['IdList']
    handle1.close()
    #print(journal, ": ", idlist)

    handle2 = Entrez.efetch(db="pmc", id=idlist, rettype="medline",retmode="text")
    records = Medline.parse(handle2)
    records = list(records)
    #print(records[0].keys())
    i = 0
    if idlist: # if journal list is not empy
        for record in records:
            print(idlist[i] + ": " + record.get("TI", "?"))
            #f = open(curdir + journal + "_" + str(idlist[i]) + ".txt", "w", encoding="utf-8")
            i += 1
            f.write("" + ''.join(record.get("AB", "?"))+ "₩n")
    else:
        print(journal + ": list empty!!!")
    f.close()
```

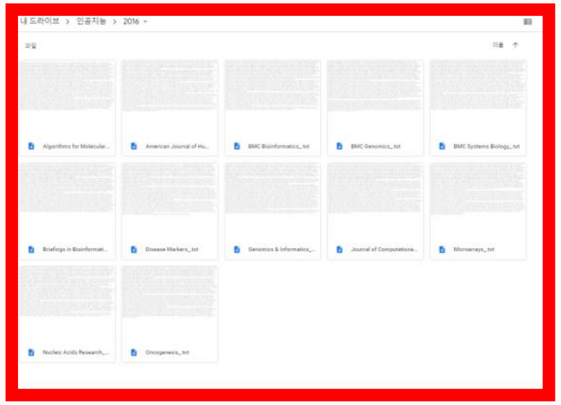
```
4697333: Multi-label multi-instance transfer learning for simultaneous reconstruction and cross-talk modeling of multiple human signalin
4696314: BigQ: a NoSQL based framework to handle genomic variants in i2b2
```

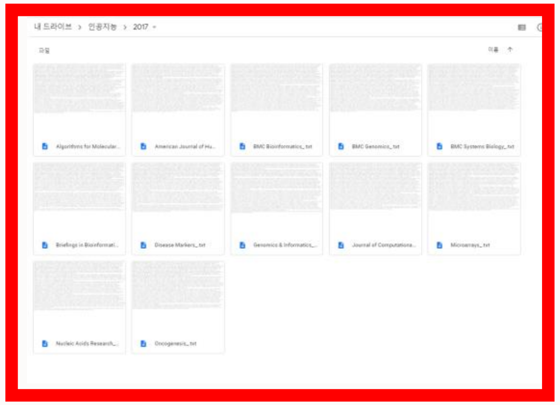


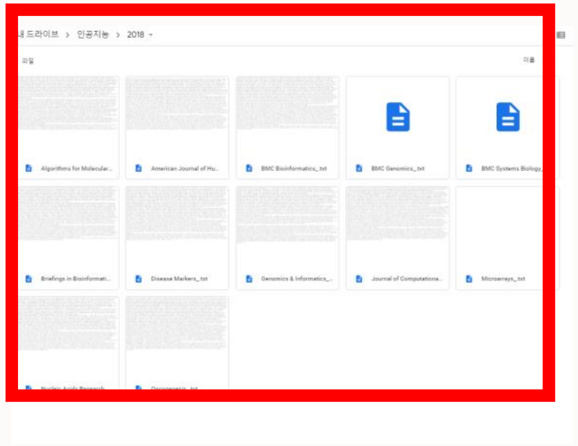
Algorithms for Molecul...
American Journal of H...
BMC Bioinformatics_.txt
BMC Genomics_.txt
BMC Systems Biology_...
Briefings in Bioinforma...
Disease Markers_.txt
Genomics & Informatic...
Journal of Computatio...
Microarrays_.txt
Nucleic Acids Researc...
Oncogenesis_.txt

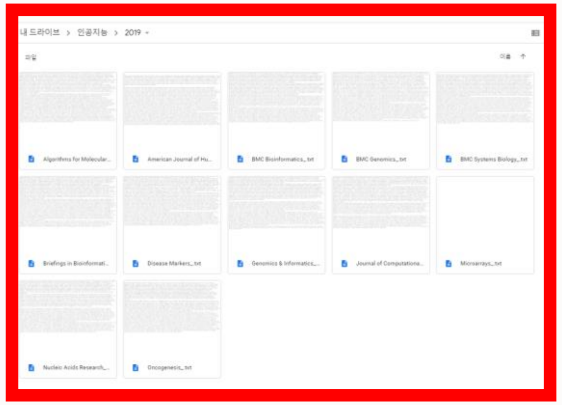
File Generator with Bio Python

[2016년]/[2017년]/[2018년]/[2019년] 도 마찬가지로
12개의 Journal List별로 논문의 abstract만을 추출하여 파일 생성


내 드라이브 › 인공지능 › 2016
파일
이름
Algorithms for Molecular...
American Journal of Hu...
BMC Bioinformatics_.txt
BMC Genomics_.txt
BMC Systems Biology_.txt
Briefings in Bioinformati...
Disease Markers_.txt
Genomics & Informatics_...
Journal of Computationa...
Microarrays_.txt
Nucleic Acids Research_...
Oncogenesis_.txt


내 드라이브 › 인공지능 › 2017
파일
이름
Algorithms for Molecular...
American Journal of Hu...
BMC Bioinformatics_.txt
BMC Genomics_.txt
BMC Systems Biology_.txt
Briefings in Bioinformati...
Disease Markers_.txt
Genomics & Informatics_...
Journal of Computationa...
Microarrays_.txt
Nucleic Acids Research_...
Oncogenesis_.txt


내 드라이브 › 인공지능 › 2018
파일
이름
Algorithms for Molecular...
American Journal of Hu...
BMC Bioinformatics_.txt
BMC Genomics_.txt
BMC Systems Biology
Briefings in Bioinformati...
Disease Markers_.txt
Genomics & Informatics_...
Journal of Computationa...
Microarrays_.txt

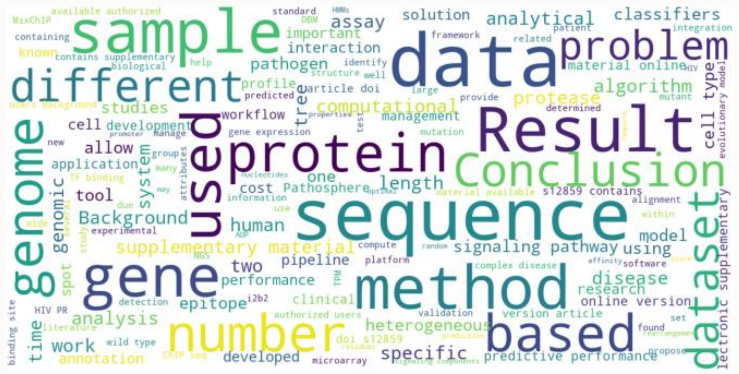

내 드라이브 › 인공지능 › 2019
파일
이름
Algorithms for Molecular...
American Journal of Hu...
BMC Bioinformatics_.txt
BMC Genomics_.txt
BMC Systems Biology_.txt
Briefings in Bioinformati...
Disease Markers_.txt
Genomics & Informatics_...
Journal of Computationa...
Microarrays_.txt
Nucleic Acids Research_...
Oncogenesis_.txt

2. Analysis

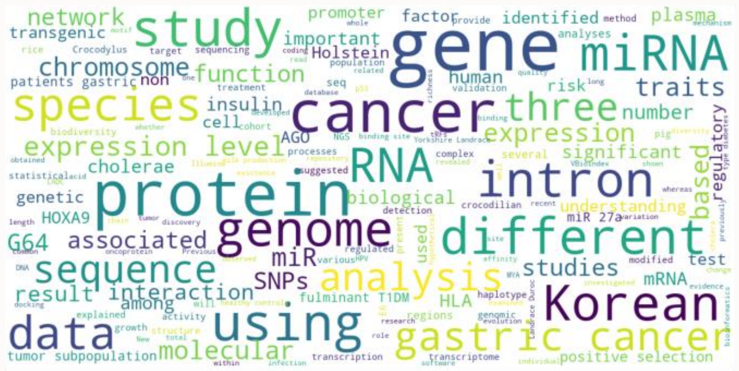
- wordcloud
- keyword 분포
- GNI와 유사한 논문지 찾기

+ 12개의 저널별 논문 accepted 기간 분석

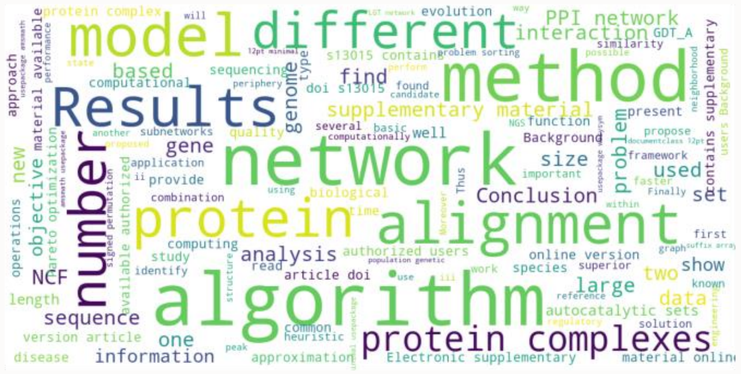
저널별 워드클라우드



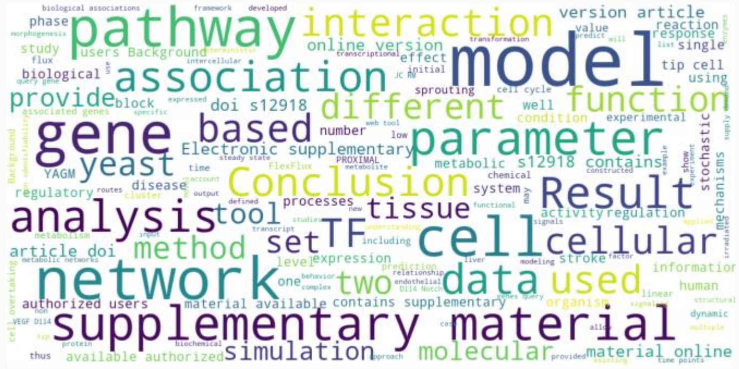
BMC Bioinformatics



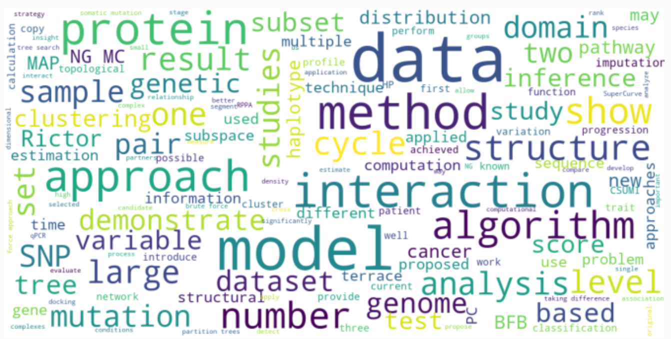
Genomics & Informatics



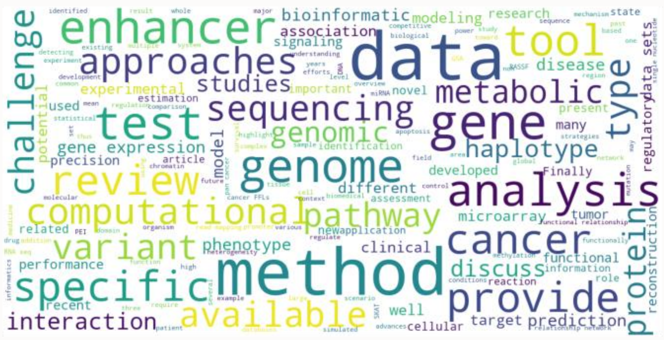
Algorithms for Molecular Biology : AMB



BMC Systems Biology

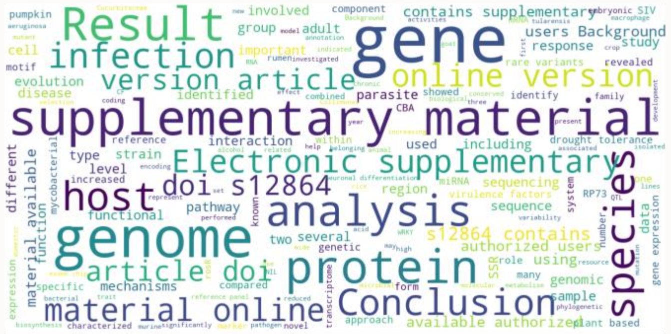


journal of Computational Biology

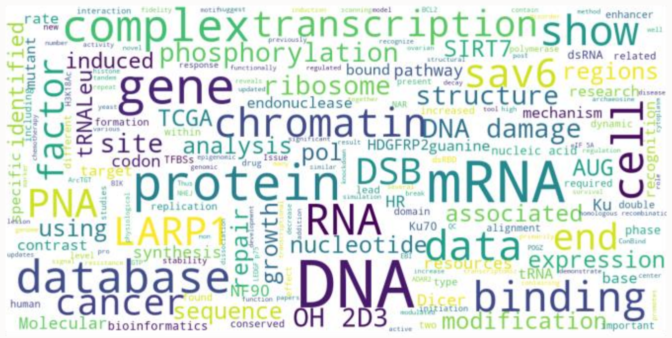


Briefings in Bioinformatics

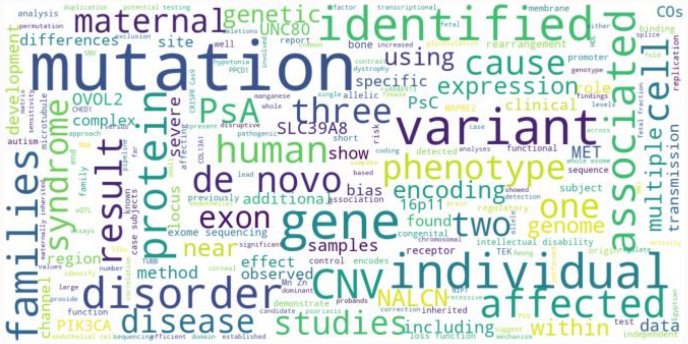
저널별 워드클라우드



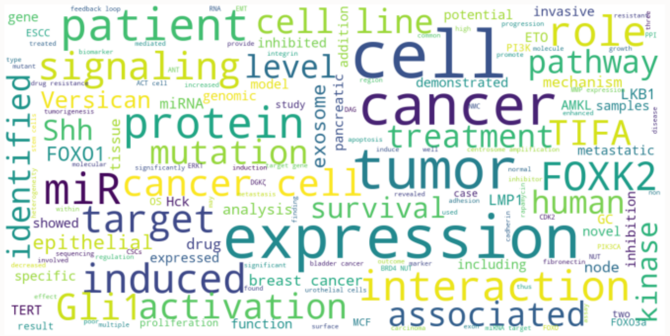
BMC Genomics



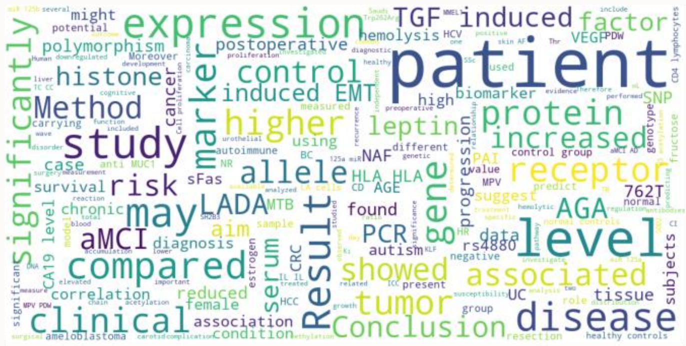
Nucleic Acids Research



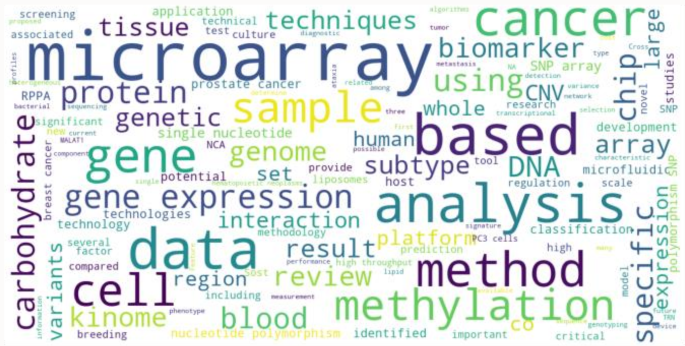
American Journal of Human Genetics



Oncogenesis

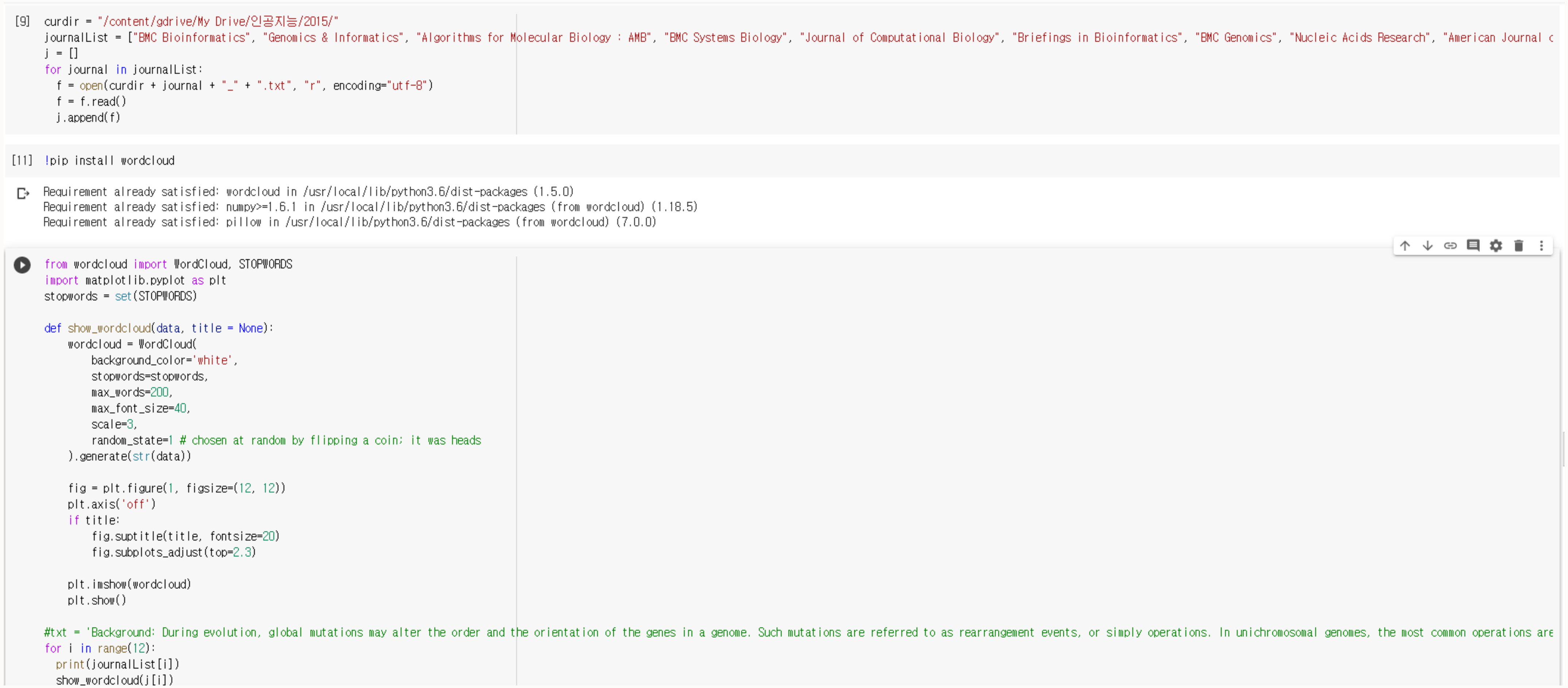


Disease Markers



Microarrays

저널별 워드클라우드 - Code



```
[9]  curdir = "/content/gdrive/My Drive/인공지능/2015/"
     journalList = ["BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefings in Bioinformatics", "BMC Genomics", "Nucleic Acids Research", "American Journal c
     j = []
     for journal in journalList:
       f = open(curdir + journal + "_" + ".txt", "r", encoding="utf-8")
       f = f.read()
       j.append(f)
```

```
[11] !pip install wordcloud
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.6/dist-packages (1.5.0)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.6/dist-packages (from wordcloud) (1.18.5)
Requirement already satisfied: pillow in /usr/local/lib/python3.6/dist-packages (from wordcloud) (7.0.0)
```

```
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
stopwords = set(STOPWORDS)

def show_wordcloud(data, title = None):
    wordcloud = WordCloud(
        background_color='white',
        stopwords=stopwords,
        max_words=200,
        max_font_size=40,
        scale=3,
        random_state=1 # chosen at random by flipping a coin; it was heads
    ).generate(str(data))

    fig = plt.figure(1, figsize=(12, 12))
    plt.axis('off')
    if title:
        fig.suptitle(title, fontsize=20)
        fig.subplots_adjust(top=2.3)

    plt.imshow(wordcloud)
    plt.show()

#txt = 'Background: During evolution, global mutations may alter the order and the orientation of the genes in a genome. Such mutations are referred to as rearrangement events, or simply operations. In unichromosomal genomes, the most common operations are
for i in range(12):
  print(journalList[i])
  show_wordcloud(j[i])
```

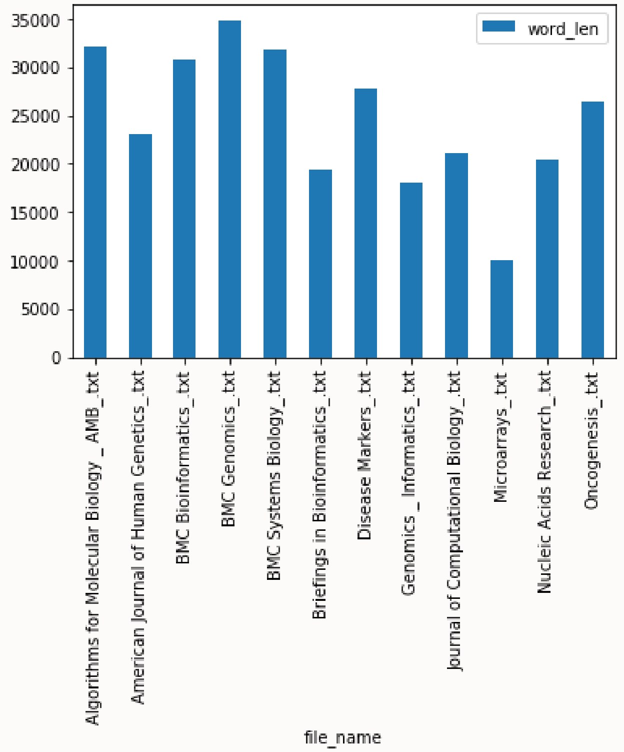
저널별 단어분포

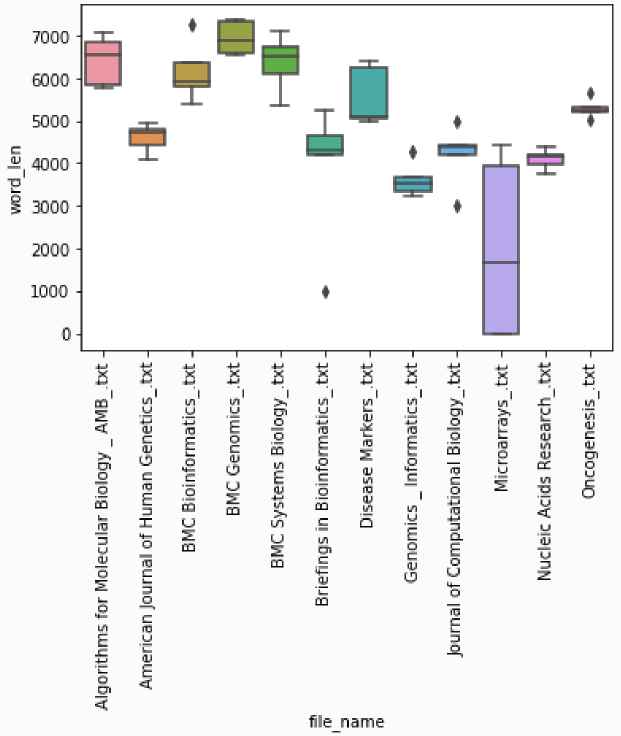
앞에서 살펴본 워드클라우드를 통해, 자주 등장하는 단어를 정리해보면 다음과 같다.

BMC Bioinformatics -	data, sequence, protein, genome
Genomics & Informatics -	algorithm, network, method, alignment
Algorithms for Molecular Biology : AMB -	protein, cancer, genome, gene, RNA, Korean
BMC Systems Biology -	model, gene, network, material, analysis, pathway, association
Journal of Computational Biology -	data, method, model, algorithm, approach, interaction, structure
Briefings in Bioinformatics -	data, gene, genome, method, cancer, analysis
BMC Genomics -	gene, genome, analysis, pretein infection
Nucleic Acids Research -	DNA, mRNA, database, gene, data, binding
American Journal of Human Genetics -	mutation, variant, gene, individual, disorder
Oncogenesis -	cell, cancer, expression, tumor
Disease Markers -	patient, study, level, result, disease
Microarrays -	microarray,based, data, analysis, method, cell

저널별 단어분포

단어분포를 살펴보기 전에 우선 저널별 단어의 길이를 살펴보면 다음과 같다.
비슷한 수의 파일을 선택했음에도 불구하고 약간의 차이가 존재하여 이를 고려하여 앞으로 분석을 할 것이다.

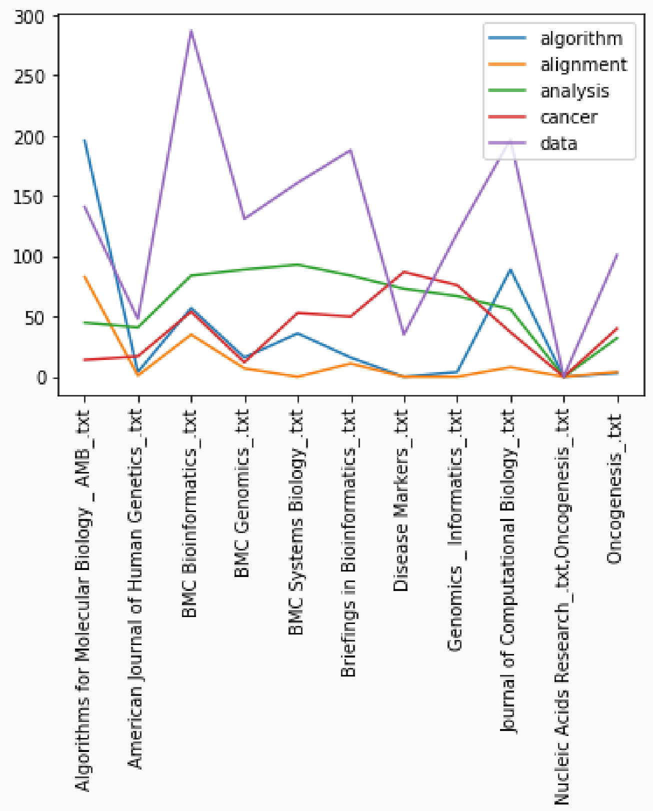


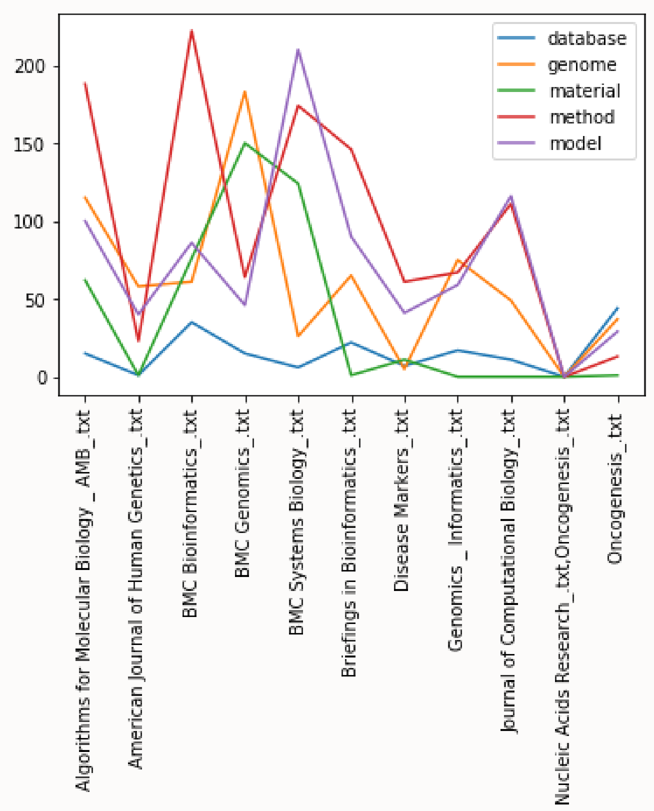


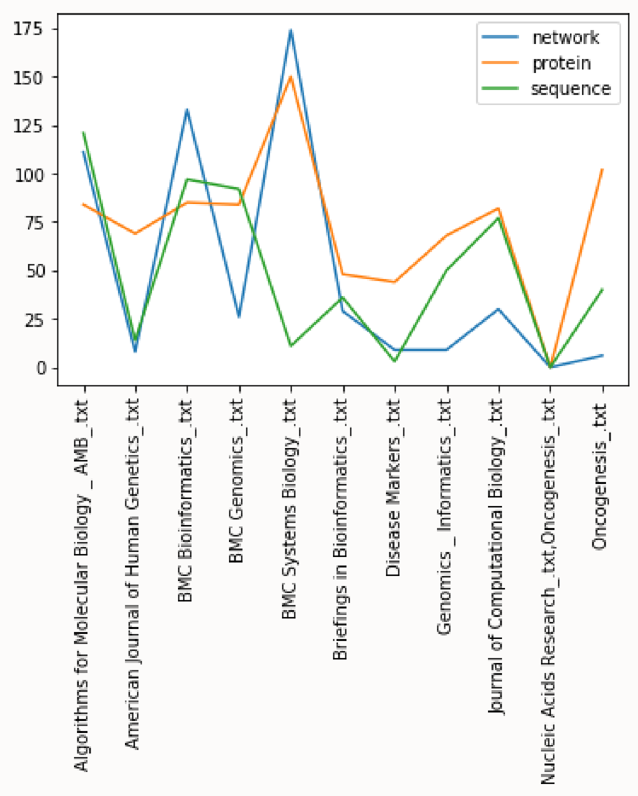
*Microarray 저널은 표본의 수가 너무 부족하기 때문에 왜곡이 될 가능성이 있어 우선 제외하고 분석함.

저널별 단어분포 - Code

앞에서 살펴본 자주 등장하는 단어들에 대해 시각화하여 저널별로 비교를 해보면 다음과 같다.

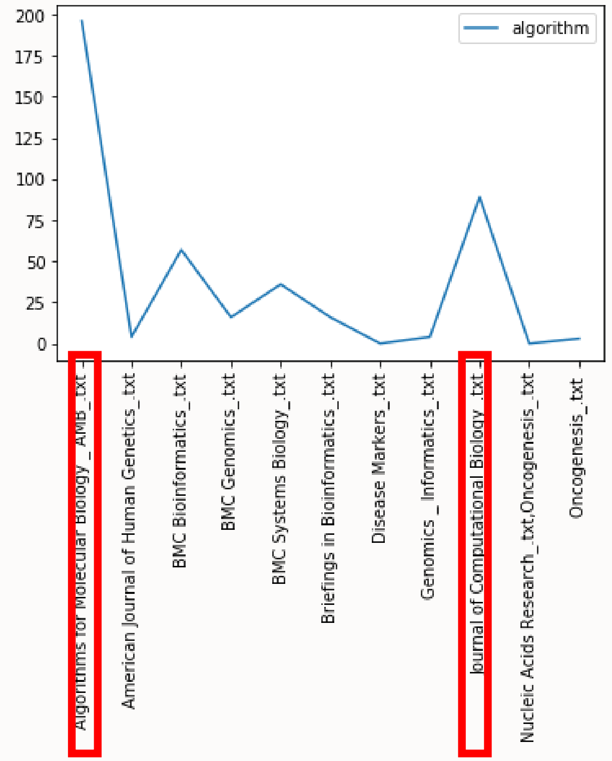


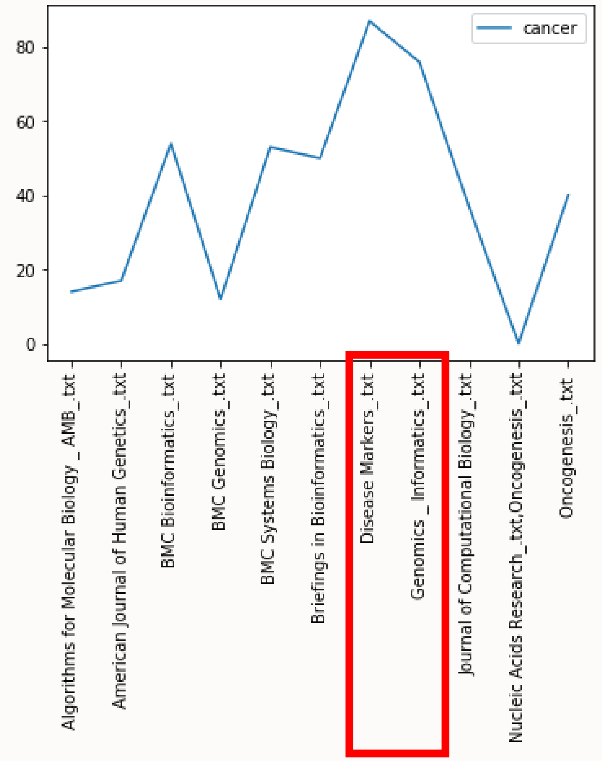




저널별 단어분포 - Code

앞의 그래프를 통해 의미있는 결과를 간단히 정리해보자.



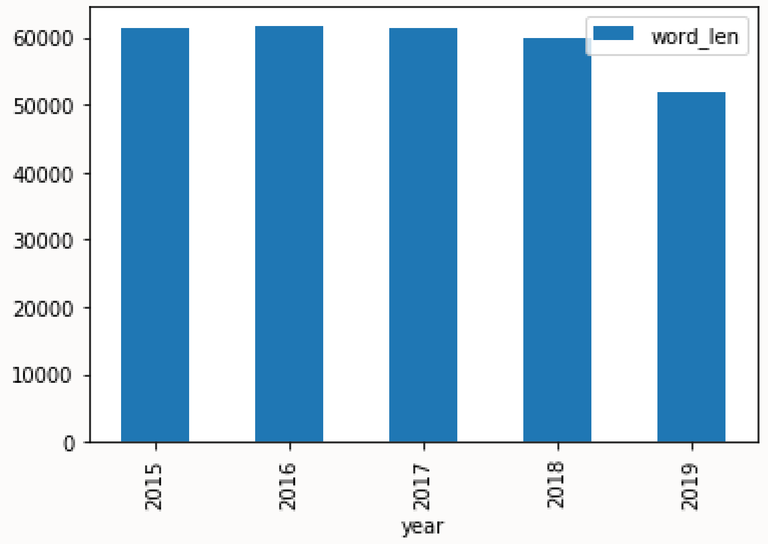


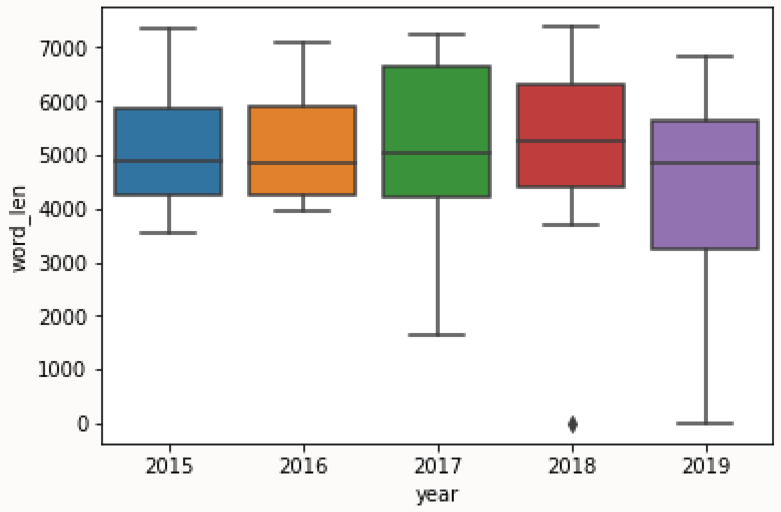
-> algorithm 단어는 Algorithms for Molecular Biology _AMB저널에서
압도적으로 많이 등장하였으며,
Journal of Conputational Biology 저널에서도 꽤 많이 등장하였다.
이는 알고리즘, 컴퓨터학문에 관련한 저널이기 때문이라고 유추할 수 있다.

-> cancer 단어는 Disease Markers 저널, Genomics_Informatics에서
자주 등장하였다.

2015~2019 시간에 따른 단어분포 변화

단어분포를 살펴보기 전에 우선 연도별 단어의 길이를 살펴보았다.

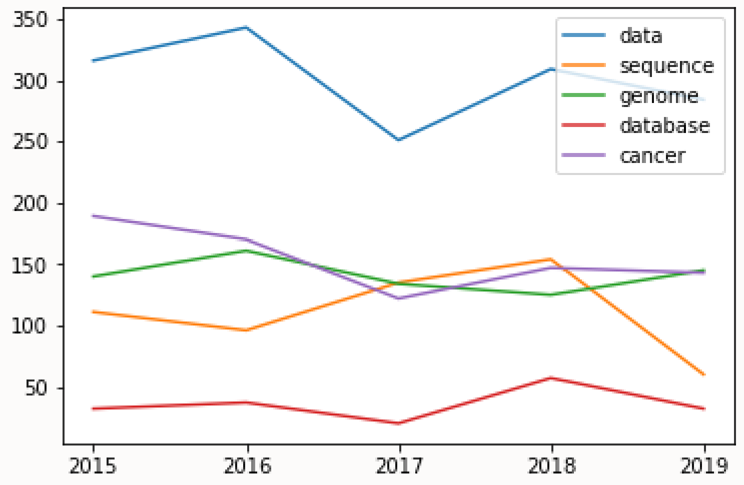


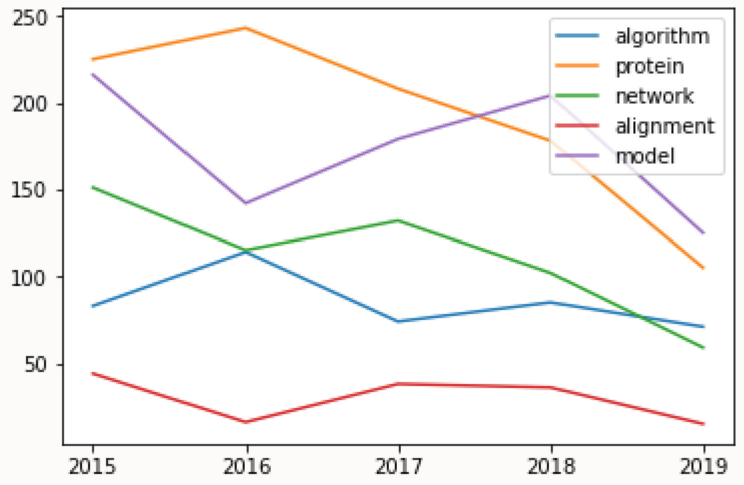


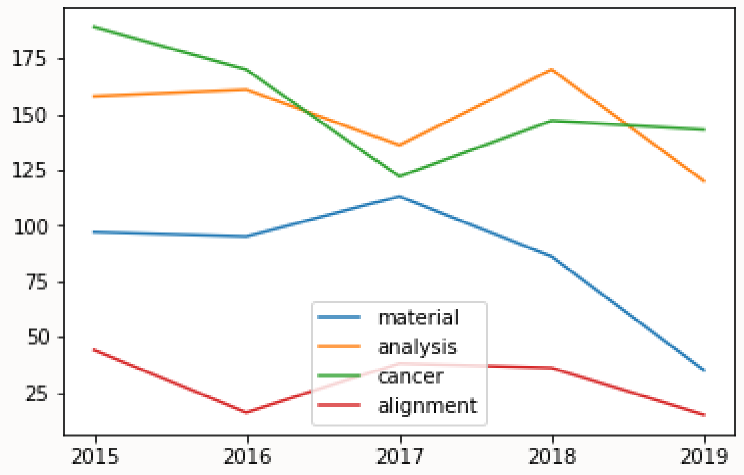
논문의 수를 비슷하게 뽑아왔기 때문에, 데이터 분포에 큰 치우침이 없어보인다.
따라서 이 데이터를 그대로 분석에 이용해도 크게 문제 없을 것 같다.
실제로 총단어갯수를 고려한 그래프를 따로 그려보았는데 기존 그래프와 큰 차이가 없었다.

2015~2019 시간에 따른 단어분포 변화

pubmed data의 2015~2019년 abstact의 단어 분포를 살펴본 결과이다.



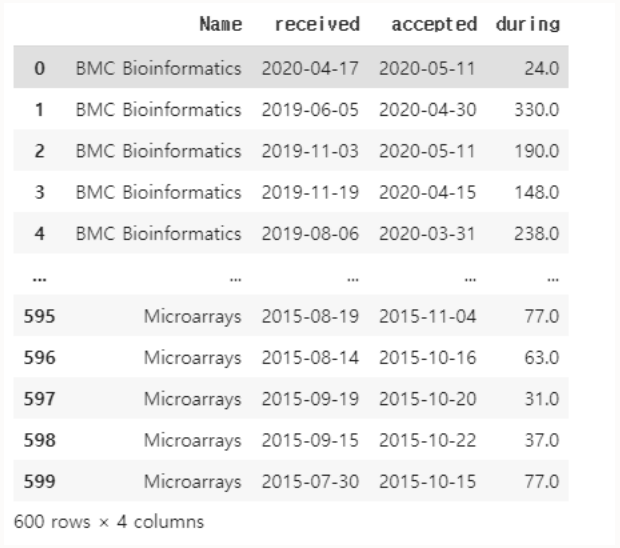




대체적으로 data 단어가 자주 사용되는 것을 확인할 수 있고 model이라는 단어가 2018년에
많이 사용된 것을 확인할 수 있다. 이외에도 각 단어별 시간에 따른 분포 변화를 확인할 수 있다.

+ 12개의 저널별 논문 accepted 기간 분석

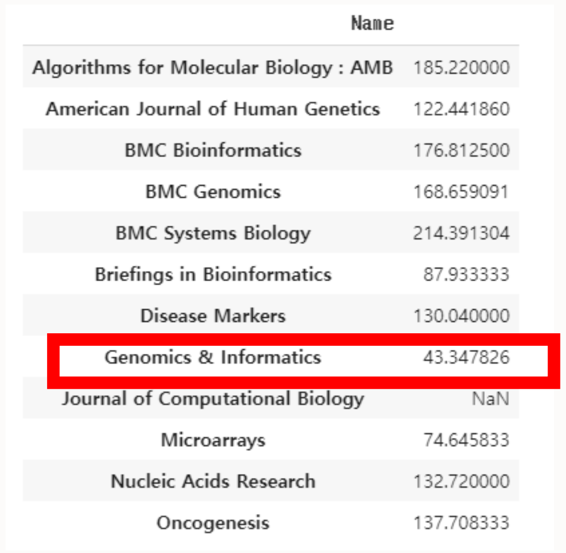
received ~ accepted 기간



	Name	received	accepted	during
0	BMC Bioinformatics	2020-04-17	2020-05-11	24.0
1	BMC Bioinformatics	2019-06-05	2020-04-30	330.0
2	BMC Bioinformatics	2019-11-03	2020-05-11	190.0
3	BMC Bioinformatics	2019-11-19	2020-04-15	148.0
4	BMC Bioinformatics	2019-08-06	2020-03-31	238.0
...	...	...	...	...
595	Microarrays	2015-08-19	2015-11-04	77.0
596	Microarrays	2015-08-14	2015-10-16	63.0
597	Microarrays	2015-09-19	2015-10-20	31.0
598	Microarrays	2015-09-15	2015-10-22	37.0
599	Microarrays	2015-07-30	2015-10-15	77.0

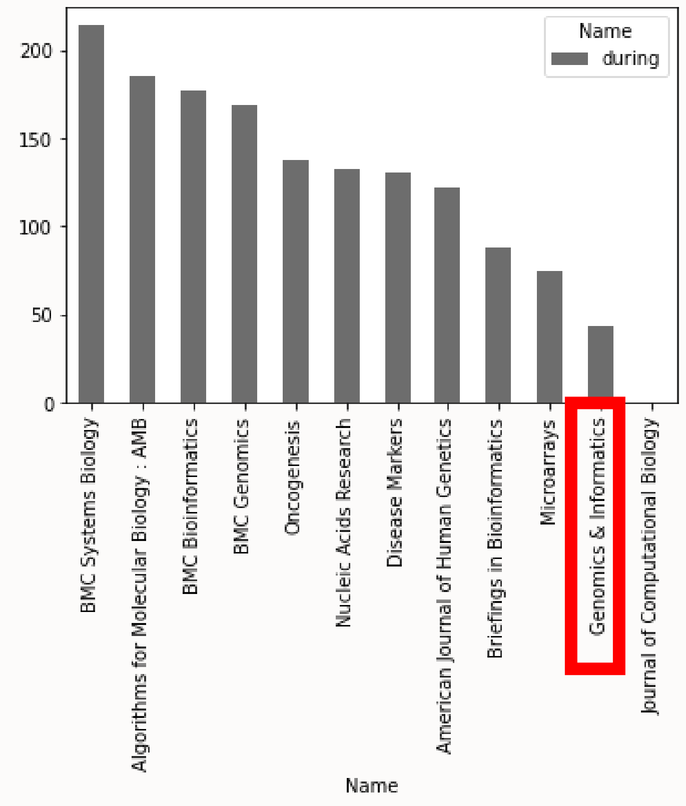
600 rows × 4 columns

저널별 received ~ accepted 기간 평균



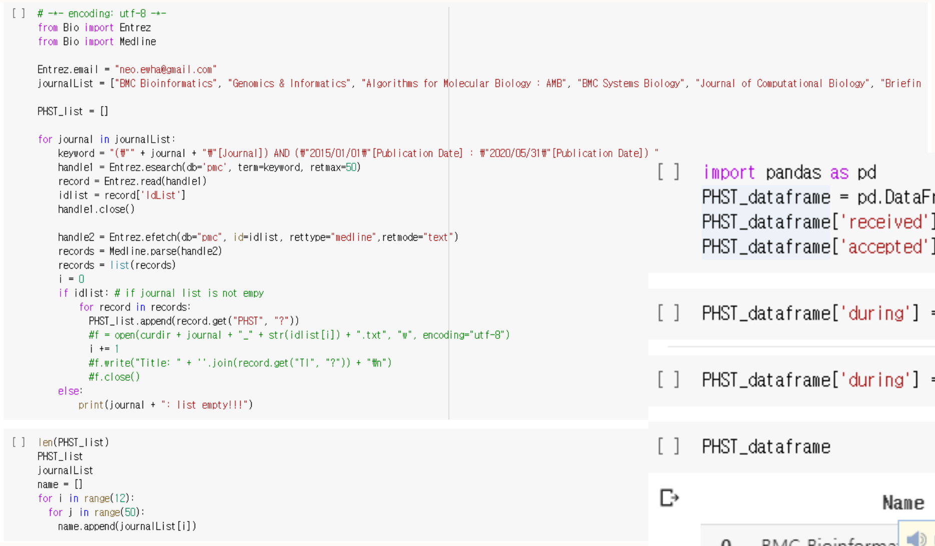
Name	
Algorithms for Molecular Biology : AMB	185.220000
American Journal of Human Genetics	122.441860
BMC Bioinformatics	176.812500
BMC Genomics	168.659091
BMC Systems Biology	214.391304
Briefings in Bioinformatics	87.933333
Disease Markers	130.040000
Genomics & Informatics	43.347826
Journal of Computational Biology	NaN
Microarrays	74.645833
Nucleic Acids Research	132.720000
Oncogenesis	137.708333

저널별 received ~ accepted 기간 평균



-> genomics & informatics의 논문들은
다른 논문들에 비해 received 하고 accepted 되는데에 걸리는 기간이 굉장히 짧은 편이었다.

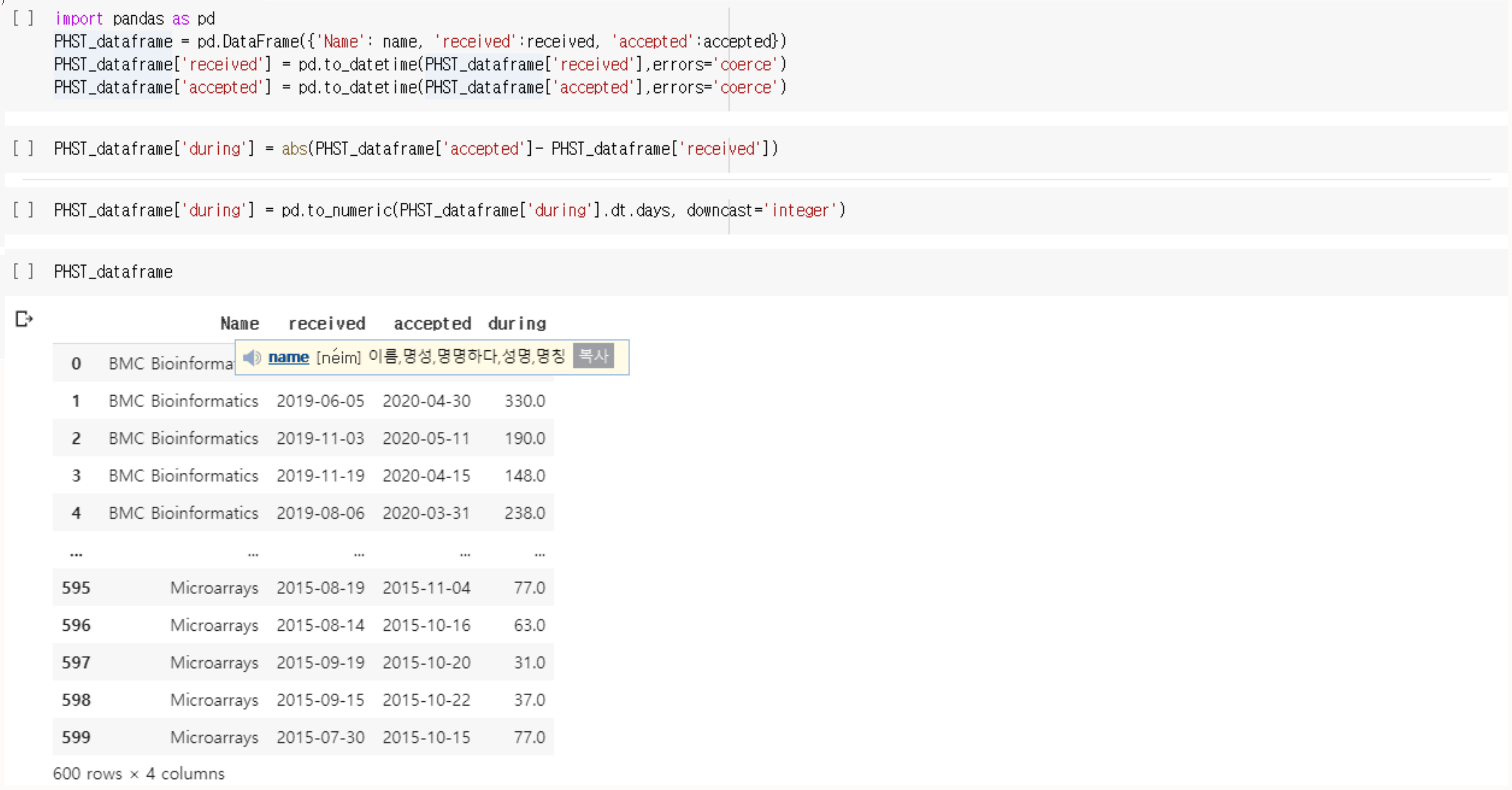
+ 12개의 저널별 논문 accepted 기간 분석 - Code



```
[ ] # -*- encoding: utf-8 -*-
    from Bio import Entrez
    from Bio import Medline

    Entrez.email = "neo.ewha@gmail.com"
    journalList = ["BMC Bioinformatics", "Genomics & Informatics", "Algorithms for Molecular Biology : AMB", "BMC Systems Biology", "Journal of Computational Biology", "Briefin

    PHST_list = []

    for journal in journalList:
        keyword = "(₩"" + journal + "₩"[Journal]) AND (₩"2015/01/01₩"[Publication Date] : ₩"2020/05/31₩"[Publication Date]) "
        handle1 = Entrez.esearch(db='pmc', term=keyword, retmax=50)
        record = Entrez.read(handle1)
        idlist = record['IdList']
        handle1.close()

        handle2 = Entrez.efetch(db="pmc", id=idlist, rettype="medline",retmode="text")
        records = Medline.parse(handle2)
        records = list(records)
        i = 0
        if idlist: # if journal list is not empy
            for record in records:
              PHST_list.append(record.get("PHST", "?"))
              #f = open(curdir + journal + "_" + str(idlist[i]) + ".txt", "w", encoding="utf-8")
              i += 1
              #f.write("Title: " + ''.join(record.get("TI", "?")) + "₩n")
              #f.close()
        else:
            print(journal + ": list empty!!!")
```

```
[ ] len(PHST_list)
    PHST_list
    journalList
    name = []
    for i in range(12):
      for j in range(50):
        name.append(journalList[i])
```

*자세한 코드는 생략



```
[ ] import pandas as pd
    PHST_dataframe = pd.DataFrame({'Name': name, 'received':received, 'accepted':accepted})
    PHST_dataframe['received'] = pd.to_datetime(PHST_dataframe['received'],errors='coerce')
    PHST_dataframe['accepted'] = pd.to_datetime(PHST_dataframe['accepted'],errors='coerce')
```

```
[ ] PHST_dataframe['during'] = abs(PHST_dataframe['accepted']- PHST_dataframe['received'])
```

```
[ ] PHST_dataframe['during'] = pd.to_numeric(PHST_dataframe['during'].dt.days, downcast='integer')
```

```
[ ] PHST_dataframe
```

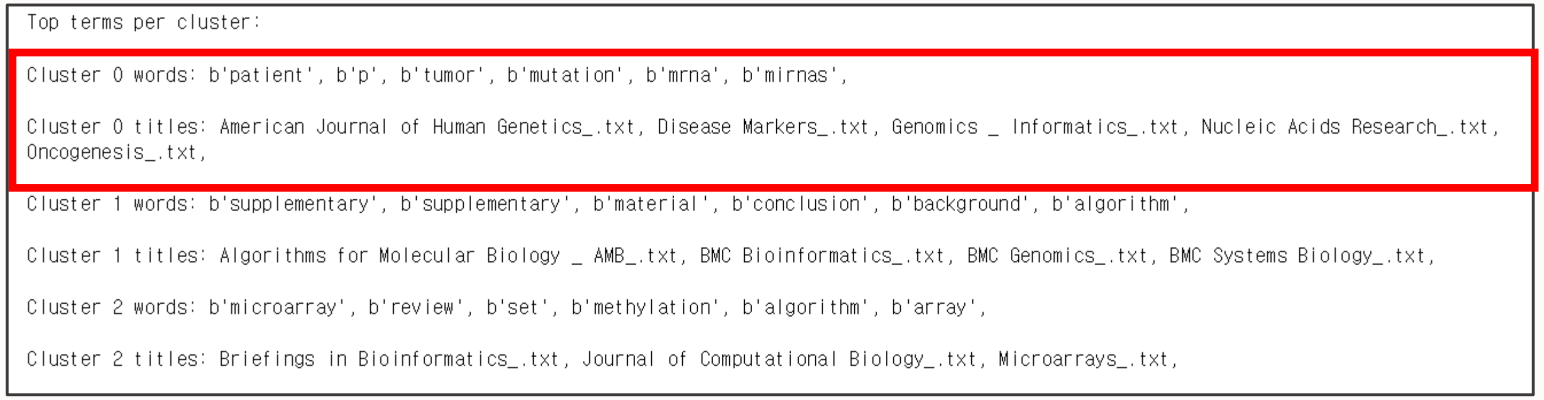
```
                  Name    received    accepted  during
  0   BMC Bioinforma  name [néim] 이름,명성,명명하다,성명,명칭 복사
  1   BMC Bioinformatics  2019-06-05  2020-04-30   330.0
  2   BMC Bioinformatics  2019-11-03  2020-05-11   190.0
  3   BMC Bioinformatics  2019-11-19  2020-04-15   148.0
  4   BMC Bioinformatics  2019-08-06  2020-03-31   238.0
 ...                 ...         ...         ...     ...
595          Microarrays  2015-08-19  2015-11-04    77.0
596          Microarrays  2015-08-14  2015-10-16    63.0
597          Microarrays  2015-09-19  2015-10-20    31.0
598          Microarrays  2015-09-15  2015-10-22    37.0
599          Microarrays  2015-07-30  2015-10-15    77.0

600 rows × 4 columns
```

3. Clustering

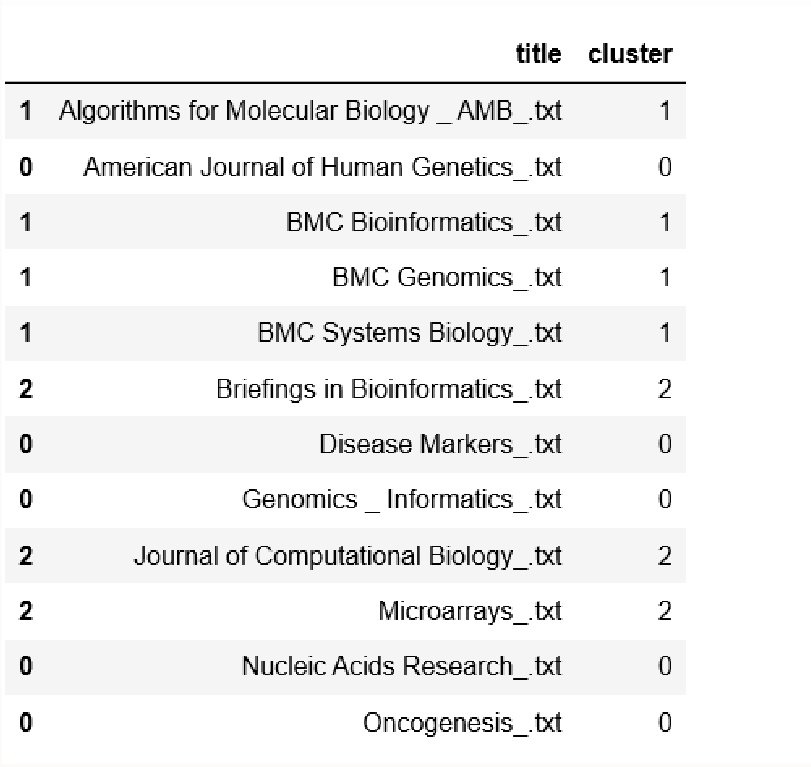
- 논문지 Kmeans clustering

Clustering

앞의 12가지 저널에 대해 유사한 저널을 찾아보기 위해 clustering을 수행해본 결과 (k=3일때)



```
Top terms per cluster:

Cluster 0 words: b'patient', b'p', b'tumor', b'mutation', b'mrna', b'mirnas',

Cluster 0 titles: American Journal of Human Genetics_.txt, Disease Markers_.txt, Genomics _ Informatics_.txt, Nucleic Acids Research_.txt,
Oncogenesis_.txt,
Cluster 1 words: b'supplementary', b'supplementary', b'material', b'conclusion', b'background', b'algorithm',

Cluster 1 titles: Algorithms for Molecular Biology _ AMB_.txt, BMC Bioinformatics_.txt, BMC Genomics_.txt, BMC Systems Biology_.txt,

Cluster 2 words: b'microarray', b'review', b'set', b'methylation', b'algorithm', b'array',

Cluster 2 titles: Briefings in Bioinformatics_.txt, Journal of Computational Biology_.txt, Microarrays_.txt,
```

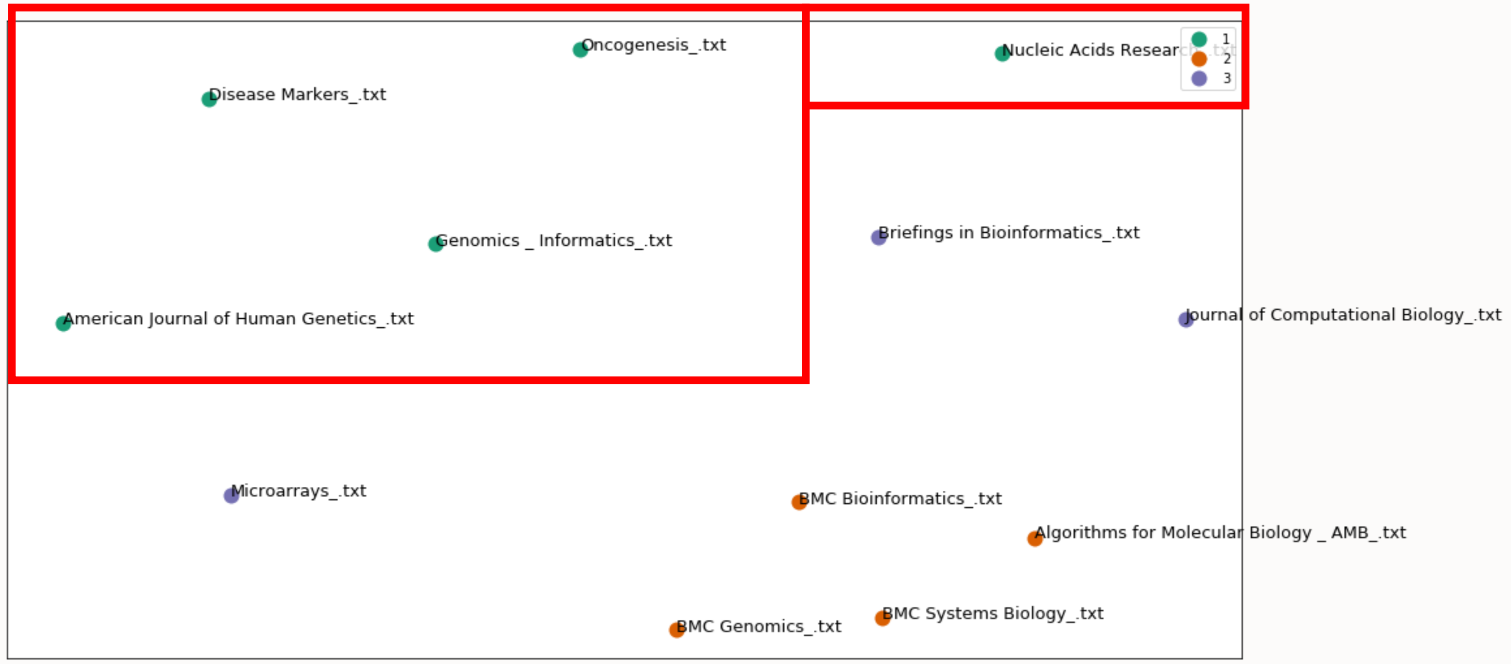


	title	cluster
1	Algorithms for Molecular Biology _ AMB_.txt	1
0	American Journal of Human Genetics_.txt	0
1	BMC Bioinformatics_.txt	1
1	BMC Genomics_.txt	1
1	BMC Systems Biology_.txt	1
2	Briefings in Bioinformatics_.txt	2
0	Disease Markers_.txt	0
0	Genomics _ Informatics_.txt	0
2	Journal of Computational Biology_.txt	2
2	Microarrays_.txt	2
0	Nucleic Acids Research_.txt	0
0	Oncogenesis_.txt	0

총 12가지 저널에 대해 GNI와 유사한 저널을 찾아보기 위하여 tf-idf를 이용하여 Kmeans Clustering을 수행해보았다.
먼저 k=3이었을 때 GNI는 patient, tumor, mutation, mran, miranas 와 같은 단어들로 이루어진 0번 클러스터로 분류가 되었다.

Clustering

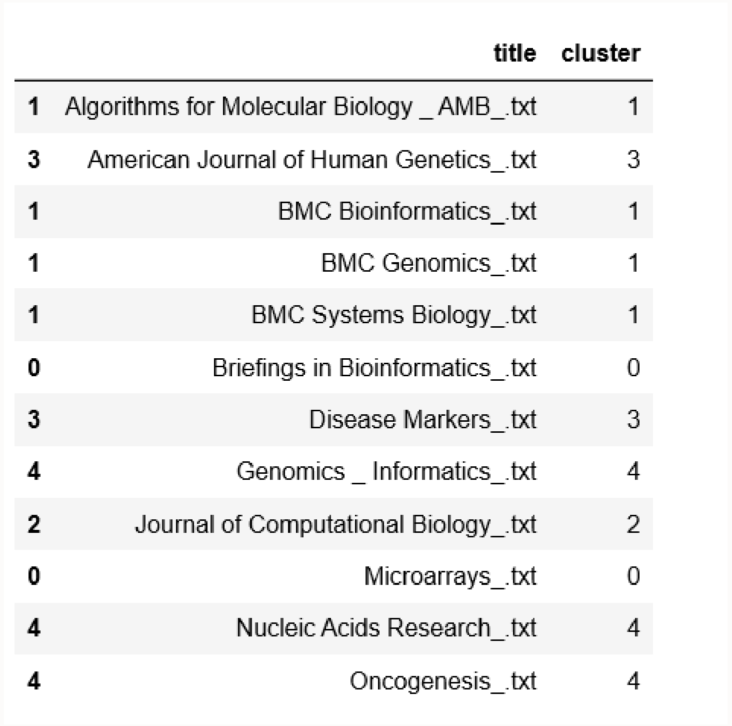
앞의 12가지 저널에 대해 유사한 저널을 찾아보기 위해 clustering을 수행해본 결과 (k=3일때)



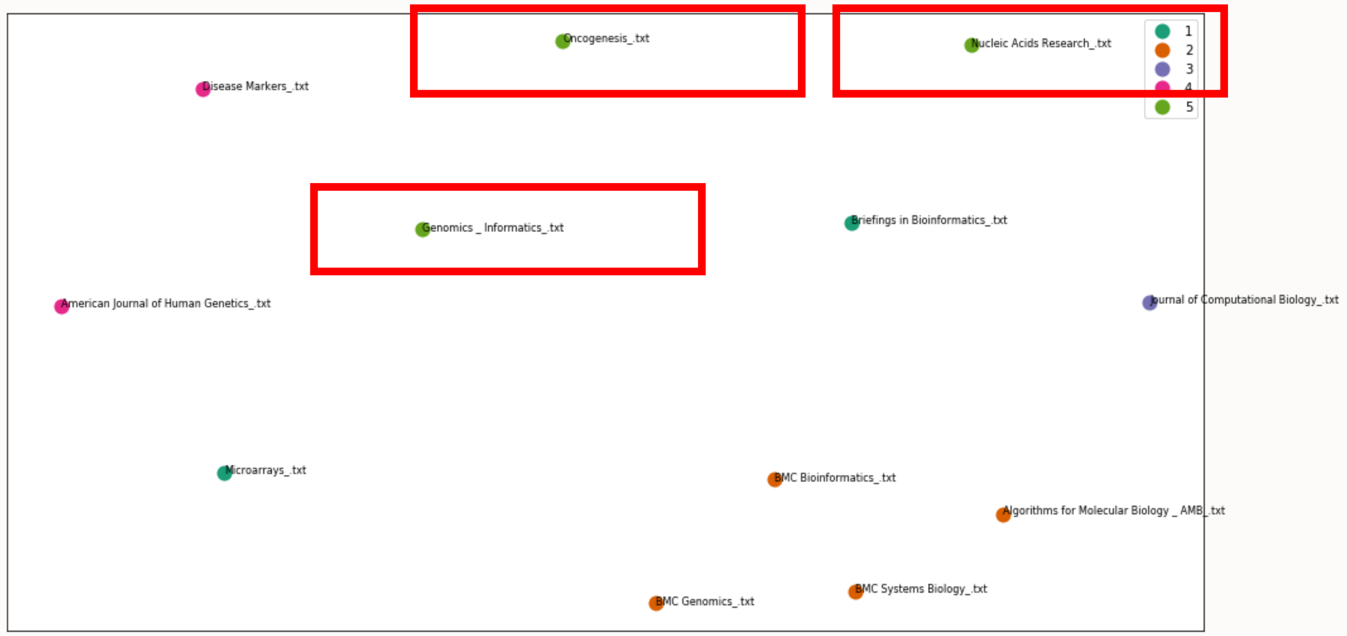
GNI와 같은 클러스터로는 American journal of Human Genetics, Diesaese Markers, Nucleic Acids Research 였다.
이는 앞에서 주요 키워드로만 유사도를 측정한 결과와 같았다.

Clustering

앞의 12가지 저널에 대해 유사한 저널을 찾아보기 위해 clustering을 수행해본 결과 (k=5일때)

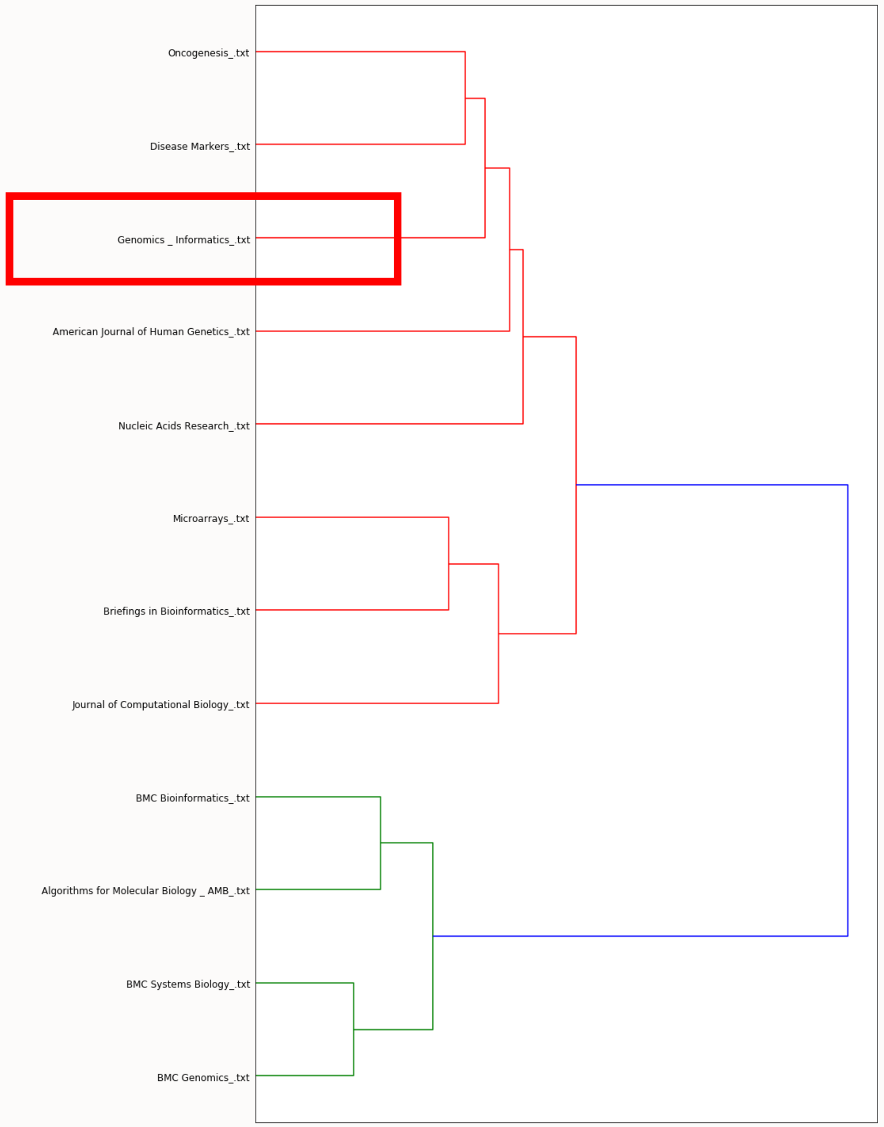


	title	cluster
1	Algorithms for Molecular Biology _ AMB_.txt	1
3	American Journal of Human Genetics_.txt	3
1	BMC Bioinformatics_.txt	1
1	BMC Genomics_.txt	1
1	BMC Systems Biology_.txt	1
0	Briefings in Bioinformatics_.txt	0
3	Disease Markers_.txt	3
4	Genomics _ Informatics_.txt	4
2	Journal of Computational Biology_.txt	2
0	Microarrays_.txt	0
4	Nucleic Acids Research_.txt	4
4	Oncogenesis_.txt	4



다음으로 k=5이었을 때 GNI는 4번 cluster로, Nucleic Acids Research, Oncogenesis와 같은 클러스터로 분류되었다.

Clustering - Hierachical doucument clustering

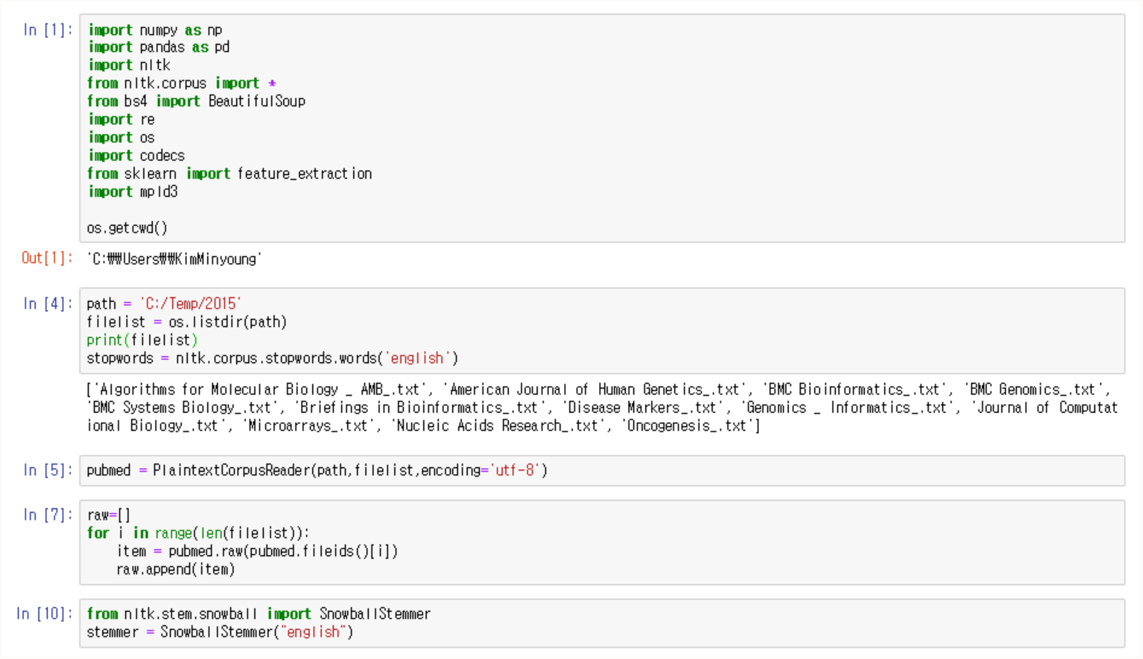


다음은 Hierachical doucument clustering 결과이다.

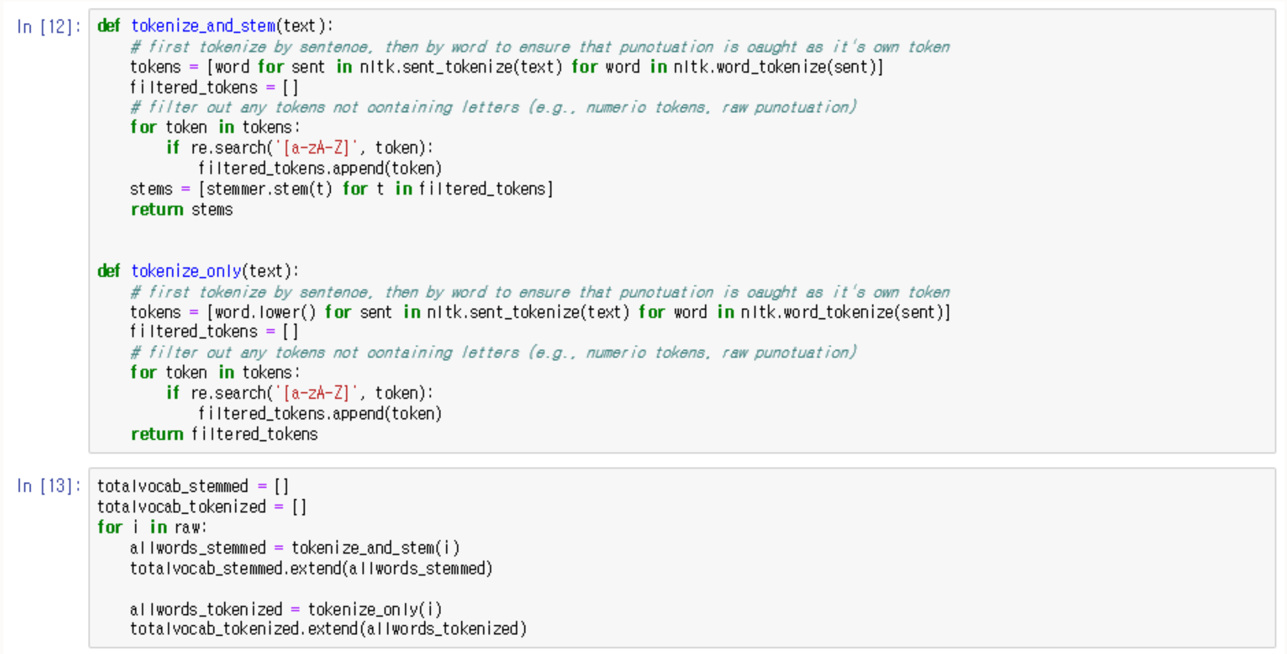
이전에 계산한 거리인 ward clustering을 사용하여 linkage_marix를 정의하고 dendrogram을 만들어 이를 시각화 하였다.

-> 그 결과 Genomics&Informatics 논문지는 Oncogenesis, Disease Markers, American journal of Human Genetics, Nucleic Acids Research 등과 유사한 것으로 확인됐다.

Clustering Code – (1)

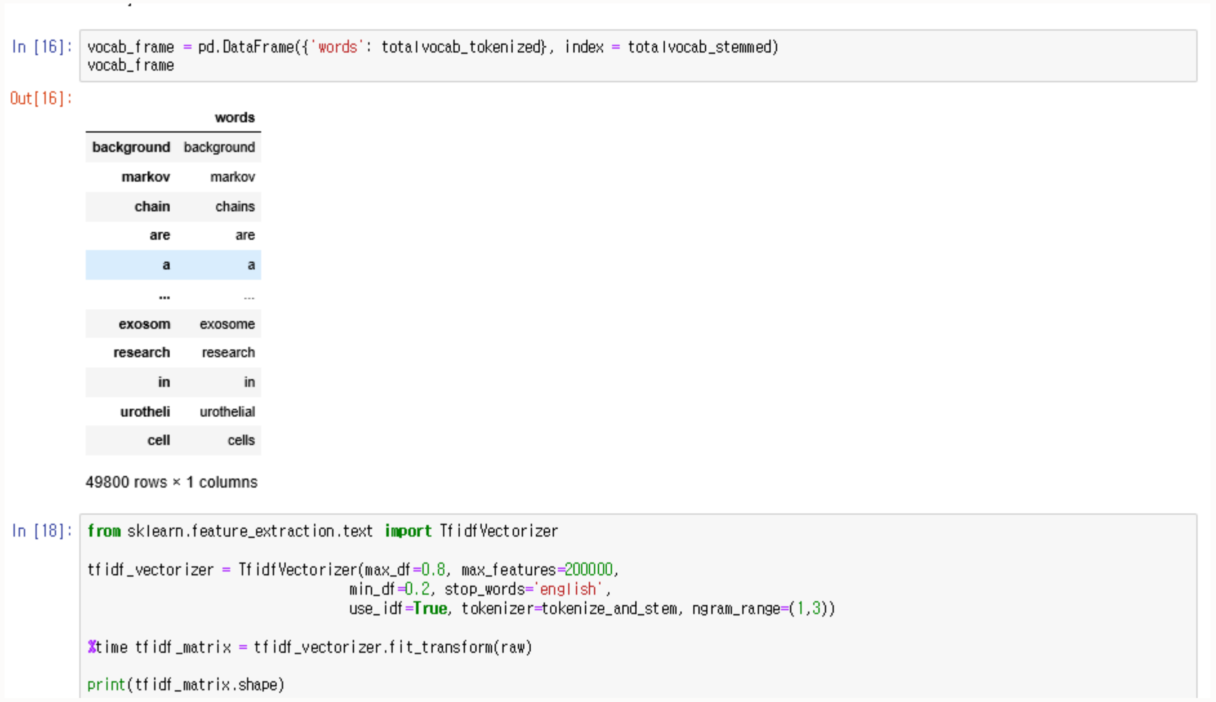


```
In [1]:  import numpy as np
         import pandas as pd
         import nltk
         from nltk.corpus import *
         from bs4 import BeautifulSoup
         import re
         import os
         import codecs
         from sklearn import feature_extraction
         import mpld3

         os.getcwd()

Out[1]:  'C:₩₩Users₩₩KimMinyoung'

In [4]:  path = 'C:/Temp/2015'
         filelist = os.listdir(path)
         print(filelist)
         stopwords = nltk.corpus.stopwords.words('english')

         ['Algorithms for Molecular Biology _ AMB_.txt', 'American Journal of Human Genetics_.txt', 'BMC Bioinformatics_.txt', 'BMC Genomics_.txt',
         'BMC Systems Biology_.txt', 'Briefings in Bioinformatics_.txt', 'Disease Markers_.txt', 'Genomics _ Informatics_.txt', 'Journal of Computat
         ional Biology_.txt', 'Microarrays_.txt', 'Nucleic Acids Research_.txt', 'Oncogenesis_.txt']

In [5]:  pubmed = PlaintextCorpusReader(path,filelist,encoding='utf-8')

In [7]:  raw=[]
         for i in range(len(filelist)):
             item = pubmed.raw(pubmed.fileids()[i])
             raw.append(item)

In [10]: from nltk.stem.snowball import SnowballStemmer
         stemmer = SnowballStemmer("english")
```



```
In [12]: def tokenize_and_stem(text):
             # first tokenize by sentence, then by word to ensure that punctuation is caught as it's own token
             tokens = [word for sent in nltk.sent_tokenize(text) for word in nltk.word_tokenize(sent)]
             filtered_tokens = []
             # filter out any tokens not containing letters (e.g., numeric tokens, raw punctuation)
             for token in tokens:
                 if re.search('[a-zA-Z]', token):
                     filtered_tokens.append(token)
             stems = [stemmer.stem(t) for t in filtered_tokens]
             return stems


         def tokenize_only(text):
             # first tokenize by sentence, then by word to ensure that punctuation is caught as it's own token
             tokens = [word.lower() for sent in nltk.sent_tokenize(text) for word in nltk.word_tokenize(sent)]
             filtered_tokens = []
             # filter out any tokens not containing letters (e.g., numeric tokens, raw punctuation)
             for token in tokens:
                 if re.search('[a-zA-Z]', token):
                     filtered_tokens.append(token)
             return filtered_tokens

In [13]: totalvocab_stemmed = []
         totalvocab_tokenized = []
         for i in raw:
             allwords_stemmed = tokenize_and_stem(i)
             totalvocab_stemmed.extend(allwords_stemmed)

             allwords_tokenized = tokenize_only(i)
             totalvocab_tokenized.extend(allwords_tokenized)
```

Clustering Code – (2)



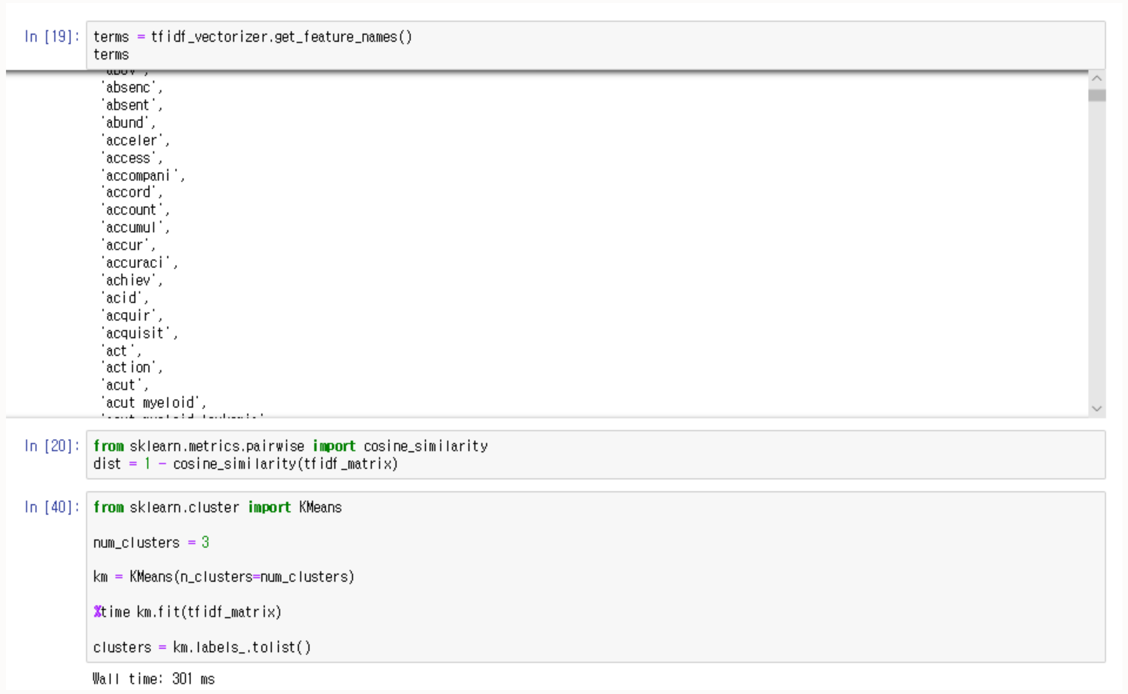
```
In [16]: vocab_frame = pd.DataFrame({'words': totalvocab_tokenized}, index = totalvocab_stemmed)
         vocab_frame
```

Out[16]:

	words
background	background
markov	markov
chain	chains
are	are
a	a
...	...
exosom	exosome
research	research
in	in
urotheli	urothelial
cell	cells

49800 rows × 1 columns

```
In [18]: from sklearn.feature_extraction.text import TfidfVectorizer

         tfidf_vectorizer = TfidfVectorizer(max_df=0.8, max_features=200000,
                                            min_df=0.2, stop_words='english',
                                            use_idf=True, tokenizer=tokenize_and_stem, ngram_range=(1,3))

         %time tfidf_matrix = tfidf_vectorizer.fit_transform(raw)

         print(tfidf_matrix.shape)
```



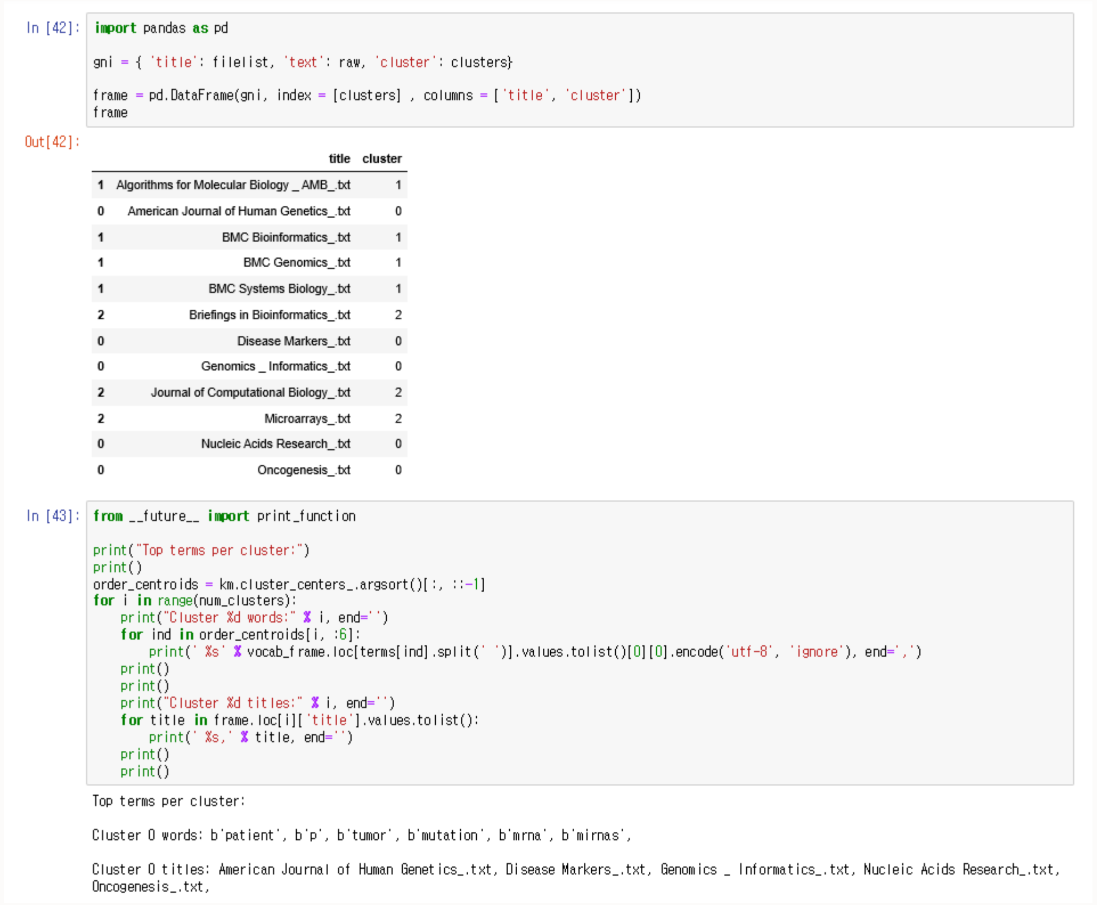
```
In [19]: terms = tfidf_vectorizer.get_feature_names()
         terms
```

```
 'absenc',
 'absent',
 'abund',
 'acceler',
 'access',
 'accompani',
 'accord',
 'account',
 'accumul',
 'accur',
 'accuraci',
 'achiev',
 'acid',
 'acquir',
 'acquisit',
 'act',
 'action',
 'acut',
 'acut myeloid',
```

```
In [20]: from sklearn.metrics.pairwise import cosine_similarity
         dist = 1 - cosine_similarity(tfidf_matrix)
```

```
In [40]: from sklearn.cluster import KMeans

         num_clusters = 3

         km = KMeans(n_clusters=num_clusters)

         %time km.fit(tfidf_matrix)

         clusters = km.labels_.tolist()

         Wall time: 301 ms
```

Clustering Code – (3)



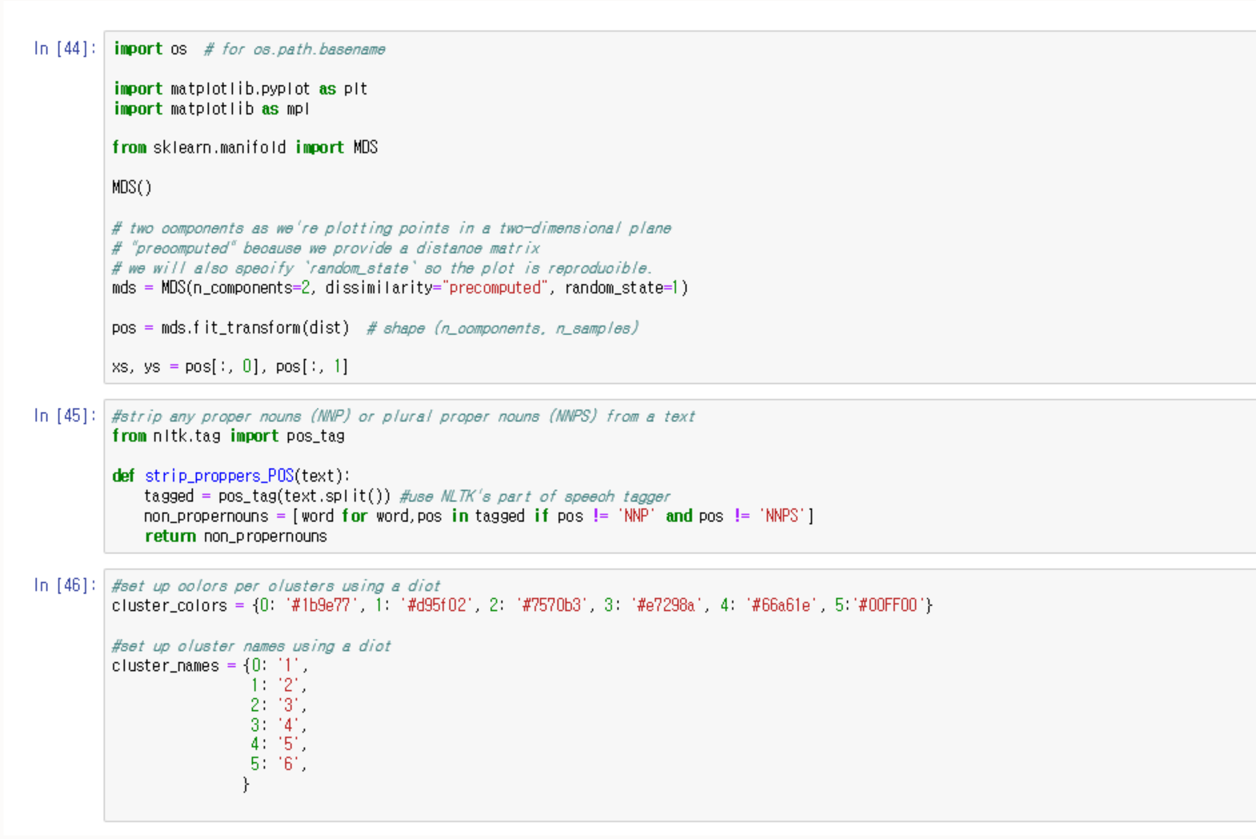
```
In [42]: import pandas as pd

         gni = { 'title': filelist, 'text': raw, 'cluster': clusters}

         frame = pd.DataFrame(gni, index = [clusters] , columns = ['title', 'cluster'])
         frame
```

Out[42]:

	title	cluster
1	Algorithms for Molecular Biology _ AMB_.txt	1
0	American Journal of Human Genetics_.txt	0
1	BMC Bioinformatics_.txt	1
1	BMC Genomics_.txt	1
1	BMC Systems Biology_.txt	1
2	Briefings in Bioinformatics_.txt	2
0	Disease Markers_.txt	0
0	Genomics _ Informatics_.txt	0
2	Journal of Computational Biology_.txt	2
2	Microarrays_.txt	2
0	Nucleic Acids Research_.txt	0
0	Oncogenesis_.txt	0

```
In [43]: from __future__ import print_function

         print("Top terms per cluster:")
         print()
         order_centroids = km.cluster_centers_.argsort()[:, ::-1]
         for i in range(num_clusters):
             print("Cluster %d words:" % i, end='')
             for ind in order_centroids[i, :6]:
                 print(' %s' % vocab_frame.loc[terms[ind].split(' ')].values.tolist()[0][0].encode('utf-8', 'ignore'), end=',')
             print()
             print()
             print("Cluster %d titles:" % i, end='')
             for title in frame.loc[i]['title'].values.tolist():
                 print(' %s,' % title, end='')
             print()
             print()
```

```
Top terms per cluster:

Cluster 0 words: b'patient', b'p', b'tumor', b'mutation', b'mrna', b'mirnas',

Cluster 0 titles: American Journal of Human Genetics_.txt, Disease Markers_.txt, Genomics _ Informatics_.txt, Nucleic Acids Research_.txt,
Oncogenesis_.txt,
```

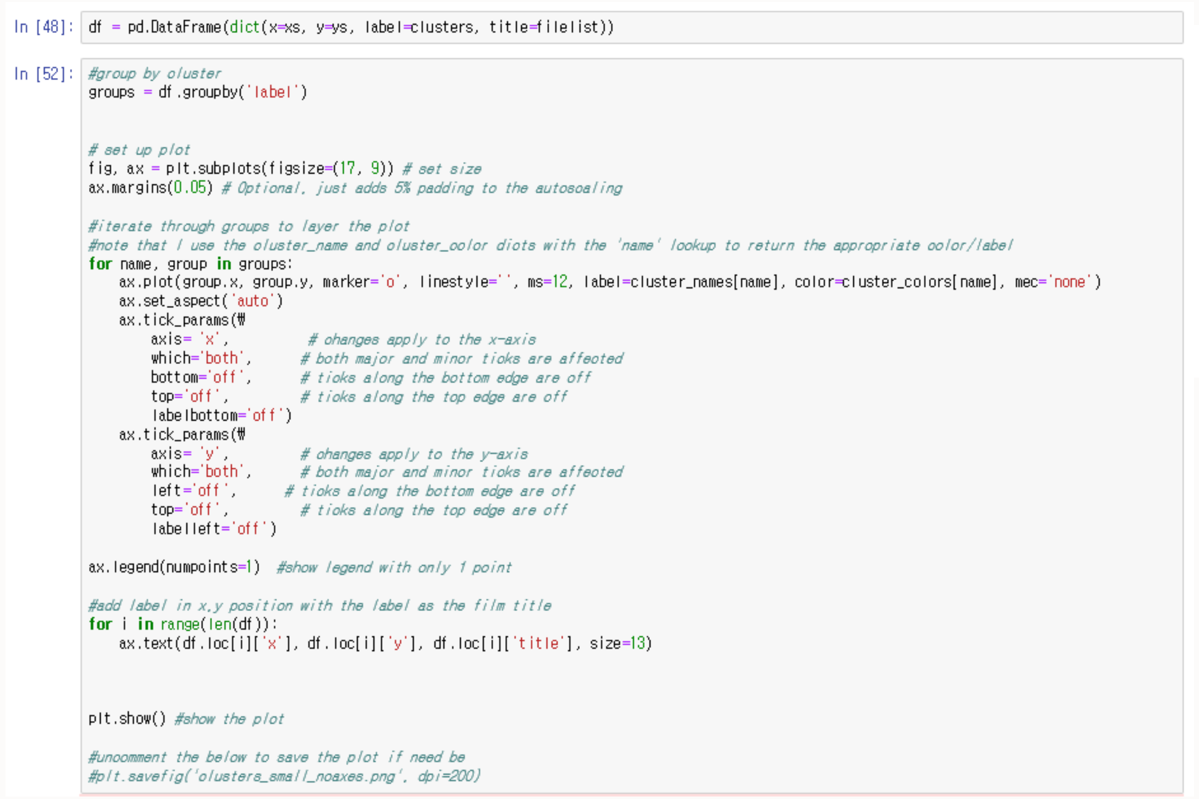


```
In [44]: import os  # for os.path.basename

         import matplotlib.pyplot as plt
         import matplotlib as mpl

         from sklearn.manifold import MDS

         MDS()

         # two components as we're plotting points in a two-dimensional plane
         # "precomputed" because we provide a distance matrix
         # we will also specify `random_state` so the plot is reproducible.
         mds = MDS(n_components=2, dissimilarity="precomputed", random_state=1)

         pos = mds.fit_transform(dist)  # shape (n_components, n_samples)

         xs, ys = pos[:, 0], pos[:, 1]
```

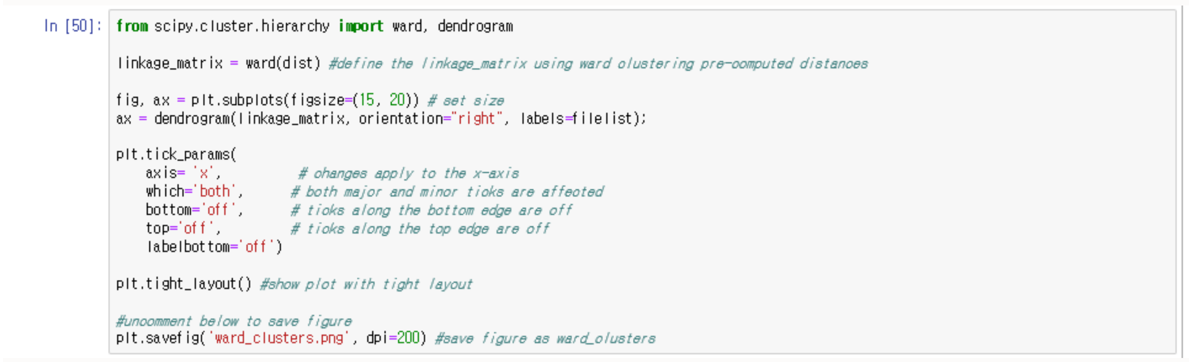
```
In [45]: #strip any proper nouns (NNP) or plural proper nouns (NNPS) from a text
         from nltk.tag import pos_tag

         def strip_proppers_POS(text):
             tagged = pos_tag(text.split()) #use NLTK's part of speech tagger
             non_propernouns = [word for word,pos in tagged if pos != 'NNP' and pos != 'NNPS']
             return non_propernouns
```

```
In [46]: #set up colors per clusters using a dict
         cluster_colors = {0: '#1b9e77', 1: '#d95f02', 2: '#7570b3', 3: '#e7298a', 4: '#66a61e', 5:'#00FF00'}

         #set up cluster names using a dict
         cluster_names = {0: '1',
                          1: '2',
                          2: '3',
                          3: '4',
                          4: '5',
                          5: '6',
                         }
```

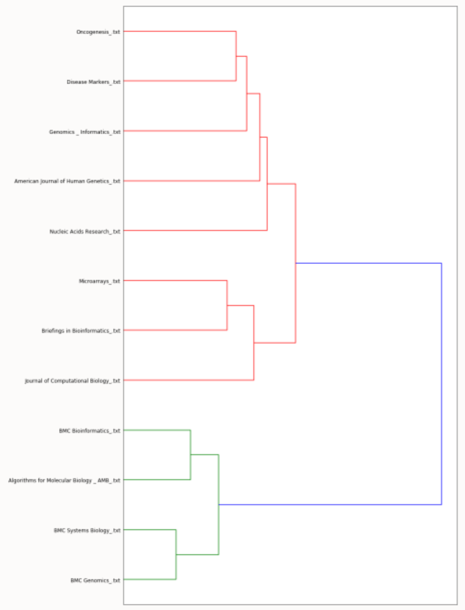
Clustering Code – (4)



```
In [48]: df = pd.DataFrame(dict(x=xs, y=ys, label=clusters, title=filelist))

In [52]: #group by cluster
         groups = df.groupby('label')


         # set up plot
         fig, ax = plt.subplots(figsize=(17, 9)) # set size
         ax.margins(0.05) # Optional, just adds 5% padding to the autoscaling

         #iterate through groups to layer the plot
         #note that I use the cluster_name and cluster_color dicts with the 'name' lookup to return the appropriate color/label
         for name, group in groups:
             ax.plot(group.x, group.y, marker='o', linestyle='', ms=12, label=cluster_names[name], color=cluster_colors[name], mec='none')
             ax.set_aspect('auto')
             ax.tick_params(₩
                 axis= 'x',          # changes apply to the x-axis
                 which='both',       # both major and minor ticks are affected
                 bottom='off',       # ticks along the bottom edge are off
                 top='off',          # ticks along the top edge are off
                 labelbottom='off')
             ax.tick_params(₩
                 axis= 'y',          # changes apply to the y-axis
                 which='both',       # both major and minor ticks are affected
                 left='off',       # ticks along the bottom edge are off
                 top='off',          # ticks along the top edge are off
                 labelleft='off')

         ax.legend(numpoints=1)  #show legend with only 1 point

         #add label in x,y position with the label as the film title
         for i in range(len(df)):
             ax.text(df.loc[i]['x'], df.loc[i]['y'], df.loc[i]['title'], size=13)



         plt.show() #show the plot

         #uncomment the below to save the plot if need be
         #plt.savefig('clusters_small_noaxes.png', dpi=200)
```



```
In [50]: from scipy.cluster.hierarchy import ward, dendrogram

         linkage_matrix = ward(dist) #define the linkage_matrix using ward clustering pre-computed distances

         fig, ax = plt.subplots(figsize=(15, 20)) # set size
         ax = dendrogram(linkage_matrix, orientation="right", labels=filelist);

         plt.tick_params(
             axis= 'x',          # changes apply to the x-axis
             which='both',      # both major and minor ticks are affected
             bottom='off',      # ticks along the bottom edge are off
             top='off',         # ticks along the top edge are off
             labelbottom='off')

         plt.tight_layout() #show plot with tight layout

         #uncomment below to save figure
         plt.savefig('ward_clusters.png', dpi=200) #save figure as ward_clusters
```


Oncogenesis_.txt
Disease Markers_.txt
Genomics _ Informatics_.txt
American Journal of Human Genetics_.txt
Nucleic Acids Research_.txt
Microarrays_.txt
Briefings in Bioinformatics_.txt
Journal of Computational Biology_.txt
BMC Bioinformatics_.txt
Algorithms for Molecular Biology _ AMB_.txt
BMC Systems Biology_.txt
BMC Genomics_.txt