Genomics & Informatics 논문지 단어 분포

1829008 김민영

0. SetUp

Print를 통해 살펴본 결과, GNI Corpus 1.0/row 폴더에는 총 345개의 논문지의 text 파일이 있었고 파일 이름 저장 방식은 왼쪽과 같았다. 또한 파일을 열어 연도를 살펴본 결과 오른쪽과 같았다.

index 0~12 : vol.15
'gi-2017-15-x-xxx.txt'
index 14~17 : vol.16
'gi-2018-16-1-x.txt'
index 18~206 : vol. 10-14
'gni-10~14-xxx.txt'
index 207~ 344: vol. 6-9
'gni-6~9-x-xx.txt'

vol.6,7,8,9 : 2008년
vol.10,11 : 2012년
vol.12 : 2014년
vol.13 : 2015년
vol.14 : 2016년
vol.15 : 2017년
vol.16 : 2018년

이들을 한눈에 보기 쉽게 하기위해, 접근을 편리하게 하기위해 dataframe으로 만들었다.

	file_name	vol	year
0	gi-2017-15-3-81.txt	15	2017
1	gi-2017-15-3-82.txt	15	2017
2	gi-2017-15-4-113.txt	15	2017
3	gi-2017-15-4-114.txt	15	2017
4	gi-2017-15-4-123.txt	15	2017
...	...	...	...
340	gni-9-4-173.txt	9	2008
341	gni-9-4-181.txt	9	2008
342	gni-9-4-189.txt	9	2008
343	gni-9-4-194.txt	9	2008
344	gni-9-4-197.txt	9	2008

345 rows × 3 columns

1. 논문의 길이

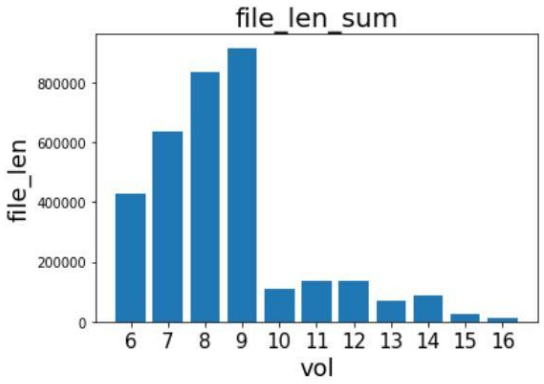
총 345개의 논문이 계산된다. 다만 쉽게 파악하기 힘드므로, 시각화하여 살펴보려고 한다. 각 논문의 길이(file_len) 또한 DataFrame에 넣어주면 다음과 같다.

	file_name	vol	year	file_len
0	gi-2017-15-3-81.txt	15	2017	367
1	gi-2017-15-3-82.txt	15	2017	1949
2	gi-2017-15-4-113.txt	15	2017	395
3	gi-2017-15-4-114.txt	15	2017	3638
4	gi-2017-15-4-123.txt	15	2017	2054

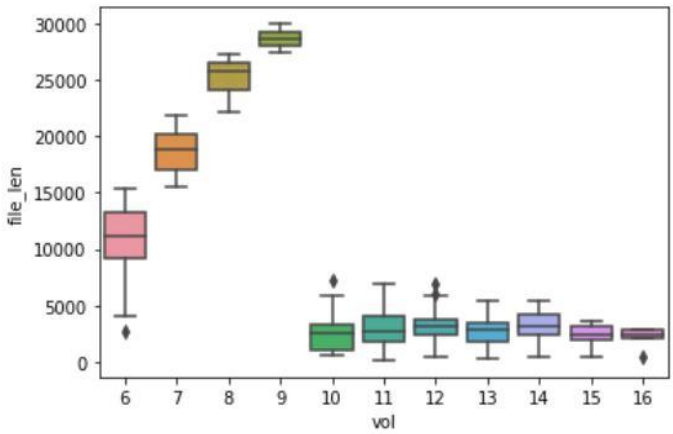
....

논문별(vol) file_len의 합

vol	file_len
6	426993
7	634884
8	834052
9	916382
10	107580
11	137952
12	136481
13	70122
14	87834
15	27282
16	10716



논문별(vol) file_len 자세한 통계

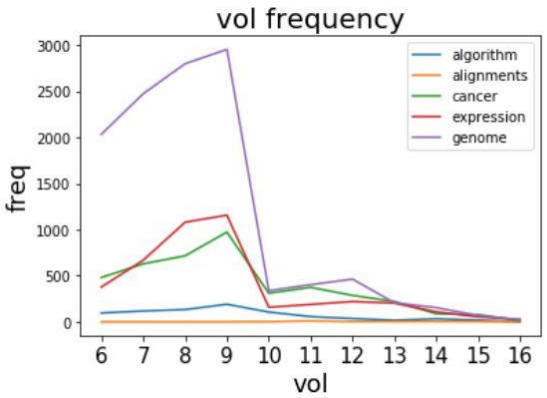


vol.6-9 논문은 대체적으로 길이가 길고, 10-16 논문은 길이가 짧다는 것을 확인할 수 있다.

2. 논문별(Vol) 단어 빈도수

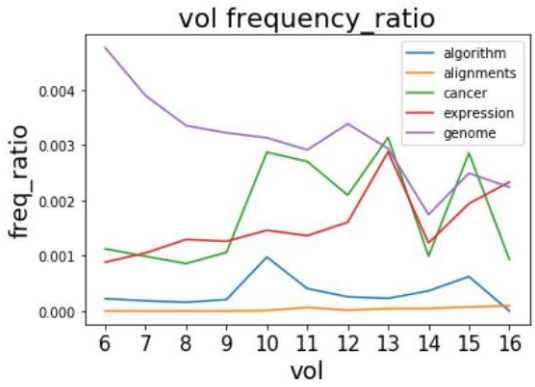
nltk.ConditionalFreqDist를 통해 단어빈도수를 구하고, vol별 sum으로 groupby하여 데이터프레임에 넣고 시각화 하여살펴보면 다음과 같다. 검색할 단어는 target_list에 넣어주면 되는데, 본 과제에서는 예시에있는 expression, genome, cancer, algorithm, alignments를 검색했다.

vol	file_len	expression	genome	cancer	algorithm	alignments
6	426993	377.0	2033.0	480.0	95.0	0.0
7	634884	667.0	2472.0	628.0	117.0	0.0
8	834052	1079.0	2797.0	715.0	132.0	0.0
9	916382	1156.0	2953.0	972.0	190.0	0.0
10	107580	157.0	337.0	309.0	105.0	1.0
11	137952	188.0	402.0	373.0	56.0	9.0
12	136481	219.0	462.0	286.0	35.0	2.0
13	70122	202.0	206.0	220.0	16.0	3.0
14	87834	108.0	153.0	87.0	32.0	4.0
15	27282	53.0	68.0	78.0	17.0	2.0
16	10716	25.0	24.0	10.0	0.0	1.0



전체적으로 봤을 때, genome 단어가 확실히 많이 등장하는 것을 볼 수 있다. 또한 cancer, expression 단어도 꽤 많이 등장한다. 반면에 alignments와 algorithm 단어는 상대적으로 적게 등장한다.

다만, 앞에서 살펴봤듯이 vol.6-9의 논문의 길이 자체가 길고, vol 11-16 논문의 길이는 짧았기 때문에 vol.6-9의 target 단어의 개수 또한 수적으로 클 수 밖에 없었다. (file_len 그래프와 분포양상이 비슷하다.) 따라서 vol별로 정확한 비교해보기위해, 각 논문의 길이를 고려하여 "특정단어빈도수/논문의길이"그래프를 그려볼 수 있다.

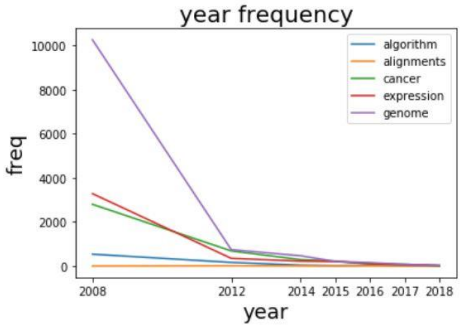


cancer : vol.6-9,14 에서는 다른 논문에 비해 등장하는 비율이 적고, vol.10 13 15 에서는 다른 논문들에 비해 cancer 단어가 많이 등장하는 것을 확인할 수 있다.
algorithm : 다른 논문에 비해 vol.10, vol15 에서 등장하는 비율이 높은 것을 확인할 수 있다.
alignments : 모든 논문에서 굉장히 적게 등장했고, vol 6~9 에서는 아예 등장하지 않았다.
genome : 모든 논문에서 많이 등장했지만, 특히 vol6 에서 두드러지는 것이 보인다.
expression :모든 논문에서 꽤 많이 등장하는데 vol.13 에서 다른 논문에 비해 등장하는 비율이 높다.

3. 연도별 단어빈도수

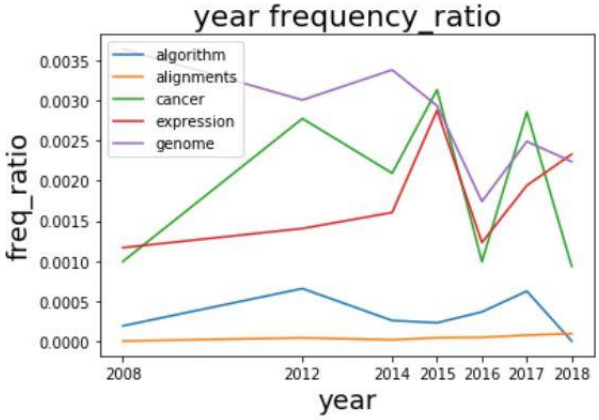
마찬가지로 nltk.ConditionalFreqDist를 통해 단어빈도수를 구하고, 연도별 sum으로 groupby하여 데이터프레임에 넣고 시각화 하여살펴보면 다음과 같다.

year	file_len	expression	genome	cancer	algorithm	alignments
2008	2812311	3279.0	10255.0	2795.0	534.0	0.0
2012	245532	345.0	739.0	682.0	161.0	10.0
2014	136481	219.0	462.0	286.0	35.0	2.0
2015	70122	202.0	206.0	220.0	16.0	3.0
2016	87834	108.0	153.0	87.0	32.0	4.0
2017	27282	53.0	68.0	78.0	17.0	2.0
2018	10716	25.0	24.0	10.0	0.0	1.0



역시 genome 단어가 확실히 많이 등장하는 것을 볼 수 있다. cancer, expression 단어도 꽤 많이 등장한다. algorithm, alignments는 적게 등장하는 것을 볼 수 있다.

연도별로 살펴볼 때에도 각 논문의 길이를 고려하여 "특정단어빈도수/논문의길이"그래프를 그려볼 수 있다.



expression : 다른 연도에 비해 2015년, 2017년에 자주 등장한 것을 볼 수 있다.
cancer : 다른 연도에 비해 2012, 2015, 2017년에 자주 등장한 것을 볼 수 있다.
genome : 전체적으로 자주 등장했지만, 2016년에는 다른 연도에비해 상대적으로 덜 등장한 것을 볼 수 있다.
alignments : 전체적으로 적게 등장했다.
algorithm : 전체적으로 적게 등장했지만, 2012년과 2017년에는 다른 연도에 비해 상대적으로 자주 등장한 것을 볼 수 있다.

+ 전체 Code(.ipynb)

0. SetUp ¶

In [2]:
```
import os
import nltk
from nltk.corpus import *
corpus_root = "C:/Users/KimMinyoung/nltk_data/corpora/Genomics-Informatics-Corpus-master/Genomics-Informatics-Corpus-master/GNI
Corpus 1.0/raw/"
giCorpus = {}
filelists = PlaintextCorpusReader(corpus_root, '.*\.txt', encoding='utf-8')
```

In [3]:
```
len(filelists.fileids())
```

Out[3]: 345

In [4]:
```
wnl = nltk.WordNetLemmatizer()
```

In [5]:
```
for i in range(345):
    print(i,filelists.fileids()[i])
```

```
0 gi-2017-15-3-81.txt
1 gi-2017-15-3-82.txt
2 gi-2017-15-4-113.txt
3 gi-2017-15-4-114.txt
4 gi-2017-15-4-123.txt
5 gi-2017-15-4-128.txt
6 gi-2017-15-4-136.txt
7 gi-2017-15-4-142.txt
8 gi-2017-15-4-147.txt
9 gi-2017-15-4-156.txt
10 gi-2017-15-4-162.txt
11 gi-2017-15-4-170.txt
12 gi-2017-15-4-178.txt
13 gi-2018-16-1-1.txt
14 gi-2018-16-1-10.txt
```

••••

총 345개의 논문지가 있었고 파일 이름 저장 방식은 다음과 같았다.

index 0~12 : vol.15

'gi-2017-15-x-xxx.txt'

index 14~17 : vol.16

'gi-2018-16-1-x.txt'

index 18~206 : vol. 10-14

'gni-10~14-xxx.txt'

index 207~ 344: vol. 6-9

'gni-6~9-x-xx.txt'

또한, 파일을 열어 연도를 확인한 결과 다음과 같았다.

vol.6,7,8,9 : 2008년

vol.10,11 : 2012년

vol.12 : 2014년

vol.13 : 2015년

vol.14 : 2016년

vol.15 : 2017년

vol.16 : 2018년

따라서 이들을 한눈에 보기 쉽게하기위해, 접근을 편리하게 하기위해 dataframe으로 만들어보자.

아래와 같이 vol, year 리스트에 각각의 값들을 저장해주면된다.

1. 논문의 길이

In [8]:
```
file_len= []
for file in filelists.fileids():
    wordlist = filelists.words(file)
    file_len.append(len(wordlist))
    print("Printing size of  " + file + " original wordlist: " + str(len(wordlist)))
    lemmatizedWordlist = [wnl.lemmatize(t) for t in wordlist]
    giCorpus[file] = lemmatizedWordlist
    fd = nltk.FreqDist(w for w in lemmatizedWordlist)
```

```
Printing size of  gi-2017-15-3-81.txt original wordlist: 367
Printing size of  gi-2017-15-3-82.txt original wordlist: 1949
Printing size of  gi-2017-15-4-113.txt original wordlist: 395
Printing size of  gi-2017-15-4-114.txt original wordlist: 3638
Printing size of  gi-2017-15-4-123.txt original wordlist: 2054
Printing size of  gi-2017-15-4-128.txt original wordlist: 2618
Printing size of  gi-2017-15-4-136.txt original wordlist: 2644
Printing size of  gi-2017-15-4-142.txt original wordlist: 1804
Printing size of  gi-2017-15-4-147.txt original wordlist: 3374
Printing size of  gi-2017-15-4-156.txt original wordlist: 1972
Printing size of  gi-2017-15-4-162.txt original wordlist: 3338
Printing size of  gi-2017-15-4-170.txt original wordlist: 3129
Printing size of  gi-2017-15-4-178.txt original wordlist: 2084
Printing size of  gi-2018-16-1-1.txt original wordlist: 415
Printing size of  gi-2018-16-1-10.txt original wordlist: 2454
Printing size of  gi-2018-16-1-14.txt original wordlist: 2858
Printing size of  gi-2018-16-1-2.txt original wordlist: 2905
Printing size of  gni-10-1.txt original wordlist: 1081
Printing size of  gni-10-106.txt original wordlist: 2071
Printing size of  gni-10-110.txt original wordlist: 2530
```

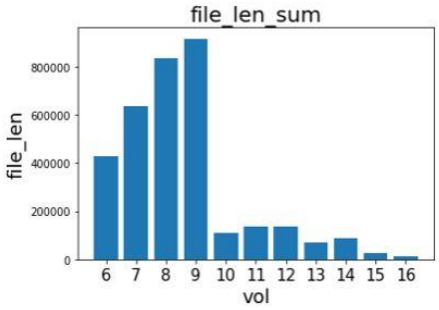
In [6]:
```
import pandas as pd
import numpy as np
import matplotlib
#df_filelists
vol=[]
year = []
for i in range(0,12):
    vol.append(15)
    year.append(2017)
for i in range(12,17):
    vol.append(16)
    year.append(2018)
for i in range(17,58):
    vol.append(10)
    year.append(2012)
for i in range(58,107):
    vol.append(11)
    year.append(2012)
for i in range(107,151):
    vol.append(12)
    year.append(2014)
for i in range(151,179):
    vol.append(13)
    year.append(2015)
for i in range(179,207):
    vol.append(14)
    year.append(2016)
for i in range(207,246):
    vol.append(6)
    year.append(2008)
for i in range(246,280):
    vol.append(7)
    year.append(2008)
for i in range(280,313):
    vol.append(8)
    year.append(2008)
for i in range(313,345):
    vol.append(9)
    year.append(2008)
```

In [7]:
```
df_filelists = pd.DataFrame({'file_name': filelists.fileids(),'vol' : vol, 'year':year})
df_filelists
```

Out[7]:

	file_name	vol	year
0	gi-2017-15-3-81.txt	15	2017
1	gi-2017-15-3-82.txt	15	2017
2	gi-2017-15-4-113.txt	15	2017
3	gi-2017-15-4-114.txt	15	2017
4	gi-2017-15-4-123.txt	15	2017
...	...	...	...

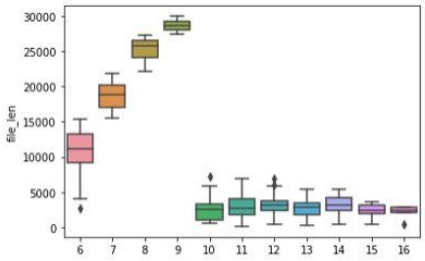
In [11]:
```
import matplotlib.pyplot as plt
vol = np.arange(6,17)
file_len_sum =  filelen_sum['file_len']
plt.bar(vol, file_len_sum)
plt.title('file_len_sum', fontsize=20)
plt.xlabel('vol', fontsize=18)
plt.ylabel('file_len', fontsize=18)
plt.xticks(vol, fontsize=15)
plt.show()
```



논문별 original worlist size 자세한 통계치

In [12]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.boxplot(x='vol', y='file_len', data=df_filelists)
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x20d4f996c88>



vol.6-9 논문은 대체적으로 길이가 길고, 10-16 논문은 길이가 짧다는 것을 확인할 수 있다.

2. Vol별 단어 빈도수

각 단어별 출현 빈도

In [13]:
```
[(fileid[:-4], target)
 for fileid in filelists.fileids()
 for word in filelists.words(fileid)
 for target in ['cancer', 'expression']
 if word.lower().startswith(target)]
```

Out[13]:
```
[('gi-2017-15-3-81', 'expression'),
 ('gi-2017-15-4-113', 'expression'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'expression'),
 ('gi-2017-15-4-114', 'cancer'),
 ('gi-2017-15-4-114', 'cancer')
```

-> 전체를 시각화하면 너무 많으므로 conditional FreqDist를 dataframe으로 만들어서 vol별로, 연도별로 통계치를 출력하려고 한다.

검색할 단어 리스트 : target_list에 넣어주면 된다.

In [14]:
```
target_list =['algorithm','alignments','cancer', 'expression','genome']

cfd = nltk.ConditionalFreqDist(
    (fileid, target)
 for fileid in filelists.fileids()
 for word in filelists.words(fileid)
 for target in target_list
 if word.lower().startswith(target)
)
```

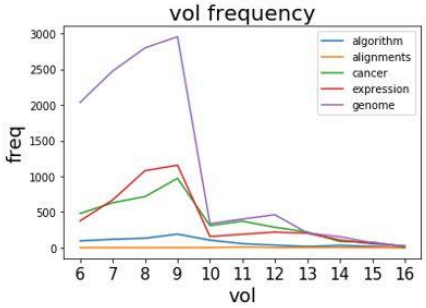
In [15]:
```
df_freq=pd.DataFrame.from_dict(cfd,orient='index')
df_freq.index.name='file_name'
df_freq
df_filelists=pd.merge(df_filelists,df_freq,how='left',on='file_name')
```

In [16]:
```
freq_sum = df_filelists.groupby('vol').sum()
freq_sum.drop(columns='year')
```

Out[16]:

vol	file_len	expression	genome	cancer	algorithm	alignments
6	426993	377.0	2033.0	480.0	95.0	0.0
7	634884	667.0	2472.0	628.0	117.0	0.0
8	834052	1079.0	2797.0	715.0	132.0	0.0
9	916382	1156.0	2953.0	972.0	190.0	0.0
10	107580	157.0	337.0	309.0	105.0	1.0
11	137952	188.0	402.0	373.0	56.0	9.0
12	136481	219.0	462.0	286.0	35.0	2.0
13	70122	202.0	206.0	220.0	16.0	3.0
14	87834	108.0	153.0	87.0	32.0	4.0
15	27282	53.0	68.0	78.0	17.0	2.0
16	10716	25.0	24.0	10.0	0.0	1.0

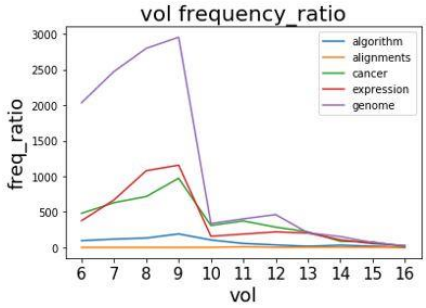
In [17]:
```
algorithm = freq_sum['algorithm']
alignments = freq_sum['alignments']
cancer = freq_sum['cancer']
expression = freq_sum['expression']
genome = freq_sum['genome']


vol = np.arange(6,17)
file_len_sum =  filelen_sum['file_len']
plt.plot(vol, algorithm,label='algorithm')
plt.plot(vol, alignments,label='alignments')
plt.plot(vol, cancer,label='cancer')
plt.plot(vol, expression,label='expression')
plt.plot(vol, genome,label='genome')

plt.title('vol frequency', fontsize=20)
plt.xlabel('vol', fontsize=18)
plt.ylabel('freq', fontsize=18)
plt.xticks(vol, fontsize=15)
plt.legend()
plt.show()
```



전체적으로 봤을 때, genome 단어가 확실히 많이 등장하는 것을 볼 수 있다. 또한 cancer, expression 단어도 꽤 많이 등장한다. 반면에 alignments와 algorithm 단어는 상대적으로 적게 등장한다.

다만, 앞에서 살펴봤듯이 vol.6-9의 논문의 길이 자체가 길고, vol 11-16 논문의 길이는 짧았기 때문에 vol.6-9의 target 단어의 개수 또한 수적으로 클 수 밖에 없었다. (file_len 그래프와 분포양상이 비슷하다.) 따라서 vol별로 정확한 비교해보기위해, 각 논문의 길이를 고려하여 "특정단어빈도수/논문의길이"그래프를 그려보자

In [18]:
```
vol = np.arange(6,17)
file_len_sum =  filelen_sum['file_len']
plt.plot(vol, algorithm,label='algorithm')
plt.plot(vol, alignments,label='alignments')
plt.plot(vol, cancer,label='cancer')
plt.plot(vol, expression,label='expression')
plt.plot(vol, genome,label='genome')
plt.title('vol frequency_ratio', fontsize=20)
plt.xlabel('vol', fontsize=18)
plt.ylabel('freq_ratio', fontsize=18)
plt.xticks(vol, fontsize=15)
plt.legend()
plt.show()
```



cancer : vol.6-9,14에서는 다른 논문에 비해 등장하는 비율이 적고, vol.10 13 15에서는 다른 논문들에 비해 cancer 단어가 많이 등장하는 것을 확인할 수 있다.

algorithm : 다른 논문에 비해 vol.10, vol15 에서 등장하는 비율이 높은 것을 확인할 수 있다.

alignments : 모든 논문에서 굉장히 적게 등장했고, vol 6~9 에서는 아예 등장하지 않았다.

genome : 모든 논문에서 많이 등장했지만, 특히 vol6에서 두드러지는 것이 보인다.

expression :모든 논문에서 꽤 많이 등장하는데 vol.13에서 다른 논문에 비해 등장하는 비율이 높다.

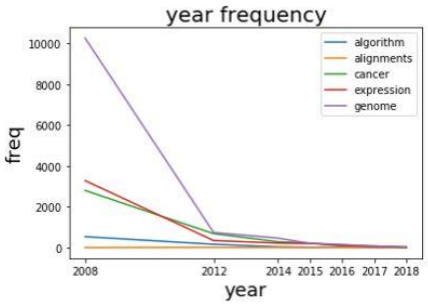
3. 연도별 단어빈도수

In [19]:
```
freq_sum_year = df_filelists.groupby('year').sum()
freq_sum_year.drop(columns='vol')
```

Out[19]:

year	file_len	expression	genome	cancer	algorithm	alignments
2008	2812311	3279.0	10255.0	2795.0	534.0	0.0
2012	245532	345.0	739.0	682.0	161.0	10.0
2014	136481	219.0	462.0	286.0	35.0	2.0
2015	70122	202.0	206.0	220.0	16.0	3.0
2016	87834	108.0	153.0	87.0	32.0	4.0
2017	27282	53.0	68.0	78.0	17.0	2.0
2018	10716	25.0	24.0	10.0	0.0	1.0

In [20]:
```
algorithm = freq_sum_year['algorithm']
alignments = freq_sum_year['alignments']
cancer = freq_sum_year['cancer']
expression = freq_sum_year['expression']
genome = freq_sum_year['genome']

year =[2008,2012,2014,2015,2016,2017,2018]
file_len_sum =  filelen_sum['file_len']
plt.plot(year, algorithm,label='algorithm')
plt.plot(year, alignments,label='alignments')
plt.plot(year, cancer,label='cancer')
plt.plot(year, expression,label='expression')
plt.plot(year, genome,label='genome')
plt.title('year frequency', fontsize=20)
plt.xlabel('year', fontsize=18)
plt.ylabel('freq', fontsize=18)
plt.xticks(year, fontsize=10)
plt.legend()
plt.show()
```

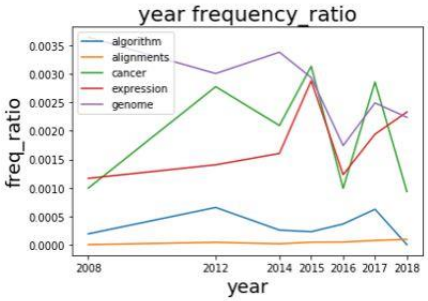


genome 단어가 확실히 많이 등장하는 것을 볼 수 있다. cancer, expression 단어도 꽤 많이 등장한다. algorithm, alignments는 적게 등장하는 것을 볼 수 있다.

연도별로 살펴볼 때에도 각 논문의 길이를 고려하여 "특정단어빈도수/논문의길이"그래프를 그려볼 수 있다.

In [21]:
```
year = [2008,2012,2014,2015,2016,2017,2018]
algorithm = freq_sum_year['algorithm']/freq_sum_year['file_len']
alignments = freq_sum_year['alignments']/freq_sum_year['file_len']
cancer = freq_sum_year['cancer']/freq_sum_year['file_len']
expression = freq_sum_year['expression']/freq_sum_year['file_len']
genome = freq_sum_year['genome']/freq_sum_year['file_len']
```

In [22]:
```
plt.plot(year, algorithm,label='algorithm')
plt.plot(year, alignments,label='alignments')
plt.plot(year, cancer,label='cancer')
plt.plot(year, expression,label='expression')
plt.plot(year, genome,label='genome')
plt.title('year frequency_ratio', fontsize=20)
plt.xlabel('year', fontsize=18)
plt.ylabel('freq_ratio', fontsize=18)
plt.xticks(year, fontsize=10)
plt.legend()
plt.show()
```



expression : 다른 연도에 비해 2015년, 2017년에 자주 등장한 것을 볼 수 있다.

cancer : 다른 연도에 비해 2012, 2015, 2017년에 자주 등장한 것을 볼 수 있다.

genome : 전체적으로 자주 등장했지만, 2016년에는 다른 연도에비해 상대적으로 덜 등장한 것을 볼 수 있다.

alignments : 전체적으로 적게 등장했다.

algorithm : 전체적으로 적게 등장했지만, 2012년과 2017년에는 다른 연도에 비해 상대적으로 자주 등장한 것을 볼 수 있다.