7/11

Date	
Property	
Weeks	Coursera 4번째 강의

Lab session: Introduction to the Uber Dataset

2주차 중간~

Lab session: Company Distances and Industry Distances

Company Distances and Industry Distances

In [58]:
```
#We can use euclidean distance to see how far away two companies are in terms of words
def findDist(company1,company2):
    return sum((company1-company2)**2)**.5
print(findDist(word_frequency['The_Bank_of_New_York_Mellon'],word_frequency['JPMorgan_Chase_%26_Co.']))
print(findDist(word_frequency['Facebook'],word_frequency['The_Bank_of_New_York_Mellon']))
```

```
0.069269383058396
0.09419186927592427
```

In [59]:
```
#We can use itertools to find the combinations
from  itertools import combinations
combinations = list(combinations(word_frequency.columns,2))
print(combinations)
```

```
[('3M', 'AT%26T'), ('3M', 'AbbVie_Inc.'), ('3M', 'Abbott_Laboratories'), ('3M', 'Accenture'), ('3M', 'Adobe_Inc.'), ('3M', 'Allergan'), ('3M', 'Allstate'), ('3M', 'Alphabet_Inc.'), ('3M', 'Altria'), ('3M', 'Amazon.com'), ('3M', 'American_Express'), ('3M', 'American_International_Group'), ('3M', 'Amgen'), ('3M', 'Apple_Inc.'), ('3M', 'Bank_of_America'), ('3M', 'Berkshire_Hathaway'), ('3M', 'Biogen'), ('3M', 'BlackRock'), ('3M', 'Boeing'), ('3M', 'Booking_Holdings'), ('3M', 'Bristol-Myers_Squibb'), ('3M', 'CVS_Health'), ('3M', 'Capital_One'), ('3M', 'Caterpillar_Inc.'), ('3M', 'Celgene'), ('3M', 'Charter_Communications'), ('3M', 'Chevron_Corporation'), ('3M', 'Cisco_Systems'), ('3M', 'Citigroup'), ('3M', 'Colgate-Palmolive'), ('3M', 'Comcast'), ('3M', 'ConocoPhillips'), ('3M', 'Costco_Wholesale_Corp.'), ('3M', 'Danaher_Corporation'), ('3M', 'DowDuPont'), ('3M', 'Dow_Inc.'), ('3M', 'Duke_Energy'), ('3M', 'Eli_Lilly_and_Company'), ('3M', 'Emerson_Electric'), ('3M', 'Exelon'), ('3M', 'ExxonMobil'), ('3M', 'Facebook'), ('3M', 'FedEx'), ('3M', 'Ford_Motor_Company'), ('3M', 'General_Dynamics'), ('3M', 'General_Electric'), ('3M', 'General_Motors'), ('3M', 'Gilead_Sciences'), ('3M', 'Goldman_Sachs'), ('3M', 'Home_Depot'), ('3M', 'Honeywell'), ('3M', 'IBM'), ('3M', 'Intel'), ('3M', 'JPMorgan_Chase_%26_Co.'), ('3M', 'Johnson_%26_Johnson'), ('3M', 'Kinder_Morgan'), ('3M', 'Kraft_Heinz'), ('3M', 'Lockheed_Martin'), ('3M', 'Lowe%27s'), ('3M', 'MasterCard'), ('3M', 'McDonald%27s'), ('3M', 'Medtronic'), ('3M', 'Merck_%26_Co.'), ('3M', 'MetLife'), ('3M', 'Microsoft'), ('3M', 'Mondel%C4%93z_International'), ('3M', 'Morgan_Stanley'), ('3M', 'Netflix'), ('3M', 'NextEra_Energy'), ('3M', 'Nike,_Inc.'), ('3M', 'Nvidia'), ('3M', 'Occidental_Petroleum'), ('3M', 'Oracle_Corporation'), ('3M', 'PayPal'), ('3M', 'PepsiCo'), ('3M', 'Pfizer_Inc'), ('3M', 'Philip_Morris_International'), ('3M', 'Procter_%26_Gamble'), ('3M', 'Qualcomm'), ('3M', 'Raytheon'), ('3M', 'Schlumberger'), ('3M', 'Simon_Property_Group'), ('3M', 'Southern_Company'), ('3M', 'Starbucks'), ('3M', 'Target_Corporation'), ('3M', 'Texas_Instruments'), ('3M', 'The_Bank_of_New_York_Mellon'), ('3M', 'The_Coca-Cola_Company'), ('3M', 'The_Walt_Disney_Company'), ('3M', 'U.S._Bancorp'), ('3M', 'Union_Pacific_Corporation'), ('3M', 'UnitedHealth_Group'), ('3M', 'United_Parcel_Service'), ('3M', 'United_Technologies'), ('3M', 'Verizon_Communications'), ('3M', 'Visa_Inc.'), ('3M', 'Walgreens_Boots_Alliance'), ('3M', 'Walmart'), ('3M',
```

similarity 구하는 방법 : 거리를 구해서 비교해본다.

BNY Mellon과 JPMorgan 간 거리가 BNY Mellon 과 Facebook 거리보다 가깝다.

(비슷한 업계니까)

ex) 3M 과 다른 기업 간 text에서 찾을 수 있는 combination 개수를 찾는다.

In [60]:
```python
#Create the distance dataframe
distance = pd.DataFrame(combinations)
distance.columns = ["Company 1", "Company 2"]
#Create the distance for each combination
distance["Distance"] = distance.apply(lambda x: findDist(word_frequency[x["Company 1"]], word_frequency[x["Company 2"]]), axis=1)
print(distance)
```

```
                         Company 1                   Company 2  Distance
0                               3M                       AT%26T  0.067785
1                               3M                   AbbVie_Inc.  0.086907
2                               3M          Abbott_Laboratories  0.082096
3                               3M                    Accenture  0.099816
4                               3M                   Adobe_Inc.  0.098538
5                               3M                     Allergan  0.091710
6                               3M                     Allstate  0.089795
7                               3M                Alphabet_Inc.  0.099336
8                               3M                       Altria  0.095903
9                               3M                   Amazon.com  0.084428
10                              3M             American_Express  0.098040
11                              3M  American_International_Group  0.087494
12                              3M                        Amgen  0.106996
13                              3M                   Apple_Inc.  0.084011
14                              3M              Bank_of_America  0.093553
15                              3M           Berkshire_Hathaway  0.079356
16                              3M                      Biogen  0.086669
17                              3M                    BlackRock  0.099466
18                              3M                       Boeing  0.091513
19                              3M             Booking_Holdings  0.107829
20                              3M        Bristol-Myers_Squibb  0.077146
21                              3M                   CVS_Health  0.086509
22                              3M                  Capital_One  0.093070
23                              3M            Caterpillar_Inc.  0.075362
24                              3M                      Celgene  0.087456
25                              3M       Charter_Communications  0.087377
26                              3M          Chevron_Corporation  0.079814
27                              3M                Cisco_Systems  0.089222
```

big data → sorting 필요하다

```python
#Turn it into a function
def get_company_industries(urls):
    industries_data = []
    for url in urls:
        r = requests.get(url)
        soup = BeautifulSoup(r.content, 'html.parser')
        infobox = soup.find("table", {"class": "infobox"})
        industries = [x.text for x in infobox.find("th", text = "Industry").parent()[1].find_all('a')]
        industries_data.append(industries)
    return industries_data
print(get_company_industries(links_unique[:5]))

#Instead of an array, let's modify to get a dataframe of dummy variables representing what industries each company is tagged with
def get_company_industries(urls):
    industries_data = []
    for url in urls:
        r = requests.get(url)
        soup = BeautifulSoup(r.content, 'html.parser')
        infobox = soup.find("table", {"class": "infobox"})
        industries = [x.text for x in infobox.find("th", text = "Industry").parent()[1].find_all('a')]
        industries = pd.Series(1, index=industries)
        industries_data.append(industries)
    industries_data = pd.concat(industries_data,axis=1,sort=False).fillna(0)  ## 행방향
    return industries_data
print(get_company_industries(links_unique[:5]))


#And clean up with transposing and putting in the index of tickers
def get_company_industries(urls):
    industries_data = []
    for url in urls:
        r = requests.get(url)
        soup = BeautifulSoup(r.content, 'html.parser')
        infobox = soup.find("table", {"class": "infobox"})
        industries = [x.text for x in infobox.find("th", text = "Industry").parent()[1].find_all('a')]
        industries = pd.Series(1, index=industries)
        industries_data.append(industries)
    industries_data = pd.concat(industries_data,axis=1,sort=False).fillna(0)
    return industries_data
industries = get_company_industries(links_unique)
industries = industries.transpose()
```

```
industries.index = index
print(industries)
```

```
                              Conglomerate  Telecommunications  Technology  ₩
3M                                     1.0                 0.0         0.0
AT%26T                                 0.0                 1.0         1.0
AbbVie_Inc.                            0.0                 0.0         0.0
Abbott_Laboratories                    0.0                 0.0         0.0
Accenture                              0.0                 0.0         0.0
Adobe_Inc.                             0.0                 0.0         0.0
Allergan                               0.0                 0.0         0.0
Allstate                               0.0                 0.0         0.0
Alphabet_Inc.                          1.0                 0.0         0.0
Altria                                 0.0                 0.0         0.0
Amazon.com                             0.0                 0.0         0.0
American_Express                       0.0                 0.0         0.0
American_International_Group           0.0                 0.0         0.0
Amgen                                  0.0                 0.0         0.0
Apple_Inc.                             0.0                 0.0         0.0
Bank_of_America                        0.0                 0.0         0.0
Berkshire_Hathaway                     1.0                 0.0         0.0
Biogen                                 0.0                 0.0         0.0
```

```
#Let's see which companies are in financial services
fin_services = industries[industries['financial services'] == 1].index
print(fin_services)
```

```
Index(['American_Express', 'American_International_Group', 'Bank_of_America',
       'Capital_One', 'Caterpillar_Inc.', 'Citigroup', 'Goldman_Sachs',
       'JPMorgan_Chase_%26_Co.', 'MasterCard', 'MetLife', 'Morgan_Stanley',
       'PayPal', 'The_Bank_of_New_York_Mellon', 'U.S._Bancorp', 'Visa_Inc.',
       'Wells_Fargo'],
      dtype='object')
```

In [78]:

```
#Let's check how similar companies are within and outside of the financials industry
print(distance.loc[fin_services_index]["Distance"].mean())
print(distance.loc[fin_services_index2]["Distance"].mean())
```

```
0.08921896425997317
0.099163444188783
```

In [79]:

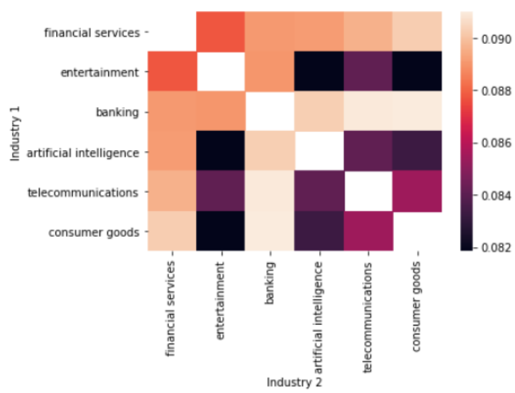
```
#And check how different industries line up
#First create the base of the dataframe, each combination of industry
from  itertools import combinations
industry_distances = pd.DataFrame(list(combinations(industries.columns,2)))
industry_distances.columns = ["Industry 1", "Industry 2"]
print(industry_distances)
```

```
      Industry 1             Industry 2
0     conglomerate     telecommunications
1     conglomerate             technology
2     conglomerate             mass media
3     conglomerate          entertainment
4     conglomerate            health care
5     conglomerate      computer software
6     conglomerate              insurance
7     conglomerate                tobacco
8     conglomerate        cloud computing
9     conglomerate  artificial intelligence
10    conglomerate   consumer electronics
11    conglomerate   digital distribution
12    conglomerate                banking
13    conglomerate     financial services
14    conglomerate          biotechnology
15    conglomerate      computer hardware
16    conglomerate         semiconductors
```

In [86]:
```python
#What about the most similar to financial technology?
print(industry_distances[industry_distances["Industry 1"] == 'financial services'].sort_values(by='Distance').dropna())
```

```
             Industry 1              Industry 2  Distance
123  financial services           entertainment  0.087768
312  financial services                 banking  0.089098
258  financial services  artificial intelligence  0.089152
42   financial services      telecommunications  0.089627
337  financial services          consumer goods  0.090286
342  financial services          pharmaceutical  0.090317
336  financial services             oil and gas  0.090589
295  financial services    digital distribution  0.091025
97   financial services              mass media  0.091230
334  financial services         pharmaceuticals  0.091732
332  financial services               aerospace  0.092209
333  financial services                 defense  0.092209
238  financial services         cloud computing  0.092336
172  financial services       computer software  0.092457
148  financial services             health care  0.092548
277  financial services    consumer electronics  0.092594
195  financial services               insurance  0.092609
340  financial services              automotive  0.092721
341  financial services       medical equipment  0.093429
331  financial services          semiconductors  0.093476
335  financial services                  retail  0.093893
330  financial services       computer hardware  0.094054
344  financial services             video games  0.094163
```

In [90]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
#Plot the heatmap
sns.heatmap(pivot_data)
plt.show()
```



Application: applying similarity analysis on corporate filings to predict returns

텍스트 분석으로 주식 수익률 계산?

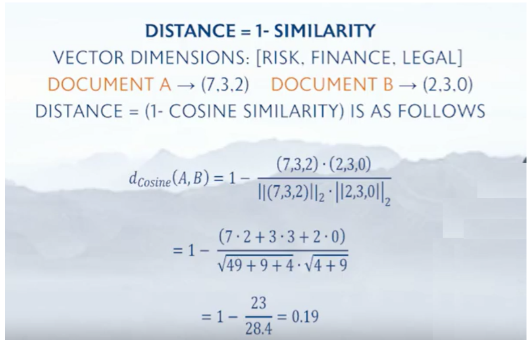
Q. 분기별/연도별 보고서에서 텍스트가 바뀌는 것이 실제로 회사의 변화를 의미하는가

A.

1. 보고서들의 유사성을 확인 (유클리디안 대신 cosine similarity 이용한다.)

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

where A_i and B_i are components of vector A and B respectively.



2. 결론 (a paper by Cohen, Malloy, and Nguyen 참조)

- 올해와 전분기 사이에 전혀 변동이 없는 포트폴리오를 사서 작년과 가장 큰 변동폭을 보인 1분기에 주식을 팔면 실제로 상당한 수익률을 보인다.
- 보고서의 법적인 부분의 변화가 제일 큰 영향을 끼친다.

Lab session: Working with 10-K Data