In [58]:
```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from os import listdir
from os.path import isfile, join
from sklearn.feature_extraction.text import CountVectorizer
from itertools import chain
%matplotlib inline
```

Part A

Question 1 - Group Info

Group Name: Plum Member: Eric Grant

Question 2 - Text Classification

(a)

Count of documents that contain X and belong to given class + 1 / count of documents that belong to given class + 2

i. 3/4

ii. 1/2

iii. 2/3

(b)

Count occurences of X in documents belonging to given class + 1 / count of total words used in all documents belonging to given class + total unique words

i. 5/28

ii. 1/14

iii. 1/11

(c)

P(Class = 1 | d5)

=

P(Class = 1) *

P(X_daffodil | Class = 1) *

P(X_crocus | Class = 1) *

P(X_daisy | Class = 1) *

P(X_tulip | Class = 1) *

P(X_clematis | Class = 1) *

P(X_peony | Class = 1)

=

$1/4 * (1/3)^3 * (2/3)^3$

=

0.0027

P(Class = 2 | d5)

=

$2/4 * (2/4)^2 * (1/4)^2 * (3/4)^2$

=

0.0044

P(Class = 3 | d5)

= 0 0 1 1 0 0

$1/4 * (1/3)^4 * (2/3)^2$

=

0.0014

We would predict that the test document is most likely to be in class 2

(d)

P(Class = 1 | d5)

=

$1/4 * (1/22)^3 * (2/22)^3$

=

$$1.76 * 10^{-8}$$

P(Class = 2 | d5)

=

$2/4 * (2/28)^2 * (1/28)^2 * (5/28)^2$

=

$$1.04 * 10^{-7}$$

P(Class = 3 | d5)

=

$1/4 * (1/21)^4 * (2/21) * (3/21)$

=

$$1.75 * 10^{-8}$$

We would predict that the test document is most likely to be in class 2

Question 3 - Text Mining

(a)

	d1	d2	d3
cat	3	0	1
bat	1	3	0
rat	1	1	1
fat	1	0	1
mat	0	1	1
pat	0	1	1
sat	0	0	1

(b)

	d1	d2	d3

	d1	d2	d3
cat	0.11	0	0.05
bat	0.05	0.11	0
rat	0	0	0
fat	0.05	0	0.05
mat	0	0.05	0.05
pat	0	0.05	0.05
sat	0	0	0.144

(c)

Sat + d3

Part B

Question 4 - College Data

(a)

In [2]:

```
dataOrig = pd.read_csv("./college_data.csv")
scaler = StandardScaler()
scaler.fit(dataOrig.iloc[:,3:21])
dataScaled = scaler.transform(dataOrig.iloc[:,3:21])
pca = PCA()
pca.fit(dataScaled)
pc = pd.DataFrame(pca.components_)
pc.columns = list(dataOrig.iloc[:,3:21].columns)
print("principal components")
display(pc)
print("\nsingular values")
print(pca.singular_values_)
print("\nexplained variance ratio")
print(pca.explained_variance_ratio_)
```

```
principal components
```

	Early Career Pay	Mid-Career Pay	Total price for in-district students living on campus 2015-16 (DRVIC2015)	Professors (S2014_SIS_RV With faculty status tenured)	Associate professors (S2014_SIS_RV With faculty status tenured)	Assistant professors (S2014_SIS_RV With faculty status on tenure track)	Average sal equated t months of f time instructio staff - profess (DRVHR2014_F
0	0.234252	0.154926	0.260735	0.223983	0.098181	0.204502	0.2916
1	-0.151252	-0.033391	-0.140536	0.365842	0.486784	0.340686	0.0022
2	-0.285380	-0.277861	0.061869	-0.003917	0.254142	0.219472	0.0443
3	-0.101911	-0.729584	0.252605	-0.016479	-0.128797	0.097640	-0.0381
4	0.219064	-0.406274	-0.249791	0.026849	0.099618	-0.165262	-0.2920

	Early Career Pay	Mid-Career Pay	Total price for in-district students living on campus 2015-16 (DRVIC2015)	Professors (S2014_SIS_RV With faculty status tenured)	Associate professors (S2014_SIS_RV With faculty status tenured)	Assistant professors (S2014_SIS_RV With faculty status on tenure track)	Average sal equated t months of fu time instructio staff - professo (DRVHR2014_F
5	-0.519585	0.324018	-0.233804	-0.066187	0.064639	0.127362	-0.132(
6	-0.363057	-0.104523	0.080913	-0.014434	-0.082103	-0.428552	0.175
7	0.214501	-0.041716	0.103240	-0.283795	-0.104630	0.270311	-0.166
8	-0.077971	0.196979	0.689844	-0.310179	0.329320	-0.167927	-0.149
9	-0.424164	-0.048141	0.231122	0.049559	-0.353480	0.461022	-0.097
10	0.104211	-0.092886	-0.086032	-0.081326	0.520413	0.116326	-0.108
11	0.275724	0.106190	0.304658	0.503050	-0.137202	0.186474	-0.187
12	0.158095	0.123996	-0.121595	-0.290002	-0.255124	0.229517	-0.331
13	0.086325	0.019834	-0.124460	-0.262070	0.088029	0.192834	-0.106
14	0.121795	-0.062216	0.086049	-0.437654	0.148996	0.203035	0.341
15	-0.120051	0.042374	0.031090	-0.020408	0.103899	-0.068538	-0.567
16	0.007712	0.000462	0.205067	0.157631	0.107465	-0.242218	-0.308
17	-0.003409	-0.014631	0.012022	-0.043903	-0.000052	0.025581	0.127

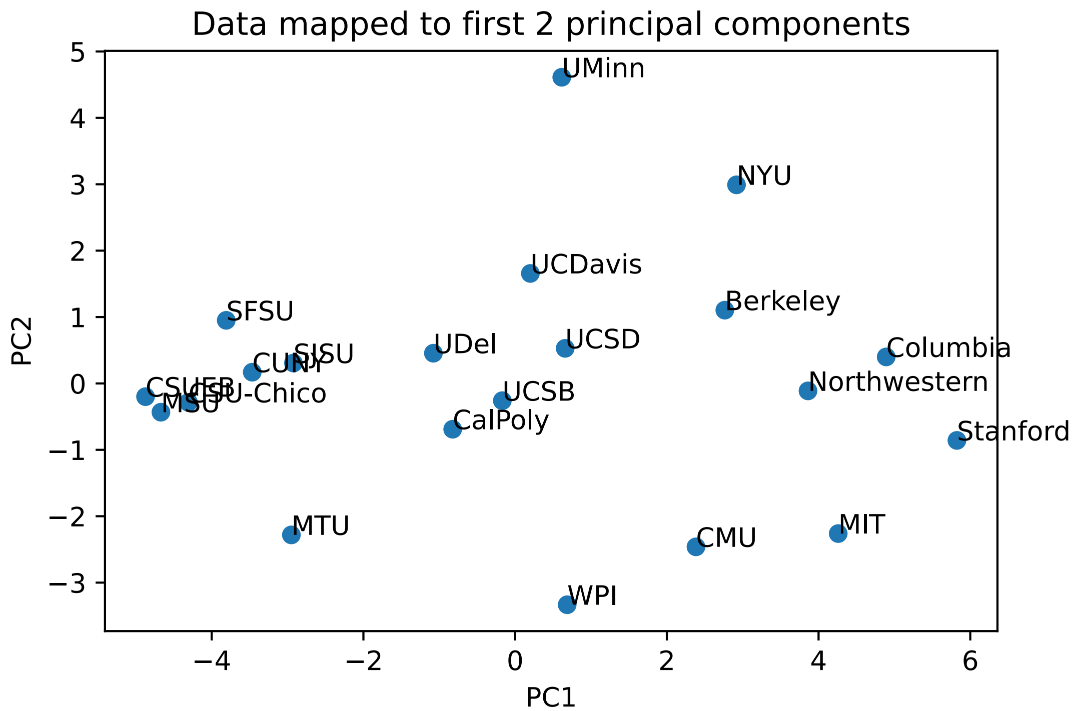
```
singular values
[14.95856579  8.0582619   5.3473243   4.30120268  3.7020738   3.09316575
  2.43355971  2.09857736  1.79023549  1.41081355  1.18186106  1.02158233
  0.77631902  0.41325291  0.32896467  0.27234371  0.1362472   0.04552867]

explained variance ratio
[5.91954208e-01 1.71787262e-01 7.56451778e-02 4.89427102e-02
 3.62575408e-02 2.53113078e-02 1.56672298e-02 1.16508649e-02
 8.47868550e-03 5.26559491e-03 3.69522639e-03 2.76092713e-03
 1.59436832e-03 4.51793562e-04 2.86290358e-04 1.96219825e-04
 4.91092564e-05 5.48375611e-06]
```

(b)

In [3]:

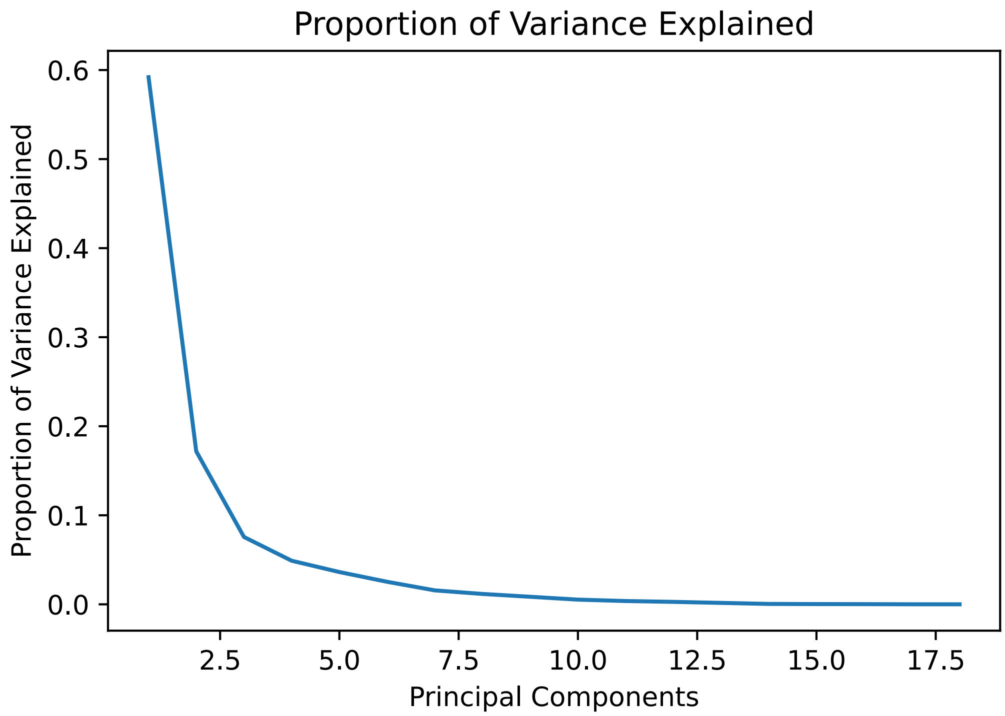
```
n = list(dataOrig["ShortHandName"])
X = pca.transform(dataScaled)
Xnew = pd.DataFrame(X)
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Data mapped to first 2 principal components')
for i, txt in enumerate(n):
    ax.annotate(txt, (X[:,0][i], X[:,1][i]))
```

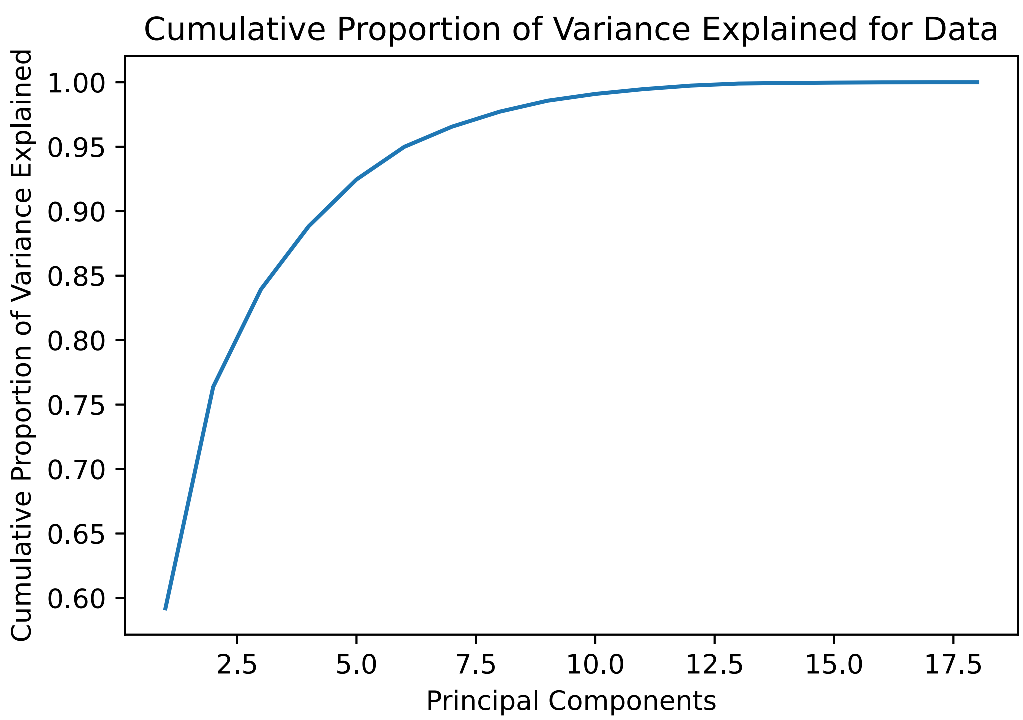


(c)

I would consider 12 principal components for future analysis

In [4]:
```
plt.plot(
    [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18],
    pca.explained_variance_ratio_)
plt.xlabel("Principal Components")
plt.ylabel("Proportion of Variance Explained")
plt.title("Proportion of Variance Explained")
plt.show()
plt.plot(
    [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18],
    pca.explained_variance_ratio_.cumsum())
plt.xlabel("Principal Components")
plt.ylabel("Cumulative Proportion of Variance Explained")
plt.title('Cumulative Proportion of Variance Explained for Data')
plt.show()
```





Question 5 - Stock Data

(a)

In [5]:

```
dataOrig = pd.read_csv("./stock_data_2020.csv")

#point for each date
dataScaled = dataOrig.copy()
scaler = StandardScaler()
scaler.fit(dataOrig.iloc[:,1:31])
dataScaled.loc[:, dataScaled.columns != 'Date'] = scaler.transform(dataOrig.iloc
```

```
pca = PCA()
pca.fit(dataScaled.loc[:, dataScaled.columns != 'Date'].to_numpy())
pc = pd.DataFrame(pca.components_)
pc.columns = list(dataOrig.iloc[:,1:31].columns)
print("principal components")
display(pc)
print("\nsingular values")
print(pca.singular_values_)
print("\nexplained variance ratio")
print(pca.explained_variance_ratio_)

#point for each stock
dataPivot = dataScaled.T
dataPivot.columns = list(dataOrig["Date"])
dataPivot = dataPivot.drop('Date')
pcap = PCA()
pcap.fit(dataPivot)
pc = pd.DataFrame(pcap.components_)
pc.columns = list(dataOrig["Date"])

print("principal components")
display(pc)
print("\nsingular values")
print(pcap.singular_values_)
print("\nexplained variance ratio")
print(pcap.explained_variance_ratio_)
```

principal components

	AAPL	AXP	BA	CAT	CSCO	CVX	DIS	DOW	GS
0	-0.162530	-0.213868	-0.159864	-0.213499	-0.104524	-0.120201	-0.232664	-0.241925	-0.237166
1	-0.243507	0.146509	0.236340	-0.142659	0.189519	0.268344	-0.006453	-0.048709	0.075507
2	-0.018969	0.173027	0.180206	0.168395	-0.409761	-0.182176	0.096782	0.089968	0.002097
3	0.091415	0.089807	0.077463	0.073059	0.332673	0.046111	0.139562	0.084602	0.203361
4	-0.020003	-0.075304	-0.110102	0.146311	0.210967	0.109763	0.235380	0.012347	0.099259
5	0.127064	-0.102560	0.023972	-0.028166	0.187763	0.088970	-0.081855	0.150950	-0.029447
6	0.224395	-0.012168	0.117998	-0.219951	0.104471	0.020369	0.136892	-0.202731	0.007539
7	0.173531	0.043077	-0.005270	0.068990	0.208040	-0.291597	0.334460	0.073826	0.137076
8	-0.092053	-0.103789	0.012253	0.044130	0.124469	-0.026683	-0.376074	-0.083936	-0.024121
9	0.015869	0.096466	-0.102702	-0.235320	-0.360760	0.242106	0.193614	-0.116732	0.046328
10	0.085569	-0.287011	0.058074	0.023893	-0.180541	-0.107231	0.193295	-0.103850	0.189746
11	-0.202894	-0.172415	-0.067146	0.067040	0.054478	0.315243	-0.034094	0.159390	0.072782
12	0.033992	0.095838	-0.185901	-0.076704	-0.067898	-0.088363	0.029961	0.001389	-0.002501
13	0.040989	-0.004747	0.093105	0.037356	-0.210719	0.208558	-0.192988	0.270465	-0.019459
14	-0.187010	0.103976	-0.176249	0.015971	0.367749	-0.314191	-0.092452	-0.143783	0.048747
15	0.188588	0.062085	0.172932	-0.016323	-0.081072	-0.073006	-0.254736	-0.149142	0.212790
16	-0.155359	0.064047	0.151496	-0.072209	0.348968	0.106368	0.075294	0.045643	-0.128287
17	0.163195	0.241084	0.031590	-0.375197	0.030786	-0.010387	-0.200798	-0.248922	0.485839
18	0.214591	-0.128636	0.171582	-0.159069	0.041822	0.170678	0.380996	-0.397601	-0.248563

	AAPL	AXP	BA	CAT	CSCO	CVX	DIS	DOW	GS
19	0.012322	0.025121	-0.479425	-0.136355	-0.043737	-0.101736	0.213345	0.079255	0.172916
20	0.028686	0.057755	-0.224114	0.184692	0.015202	0.181430	0.099506	-0.272362	0.022427
21	0.001873	-0.227409	0.125468	-0.301380	-0.008183	-0.036628	-0.092048	0.166167	0.356352
22	-0.120646	-0.304833	-0.308985	-0.221940	0.024685	0.012868	-0.107689	-0.106577	0.115128
23	-0.098485	-0.202075	0.005227	-0.334317	-0.025630	-0.299243	0.235578	0.448616	-0.030800
24	-0.051828	-0.016496	-0.016989	-0.237108	-0.044757	-0.051720	0.061125	-0.112591	-0.310119
25	-0.331106	-0.106630	-0.209611	-0.040871	0.029529	0.201701	0.092047	-0.054125	0.005288
26	0.016366	-0.545074	0.179448	0.324413	0.006840	-0.255809	-0.031579	-0.262856	0.096795
27	0.462059	-0.006815	-0.085333	-0.213221	0.186632	-0.096220	-0.216689	0.144815	-0.416416
28	0.472876	-0.149504	-0.376631	0.209539	0.017722	0.279381	-0.104071	0.117587	0.086496
29	0.078903	-0.348125	0.245809	-0.162764	0.050765	0.276197	0.046148	0.138236	0.052223

30 rows × 30 columns

```
singular values
[62.58332424 47.78159344 21.05451808 15.70104973 13.95571107 10.31633121
  8.25470069  7.18045997  6.70052029  5.98951703  5.44956696  4.80423941
  4.64066922  3.7843043   3.64036563  3.29623688  2.86641655  2.78782939
  2.77924568  2.47483302  2.41025947  1.97917258  1.91486079  1.783583
  1.50814041  1.45519149  1.23042049  1.19284067  1.17062859  0.95320411]

explained variance ratio
[5.18078369e-01 3.01994798e-01 5.86366047e-02 3.26088575e-02
 2.57621523e-02 1.40776045e-02 9.01323856e-03 6.81997427e-03
 5.93875293e-03 4.74527966e-03 3.92827778e-03 3.05300481e-03
 2.84865222e-03 1.89430675e-03 1.75294469e-03 1.43719280e-03
 1.08681797e-03 1.02804137e-03 1.02172044e-03 8.10158529e-04
 7.68432633e-04 5.18138109e-04 4.85012148e-04 4.20789459e-04
 3.00858134e-04 2.80103475e-04 2.00255897e-04 1.88210166e-04
 1.81266043e-04 1.20184930e-04]
principal components
```

	1/2/20	1/3/20	1/6/20	1/7/20	1/8/20	1/9/20	1/10/20	1/13/20	1/14/20
0	0.114573	0.115087	0.115403	0.113877	0.105315	0.104574	0.102510	0.104552	0.104014
1	-0.084707	-0.088219	-0.087742	-0.088352	-0.092950	-0.094888	-0.092950	-0.092254	-0.093606
2	0.086500	0.073675	0.072965	0.055811	0.056401	0.051765	0.064405	0.061314	0.070719
3	0.022492	0.014442	0.012387	0.023720	0.047911	0.047683	0.050507	0.055872	0.059118
4	0.109884	0.109505	0.108165	0.102999	0.078392	0.054497	0.057107	0.051369	0.062209
5	0.162253	0.163341	0.174698	0.145313	0.129835	0.126410	0.119664	0.122856	0.114885
6	0.062613	0.054060	0.049090	0.041730	0.070929	0.064645	0.059199	0.059495	0.059496
7	-0.022734	0.002734	0.010900	0.016212	0.036366	0.039097	0.048613	0.010689	0.024442
8	-0.030552	-0.013796	-0.017027	-0.000098	0.000744	0.013784	0.035382	0.046076	0.053066
9	-0.005787	-0.020034	-0.018556	-0.008164	0.010772	-0.003393	-0.010236	0.012471	-0.004264
10	0.001754	-0.011312	-0.030801	-0.062047	-0.049944	-0.101388	-0.084708	-0.076563	-0.090869
11	-0.008987	-0.008092	-0.005470	0.056976	0.039397	0.005819	0.013229	0.004191	0.013447

	1/2/20	1/3/20	1/6/20	1/7/20	1/8/20	1/9/20	1/10/20	1/13/20	1/14/20
12	0.066885	0.070415	0.053335	0.094292	0.079981	0.058898	0.028340	0.034323	0.002409
13	-0.056723	-0.060538	-0.070123	-0.055461	-0.042607	0.020914	0.016947	0.072118	0.082714
14	0.028637	0.028592	0.032673	0.003816	-0.006828	0.027408	0.018912	-0.022521	-0.024218
15	0.087682	0.075810	0.074558	0.065917	0.056450	0.073941	0.064693	0.024080	0.036745
16	-0.030312	-0.019665	-0.040831	-0.079144	-0.047584	-0.037155	-0.022564	-0.020298	-0.030388
17	-0.106979	-0.094519	-0.074449	-0.062324	-0.002873	0.059602	0.057129	0.043192	0.035919
18	-0.014079	0.011629	-0.001774	0.041938	0.045751	0.054173	0.057369	0.049749	0.033224
19	-0.011714	-0.032637	-0.038783	-0.095026	-0.020206	0.024212	0.065282	0.072862	0.032070
20	-0.050160	-0.044456	-0.030401	-0.040016	-0.091958	-0.031629	-0.009389	0.057277	0.031989
21	-0.051996	-0.042672	-0.015685	-0.028291	-0.032073	-0.030481	-0.041242	0.000309	0.015747
22	0.012424	0.016460	0.008281	-0.025032	-0.037641	0.016762	-0.015078	-0.002532	0.010193
23	-0.004634	0.023773	-0.017302	-0.009913	-0.001131	0.039429	0.015092	0.006417	0.029097
24	-0.031835	-0.031465	-0.072899	-0.116606	-0.027529	-0.050412	-0.058877	-0.011144	-0.048123
25	-0.118674	-0.125273	-0.115141	-0.074171	0.025971	0.047675	0.052044	0.025503	-0.015168
26	-0.031800	-0.013552	-0.042233	-0.046612	-0.066118	-0.013149	-0.019260	0.053858	0.036292
27	0.121097	0.027504	0.011227	-0.037865	0.034323	0.033020	0.006518	0.028710	0.096963
28	-0.105824	-0.081769	-0.072086	-0.055244	-0.044719	0.049726	0.052347	0.024340	0.004605
29	-0.124855	0.163791	0.101366	0.029317	-0.360608	0.000500	0.080790	-0.090596	-0.046509

30 rows × 252 columns

```
singular values
[4.82641991e+01 2.36757507e+01 1.59584555e+01 1.48485011e+01
 1.38198841e+01 1.02433544e+01 8.11700365e+00 7.17653956e+00
 6.33817229e+00 5.79700974e+00 5.43916394e+00 4.64807042e+00
 4.32776975e+00 3.74888227e+00 3.32272668e+00 3.17776516e+00
 2.85218279e+00 2.78753327e+00 2.48102629e+00 2.41062664e+00
 2.05982116e+00 1.92838661e+00 1.85179514e+00 1.52105130e+00
 1.46506781e+00 1.44810472e+00 1.22822876e+00 1.18001516e+00
 1.04957734e+00 7.71013670e-15]

explained variance ratio
[5.81320603e-01 1.39885605e-01 6.35546338e-02 5.50212861e-02
 4.76622247e-02 2.61848389e-02 1.64420888e-02 1.28527386e-02
 1.00252125e-02 8.38636128e-03 7.38294792e-03 5.39151611e-03
 4.67405481e-03 3.50726928e-03 2.75521025e-03 2.52004970e-03
 2.03011303e-03 1.93912420e-03 1.53613095e-03 1.45019168e-03
 1.05882585e-03 9.28012279e-04 8.55758830e-04 5.77368691e-04
 5.35649818e-04 5.23317723e-04 3.76464502e-04 3.47488695e-04
 2.74912444e-04 1.48350722e-32]
```

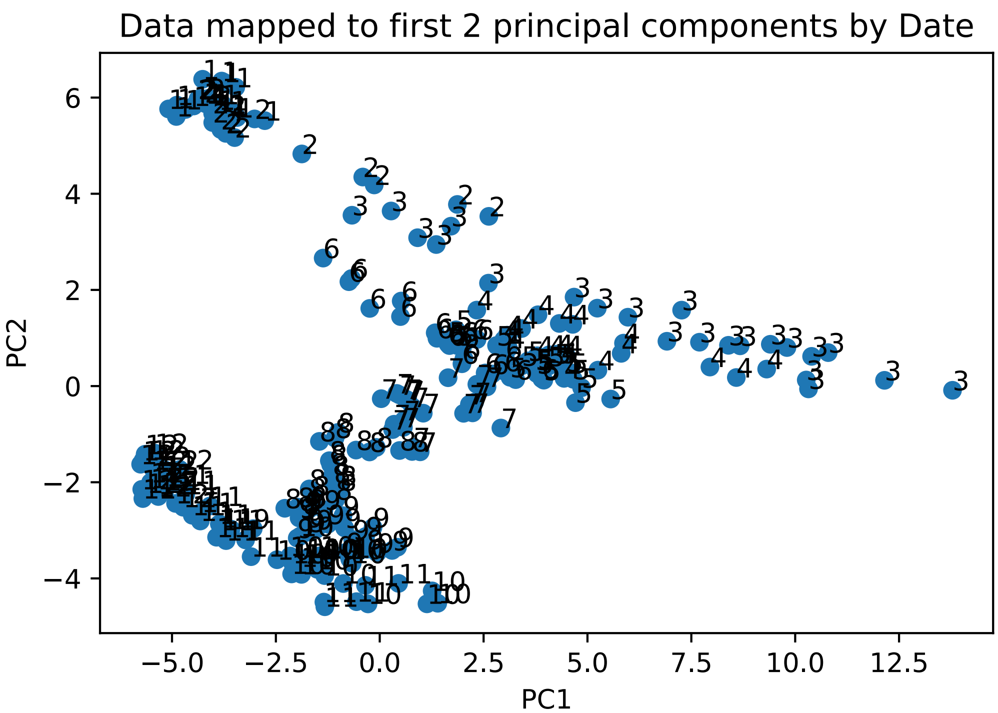
(b)

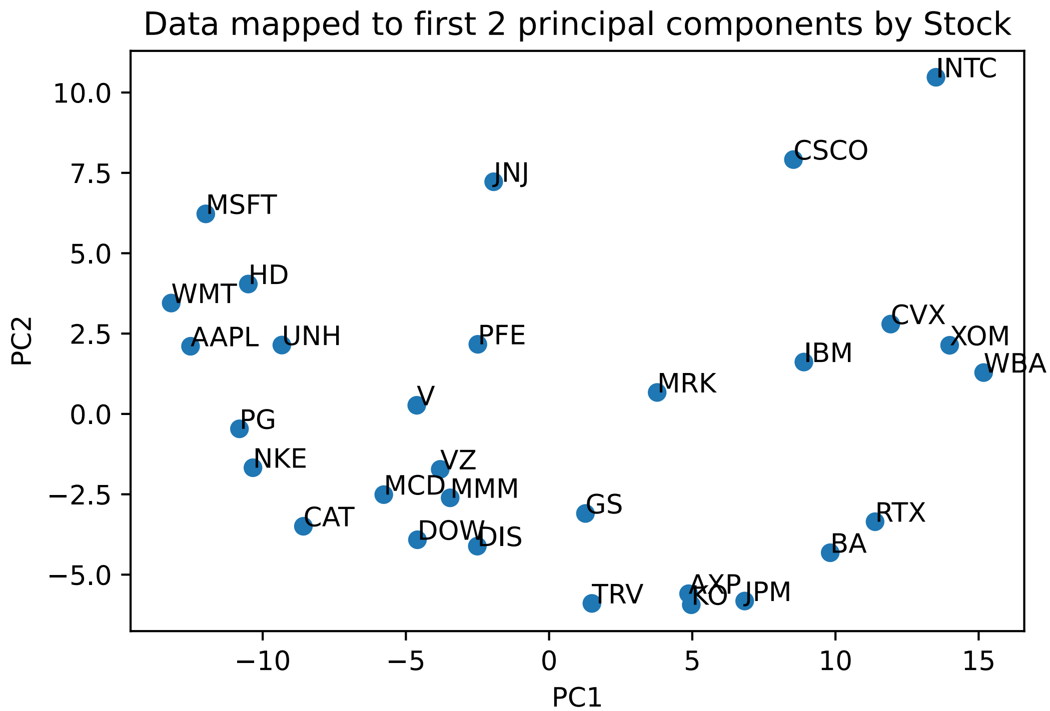
In [6]:

```
#point for each date
dates = []
for date in list(dataOrig["Date"]):
    dates.append(date.split('/')[0])
```

```
X = pca.transform(dataScaled.loc[:, dataScaled.columns != 'Date'].to_numpy())
Xnew = pd.DataFrame(X)
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Data mapped to first 2 principal components by Date')
for i, txt in enumerate(dates):
    ax.annotate(txt, (X[:,0][i], X[:,1][i]))
plt.show()

#point for each stock
n = list(dataOrig.iloc[:,1:31].columns)
X = pcap.transform(dataPivot.to_numpy())
Xnew = pd.DataFrame(X)
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Data mapped to first 2 principal components by Stock')
for i, txt in enumerate(n):
    ax.annotate(txt, (X[:,0][i], X[:,1][i]))
plt.show()
```





(c)

The majority of stocks are in the lower right quardrent with a few scattering to the top right.

At the start of the year most stocks began in the top left and traveled down right then down left setteling in a cluster at the bottom left.

(d)

In [77]:

```
dataOrig = pd.read_csv("./stock_data_2019.csv")

#point for each date
dataScaled = dataOrig.copy()
scaler = StandardScaler()
scaler.fit(dataOrig.iloc[:,1:31])
dataScaled.loc[:, dataScaled.columns != 'Date'] = scaler.transform(dataOrig.iloc
pca = PCA()
pca.fit(dataScaled.loc[:, dataScaled.columns != 'Date'].to_numpy())
pc = pd.DataFrame(pca.components_)
pc.columns = list(dataOrig.iloc[:,1:31].columns)
print("principal components")
display(pc)
print("\nsingular values")
print(pca.singular_values_)
print("\nexplained variance ratio")
print(pca.explained_variance_ratio_)

#point for each stock
dataPivot = dataScaled.T
dataPivot.columns = list(dataOrig["Date"])
dataPivot = dataPivot.drop('Date')
pcap = PCA()
pcap.fit(dataPivot)
pc = pd.DataFrame(pcap.components_)
```

```
pc.columns = list(dataOrig["Date"])

print("principal components")
display(pc)
print("\nsingular values")
print(pcap.singular_values_)
print("\nexplained variance ratio")
print(pcap.explained_variance_ratio_)

#point for each date
dates = []
for date in list(dataOrig["Date"]):
    dates.append(date.split('/')[0])
X = pca.transform(dataScaled.loc[:, dataScaled.columns != 'Date'].to_numpy())
Xnew = pd.DataFrame(X)
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Data mapped to first 2 principal components by Date')
for i, txt in enumerate(dates):
    ax.annotate(txt, (X[:,0][i], X[:,1][i]))
plt.show()

#point for each stock
n = list(dataOrig.iloc[:,1:31].columns)
X = pcap.transform(dataPivot.to_numpy())
Xnew = pd.DataFrame(X)
fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('Data mapped to first 2 principal components by Stock')
for i, txt in enumerate(n):
    ax.annotate(txt, (X[:,0][i], X[:,1][i]))
plt.show()
```

principal components

	AAPL	AXP	BA	CAT	CSCO	CVX	DIS	DD
0	-0.239378	-2.230095e-01	1.136913e-01	-4.271602e-02	2.132816e-02	-3.904224e-02	-2.310544e-01	2.060236e-01
1	-0.120297	-3.437179e-02	-1.672058e-01	-3.680292e-01	-1.445037e-01	-2.395874e-01	-1.330925e-02	-1.230730e-01
2	0.144383	-2.175893e-01	-3.075029e-02	1.718042e-01	-4.259425e-01	-2.945722e-01	-9.366280e-02	-7.074922e-02
3	0.008162	-1.089321e-01	3.478037e-01	3.314274e-02	3.075756e-03	1.329467e-01	-1.713152e-01	2.073256e-01
4	-0.063124	1.771221e-01	-8.086534e-02	1.429941e-01	4.942066e-02	1.889766e-01	1.423651e-01	8.832845e-02
5	0.057306	1.559226e-02	-6.359346e-01	2.540918e-01	1.871143e-01	-2.296807e-02	1.343679e-01	2.379826e-01
6	0.048831	1.862273e-02	2.366817e-01	9.104625e-02	4.573066e-02	-4.628428e-01	3.247490e-01	-1.622092e-02
7	0.113663	5.174601e-02	-2.622531e-01	-2.531390e-01	-2.907823e-02	1.521555e-01	-1.527369e-01	-1.916664e-01

	AAPL	AXP	BA	CAT	CSCO	CVX	DIS	DD
8	0.040007	7.091671e-02	2.094998e-01	8.233537e-02	5.927579e-02	3.514762e-01	-4.927151e-03	-5.634827e-01
9	0.061096	7.896653e-02	2.576431e-01	1.634395e-01	-1.292009e-01	2.317940e-01	2.196912e-01	3.014736e-01
10	0.103380	1.727083e-02	-2.514903e-02	6.248996e-02	1.816583e-01	-3.544005e-01	-8.727859e-02	-1.445712e-01
11	0.091832	9.305687e-02	-2.545588e-01	-8.422741e-03	1.405995e-02	3.022488e-01	-2.531161e-01	-3.496450e-02
12	-0.082416	5.052740e-02	9.720987e-02	-8.076475e-02	2.877352e-02	6.020080e-02	1.548999e-01	-1.861580e-01
13	0.027764	2.329948e-02	1.399946e-01	-1.890362e-01	2.511505e-01	-4.096785e-02	-2.172021e-01	4.407242e-01
14	0.023816	2.801166e-01	-9.980066e-02	7.571143e-02	1.928513e-01	-8.842887e-02	-1.454928e-01	7.193376e-02
15	-0.077535	8.644791e-02	6.970314e-03	3.094997e-01	-3.442417e-02	-1.542108e-01	-3.559405e-01	-8.048151e-02
16	0.021959	1.393750e-02	-2.495271e-02	-8.998330e-02	-1.233117e-01	1.661632e-01	1.288115e-01	2.626187e-01
17	0.001342	-1.141548e-02	-1.737985e-01	-1.728437e-01	-6.292471e-02	-3.462158e-02	2.328535e-01	1.229270e-01
18	-0.062469	3.598241e-01	1.563770e-01	-1.649011e-01	4.929766e-01	-1.390347e-01	-1.978306e-01	-2.877343e-02
19	-0.231776	1.771623e-01	7.499227e-03	1.725842e-01	1.995869e-01	-6.916970e-02	3.692877e-01	-6.959350e-02
20	0.039909	2.944542e-01	2.897144e-02	4.583988e-01	-2.155339e-01	-7.647758e-02	-2.527834e-01	4.244286e-02
21	-0.123218	1.829743e-02	-7.351353e-02	-2.252485e-01	-9.638783e-02	-2.209891e-01	-1.381998e-01	-5.323696e-03
22	-0.059748	-2.120851e-01	8.688156e-02	-4.027510e-03	1.419243e-01	1.331975e-02	1.152682e-01	3.863027e-02
23	-0.196793	5.129316e-01	5.101190e-02	-2.317182e-01	-3.659388e-01	-6.529456e-02	1.259814e-01	4.135507e-02
24	-0.085035	7.111692e-02	2.556204e-02	-2.154510e-01	-1.252999e-01	1.091792e-01	-1.172819e-01	7.455854e-02
25	0.541479	4.974019e-02	1.118859e-01	-1.228648e-01	-9.366467e-02	-6.101859e-02	-7.434466e-02	7.647482e-02
26	0.452690	-3.685045e-02	2.069081e-02	-1.361313e-01	1.896250e-01	-5.431750e-02	1.306602e-01	-9.291805e-02
27	0.331706	4.122459e-01	3.242486e-02	-5.595115e-02	-1.430319e-01	-2.347867e-02	6.432301e-02	7.979380e-02
28	-0.337014	1.041115e-01	-1.768016e-02	-2.803305e-02	-9.185347e-02	-2.637184e-02	-5.677897e-02	-2.688983e-02
29	-0.000000	-7.978351e-17	-9.098255e-17	5.472836e-17	-4.622489e-16	2.418135e-16	-6.674109e-17	2.524241e-17

30 rows × 30 columns

```
singular values
[5.93356819e+01 3.61923238e+01 3.36749723e+01 2.27208732e+01
 1.60401348e+01 1.38764139e+01 1.16767407e+01 1.00416541e+01
 8.64603983e+00 8.14910098e+00 6.20841225e+00 5.53076307e+00
 4.91690313e+00 4.64518731e+00 4.32249925e+00 4.03319616e+00
 3.92216910e+00 3.40087252e+00 3.00679038e+00 2.50075051e+00
 2.41683391e+00 2.22796892e+00 2.08804385e+00 1.94813802e+00
 1.76936805e+00 1.54171442e+00 1.37301467e+00 1.25650865e+00
 9.76405909e-01 5.27508143e-15]

explained variance ratio
[4.67559515e-01 1.73955418e-01 1.50598109e-01 6.85575136e-02
 3.41681175e-02 2.55716949e-02 1.81070748e-02 1.33910781e-02
 9.92749068e-03 8.81910316e-03 5.11877592e-03 4.06232937e-03
 3.21061572e-03 2.86557306e-03 2.48127486e-03 2.16024851e-03
 2.04294959e-03 1.53598060e-03 1.20063590e-03 8.30511703e-04
 7.75708652e-04 6.59209230e-04 5.79007588e-04 5.04016167e-04
 4.15758737e-04 3.15655161e-04 2.50354486e-04 2.09669851e-04
 1.26609363e-04 3.69541622e-33]
principal components
```

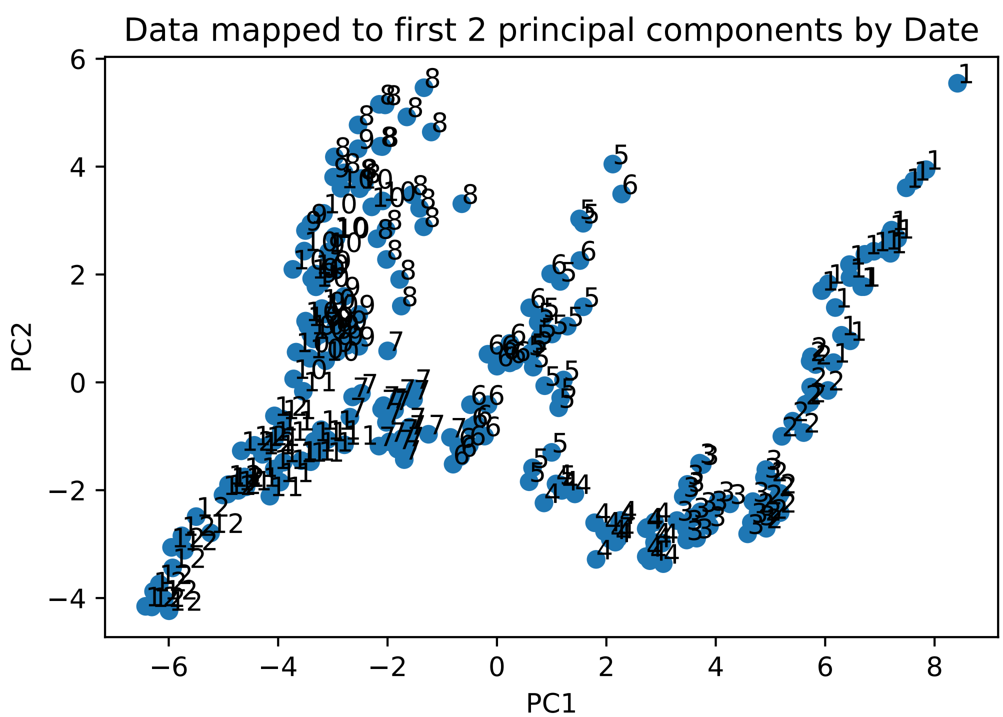
	1/2/19	1/3/19	1/4/19	1/7/19	1/8/19	1/9/19	1/10/19	1/11/19	1/14/19
0	0.078200	0.069770	0.078344	0.078083	0.082292	0.085680	0.086266	0.084236	0.081922
1	-0.066791	-0.065400	-0.061272	-0.058756	-0.067499	-0.062567	-0.066192	-0.067213	-0.069475
2	-0.038210	-0.049802	-0.048985	-0.057918	-0.045313	-0.054099	-0.057381	-0.054849	-0.063572
3	0.156000	0.172866	0.149883	0.139935	0.135973	0.127469	0.115641	0.124453	0.121941
4	0.038162	0.056218	0.041328	0.028446	0.079247	0.038067	0.049769	0.057665	0.043036
5	-0.157869	-0.171455	-0.152227	-0.160115	-0.153771	-0.130988	-0.122670	-0.126855	-0.142553
6	0.020329	0.025175	0.025168	0.038201	0.030398	0.020699	0.003833	0.007518	-0.011865
7	-0.022796	-0.014474	-0.024051	0.005502	0.025005	0.021226	0.024203	0.033471	0.039946
8	-0.076323	-0.035982	-0.057125	-0.048993	-0.017020	-0.055886	-0.055296	-0.037100	-0.027386
9	-0.044827	-0.007334	-0.057692	-0.042833	-0.033519	-0.041167	-0.014003	-0.012519	0.026054
10	-0.087376	-0.052410	-0.050564	-0.032541	-0.000840	-0.002603	0.041000	0.010343	0.019769
11	0.082292	0.089757	0.061034	0.075692	0.023189	0.049586	0.009208	-0.023187	-0.031896
12	0.001851	0.014407	0.015856	0.065920	0.011467	0.043642	0.036188	0.033622	0.051943
13	-0.049256	-0.080485	-0.112262	-0.112877	-0.109428	-0.127924	-0.139909	-0.124360	-0.073429
14	0.052677	0.048475	0.051020	0.011503	0.004030	-0.036388	-0.034513	-0.005451	-0.004262
15	-0.012976	-0.007178	0.037844	0.012375	-0.026771	-0.037889	-0.071180	-0.055070	-0.037565
16	-0.067427	-0.058419	-0.081552	-0.071663	-0.017140	-0.052297	0.036721	0.069348	0.066963
17	-0.027213	-0.026389	-0.008771	-0.025347	0.019729	-0.050939	-0.042928	0.005040	0.002753
18	-0.038400	0.026071	0.054890	0.030257	0.081342	-0.004935	0.022419	0.012650	0.020339
19	-0.043927	-0.095541	0.016533	-0.043954	0.015236	-0.020661	0.120719	0.082079	0.092039
20	0.018644	0.115069	0.053070	0.042773	0.011116	-0.013873	-0.079803	-0.079408	-0.045106
21	-0.038464	0.003018	0.006799	-0.046114	-0.112982	-0.108517	-0.111945	-0.091497	-0.034635
22	-0.179419	-0.104372	-0.006564	0.017252	0.118872	0.074414	0.032904	0.066916	0.060493
23	-0.023429	-0.026503	-0.051905	0.021733	0.087014	0.096962	0.030410	0.035665	0.060482
24	-0.007443	0.005785	-0.005318	-0.010426	-0.021497	0.047945	0.006910	-0.046878	-0.019510

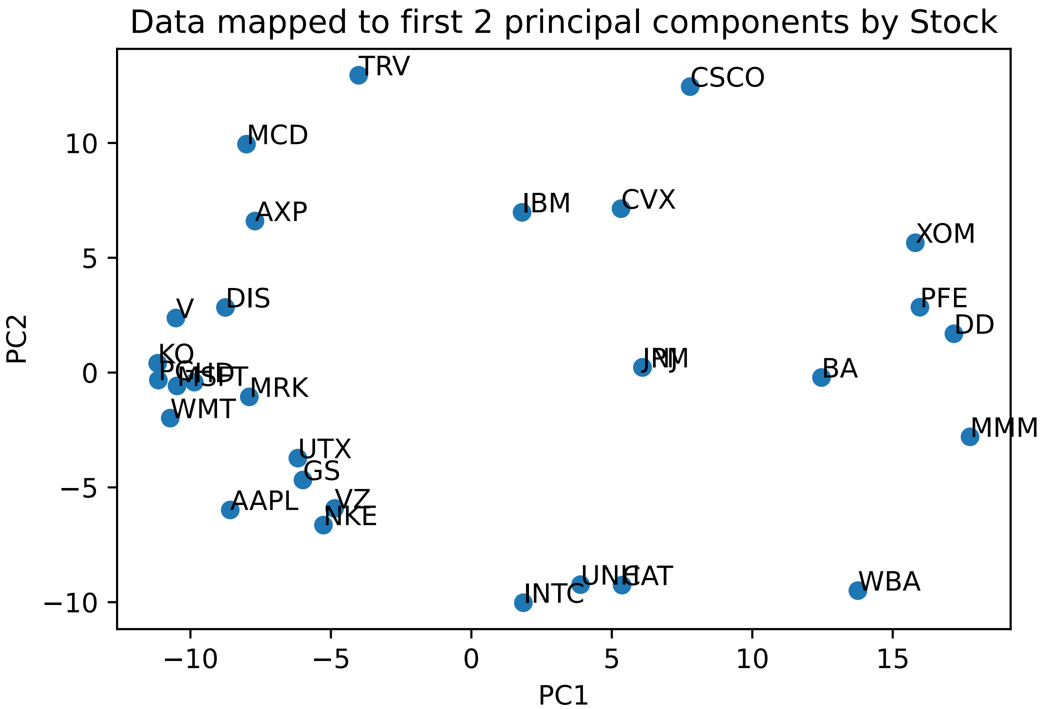
	1/2/19	1/3/19	1/4/19	1/7/19	1/8/19	1/9/19	1/10/19	1/11/19	1/14/19
25	-0.047165	-0.081872	-0.036076	0.025285	0.014283	0.080405	0.104732	0.070389	0.076886
26	-0.028474	0.012738	-0.054289	-0.038860	-0.040001	0.007072	0.022227	-0.086589	-0.067766
27	-0.019933	0.072623	0.127567	0.051120	-0.033658	-0.003462	-0.092295	-0.087357	-0.060118
28	0.382409	-0.119432	-0.164963	0.474768	-0.095169	-0.030176	-0.023620	-0.063097	-0.290629
29	0.033465	0.061424	-0.000336	0.035551	0.107071	-0.022377	-0.051419	0.044697	-0.114220

30 rows × 251 columns

```
singular values
[5.32200997e+01 3.38794039e+01 2.40292315e+01 2.06582048e+01
 1.51327679e+01 1.36099321e+01 1.15317327e+01 9.32692169e+00
 8.20381130e+00 6.66771620e+00 6.09049569e+00 5.41523701e+00
 4.81911792e+00 4.63192131e+00 4.32219570e+00 4.02784645e+00
 3.63260947e+00 3.07016722e+00 2.52251419e+00 2.42313441e+00
 2.37250957e+00 2.13230154e+00 1.98373022e+00 1.93010901e+00
 1.66085270e+00 1.41381105e+00 1.25918073e+00 9.76488214e-01
 6.95629552e-15 4.59495942e-15]

explained variance ratio
[4.77127570e-01 1.93354669e-01 9.72664144e-02 7.18899701e-02
 3.85762393e-02 3.12029075e-02 2.24012514e-02 1.46541221e-02
 1.13374261e-02 7.48923325e-03 6.24868200e-03 4.93989915e-03
 3.91217480e-03 3.61414431e-03 3.14696562e-03 2.73293282e-03
 2.22290328e-03 1.58784170e-03 1.07189040e-03 9.89095334e-04
 9.48198118e-04 7.65914664e-04 6.62900526e-04 6.27547810e-04
 4.64670796e-04 3.36717656e-04 2.67091015e-04 1.60626485e-04
 8.15153113e-33 3.55669402e-33]
```





Compared to the 2020 data, in 2019 stocks were more spread out with a smaller cluster on the left.

Stocks started very good in the top right and traveled in waves down and left with upward spikes.

In [ ]:

```
## Question 6 - Text Classification
```

(a)

In [213…

```
partyData = pd.read_csv("./sotu/party.txt", delimiter=", ", names=['Party','Pres

listfiles = [f for f in listdir("files/") if isfile(join("files/", f))]
listfiles.sort()

corpusPre = []
for fi in listfiles:
    file_path = "files/%s" % (fi)
    with open(file_path) as f:
        corpusPre.append(f.read().splitlines())
        f.close()

corpus = []
i = 0
for words in corpusPre:
    corpus.append(" ".join(corpusPre[i]))
    i+=1

corpusExtra = []
i = 0
for words in corpusPre:
    corpusExtra.append([
        partyData.loc[i][0],
        " ".join(corpusPre[i]),
        partyData.loc[i][1],
        partyData.loc[i][2]])
```

```
    i+=1

data = pd.DataFrame(corpusExtra, columns=['Party','Document','President','Year']

corpusSmall = []
names = ["trump","obama","bush","clinton","kennedy"]
corpusSmall.append(data.loc[(data['President'] == 'trump') & (data['Year'] == 20
corpusSmall.append(data.loc[(data['President'] == 'obama') & (data['Year'] == 20
corpusSmall.append(data.loc[(data['President'] == 'bush') & (data['Year'] == 200
corpusSmall.append(data.loc[(data['President'] == 'clinton') & (data['Year'] ==
corpusSmall.append(data.loc[(data['President'] == 'kennedy') & (data['Year'] ==
```

(b)

In [214…

```
stopwds = []
with open("./sotu/stopwords.txt") as f:
    stopwds = f.read().lower().splitlines()
    f.close()
```

(c)

In [219…

```
TokenPattern = r'\b[a-zA-Z]{1,}\b'
vectorizer = CountVectorizer(input='content', token_pattern=TokenPattern,
                             stop_words = stopwds)

X = vectorizer.fit_transform(corpus)

# create Document-Term Matrix / DataFrame
Xframe = pd.DataFrame(X.toarray(),
                      index=listfiles,
                      columns=vectorizer.get_feature_names())

Xframe.iloc[0:10,0:5]
```

```
/home/ericgi231/.local/lib/python3.7/site-packages/sklearn/feature_extraction/te
xt.py:391: UserWarning: Your stop_words may be inconsistent with your preprocess
ing. Tokenizing the stop words generated tokens ['ain', 'aren', 'couldn', 'dare
n', 'didn', 'doesn', 'don', 'hadn', 'hasn', 'haven', 'isn', 'll', 'mayn', 'might
n', 'mon', 'mustn', 'needn', 'oughtn', 'shan', 'shouldn', 've', 'wasn', 'weren',
'won', 'wouldn'] not in stop_words.
  'stop_words.' % sorted(inconsistent))
```

Out[219…

	aaa	aana	aaron	abandon	abandoned
a1.txt	0	0	0	0	0
a10.txt	0	0	0	1	0
a100.txt	0	0	0	0	0
a101.txt	0	0	0	0	1
a102.txt	0	0	0	0	1
a103.txt	0	0	0	1	0
a104.txt	0	0	0	0	0
a105.txt	0	0	0	0	0
a106.txt	0	2	0	2	1

	aaa	aana	aaron	abandon	abandoned
a107.txt	0	0	0	0	0

Remainder of question left un answered