PROJECT - 4 - Application of Clustering Models

By Prakash Ghosh

Problem Statement: Application of Clustering Models

* Dataset Link: https://drive.google.com/file/d/1pP0Rr83ri0voscgr95-YnVCBv6BYV22w/view (https://drive.google.com/file/d/1pP0Rr83ri0voscgr95-YnVCBv6BYV22w/view)

Problem 1: There are various stocks for which we have collected a data set, which all stocks are apparently similar in performance.

Problem 2: How many Unique patterns that exist in the historical stock data set, based on fluctuations in price.

Problem 3: Identify which all stocks are moving together and which all stocks are different from each other.

Solution:

In [10]:
```
# Import Required Libraries
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

from sklearn import decomposition, datasets
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import seaborn as sns
sns.set()
sns.set_style('whitegrid')

%matplotlib inline
from datetime import datetime
```

Load Data

In [3]:
```
# Load data from link
# File downloaded as data_stocks.csv from the URL="https://drive.google.com/file/d/1pP0Rr83ri0voscgr95-YnVCBv6BYV22w/view"
file_name="data_stocks.csv"
df = pd.read_csv(file_name)
df.head(10)
```

Out[3]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AK
0	1491226200	2363.6101	42.3300	143.6800	129.6300	82.040	102.2300	85.2200	59.760
1	1491226260	2364.1001	42.3600	143.7000	130.3200	82.080	102.1400	85.6500	59.840
2	1491226320	2362.6799	42.3100	143.6901	130.2250	82.030	102.2125	85.5100	59.795
3	1491226380	2364.3101	42.3700	143.6400	130.0729	82.000	102.1400	85.4872	59.620
4	1491226440	2364.8501	42.5378	143.6600	129.8800	82.035	102.0600	85.7001	59.620
5	1491226500	2365.6201	42.5399	143.7800	130.0700	82.040	102.0400	85.9200	59.610
6	1491226560	2365.2000	42.4700	143.8640	130.1800	82.120	102.3300	85.9120	59.540
7	1491226620	2365.2900	42.4700	143.8100	130.1400	82.190	102.3700	85.8200	59.410
8	1491226680	2364.3201	42.3900	143.8150	130.1000	82.230	102.3800	85.8800	59.430
9	1491226740	2364.6399	42.3300	143.8000	130.2100	82.165	102.3300	85.8600	59.260

10 rows × 502 columns

Problem 1: There are various stocks for which we have collected a data set, which all stocks are apparently similar in performance.

Analyze Data

In [21]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41266 entries, 0 to 41265
Columns: 502 entries, DATE to NYSE.ZTS
dtypes: float64(501), int64(1)
memory usage: 158.0 MB
```

In [22]:
```
df.describe()
```

Out[22]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NA
count	4.126600e+04	41266.000000	41266.000000	41266.000000	41266.00000	41266.000000	41266.000000	41266.000000	41
mean	1.497749e+09	2421.537882	47.708346	150.453566	141.31793	79.446873	103.480398	102.998608	50
std	3.822211e+06	39.557135	3.259377	6.236826	6.91674	2.000283	4.424244	9.389788	4.8
min	1.491226e+09	2329.139900	40.830000	140.160000	128.24000	74.800000	95.870000	83.000000	44
25%	1.494432e+09	2390.860100	44.945400	144.640000	135.19500	78.030000	101.300000	94.820000	47
50%	1.497638e+09	2430.149900	48.360000	149.945000	142.26000	79.410000	102.440000	106.820000	49
75%	1.501090e+09	2448.820100	50.180000	155.065000	147.10000	80.580000	104.660000	110.490000	52
max	1.504210e+09	2490.649900	54.475000	164.510000	155.33000	90.440000	121.770000	119.270000	62

8 rows × 502 columns

In [23]:
```
# Check Null
df.isnull().values.any()
```

Out[23]: False

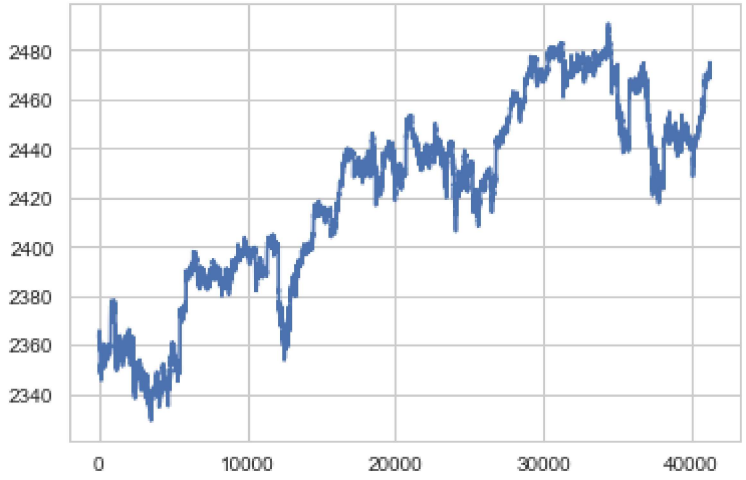
In [14]:
```
# Drop date variable
df_new = df.copy()
df_new = df_new.drop(['DATE'], 1)
# Dimensions of dataset
n = df_new.shape[0]
p = df_new.shape[1]
# Make data a numpy array
df_new = df_new.values
```

In [15]:
```
df_new
```

Out[15]:
```
array([[2363.6101,   42.33  ,  143.68  , ...,   63.86  ,  122.    ,
           53.35  ],
       [2364.1001,   42.36  ,  143.7   , ...,   63.74  ,  121.77  ,
           53.35  ],
       [2362.6799,   42.31  ,  143.6901, ...,   63.75  ,  121.7   ,
           53.365 ],
       ...,
       [2470.03  ,   44.74  ,  164.01  , ...,   76.88  ,  114.31  ,
           62.685 ],
       [2471.49  ,   44.71  ,  163.88  , ...,   76.83  ,  114.23  ,
           62.6301],
       [2471.49  ,   44.74  ,  163.98  , ...,   76.81  ,  114.28  ,
           62.68  ]])
```

In [18]:
```
plt.plot(df['SP500'])
```

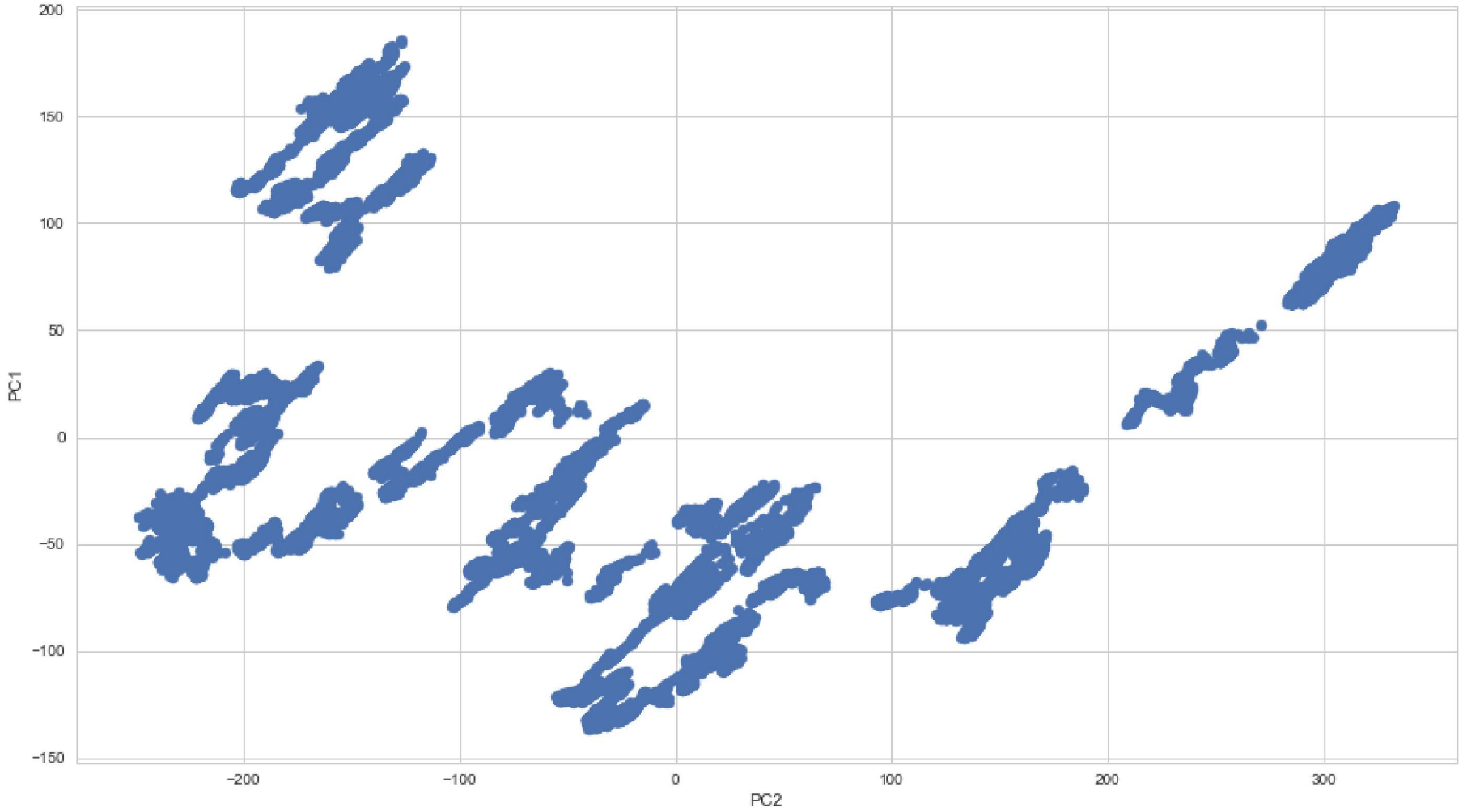
Out[18]:
```
[<matplotlib.lines.Line2D at 0x20d1bdbea90>]
```



Principal Component Analysis

In [19]:
```
pca = PCA(n_components=3)
pca.fit(df_new)
df_pca = pca.transform(df_new)

#Scatter Plot
plt.figure(figsize=(16,9))
plt.scatter(df_pca[:,0],df_pca[:,1])
plt.ylabel('PC1')
plt.xlabel('PC2')
plt.show()
```



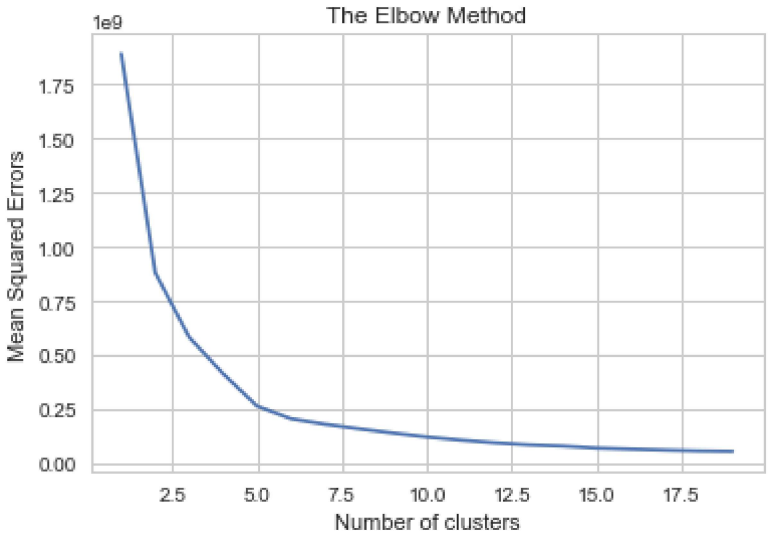
KMeans Clustering

In [20]:
```
df_pca
```

Out[20]:
```
array([[ 312.06471872,   82.16326685,   61.13922645],
       [ 308.40806514,   78.09348261,   61.7988178 ],
       [ 306.77364778,   77.32427431,   59.11828842],
       ...,
       [-201.54427195,  118.68060234,  -59.00831309],
       [-200.92133737,  119.16896696,  -58.55435589],
       [-202.42056915,  117.64446307,  -59.82857205]])
```

In [24]:
```
# Finding optimum number of clusters for KMEANS cluster - Elbow Method
k = []
inertia = []
for i in range(1,20):
    k_means = KMeans(n_clusters = i, random_state = 0)
    k_means.fit(df_new)
    k.append(i)
    inertia.append(k_means.inertia_)
```

In [25]:
```
# Plot to find the Number of clusters
plt.plot(k,inertia)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('Mean Squared Errors')
plt.show()
```

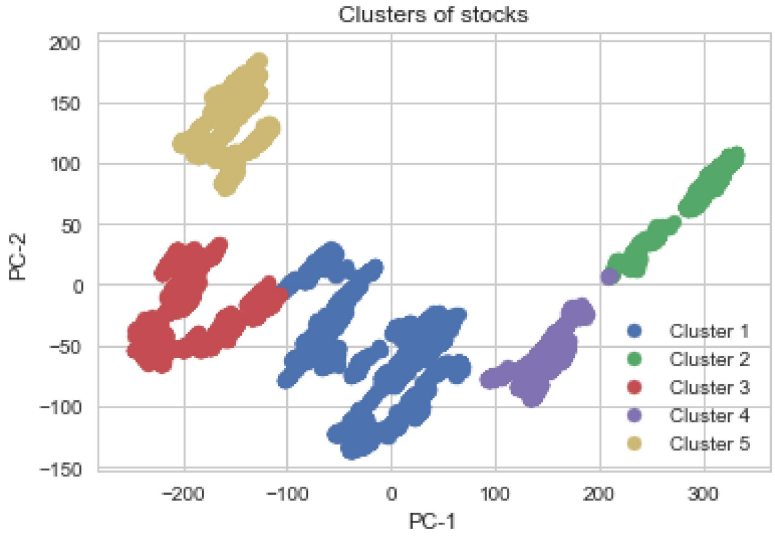


Observation - 1 : Optimum number of cluster from the above graph can be determined as 5

In [28]:
```
# with the number of KMeans cluster =5
k_means = KMeans(n_clusters=5,random_state=0,init='k-means++')
k_means.fit(df_pca)
y_kmeans = k_means.fit_predict(df_pca)
labels = k_means.labels_
labels
```

Out[28]:
```
array([1, 1, 1, ..., 4, 4, 4])
```

In [31]:
```
plt.scatter(df_pca[y_kmeans == 0, 0], df_pca[y_kmeans == 0, 1], label = 'Cluster 1')
plt.scatter(df_pca[y_kmeans == 1, 0], df_pca[y_kmeans == 1, 1], label = 'Cluster 2')
plt.scatter(df_pca[y_kmeans == 2, 0], df_pca[y_kmeans == 2, 1], label = 'Cluster 3')
plt.scatter(df_pca[y_kmeans == 3, 0], df_pca[y_kmeans == 3, 1], label = 'Cluster 4')
plt.scatter(df_pca[y_kmeans == 4, 0], df_pca[y_kmeans == 4, 1], label = 'Cluster 5')
plt.title('Clusters of stocks')
plt.xlabel('PC-1')
plt.ylabel('PC-2')
plt.legend()
plt.show()
```



Answer of Problem -1 : The above 5 clusters shows the stocks which are similar in stock performance

---

Problem 2: How many Unique patterns that exist in the historical stock data set, based on fluctuations in price.

In [35]:
```
df_pc = df.copy()
df_pc = df_pc.drop(['DATE'], 1)
df_pc = pd.DataFrame(pca.components_,columns=df_pc.columns)
df_pc.head()
```

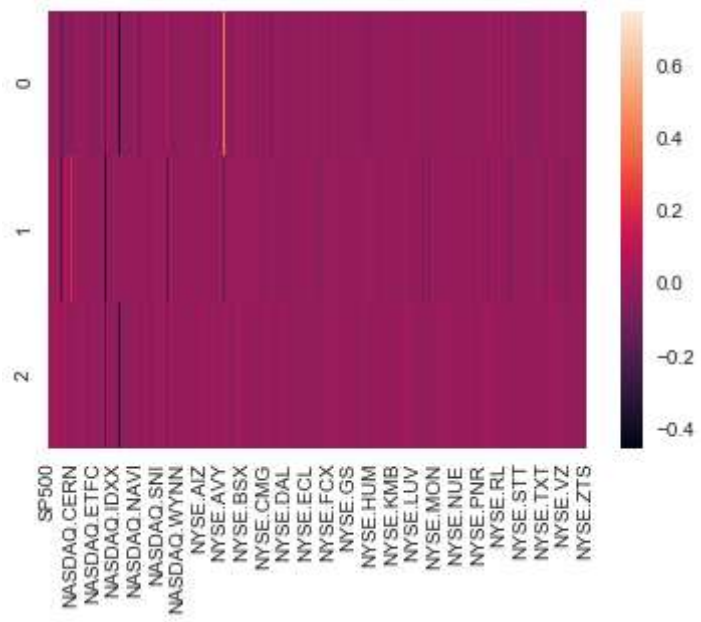
Out[35]:

	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AKAM	NASDAQ
0	-0.216015	-0.013154	-0.019954	-0.037492	-0.002238	-0.012834	-0.046234	0.020698	-0.02990
1	-0.030758	-0.017003	0.015011	0.011668	-0.006525	0.020935	-0.020952	0.004190	0.077607
2	0.047246	0.007741	-0.025218	-0.014318	-0.010718	0.031483	-0.043681	0.029598	0.064317

3 rows × 501 columns

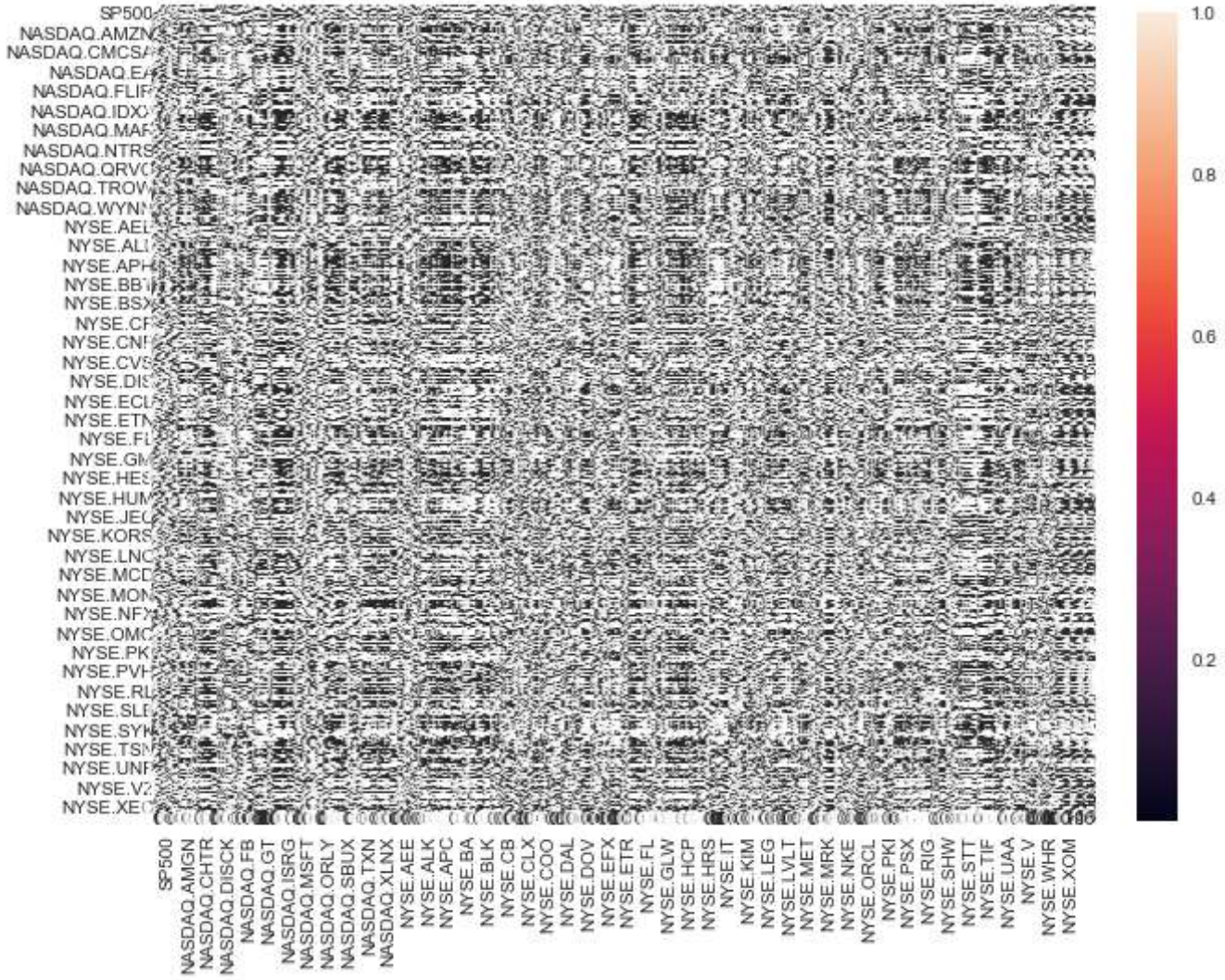
In [36]:
```
sns.set_style('whitegrid')
sns.heatmap(df_pc)
```

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x20d1c16cdd8>



In [37]:
```
plt.figure(figsize=(11,8))
df_corr = df_pc.corr().abs()
sns.heatmap(df_corr,annot=True)
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x20d1be81c50>



Problem 3: Identify which all stocks are moving together and which all stocks are different from each other.

In [40]:
```
df['labels'] = labels
df.head()
```

Out[40]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AK
0	1491226200	2363.6101	42.3300	143.6800	129.6300	82.040	102.2300	85.2200	59.760
1	1491226260	2364.1001	42.3600	143.7000	130.3200	82.080	102.1400	85.6500	59.840
2	1491226320	2362.6799	42.3100	143.6901	130.2250	82.030	102.2125	85.5100	59.795
3	1491226380	2364.3101	42.3700	143.6400	130.0729	82.000	102.1400	85.4872	59.620
4	1491226440	2364.8501	42.5378	143.6600	129.8800	82.035	102.0600	85.7001	59.620

5 rows × 503 columns

In [41]:
```
df['labels'].unique().tolist()
```

Out[41]:
```
[1, 3, 0, 2, 4]
```

In [42]:
```
for i in df['labels'].unique().tolist():
    count = df[df['labels'] == i].shape[0]
    print('For lablel {}: The number of stock with similar performances is: {} '.format(i,count))
```

```
For lablel 1: The number of stock with similar performances is: 7024
For lablel 3: The number of stock with similar performances is: 6661
For lablel 0: The number of stock with similar performances is: 12065
For lablel 2: The number of stock with similar performances is: 8869
For lablel 4: The number of stock with similar performances is: 6647
```