In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
data = pd.read_csv('data_stocks.csv')
```

In [3]:

```
data.head()
```

Out[3]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.AD
0	1491226200	2363.6101	42.3300	143.6800	129.6300	82.040
1	1491226260	2364.1001	42.3600	143.7000	130.3200	82.080
2	1491226320	2362.6799	42.3100	143.6901	130.2250	82.030
3	1491226380	2364.3101	42.3700	143.6400	130.0729	82.000
4	1491226440	2364.8501	42.5378	143.6600	129.8800	82.035

5 rows × 502 columns

```
In [4]:
data.columns.tolist()[0]
Out[4]:
'DATE'
In [5]:
data.set_index(data.columns.tolist()[0],inplace=True)
In [6]:
data.head()
Out[6]:
```

	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI
DATE					
1491226200	2363.6101	42.3300	143.6800	129.6300	82.040
1491226260	2364.1001	42.3600	143.7000	130.3200	82.080
1491226320	2362.6799	42.3100	143.6901	130.2250	82.030
1491226380	2364.3101	42.3700	143.6400	130.0729	82.000
1491226440	2364.8501	42.5378	143.6600	129.8800	82.035

5 rows × 501 columns

In [7]:
```
data.drop('SP500', axis= 1, inplace= True)
```
In [8]:
```
from sklearn import cluster, datasets as dt
```
In [9]:
```
n_clusters = 5
```
In [10]:
```
np.random.seed(0)
k_Means= cluster.KMeans(n_clusters=n_clusters, random_state= 25)
k_Means.fit(data)
print('Over')
```

```
Over
```
In [11]:
```
k_Means.labels_
```
Out[11]:
```
array([4, 4, 4, ..., 3, 3, 3])
```
In [12]:
```
len(k_Means.labels_)
```
Out[12]:
```
41266
```
In [13]:
```
k_Means.cluster_centers_
```
Out[13]:
```
array([[  51.72587735,  151.06200533,  146.92356499, ...,
74.45093304,
         125.61319585,   62.33830194],
       [  45.09849074,  151.07393826,  135.6250932 , ...,
68.61520574,
         118.89374393,   58.85615306],
       [  49.26457875,  148.64841274,  141.54205293, ...,
73.30668488,
         124.86423224,   62.56597652],
       [  46.2734716 ,  160.11520883,  150.24007608, ...,
76.00650235,
         112.7923117 ,   61.19815181],
       [  43.77358132,  142.98696747,  130.74934798, ...,
64.62368362,
         120.79942516,   53.63660808]])
```
In [14]:
```
data['Cluster_No'] = k_Means.labels_
```
In [15]:
```
data.head()
```
Out[15]:

	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.A

DATE					
1491226200	42.3300	143.6800	129.6300	82.040	102.2300
1491226260	42.3600	143.7000	130.3200	82.080	102.1400
1491226320	42.3100	143.6901	130.2250	82.030	102.2125
1491226380	42.3700	143.6400	130.0729	82.000	102.1400
1491226440	42.5378	143.6600	129.8800	82.035	102.0600

5 rows × 501 columns

Problem 1:¶

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

```
In [16]:
companyMean = data.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(data.columns)-1]

companyName = data.columns.tolist()[:len(data.columns)-1]

companyStd= data.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(data.columns)-1]

companyData = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
In [17]:
companyData.sort_values('cStd', ascending=False)
Out[17]:
```

	Company	cMean	cStd
91	NASDAQ.PCLN	1867.722306	78.341014
163	NYSE.AZO	598.324633	76.696839
65	NASDAQ.ISRG	901.320301	66.399068
199	NYSE.CMG	417.670807	63.347703
96	NASDAQ.REGN	457.676948	46.809471
52	NASDAQ.GOOGL	941.369665	43.450668
51	NASDAQ.GOOG	922.131875	42.227541
11	NASDAQ.AMZN	968.747188	38.966212
87	NASDAQ.ORLY	224.415988	27.904559
21	NASDAQ.CHTR	348.719389	25.256789
110	NASDAQ.ULTA	274.241713	25.112028
84	NASDAQ.NVDA	141.419298	25.003991
164	NYSE.BA	202.534020	22.736921
169	NYSE.BCR	306.158251	22.395830
208	NYSE.COO	228.707758	20.094502
175	NYSE.BLK	409.701703	19.228577
124	NYSE.AAP	123.296657	19.017181
280	NYSE.GWW	180.151759	18.426612
37	NASDAQ.EQIX	432.201844	17.370001
410	NYSE.PXD	161.082634	17.154405
447	NYSE.TDG	263.070526	17.060790

114	NASDAQ.VRTX	131.767415	16.472286
176	NYSE.BLL	51.297688	15.874231
15	NASDAQ.BIIB	272.957640	13.763828
428	NYSE.SHW	338.119248	13.586502
263	NYSE.FL	56.910060	13.280365
13	NASDAQ.AVGO	238.598238	12.824722
130	NYSE.ADS	244.854817	12.653201
81	NASDAQ.NFLX	160.850341	12.570000
162	NYSE.AYI	183.341470	12.518941
...	...	...	...
485	NYSE.WMB	30.102558	0.927698
63	NASDAQ.INTC	35.489340	0.915785
53	NASDAQ.GRMN	51.344658	0.915121
489	NYSE.WY	33.248472	0.907998
320	NYSE.KMI	19.757229	0.906750
423	NYSE.RSG	63.788602	0.896241
245	NYSE.EMR	59.387318	0.887733
338	NYSE.LUK	25.528150	0.872308
62	NASDAQ.INFO	46.288736	0.850310
327	NYSE.L	47.246764	0.816147
359	NYSE.MON	116.911295	0.809861
292	NYSE.HPE	17.744404	0.763962
470	NYSE.USB	51.863284	0.760711
402	NYSE.PPL	38.609105	0.734429
165	NYSE.BAC	23.701408	0.674120
194	NYSE.CHK	4.990837	0.654819
29	NYSE.HPQ	18.534090	0.648805

3			
203	NYSE.CNP	28.212311	0.647618
389	NYSE.PFE	33.269778	0.593905
317	NYSE.KEY	18.160407	0.577986
85	NASDAQ.NWS	13.715717	0.556995
86	NASDAQ.NWSA	13.316591	0.532835
101	NASDAQ.SPLS	9.666552	0.506469
297	NYSE.HST	18.272814	0.424607
89	NASDAQ.PBCT	17.362005	0.420245
414	NYSE.RF	14.260065	0.378946
488	NYSE.WU	19.272765	0.362158
56	NASDAQ.HBAN	13.051218	0.358746
133	NYSE.AES	11.347591	0.281469
257	NYSE.F	11.139757	0.262379

500 rows × 3 columns

In [18]:
```
df_1 = data.loc[data.Cluster_No == 0,:]
df_2 = data.loc[data.Cluster_No == 1,:]
df_3 = data.loc[data.Cluster_No == 2,:]
df_4 = data.loc[data.Cluster_No == 3,:]
df_5 = data.loc[data.Cluster_No == 4,:]
```

In [19]:
```
companyMean = df_1.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(df_1.columns)-1]

companyName = df_1.columns.tolist()[:len(df_1.columns)-1]

companyStd= df_1.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(df_1.columns)-1]

companydf_1 = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
```

In [20]:
```
companydf_1.shape
```

Out[20]:
```
(500, 3)
```

In [21]:
```
companydf_1.head()
```

Out[21]:

	Company	cMean	cStd

0	NASDAQ.AAL	51.725877	1.430593
1	NASDAQ.AAPL	151.062005	4.324873
2	NASDAQ.ADBE	146.923565	2.093385
3	NASDAQ.ADI	79.556993	1.031093
4	NASDAQ.ADP	107.410306	5.979057

In [22]:
```
pd.Series(k_Means.labels_).value_counts()
```
Out[22]:
```
2    12029
0     8905
4     6976
1     6709
3     6647
dtype: int64
```
In [23]:
```
companyData.sort_values('cStd', ascending=False, inplace= True)
```
In [24]:
```
companydf_1.sort_values('cStd', ascending=False, inplace= True)
```
In [25]:
```
companyData.head()
```
Out[25]:

	Company	cMean	cStd
91	NASDAQ.PCLN	1867.722306	78.341014
163	NYSE.AZO	598.324633	76.696839
65	NASDAQ.ISRG	901.320301	66.399068
199	NYSE.CMG	417.670807	63.347703
96	NASDAQ.REGN	457.676948	46.809471

In [26]:
```
companydf_1.head()
```
Out[26]:

	Company	cMean	cStd
91	NASDAQ.PCLN	1992.227976	41.105572
199	NYSE.CMG	364.615179	25.146872
11	NASDAQ.AMZN	1010.326946	22.047485
21	NASDAQ.CHTR	358.311887	19.844799
52	NASDAQ.GOOGL	963.180313	18.290805

In [27]:
```
companydf_1.set_index('Company', inplace=True)
```
In [28]:
```
companyData.set_index('Company', inplace=True)
```
In [29]:
```
companyMean = df_2.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(df_2.columns)-1]

companyName = df_2.columns.tolist()[:len(df_2.columns)-1]
```

```
companyStd= df_2.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(df_2.columns)-1]

companydf_2 = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
In [30]:
companyMean = df_3.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(df_3.columns)-1]

companyName = df_3.columns.tolist()[:len(df_3.columns)-1]

companyStd= df_3.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(df_3.columns)-1]

companydf_3 = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
In [31]:
companyMean = df_4.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(df_4.columns)-1]

companyName = df_4.columns.tolist()[:len(df_4.columns)-1]

companyStd= df_4.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(df_4.columns)-1]

companydf_4 = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
In [32]:
companyMean = df_5.apply(np.mean, axis= 0)
companyM = companyMean.tolist()[:len(df_5.columns)-1]

companyName = df_5.columns.tolist()[:len(df_5.columns)-1]

companyStd= df_5.apply(np.std, axis= 0)
companyStdD= companyStd.tolist()[:len(df_5.columns)-1]

companydf_5 = pd.DataFrame({'Company':companyName ,
'cMean':companyM, 'cStd': companyStdD})
In [33]:
companydf_2.set_index('Company', inplace=True)
companydf_3.set_index('Company', inplace=True)
companydf_4.set_index('Company', inplace=True)
companydf_5.set_index('Company', inplace=True)
In [34]:
companyData['Cluster_1_Mean'] = companydf_1.cMean
companyData['Cluster_2_Mean'] = companydf_2.cMean
companyData['Cluster_3_Mean'] = companydf_3.cMean
companyData['Cluster_4_Mean'] = companydf_4.cMean
companyData['Cluster_5_Mean'] = companydf_5.cMean
In [35]:
companyData['Cluster_1_Std'] = companydf_1.cStd
companyData['Cluster_2_Std'] = companydf_2.cStd
companyData['Cluster_3_Std'] = companydf_3.cStd
```

```
companyData['Cluster_4_Std'] = companydf_4.cStd
companyData['Cluster_5_Std'] = companydf_5.cStd
In [36]:
companyData.head()
Out[36]:
```

	cMean	cStd	Cluster_1_Mean	Cluster_2_Mean	Cluster_
Company					
NASDAQ.PCLN	1867.722306	78.341014	1992.227976	1848.808425	1861.036
NYSE.AZO	598.324633	76.696839	516.246706	687.793454	590.770(
NASDAQ.ISRG	901.320301	66.399068	942.628129	850.114460	924.396[
NYSE.CMG	417.670807	63.347703	364.615179	482.611264	449.560[
NASDAQ.REGN	457.676948	46.809471	496.393374	430.232695	483.194[

```
In [37]:
companyData['lift_1'] =
pd.Series(companyData.cStd/companyData.Cluster_1_Std)
In [38]:
companyData.sort_values('lift_1', inplace=True, ascending=False)
companyData.lift_1.head()
Out[38]:
Company
NYSE.BHI              inf
NASDAQ.WFM     34.848157
NYSE.BLL       25.800289
NYSE.BCR       16.166194
NASDAQ.SPLS    14.218695
Name: lift_1, dtype: float64
In [39]:
companyData['lift_2'] =
pd.Series(companyData.cStd/companyData.Cluster_2_Std)
companyData['lift_3'] =
pd.Series(companyData.cStd/companyData.Cluster_3_Std)
companyData['lift_4'] =
pd.Series(companyData.cStd/companyData.Cluster_4_Std)
companyData['lift_5'] =
pd.Series(companyData.cStd/companyData.Cluster_5_Std)
In [40]:
companyData.loc['NASDAQ.ISRG']
Out[40]:
cMean              901.320301
cStd                66.399068
Cluster_1_Mean     942.628129
Cluster_2_Mean     850.114460
Cluster_3_Mean     924.396511
Cluster_4_Mean     976.244296
Cluster_5_Mean     786.654340
Cluster_1_Std        9.697559
Cluster_2_Std       13.179252
Cluster_3_Std       16.860209
Cluster_4_Std       19.677591
Cluster_5_Std       28.612941
lift_1               6.846988
```

```
lift_2              5.038152
lift_3              3.938211
lift_4              3.374349
lift_5              2.320596
Name: NASDAQ.ISRG, dtype: float64
In [41]:
companyData.loc['NYSE.ROP']
Out[41]:
cMean             226.125881
cStd                9.239688
Cluster_1_Mean    234.816249
Cluster_2_Mean    220.412763
Cluster_3_Mean    230.114628
Cluster_4_Mean    230.905701
Cluster_5_Mean    209.094555
Cluster_1_Std       1.622447
Cluster_2_Std       2.230649
Cluster_3_Std       2.774568
Cluster_4_Std       1.811958
Cluster_5_Std       3.660084
lift_1              5.694909
lift_2              4.142153
lift_3              3.330135
lift_4              5.099283
lift_5              2.524447
Name: NYSE.ROP, dtype: float64
In [42]:
companyLifts =
companyData[['lift_1','lift_2','lift_3','lift_4','lift_5']]
In [43]:
max = companyLifts.loc[companyLifts.lift_1 != np.inf].max()
companyLifts.replace(np.inf, max, inplace=True)

C:\Anaconda3\lib\site-packages\ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy

In [44]:
companyLifts.head()
print(np.std(companyLifts.lift_1))
print(np.mean(companyLifts.lift_1))

2.769257065856375
2.6237397287442903
In [45]:
k_Means= cluster.KMeans(n_clusters = 30, random_state= 25,
max_iter=200)
k_Means.fit(companyLifts)
Out[45]:
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++',
max_iter=200,
    n_clusters=30, n_init=10, n_jobs=1,
precompute_distances='auto',
    random_state=25, tol=0.0001, verbose=0)
```

In [46]:

```
companyLifts['Cluster_No'] = k_Means.labels_
```

```
C:\Anaconda3\lib\site-packages\ipykernel_launcher.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a
DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
```

In [47]:

```
companyLifts.head()
```

Out[47]:

	lift_1	lift_2	lift_3	lift_4	lift_5	Cluster_No
Company						
NYSE.BHI	34.848157	11.158152	2.150664	1.669370e+13	9.714463	1
NASDAQ.WFM	34.848157	12.926240	0.993825	3.319598e+01	1.598287	15
NYSE.BLL	25.800289	0.954455	21.275511	2.757203e+01	12.416564	11
NYSE.BCR	16.166194	25.007256	8.046074	2.837327e+01	1.009736	10
NASDAQ.SPLS	14.218695	1.337905	1.207683	2.683804e+01	1.820849	18

In [48]:

```
pd.Series(k_Means.labels_).value_counts()
```

Out[48]:

```
0     98
27    73
16    67
8     62
22    53
12    42
19    41
29    12
23    10
21     8
20     8
28     7
24     2
25     2
26     2
6      1
1      1
2      1
3      1
4      1
```

```
5      1
10     1
7      1
11     1
13     1
15     1
18     1
14     1
dtype: int64
In [147]:
companyCluster_1 = companyLifts.loc[companyLifts.Cluster_No ==
0,:]
companyCluster_27 = companyLifts.loc[companyLifts.Cluster_No ==
27,:]
companyCluster_16 = companyLifts.loc[companyLifts.Cluster_No ==
16,:]
companyCluster_8 = companyLifts.loc[companyLifts.Cluster_No ==
8,:]
companyCluster_22 = companyLifts.loc[companyLifts.Cluster_No ==
22,:]
companyCluster_12 = companyLifts.loc[companyLifts.Cluster_No ==
12,:]
companyCluster_19 = companyLifts.loc[companyLifts.Cluster_No ==
19,:]
companyCluster_29 = companyLifts.loc[companyLifts.Cluster_No ==
29,:]
companyCluster_23 = companyLifts.loc[companyLifts.Cluster_No ==
23,:]
writer = pd.ExcelWriter('companyLiftCluster.xlsx')
companyCluster_1.to_excel(writer,'Sheet1')
companyCluster_27.to_excel(writer,'Sheet2')
companyCluster_16.to_excel(writer,'Sheet3')
companyCluster_8.to_excel(writer,'Sheet4')
companyCluster_22.to_excel(writer,'Sheet5')
companyCluster_12.to_excel(writer,'Sheet6')
companyCluster_19.to_excel(writer,'Sheet7')
companyCluster_29.to_excel(writer,'Sheet8')
companyCluster_23.to_excel(writer,'Sheet9')
writer.save()
In [152]:
companyData.loc['NASDAQ.WYNN']
Out[152]:
cMean             128.505151
cStd                6.975419
Cluster_1_Mean    131.644940
Cluster_2_Mean    124.095096
Cluster_3_Mean    132.623519
Cluster_4_Mean    133.423110
Cluster_5_Mean    116.950926
Cluster_1_Std       3.519493
Cluster_2_Std       1.971915
Cluster_3_Std       4.146181
Cluster_4_Std       3.581380
```

```
Cluster_5_Std       3.054034
lift_1              1.981939
lift_2              3.537384
lift_3              1.682372
lift_4              1.947690
lift_5              2.284002
Name: NASDAQ.WYNN, dtype: float64
```

1.There are various stocks for which we have collected a data set, which all stocks are apparently similar in performance
After doing the above analysis, I am grouping the companies which are apparently similar in performance. Group 1 :
NYSE.AIZ,NYSE.CRM,NASDAQ.CSCO,NASDAQ.JBHT,NYSE.LUK,NYSE.DFS,NYSE.ETR,NYSE.BF.B,NYSE.SLG,NYSE.CCI,NYSE.WRK,NYSE.PSA,NYSE.UPS,NYSE.IVZ,NYSE.DTE,NYSE.ARNC,NASDAQ.AAL,NYSE.ES,NYSE.MPC,NASDAQ.BIIB,NYSE.PNC,NYSE.VMC,NYSE.BMY,NYSE.LNC,NASDAQ.FITB,NYSE.D,NASDAQ.SNPS,NYSE.PEP,NYSE.COP,NYSE.APA,NYSE.DOV,NASDAQ.WYNN,NYSE.LEN,NASDAQ.XLNX,NYSE.WMB,NYSE.LYB,NYSE.AIG,NASDAQ.DLTR,NYSE.MLM,NYSE.MS,NYSE.PSX,NYSE.MTB,NYSE.GM,NASDAQ.FISV,NYSE.CLX,NYSE.UAA,NYSE.PPL,NYSE.ALB,NYSE.EOG,NYSE.CMI,NYSE.EIX,NYSE.AVB,NYSE.AES,NASDAQ.PAYX,NASDAQ.EBAY,NYSE.SYY,NYSE.HRL,NYSE.PWR,NYSE.GGP,NYSE.TIF,NYSE.CXO,NYSE.DAL,NYSE.SWK,NYSE.GIS,NYSE.CNP,NYSE.UA,NYSE.WY,NYSE.SRE,NASDAQ.NWS,NYSE.UNM,NYSE.FTV,NYSE.PRGO,NYSE.FRT,NASDAQ.SYMC,NYSE.OKE,NASDAQ.NWSA,NYSE.COTY,NYSE.HD,NYSE.EVHC,NYSE.EFX,NYSE.PCG,NASDAQ.MU,NYSE.DHR,NYSE.DRI,NYSE.DG,NYSE.HPQ,NASDAQ.IDXX,NYSE.PEG,NYSE.WMT,NASDAQ.CSX,NYSE.NEM,NASDAQ.NTAP,NYSE.AIV,NASDAQ.ILMN,NYSE.TGT,NASDAQ.WDC,NYSE.NLSN,NYSE.CVX
Group 2:
NYSE.AYI,NYSE.SCHW,NASDAQ.MDLZ,NASDAQ.PCAR,NYSE.RSG,NYSE.VLO,NYSE.JWN,NYSE.PRU,NASDAQ.QRVO,NYSE.CBG,NASDAQ.PBCT,NYSE.WM,NASDAQ.AMD,NYSE.ALLE,NYSE.TMK,NYSE.ARE,NYSE.PFE,NYSE.PFG,NYSE.TXT,NYSE.HCN,NYSE.HBI,NYSE.PH,NYSE.IFF,NYSE.MON,NASDAQ.MCHP,NYSE.CMA,NYSE.URI,NASDAQ.INCY,NYSE.LLY,NYSE.VTR,NASDAQ.ADI,NYSE.SPG,NYSE.CFG,NYSE.CVS,NYSE.MRK,NYSE.BAC,NYSE.RF,NYSE.CBS,NYSE.DLR,NYSE.,TI,NYSE.EMR,NYSE.DD,NASDAQ.DISH,NYSE.NUE,NYSE.WFC,NYSE.HST,NASDAQ.HBAN,NYSE.GPS,NYSE.USB,NYSE.DOW,NASDAQ.QCOM,NASDAQ.VRSK,NYSE.REG,NY,E.CF,NYSE.TSN,NASDAQ.SWKS,NYSE.JEC,NYSE.EXR,NYSE.OXY,NASDAQ.GRMN,NASDAQ.TXN,NYSE.GS,NYSE.KEY,NYSE.WU,NYSE.COF,NASDAQ.INTC,NYSE.L,NYSE.UNP,NYSE.SEE,NYSE.PPG,NYSE.F,NYSE.KSS,NYSE.NSC
Group 3:
NASDAQ.ROST,NYSE.HSY,NYSE.COH,NYSE.DLPH,NASDAQ.CELG,NYSE.DE,NYSE.SIG,NYSE.DXC,NASDAQ.ATVI,NASDAQ.ADSK,NYSE.AMT,NYSE.NKE,NYSE.NWL,NYSE.SYF,NYSE.HRS,NYSE.TMO,NYSE.EL,NYSE.TDG,NASDAQ.TRIP,NASDAQ.AVGO,NYSE.MAC,NASDAQ.CERN,NYSE.CMS,NYSE.PX,NASDAQ.HSIC,NYSE,MCK,NASDAQ.EQIX,NASDAQ.XRAY,NASDAQ.AMGN,NYSE.AMG,NASDAQ.FOX,NYSE.HUM,NASDAQ.FOXA,NASDAQ.CHRW,NYSE.AEP,NASDAQ.TSCO,NASDAQ.CTXS,NYSE.,KI,NYSE.AGN,NYSE.ALL,NYSE.LNT,NASDAQ.PCLN,NYSE.RL,NYSE.KO,NYSE.XEL,NYSE.MCO,NYSE.DUK,NASDAQ.GILD,NASDAQ.AMZN,NYSE.KMI,NYSE.K,NY,E.AJG,NYSE.UTX,NYSE.VAR,NYSE.FE,NYSE.WEC,NASDAQ.AAPL,NYSE.AEE,NASDAQ.MYL,NYSE.KORS,NY

SE.FMC,NYSE.AMP,NYSE.FCX,NASDAQ.CINF,NYSE.UAL,NYSE.LEG,NASDAQ.ADP
Group 4:
NYSE.FL,NYSE.AET,NYSE.MCD,NYSE.CAG,NASDAQ.ISRG,NYSE.APC,NYSE.RHT,NYSE.HRB,NYSE.STZ,NYSE.AXP,NYSE.ROP,NASDAQ.CBOE,NYSE.YUM,NYSE.KSU,NYSE.CI,NYSE.EW,NYSE.NFX,NYSE.NOC,NASDAQ.NVDA,NYSE.FTI,NYSE.GWW,NASDAQ.WLTW,NYSE.XEC,NYSE.BLK,NYSE.CCL,NYSE.M,NYSE.STT,NYSE.IT,NYSE.ICE,NYSE.DVN,NYSE.ANTM,NYSE.C,NASDAQ.ADBE,NYSE.LMT,NYSE.SPGI,NYSE.GE,NYSE.UNH,NYSE.BK,NYSE.ABT,NASDAQ.MNST,NYSE.BAX,NYSE,MMC,NYSE.BDX,NASDAQ.REGN,NYSE.NBL,NASDAQ.EA,NYSE.NI,NYSE.NEE,NASDAQ.ETFC,NYSE.ESS,NASDAQ.VRSN,NYSE.HP,NYSE.KMB,NYSE.PVH,NYSE.TAP,NYSE.XYL,NYSE.RCL,NASDAQ.NDAQ,NASDAQ.PDCO,NASDAQ.FB,NASDAQ.CHTR,NYSE.IPG
Group 5:
NYSE.TJX,NASDAQ.COST,NYSE.CL,NYSE.JNJ,NASDAQ.MAR,NYSE.AME,NYSE.COG,NASDAQ.GOOG,NASDAQ.GOOGL,NASDAQ.AKAM,NYSE.APD,NASDAQ.ALXN,NYSE.ADM,NYSE.DIS,NYSE.LLL,NYSE.PM,NASDAQ.WBA,NYSE.EQT,NYSE.O,NASDAQ.EXPD,NYSE.MSI,NYSE.KIM,NASDAQ.AMAT,NASDAQ.FFIV,NYSE.BRK.B,NYSE.JNPR,NYSE.AWK,NYSE.IP,NYSE.MET,NYSE.BBT,NYSE.CHD,NYSE.BEN,NYSE.EXC,NYSE.HCP,NASDAQ.FLIR,NYSE.ABC,NASDAQ.HOLX,NYSE.VFC,NYSE.ADS,NYSE.WHR,NASDAQ.NAVI,NYSE.PNR,NYSE.ZBH,NYSE.T,NYSE.AN,NYSE.TWX,NASDAQ.GT,NYSE.HCA,NYSE.UHS,NYSE.RHI,NASDAQ.CA,NASDAQ.NTRS,NYSE.VZ
Group 6:
NYSE.LB,NYSE.ECL,NYSE.CTL,NASDAQ.FAST,NYSE.HES,NYSE.RJF,NYSE.XOM,NYSE.LVLT,NYSE.CHK,NYSE.DHI,NYSE.ROK,NASDAQ.ESRX,NYSE.KMX,NYSE,BWA,NYSE.JPM,NYSE.BXP,NYSE.SNA,NASDAQ.EXPE,NYSE.EMN,NYSE.HAL,NASDAQ.ZION,NASDAQ.CMCSA,NYSE.DPS,NYSE.SYK,NYSE.TEL,NASDAQ.KHC,NYSE.ED,NYSE.FBHS,NYSE.PHM,NYSE.MAA,NYSE.MHK,NYSE.TRV,NYSE.HPE,NASDAQ.LRCX,NYSE.XL,NYSE.ITW,NYSE.GD,NYSE.PNW,NYSE.SHW,NYSE.GLW,NASDAQ.KLAC,NYSE.IR
Group 7:
NYSE.MAS,NYSE.MRO,NYSE.CPB,NYSE.CSRA,NASDAQ.SRCL,NYSE.DGX,NYSE.CNC,NYSE.LOW,NYSE.ABBV,NYSE.FDX,NYSE.UDR,NYSE.CB,NYSE.OMC,NYSE.M,C,NYSE.ACN,NYSE.LH,NASDAQ.CME,NYSE.BSX,NYSE.CAT,NYSE.SCG,NYSE.WAT,NASDAQ.MSFT,NYSE.APH,NYSE.PLD,NYSE.AFL,NASDAQ.SBUX,NYSE.MDT,N,SE.SO,NYSE.PXD,NYSE.MMM,NYSE.EQR,NYSE.ALK,NASDAQ.HAS,NYSE.CAH,NYSE.PG,NYSE.IRM,NYSE.LUV,NASDAQ.CTAS,NASDAQ.NFLX,NYSE.ETN,NYSE.J,I
Group 8:
NYSE.ZTS,NASDAQ.CTSH,NYSE.RTN,NYSE.MOS,NASDAQ.VIAB,NYSE.FLR,NYSE.IBM,NYSE.GPN,NASDAQ.DISCA,NASDAQ.DISCK,NYSE.DVA,NASDAQ.SNI
Group 9:
NYSE.AVY,NYSE.SLB,NYSE.NOV,NYSE.HOG,NASDAQ.ORLY,NASDAQ.TROW,NYSE.RRC,NASDAQ.LKQ,NASDAQ.STX,NYSE.FLS
Attaching the companyListCluster.xslx sheet containing the list of all the companies in separate sheets which are similar in performance.

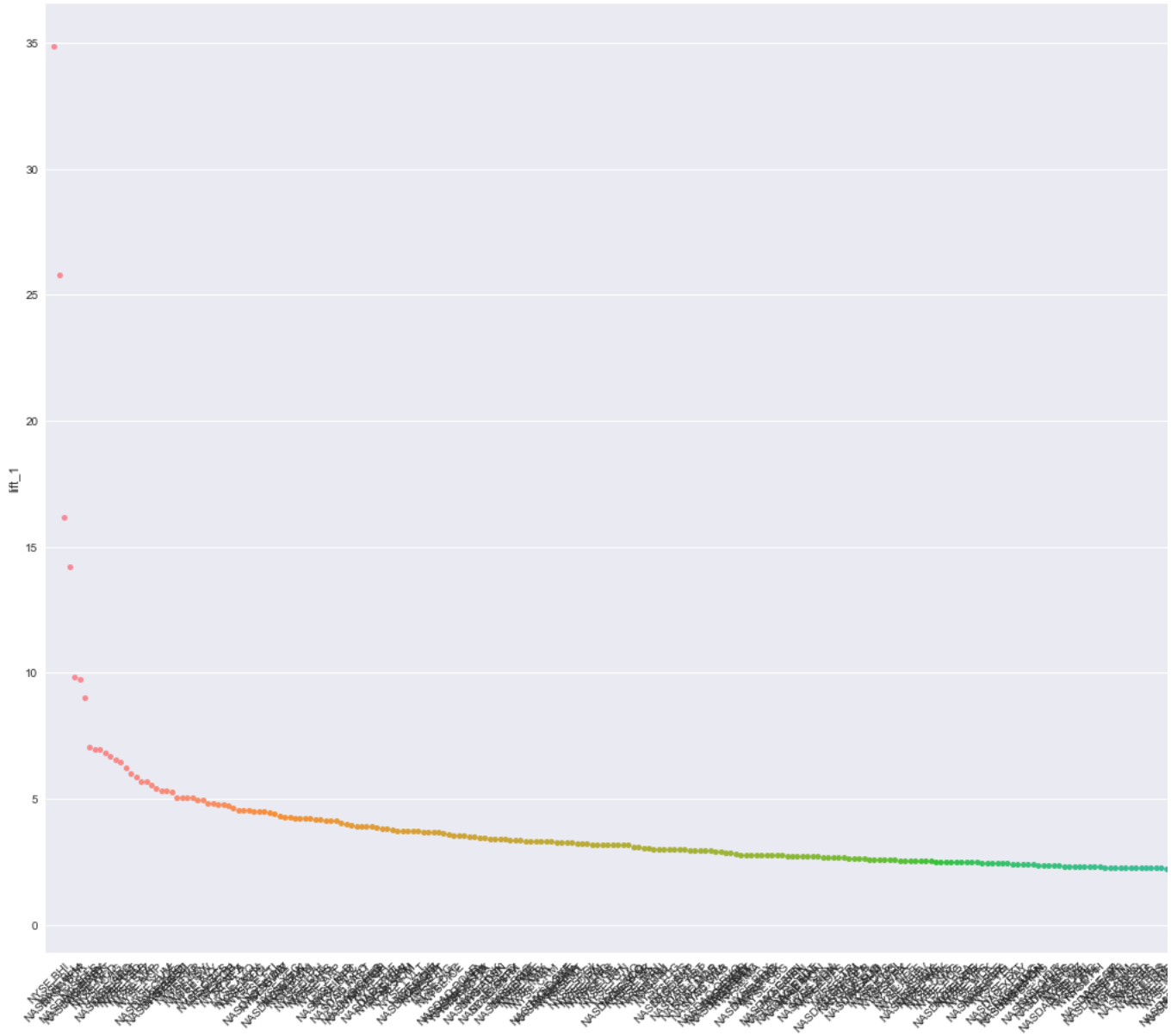
In [51]:
```
companyData.reset_index(inplace=True)
```

Problem 2:¶

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

In [52]:

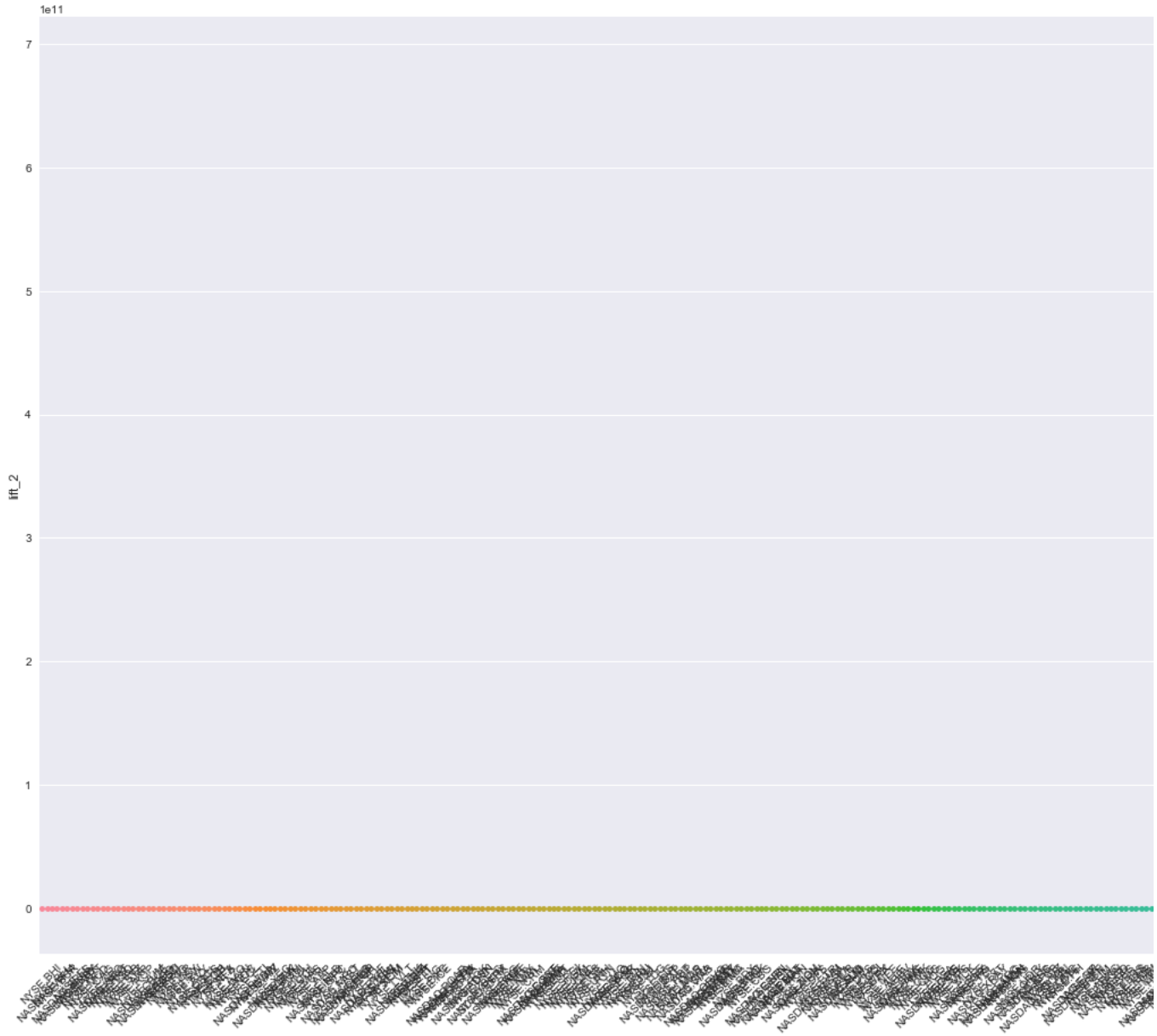
```
import seaborn as sns
plt.figure(2, figsize=(40, 15.2))
g =  sns.swarmplot(x="Company", y="lift_1", data=companyData);
#g.set_xticklabels(rotation=90)
plt.xticks(rotation = 45)

#plt.scatter(companyData.index, companyData.lift_1)
plt.show()
```



In [53]:

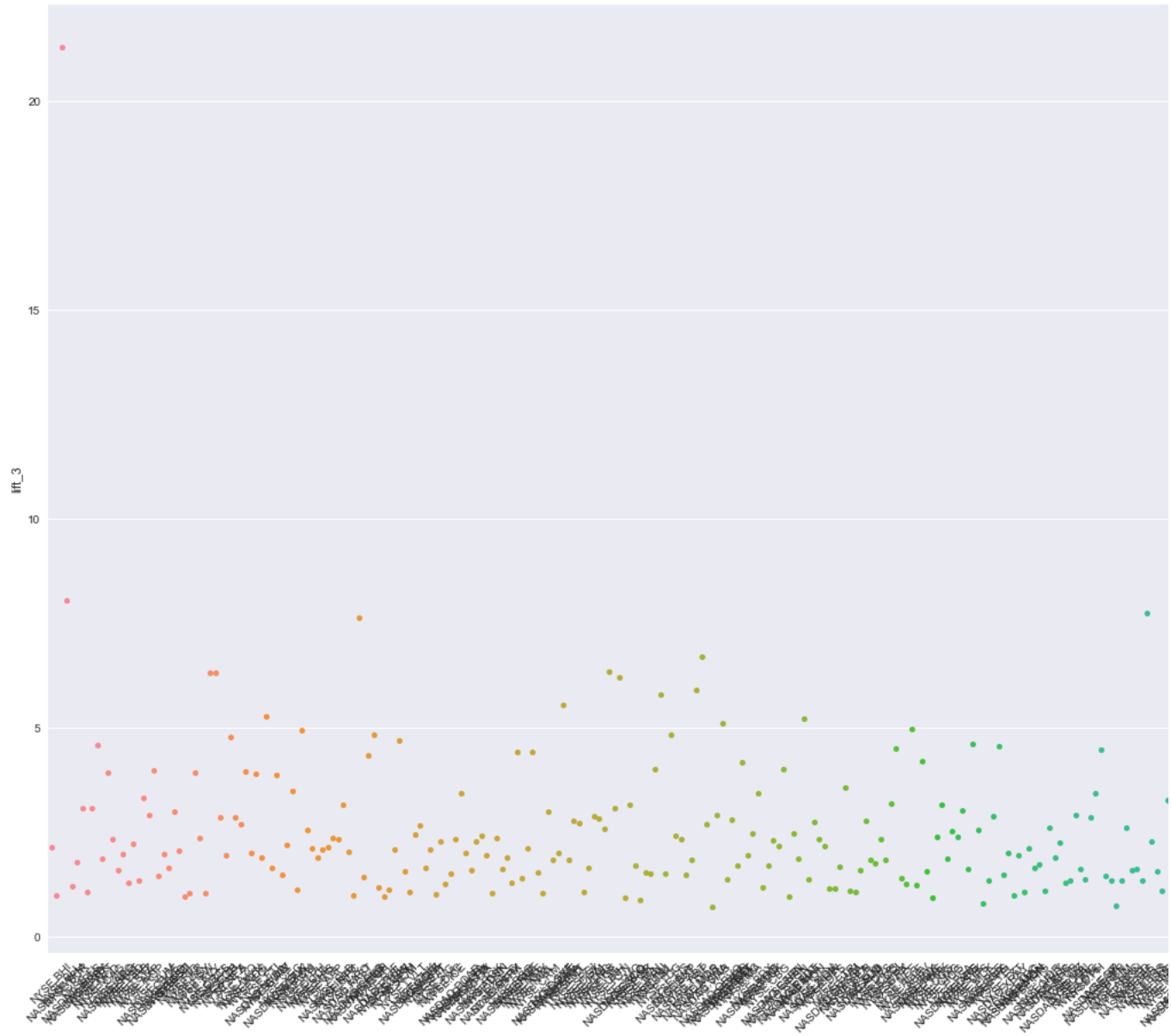
```
import seaborn as sns
plt.figure(2, figsize=(40, 15.2))
g =  sns.swarmplot(x="Company", y="lift_2", data=companyData);
#g.set_xticklabels(rotation=90)
plt.xticks(rotation = 45)
```

```
#plt.scatter(companyData.index, companyData.lift_1)
plt.show()
```



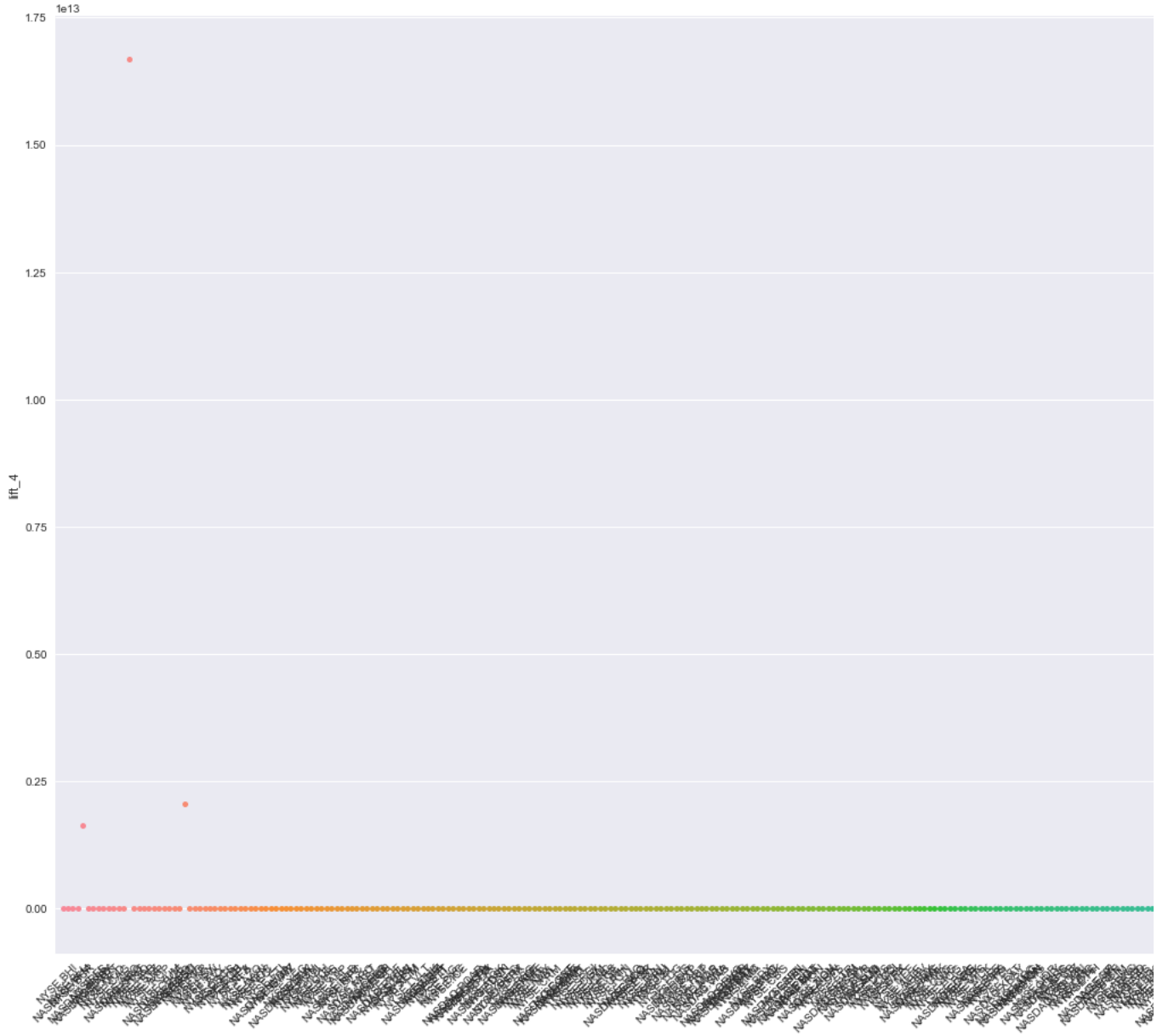
In [54]:

```
import seaborn as sns
plt.figure(2, figsize=(40, 15.2))
g =  sns.swarmplot(x="Company", y="lift_3", data=companyData);
#g.set_xticklabels(rotation=90)
plt.xticks(rotation = 45)

#plt.scatter(companyData.index, companyData.lift_1)
plt.show()
```



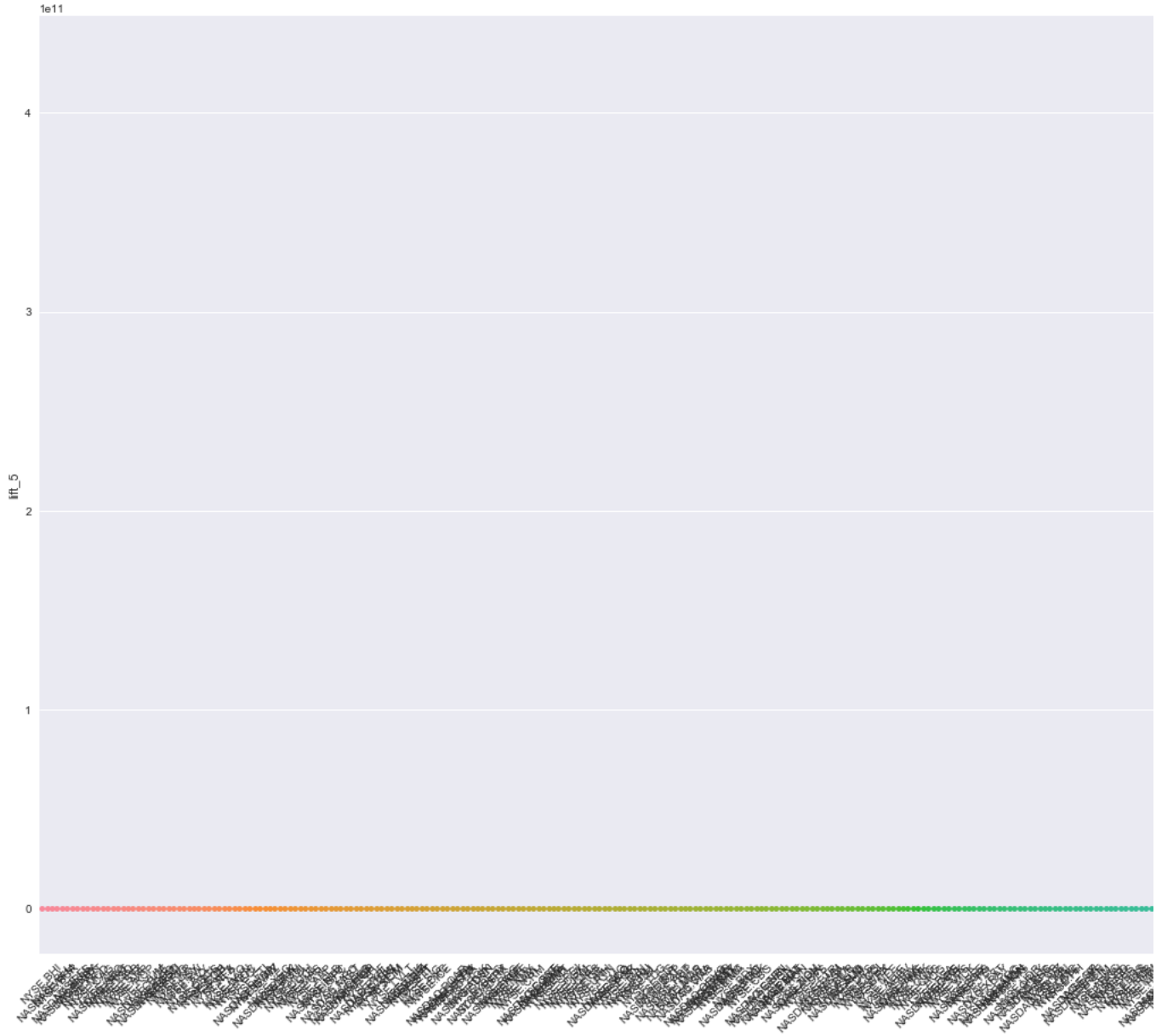
In [55]:

```
import seaborn as sns
plt.figure(2, figsize=(40, 15.2))
g =  sns.swarmplot(x="Company", y="lift_4", data=companyData);
#g.set_xticklabels(rotation=90)
plt.xticks(rotation = 45)

#plt.scatter(companyData.index, companyData.lift_1)
plt.show()
```



In [56]:

```
import seaborn as sns
plt.figure(2, figsize=(40, 15.2))
g =  sns.swarmplot(x="Company", y="lift_5", data=companyData);
#g.set_xticklabels(rotation=90)
plt.xticks(rotation = 45)

#plt.scatter(companyData.index, companyData.lift_1)
plt.show()
```



As per the graph displaying above the lifts in the standard deviation against all the companies, Lift_2 & Lift_5 is the most unique pattern that existed in historical stock data set, based on fluctuations in price, since the lift remains consistence after clustering.

Problem 3:¶

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

In [57]:
```
companyData.set_index('Company', inplace= True)
```
In [58]:
```
data.drop('Cluster_No',axis = 1, inplace=True)
```
In [59]:

```
import seaborn as sns

f, ax = plt.subplots(figsize=(100, 100))
corr = data.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool),
cmap=sns.diverging_palette(220, 10, as_cmap=True),
            square=True, ax=ax)
```

Out[59]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1dd0bf84160>
```


NASDAQ.AAL
NASDAQ.AAPL
NASDAQ.ADBE
NASDAQ.ADI
NASDAQ.ADP
NASDAQ.ADSK
NASDAQ.AKAM
NASDAQ.ALXN
NASDAQ.AMAT
NASDAQ.AMD
NASDAQ.AMGN
NASDAQ.AMZN
NASDAQ.ATVI
NASDAQ.AVGO
NASDAQ.BBBY
NASDAQ.BIIB
NASDAQ.CA
NASDAQ.CBOE
NASDAQ.CELG
NASDAQ.CERN
NASDAQ.CHRW
NASDAQ.CHTR
NASDAQ.CINF
NASDAQ.CMCSA
NASDAQ.CME
NASDAQ.COST
NASDAQ.CSCO
NASDAQ.CSX
NASDAQ.CTAS
NASDAQ.CTSH
NASDAQ.CTXS
NASDAQ.DISCA
NASDAQ.DISCK
NASDAQ.DLTR
NASDAQ.EA
NASDAQ.EBAY
NASDAQ.EQIX
NASDAQ.ESRX
NASDAQ.EXPD
NASDAQ.EXPE
NASDAQ.FAST
NASDAQ.FB
NASDAQ.FFIV
NASDAQ.FISV
NASDAQ.FLIR
NASDAQ.FOX
NASDAQ.FOXA
NASDAQ.GILD
NASDAQ.GOOG
NASDAQ.GOOGL
NASDAQ.HAL
NASDAQ.HAS
NASDAQ.HBAN
NASDAQ.HOLX
NASDAQ.HSIC
NASDAQ.ILMN
NASDAQ.INCY
NASDAQ.INFO
NASDAQ.INTC
NASDAQ.INTU
NASDAQ.ISRG
NASDAQ.JBHT
NASDAQ.KHC
NASDAQ.KLAC
NASDAQ.LKQ
NASDAQ.LRCX
NASDAQ.MAR
NASDAQ.MAT
NASDAQ.MCHP
NASDAQ.MDLZ
NASDAQ.MNST
NASDAQ.MSFT
NASDAQ.MU
NASDAQ.MYL
NASDAQ.NAVI
NASDAQ.NCLH
NASDAQ.NFLX
NASDAQ.NTAP
NASDAQ.NTRS
NASDAQ.NVDA
NASDAQ.NWL
NASDAQ.NWSA
NASDAQ.ORLY
NASDAQ.PAYX
NASDAQ.PBCT
NASDAQ.PCAR
NASDAQ.PCLN
NASDAQ.PDCO
NASDAQ.PEP
NASDAQ.QCOM
NASDAQ.QRVO
NASDAQ.REGN
NASDAQ.ROST
NASDAQ.SBUX
NASDAQ.SHPG
NASDAQ.SNPS
NASDAQ.SPLS
NASDAQ.SRCL
NASDAQ.STX
NASDAQ.SWKS
NASDAQ.SYMC
NASDAQ.TMUS
NASDAQ.TROW
NASDAQ.TSCO
NASDAQ.TXN
NASDAQ.ULTA
NASDAQ.VIAB
NASDAQ.VRSK
NASDAQ.VRTX
NASDAQ.WBA
NASDAQ.WDC
NASDAQ.WFM
NASDAQ.WLTW
NASDAQ.WYNN
NASDAQ.XLNX
NASDAQ.XRAY
NASDAQ.ZION
NYSE:A
NYSE:AAP
NYSE:ABBV
NYSE:ABC
NYSE:ABT
NYSE:ACN
NYSE:ADM
NYSE:ADS
NYSE:AEE
NYSE:AEP
NYSE:AES
NYSE:AFL
NYSE:AGN
NYSE:AIG
NYSE:AIV
NYSE:AIZ
NYSE:ALB
NYSE:ALK
NYSE:ALL
NYSE:ALLE
NYSE:AME
NYSE:AMG
NYSE:AMP
NYSE:AMT
NYSE:AN
NYSE:ANTM
NYSE:AON
NYSE:APA
NYSE:APC
NYSE:APD
NYSE:APH
NYSE:ARE
NYSE:ARNC
NYSE:AVB
NYSE:AVY
NYSE:AXP
NYSE:AYI
NYSE:AZO
NYSE:BA
NYSE:BAC
NYSE:BAX
NYSE:BBT
NYSE:BBY
NYSE:BCR
NYSE:BDX
NYSE:BEN
NYSE:BF.B
NYSE:BHI
NYSE:BK
NYSE:BLL
NYSE:BMY
NYSE:BRK.B
NYSE:BSX
NYSE:BWA
NYSE:BXP
NYSE:C
NYSE:CAG
NYSE:CAH
NYSE:CAT
NYSE:CB
NYSE:CBG
NYSE:CBS
NYSE:CCI
NYSE:CCL
NYSE:CF
NYSE:CFG
NYSE:CHD
NYSE:CHK
NYSE:CI
NYSE:CL
NYSE:CLX
NYSE:CMA
NYSE:CMG
NYSE:CMI
NYSE:CMS
NYSE:CNC
NYSE:CNP
NYSE:COF
NYSE:COG
NYSE:COH
NYSE:COL
NYSE:COO
NYSE:COP
NYSE:COTY
NYSE:CPB
NYSE:CRM
NYSE:CSRA
NYSE:CSX
NYSE:CVS
NYSE:CVX
NYSE:CXO
NYSE:D
NYSE:DAL
NYSE:DD
NYSE:DFS
NYSE:DG
NYSE:DGX
NYSE:DHI
NYSE:DHR
NYSE:DIS
NYSE:DLPH
NYSE:DLR
NYSE:DOV
NYSE:DOW
NYSE:DPS
NYSE:DRI
NYSE:DTE
NYSE:DUK
NYSE:DVA
NYSE:DVN
NYSE:DXC
NYSE:ECL
NYSE:ED
NYSE:EFX
NYSE:EIX
NYSE:EL
NYSE:EMN
NYSE:EMR
NYSE:ENDP
NYSE:EOG
NYSE:EQR
NYSE:ES
NYSE:ESS
NYSE:ETN
NYSE:ETR
NYSE:EVHC
NYSE:EW
NYSE:EXC
NYSE:EXR
NYSE:F
NYSE:FBHS
NYSE:FDX
NYSE:FDX

```
In [60]:
corrMatrix = data.corr()
In [61]:
corrMatrix
Out[61]:
```

	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NAS
NASDAQ.AAL	1.000000	0.082065	0.542213	0.209446	0.24
NASDAQ.AAPL	0.082065	1.000000	0.714578	0.264269	0.26
NASDAQ.ADBE	0.542213	0.714578	1.000000	0.259282	0.47
NASDAQ.ADI	0.209446	0.264269	0.259282	1.000000	-0.08
NASDAQ.ADP	0.245801	0.265641	0.476496	-0.085074	1.00

NASDAQ.ADSK	0.610881	0.645233	0.872353	0.470756	0.
NASDAQ.AKAM	-0.441142	-0.712522	-0.759604	-0.323603	-0
NASDAQ.ALXN	0.009581	0.326712	0.488717	-0.409747	0.
NASDAQ.AMAT	0.681863	0.545541	0.733609	0.504019	0.09
NASDAQ.AMD	0.272518	-0.238091	0.201821	0.042083	0.44

NASDAQ.AMGN	0.528900	0.131275	0.644373	-0.142261	0.
NASDAQ.AMZN	0.803319	0.370111	0.730719	0.386619	0.
NASDAQ.ATVI	0.593574	0.704358	0.961857	0.321411	0.
NASDAQ.AVGO	0.660926	0.671936	0.906947	0.354330	0.
NASDAQ.BBBY	-0.639629	-0.531976	-0.919636	-0.117133	-0
NASDAQ.BIIB	0.096628	0.279693	0.546833	-0.199210	0.

NASDAQ.CA	0.468258	-0.166888	0.365780	-0.045582	-0
NASDAQ.CBOE	0.481325	0.637467	0.931362	0.104149	0.
NASDAQ.CELG	0.404745	0.133267	0.607426	-0.170338	0.
NASDAQ.CERN	0.567155	0.314661	0.584149	0.281608	-0
NASDAQ.CHRW	-0.681113	-0.514152	-0.751792	-0.224910	-0
NASDAQ.CHTR	0.102313	0.671804	0.715841	-0.044414	0.

NASDAQ.CINF	0.211496	0.546814	0.694139	-0.146740	0.
NASDAQ.CMCSA	0.354914	0.586400	0.636358	0.403285	0.
NASDAQ.CME	0.418846	0.281176	0.671714	0.003393	0.
NASDAQ.COST	-0.409006	-0.255595	-0.661589	0.037644	-0.4
NASDAQ.CSCO	-0.583706	-0.268124	-0.635736	-0.290632	-0.3

	NASDAQ.CSX	0.662241	-0.065778	0.227743	0.274731	-0
	NASDAQ.CTAS	0.351587	0.511664	0.806725	0.097947	0.
	NASDAQ.CTSH	0.639932	0.693353	0.943406	0.273389	0.
	...	...	...	...	...	...
	NYSE.USB	0.523509	0.297190	0.514964	-0.059792	0.
NYSE.UTX		0.698371	0.068743	0.356371	0.243140	-0

NYSE.V	0.385453	0.781096	0.951404	0.140149	0.
NYSE.VAR	0.717439	0.308028	0.755542	0.367813	0.
NYSE.VFC	0.174225	0.559299	0.719638	-0.136562	0.
NYSE.VLO	0.349937	0.107434	0.434775	-0.193338	0.
NYSE.VMC	0.382945	-0.360217	-0.179305	0.202237	-0
NYSE.VNO	-0.267094	-0.792875	-0.880266	-0.056424	-0

NYSE.VTR	0.494899	-0.061758	0.460024	0.357921	0.
NYSE.VZ	-0.586295	0.211194	-0.126482	-0.140572	0.
NYSE.WAT	0.688989	0.388563	0.719185	0.263736	0.
NYSE.WEC	0.211689	0.666012	0.783377	0.234131	0.
NYSE.WFC	0.205066	-0.577962	-0.317679	-0.246926	-0
NYSE.WHR	0.665643	-0.259964	0.102225	0.125038	-0

NYSE.WM	0.275277	0.484494	0.754316	-0.064980	0.
NYSE.WMB	0.231421	-0.059534	0.015326	-0.273027	0.
NYSE.WMT	0.335280	0.761047	0.684392	0.164550	0.
NYSE.WRK	0.836602	0.198634	0.719864	0.163785	0.37
NYSE.WU	-0.409519	-0.227968	-0.470321	-0.237112	0.19

NYSE.WY	-0.166749	-0.739510	-0.668333	-0.060394	-0
NYSE.WYN	0.798624	0.306664	0.718851	0.203224	0.
NYSE.XEC	-0.744085	-0.250959	-0.792876	-0.160317	-0
NYSE.XEL	0.284370	0.738655	0.850709	0.308948	0.
NYSE.XL	0.810955	0.266994	0.722161	0.050697	0.
NYSE.XOM	-0.070950	-0.671488	-0.758967	-0.024428	-0

NYSE.XRX	0.522341	0.341687	0.806615	-0.017967	0.58
NYSE.XYL	0.395730	0.693062	0.911106	0.067258	0.63
NYSE.YUM	0.642336	0.627531	0.939185	0.261173	0.41
NYSE.ZBH	0.588241	-0.580055	-0.067235	0.130786	-0.2(
NYSE.ZTS	0.753567	0.442600	0.776052	0.351893	0.16(

500 rows × 500 columns

```
In [127]:
corr = corrMatrix
c1 = corr.unstack()
sorted = c1.sort_values(ascending = False)
stdf = pd.DataFrame({'sorted':sorted})
In [128]:
sortedCorr = stdf.sort_values('sorted', ascending=True)
In [142]:
sortCorr = stdf.sorted
sortCorr = sortCorr.reset_index()
```

The stocks which are moving together i.e. they are closely related

```
In [144]:
sortCorr.loc[(sortCorr.sorted > 0.7) & (sortCorr.sorted < 1.0)]
Out[144]:
```

	level_0	level_1	sorted
500	NASDAQ.GOOG	NASDAQ.GOOGL	0.998352
501	NASDAQ.GOOGL	NASDAQ.GOOG	0.998352
502	NASDAQ.DISCA	NASDAQ.DISCK	0.996855
503	NASDAQ.DISCK	NASDAQ.DISCA	0.996855
504	NASDAQ.NWSA	NASDAQ.NWS	0.996612
505	NASDAQ.NWS	NASDAQ.NWSA	0.996612
506	NASDAQ.FOXA	NASDAQ.FOX	0.996182
507	NASDAQ.FOX	NASDAQ.FOXA	0.996182
508	NYSE.DD	NYSE.DOW	0.986025
509	NYSE.DOW	NYSE.DD	0.986025
510	NYSE.UA	NYSE.UAA	0.983613
511	NYSE.UAA	NYSE.UA	0.983613
512	NYSE.DVN	NYSE.MRO	0.983269
513	NYSE.MRO	NYSE.DVN	0.983269
514	NYSE.CMS	NYSE.ES	0.981819
515	NYSE.ES	NYSE.CMS	0.981819
516	NYSE.DTE	NYSE.CMS	0.981701
517	NYSE.CMS	NYSE.DTE	0.981701
518	NYSE.LMT	NYSE.RTN	0.981134
519	NYSE.RTN	NYSE.LMT	0.981134
520	NYSE.MA	NYSE.V	0.980574
521	NYSE.V	NYSE.MA	0.980574
522	NYSE.RIG	NYSE.DVN	0.980150
523	NYSE.DVN	NYSE.RIG	0.980150
524	NASDAQ.ADBE	NYSE.MA	0.979629
525	NYSE.MA	NASDAQ.ADBE	0.979629
526	NASDAQ.ETFC	NYSE.BK	0.978342
527	NYSE.BK	NASDAQ.ETFC	0.978342
528	NYSE.DTE	NYSE.ES	0.978332
529	NYSE.ES	NYSE.DTE	0.978332

...	...	...	...
34018	NYSE.MHK	NYSE.UNM	0.700162
34019	NYSE.UNM	NYSE.MHK	0.700162
34020	NYSE.MOS	NYSE.HCA	0.700150
34021	NYSE.HCA	NYSE.MOS	0.700150
34022	NASDAQ.ROST	NYSE.HOG	0.700144
34023	NYSE.HOG	NASDAQ.ROST	0.700144
34024	NYSE.SNA	NASDAQ.KHC	0.700137
34025	NASDAQ.KHC	NYSE.SNA	0.700137
34026	NYSE.AGN	NASDAQ.PBCT	0.700136
34027	NASDAQ.PBCT	NYSE.AGN	0.700136
34028	NYSE.AEP	NYSE.AME	0.700134
34029	NYSE.AME	NYSE.AEP	0.700134
34030	NYSE.CAT	NYSE.AEP	0.700102
34031	NYSE.AEP	NYSE.CAT	0.700102
34032	NASDAQ.EBAY	NYSE.YUM	0.700083
34033	NYSE.YUM	NASDAQ.EBAY	0.700083
34034	NYSE.CB	NYSE.PM	0.700078
34035	NYSE.PM	NYSE.CB	0.700078
34036	NYSE.MMM	NYSE.COH	0.700066
34037	NYSE.COH	NYSE.MMM	0.700066
34038	NYSE.NOC	NYSE.SWK	0.700064
34039	NYSE.SWK	NYSE.NOC	0.700064
3404	NYSE.FTV	NYSE.ICE	0.700055

0			
34041	NYSE.ICE	NYSE.FTV	0.700055
34042	NYSE.DHI	NYSE.NEE	0.700049
34043	NYSE.NEE	NYSE.DHI	0.700049
34044	NYSE.IVZ	NYSE.ARE	0.700049
34045	NYSE.ARE	NYSE.IVZ	0.700049
34046	NYSE.ALB	NASDAQ.NVDA	0.700005
34047	NASDAQ.NVDA	NYSE.ALB	0.700005

33548 rows × 3 columns

The stocks which are moving different to each other i.e. they are negatively co-related

In [145]:
```
sortCorr.loc[(sortCorr.sorted < -0.7)]
```

Out[145]:

	level_0	level_1	sorted
231474	NASDAQ.DISCK	NYSE.LLL	-0.700006
231475	NYSE.LLL	NASDAQ.DISCK	-0.700006
231476	NYSE.LNT	NYSE.M	-0.700007
231477	NYSE.M	NYSE.LNT	-0.700007
231478	NYSE.DPS	NYSE.BAX	-0.700029
231479	NYSE.BAX	NYSE.DPS	-0.700029
231480	NYSE.MMC	NYSE.GWW	-0.700033
231481	NYSE.GWW	NYSE.MMC	-0.700033
231482	NYSE.PNC	NASDAQ.DISCK	-0.700052
231483	NASDAQ.DISCK	NYSE.PNC	-0.700052
231484	NYSE.FMC	NYSE.JCI	-0.700073
231485	NYSE.JCI	NYSE.FMC	-0.700073

231486	NYSE.AMT	NYSE.UHS	-0.700079
231487	NYSE.UHS	NYSE.AMT	-0.700079
231488	NYSE.PHM	NYSE.LB	-0.700083
231489	NYSE.LB	NYSE.PHM	-0.700083
231490	NYSE.CBG	NYSE.BHI	-0.700088
231491	NYSE.BHI	NYSE.CBG	-0.700088
231492	NYSE.CXO	NYSE.CLX	-0.700119
231493	NYSE.CLX	NYSE.CXO	-0.700119
231494	NYSE.KR	NYSE.IVZ	-0.700146
231495	NYSE.IVZ	NYSE.KR	-0.700146
231496	NYSE.AFL	NYSE.COP	-0.700167
231497	NYSE.COP	NYSE.AFL	-0.700167
231498	NYSE.T	NYSE.IVZ	-0.700201
231499	NYSE.IVZ	NYSE.T	-0.700201
231500	NASDAQ.AKAM	NYSE.TDG	-0.700221
231501	NYSE.TDG	NASDAQ.AKAM	-0.700221
231502	NASDAQ.NDAQ	NYSE.UHS	-0.700238
231503	NYSE.UHS	NASDAQ.NDAQ	-0.700238
...	...	...	...
249970	NASDAQ.ISRG	NYSE.APC	-0.957076
249971	NYSE.APC	NASDAQ.ISRG	-0.957076
249972	NYSE.CMG	NYSE.PVH	-0.957122
249973	NYSE.PVH	NYSE.CMG	-0.957122
24997	NYSE.YUM	NYSE.APC	-0.957417

4			
249975	NYSE.APC	NYSE.YUM	-0.957417
249976	NYSE.NEE	NYSE.HP	-0.957630
249977	NYSE.HP	NYSE.NEE	-0.957630
249978	NYSE.NOC	NYSE.RRC	-0.957893
249979	NYSE.RRC	NYSE.NOC	-0.957893
249980	NYSE.MCD	NYSE.SLB	-0.959518
249981	NYSE.SLB	NYSE.MCD	-0.959518
249982	NYSE.ANTM	NYSE.APC	-0.959559
249983	NYSE.APC	NYSE.ANTM	-0.959559
249984	NYSE.BLK	NYSE.XEC	-0.962016
249985	NYSE.XEC	NYSE.BLK	-0.962016
249986	NYSE.NEE	NYSE.RRC	-0.962443
249987	NYSE.RRC	NYSE.NEE	-0.962443
249988	NYSE.HUM	NYSE.NBL	-0.963759
249989	NYSE.NBL	NYSE.HUM	-0.963759
249990	NYSE.MCD	NYSE.APC	-0.963995
249991	NYSE.APC	NYSE.MCD	-0.963995
249992	NYSE.MRO	NASDAQ.ISRG	-0.965143
249993	NASDAQ.ISRG	NYSE.MRO	-0.965143
249994	NASDAQ.ISRG	NYSE.SLB	-0.966186
249995	NYSE.SLB	NASDAQ.ISRG	-0.966186
249996	NYSE.KSU	NYSE.NFX	-0.967553
24999	NYSE.NFX	NYSE.KSU	-0.967553

7			
249998	NYSE.APC	NYSE.AET	-0.976916
249999	NYSE.AET	NYSE.APC	-0.976916