In [1]:

```
#Your objective is to identify the 5 strongest pairs for every year in the dataset (eg.
5strongest pairs for 2014, 2015 and so on)
```

In [ ]:

```
import pandas as pd
import datetime as dt
import seaborn as sns
from matplotlib import pyplot as plt
plt.style.use('ggplot')
%matplotlib inline
```

In [2]:

```
cd C:\Users\harsha.teja\Desktop\myg\congitensor
```

```
C:\Users\harsha.teja\Desktop\myg\congitensor
```

In [3]:

```
df=pd.read_csv("cs-1.csv")
```

In [4]:

```
df['date'] = pd.to_datetime(df['date'])
df['year'] =df['date'].dt.year
```

In [50]:

```
df.isnull().sum()
```

Out[50]:

```
date       0
open      11
high       8
low        8
close      0
volume     0
Name       0
year       0
dtype: int64
```

In [52]:

```
df = df.dropna()
```

In [53]:

```
#To compare two stocks, both should follow the same date range of data. I found that so
me stocks had a smaller date range:
```

In [54]:

```
df.Name.value_counts()
```

Out[54]:

```
PCLN    1259
AES     1259
PBCT    1259
CINF    1259
FRT     1259
        ... 
DXC      215
BHGE     152
BHF      142
DWDP     109
APTV      44
Name: Name, Length: 505, dtype: int64
```

In [55]:

```
#Let’s remove these sets of stocks to continue with further processing:
```

In [56]:

```
count_df=pd.DataFrame(df.Name.value_counts()[:470], columns=["Name", "Count"]).reset_in
dex()
list_valid_shares=list(count_df["index"])
final_df=df[df.Name.isin(list_valid_shares)]
```

In [57]:

```
final_df.head()
```

Out[57]:

	date	open	high	low	close	volume	Name	year
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL	2013
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL	2013
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL	2013
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL	2013
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL	2013

In [58]:

```
#We have the data from 2013-2019. Our goal is to find the most similar stock for any sp
ecific year. We will take the data for the year 2018:
```

In [59]:

```
data_by_year=final_df.groupby("year")
```

In [60]:

```
data_2018=data_by_year.get_group(2018)
#Let’s make the date column the index and make our data pivot for comparing different stocks:
pivot_df=data_2018.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:475]
print("5 strongest pairs for every year 2018")
sc
```

```
5 strongest pairs for every year 2018
```

Out[60]:

		Value
Name	**Name**	
DISCA	**DISCK**	0.998295
DISCK	**DISCA**	0.998295
FRT	**REG**	0.993388
REG	**FRT**	0.993388
UPS	**PH**	0.989953

In [ ]:

```

```

In [61]:

```
data_2013=data_by_year.get_group(2013)
#Let’s make the date column the index and make our data pivot for comparing different stocks:
pivot_df=data_2013.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:475]
print("5 strongest pairs for every year 2013")
sc
```

```
5 strongest pairs for every year 2013
```

Out[61]:

		Value
Name	**Name**	
LMT	**RTN**	0.993692
RTN	**LMT**	0.993692
LMT	**NOC**	0.992921
NOC	**LMT**	0.992921
RTN	**NOC**	0.991203

In [67]:

```
corr_mat
```

Out[67]:

Name	A	AAL	AAP	AAPL	ABBV	ABC	ABT	ACN	A
Name									
A	1.000000	0.161697	0.372094	0.443275	0.873121	0.412925	0.354650	0.780226	0.82
AAL	0.161697	1.000000	0.291644	0.576478	0.278931	0.125100	0.074914	0.022125	0.14
AAP	0.372094	0.291644	1.000000	0.226897	0.416673	0.045663	0.245645	0.412514	0.20
AAPL	0.443275	0.576478	0.226897	1.000000	0.298168	0.120886	0.339397	0.474318	0.71
ABBV	0.873121	0.278931	0.416673	0.298168	1.000000	0.164412	0.521742	0.628278	0.65
...	...	...	...	...	...	...	...	...	
XYL	0.894588	0.111022	0.370479	0.636623	0.736402	0.501213	0.215704	0.794941	0.87
YUM	0.407226	0.639214	0.029528	0.001899	0.542437	0.128417	0.580547	0.160886	0.19
ZBH	0.740056	0.566519	0.089219	0.185397	0.802021	0.107214	0.569263	0.468874	0.54
ZION	0.605753	0.556991	0.464868	0.684607	0.404937	0.465808	0.094535	0.695618	0.74
ZTS	0.885419	0.067568	0.369966	0.617237	0.815330	0.287857	0.394482	0.743293	0.87

470 rows × 470 columns

In [66]:

```
sorted_corr
```

Out[66]:

```
Name  Name
ZTS   ZTS     1.000000
NEM   NEM     1.000000
NUE   NUE     1.000000
NTRS  NTRS    1.000000
NTAP  NTAP    1.000000
                ...
HD    JNPR    0.000033
CAT   CNC     0.000014
CNC   CAT     0.000014
GIS   HUM     0.000013
HUM   GIS     0.000013
Length: 220900, dtype: float64
```

In [70]:

```
data_2014=data_by_year.get_group(2014)
#Let’s make the date column the index and make our data pivot for comparing different stocks:
pivot_df=data_2014.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:480]
print("5 strongest pairs for every year 2013")
sc
```

```
5 strongest pairs for every year 2013
```

Out[70]:

		Value
Name	**Name**	
DISCK	**DISCA**	0.996228
DISCA	**DISCK**	0.996228
XEL	**CMS**	0.989159
CMS	**XEL**	0.989159
UDR	**ESS**	0.987893
ESS	**UDR**	0.987893
EQR	**AVB**	0.986714
AVB	**EQR**	0.986714
ESS	**EQR**	0.985961
EQR	**ESS**	0.985961

In [69]:

```
data_2015=data_by_year.get_group(2015)
#Let’s make the date column the index and make our data pivot for comparing different stocks:
pivot_df=data_2015.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:475]
print("5 strongest pairs for every year 2015")
sc
```

```
5 strongest pairs for every year 2015
```

Out[69]:

		Value
Name	**Name**	
DVN	**MRO**	0.984758
MRO	**DVN**	0.984758
AMZN	**TSS**	0.981945
TSS	**AMZN**	0.981945
NRG	**CMI**	0.979897

In [64]:

```
data_2016=data_by_year.get_group(2016)
#Let’s make the date column the index and make our data pivot for comparing different s
tocks:
pivot_df=data_2016.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair
trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:475]
print("5 strongest pairs for every year 2016")
sc
```

```
5 strongest pairs for every year 2016
```

Out[64]:

		Value
Name	**Name**	
CMA	**ZION**	0.991495
ZION	**CMA**	0.991495
	JPM	0.990534
JPM	**ZION**	0.990534
RF	**JPM**	0.989969

In [65]:

```
data_2017=data_by_year.get_group(2016)
#Let’s make the date column the index and make our data pivot for comparing different stocks:
pivot_df=data_2017.pivot(index="date",columns="Name", values="close")
#Finding Similarities
#To find the similarities, we will use pandas’s corr method:
corr_mat=pivot_df.corr(method ='pearson').apply(lambda x : x.abs())
#select the top ten pairs to give us the top five relationships between stocks for pair trading:
sorted_corr = corr_mat.unstack().sort_values(kind="quicksort", ascending=False)
sc=pd.DataFrame(sorted_corr, columns=["Value"])[470:475]
print("5 strongest pairs for every year 2017")
sc
```

```
5 strongest pairs for every year 2017
```

Out[65]:

		Value
Name	**Name**	
CMA	**ZION**	0.991495
ZION	**CMA**	0.991495
	JPM	0.990534
JPM	**ZION**	0.990534
RF	**JPM**	0.989969

In [ ]:

```
BY Harsha
```