Finance Capstone Project

January 16, 2020

In this data project, I focus on exploratory data analysis of bank stock prices and see how they progressed throughout the financial crisis all the way to early 2016.

[6]:
```
from pandas_datareader import data, wb
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
%matplotlib inline

plt.style.use('ggplot')
```

1 Data

I use stock information for the following banks: * Bank of America * CitiGroup * Goldman Sachs * JPMorgan Chase * Morgan Stanley * Wells Fargo

** Create a list of the ticker symbols (as strings) in alphabetical order.**

[7]:
```
tickers = ['BAC', 'C', 'GS', 'JPM', 'MS', 'WFC']
```

[8]:
```
bank_stocks = pd.read_pickle('all_banks')
```

[9]:
```
bank_stocks.head()
```

[9]:
```
Bank Ticker    BAC                                    C                      \
Stock Info    Open   High    Low  Close    Volume   Open   High    Low  Close
Date
2006-01-03   46.92  47.18  46.15  47.08  16296700  490.0  493.8  481.1  492.9
2006-01-04   47.00  47.24  46.45  46.58  17757900  488.6  491.0  483.5  483.8
2006-01-05   46.58  46.83  46.32  46.64  14970900  484.4  487.8  484.0  486.2
2006-01-06   46.80  46.91  46.35  46.57  12599800  488.8  489.0  482.0  486.2
2006-01-09   46.72  46.97  46.36  46.60  15620000  486.0  487.4  483.0  483.9

Bank Ticker           …     MS                                  WFC         \
Stock Info    Volume  …   Open   High    Low  Close    Volume   Open   High
Date                  …
```

```
2006-01-03   1537660  …  57.17  58.49  56.74  58.31  5377000  31.60  31.98
2006-01-04   1871020  …  58.70  59.28  58.35  58.35  7977800  31.80  31.82
2006-01-05   1143160  …  58.55  58.59  58.02  58.51  5778000  31.50  31.56
2006-01-06   1370250  …  58.77  58.85  58.05  58.57  6889800  31.58  31.78
2006-01-09   1680740  …  58.63  59.29  58.62  59.19  4144500  31.68  31.82

Bank Ticker
Stock Info     Low  Close    Volume
Date
2006-01-03   31.20  31.90  11016400
2006-01-04   31.36  31.53  10871000
2006-01-05   31.31  31.50  10158000
2006-01-06   31.38  31.68   8403800
2006-01-09   31.56  31.68   5619600

[5 rows x 30 columns]
```

2 EDA on stock returns

** What is the max Close price for each bank's stock throughout the time period?**

[10]:
```
bank_stocks.xs('Close', axis=1, level='Stock Info').max(axis=0)
```

[10]:
```
Bank Ticker
BAC     54.90
C      564.10
GS     247.92
JPM     70.08
MS      89.30
WFC     58.52
dtype: float64
```

** Create a new empty DataFrame called returns. This dataframe will contain the returns for each bank's stock. Returns are typically defined by:**

$$r_t = \frac{p_t - p_{t-1}}{p_{t-1}} = \frac{p_t}{p_{t-1}} - 1$$

[11]:
```
returns = pd.DataFrame()
```

** Create a for loop that goes and for each Bank Stock Ticker creates this returns column and set's it as a column in the returns DataFrame.**

[12]:
```
for ticker in tickers:
    returns[ticker+' Return'] = bank_stocks.xs('Close', axis=1, level='Stock␣
 ↪Info')[ticker].pct_change()
```

[13]:
```
returns.head()
```

[13]:

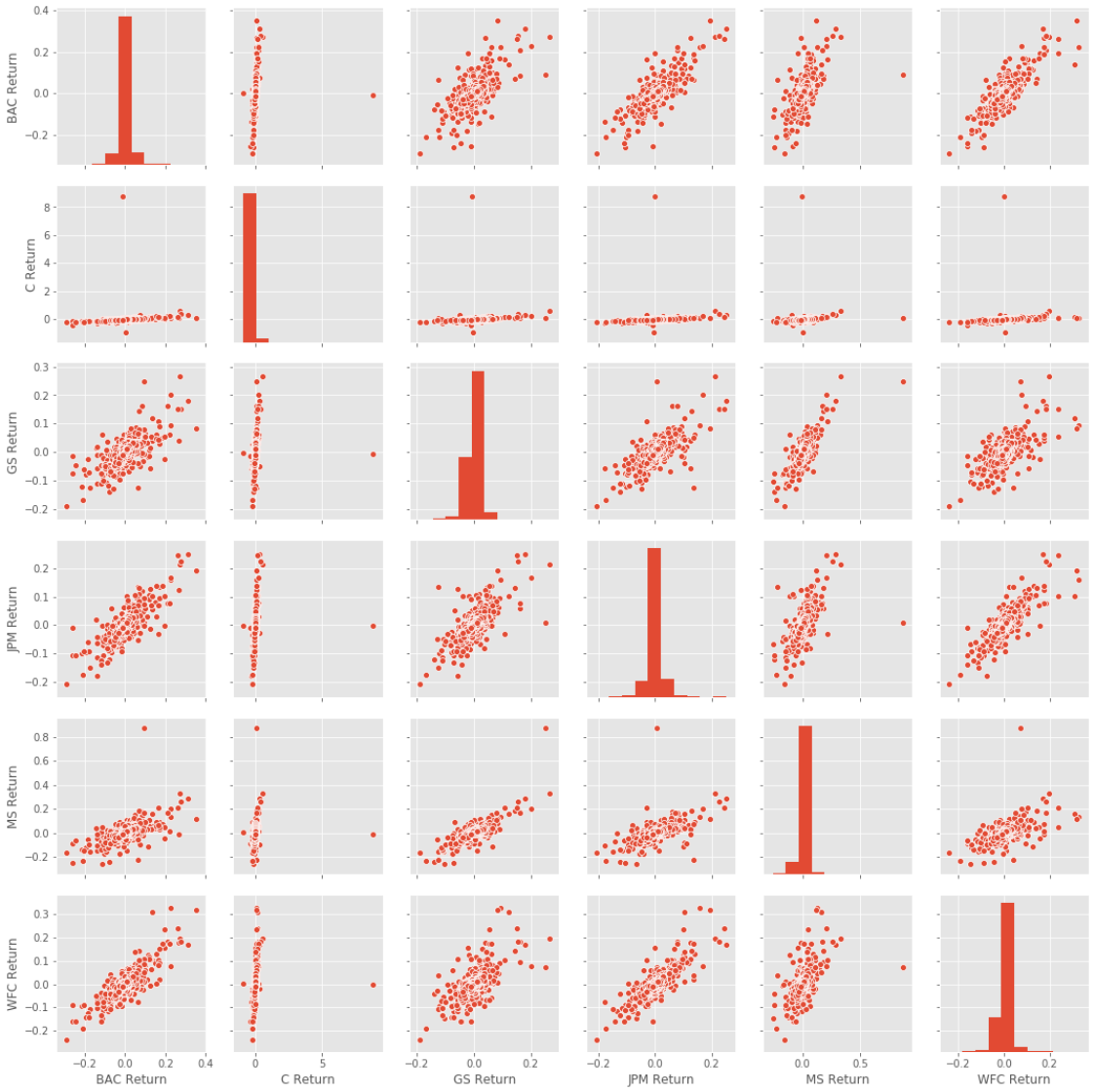
Date	BAC Return	C Return	GS Return	JPM Return	MS Return	WFC Return
2006-01-03	NaN	NaN	NaN	NaN	NaN	NaN
2006-01-04	-0.010620	-0.018462	-0.013812	-0.014183	0.000686	-0.011599
2006-01-05	0.001288	0.004961	-0.000393	0.003029	0.002742	-0.000951
2006-01-06	-0.001501	0.000000	0.014169	0.007046	0.001025	0.005714
2006-01-09	0.000644	-0.004731	0.012030	0.016242	0.010586	0.000000

** Create a pairplot using seaborn of the returns dataframe.**

[14]:
```
sns.pairplot(returns)
```

```
/Users/stella/opt/anaconda3/lib/python3.7/site-
packages/numpy/lib/histograms.py:829: RuntimeWarning: invalid value encountered
in greater_equal
  keep = (tmp_a >= first_edge)
/Users/stella/opt/anaconda3/lib/python3.7/site-
packages/numpy/lib/histograms.py:830: RuntimeWarning: invalid value encountered
in less_equal
  keep &= (tmp_a <= last_edge)
```

[14]:
```
<seaborn.axisgrid.PairGrid at 0x1a19c56390>
```



Noticed that Citigroup has returns concentrated below 0, the variance of its returns data is very small compared to other banks.

** Using this returns DataFrame, figure out on what dates each bank stock had the best and worst single day returns. **

[15]:
```
returns.idxmin()
```

[15]:
```
BAC Return    2009-01-20
C Return      2011-05-06
GS Return     2009-01-20
JPM Return    2009-01-20
MS Return     2008-10-09
WFC Return    2009-01-20
```

```
dtype: datetime64[ns]
```

[16]:
```
returns.idxmax()
```

[16]:
```
BAC Return   2009-04-09
C Return     2011-05-09
GS Return    2008-11-24
JPM Return   2009-01-21
MS Return    2008-10-13
WFC Return   2008-07-16
dtype: datetime64[ns]
```

** Take a look at the standard deviation of the returns, which stock would you classify as the riskiest over the entire time period? Which would you classify as the riskiest for the year 2015?**

[17]:
```
returns.std()
```

[17]:
```
BAC Return    0.036650
C Return      0.179969
GS Return     0.025346
JPM Return    0.027656
MS Return     0.037820
WFC Return    0.030233
dtype: float64
```

The riskiest stock over the entire 10 years seems to be Citigroup, since the standard deviation of its returns is significantly bigger than the rest.

[18]:
```
returns.loc['2015-01-01':'2015-12-31'].std()
```

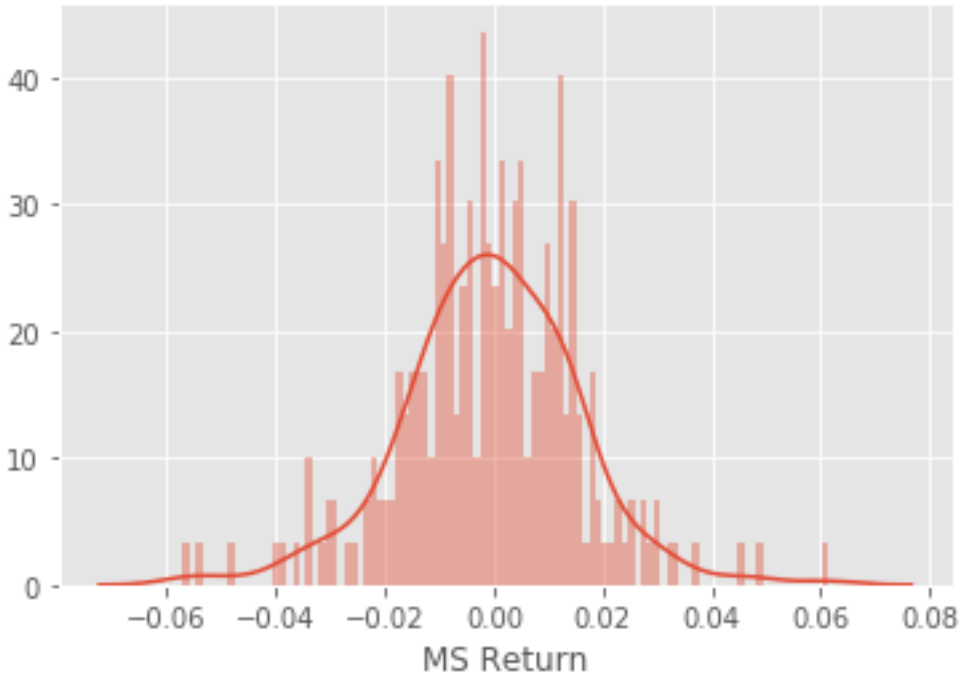
[18]:
```
BAC Return    0.016163
C Return      0.015289
GS Return     0.014046
JPM Return    0.014017
MS Return     0.016249
WFC Return    0.012591
dtype: float64
```

For the year 2015, all banks have similar standard deviations in returns, thus the risks are similar.

** Create a distplot of the 2015 returns for Morgan Stanley **

[19]:
```
sns.distplot(returns['MS Return'].loc['2015-01-01':'2015-12-31'], bins=100)
```

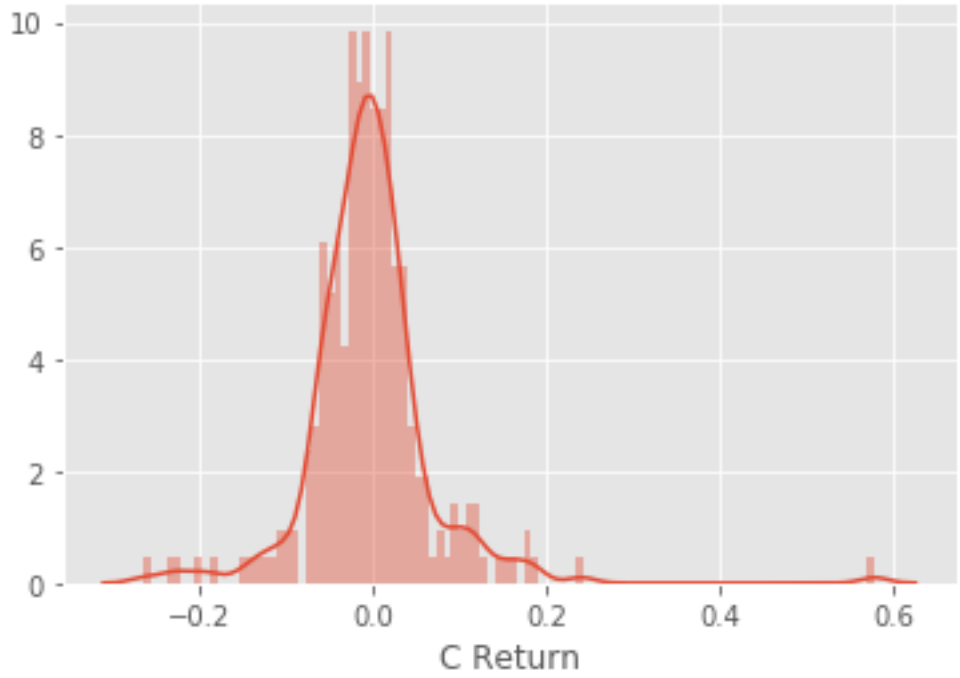
[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1b87a290>
```



** Create a distplot of the 2008 returns for CitiGroup **

[20]:
```
sns.distplot(returns['C Return'].loc['2008-01-01':'2008-12-31'], bins=100)
```

[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1befcc50>
```



3 More Visualization

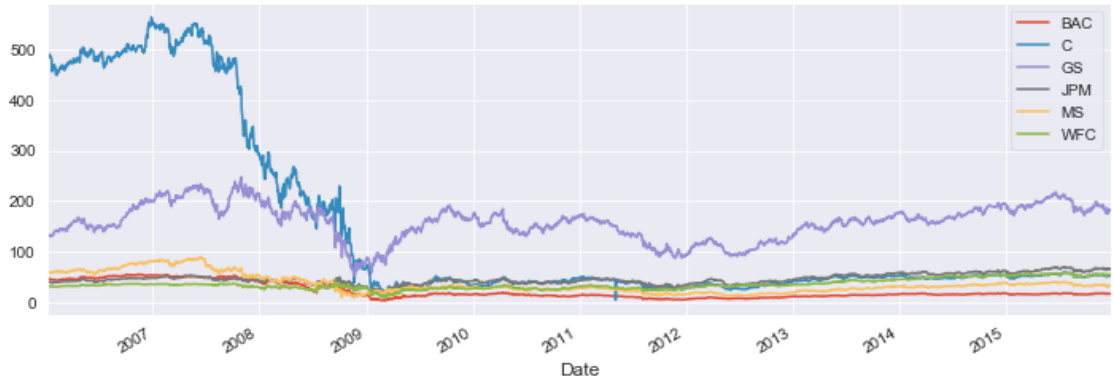
[21]:
```
sns.set_style('darkgrid')

import plotly
import cufflinks as cf
cf.go_offline()
```

** Create a line plot showing Close price for each bank for the entire index of time.**

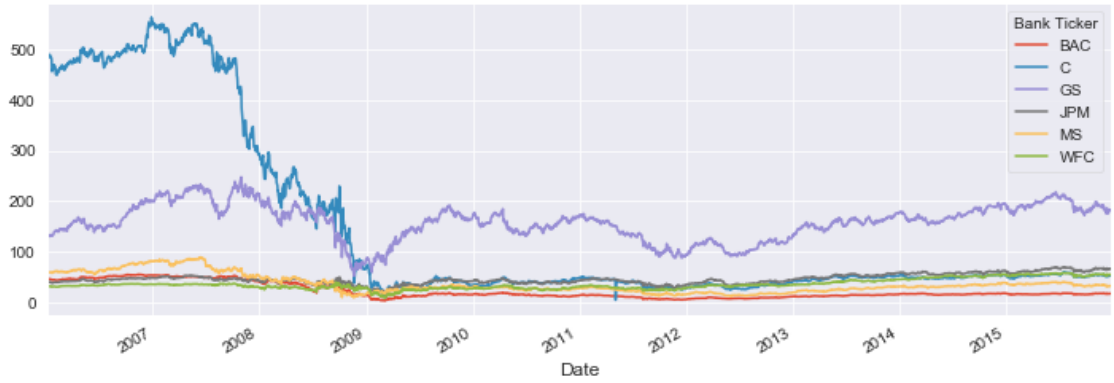
[22]:
```
for tick in tickers:
    bank_stocks[tick]['Close'].plot(figsize=(12,4), label=tick)
plt.legend()
```

[22]:
```
<matplotlib.legend.Legend at 0x1c1f04f050>
```



[23]:
```
bank_stocks.xs(key='Close', axis=1, level='Stock Info').plot(figsize=(12,4))
```

[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c1f31f710>
```



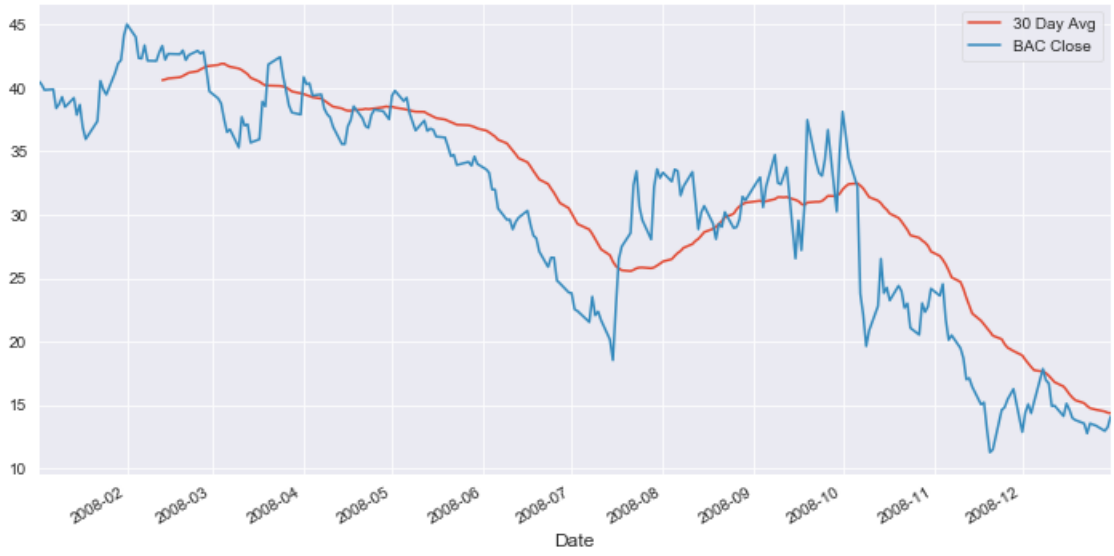
[24]:
```
bank_stocks.xs(key='Close', axis=1, level='Stock Info').iplot()
```

4 Moving Averages

** Plot the rolling 30 day average against the Close Price for Bank Of America's stock for the year 2008**

[25]:
```
bank_stocks['BAC']['Close'].loc['2008-01-01':'2008-12-31'].rolling(window=30).
 ↪mean().plot(figsize=(12,6), label='30 Day Avg')
bank_stocks['BAC']['Close'].loc['2008-01-01':'2008-12-31'].plot(label='BAC␣
 ↪Close')
plt.legend()
```

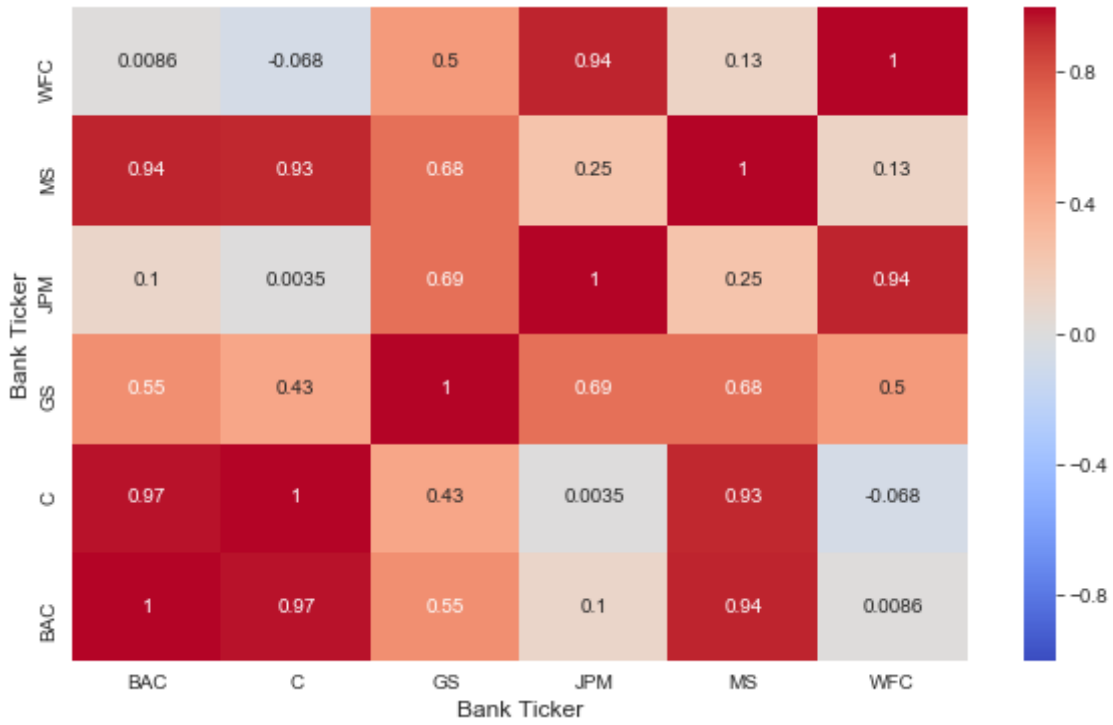
[25]:
```
<matplotlib.legend.Legend at 0x116482890>
```



** Create a heatmap of the correlation between the stocks Close Price.**

[26]:
```
plt.figure(figsize=(10,6))
sns.heatmap(bank_stocks.xs(key='Close', axis=1, level='Stock Info').corr(),␣
 ↪vmin=-1, annot=True, cmap='coolwarm')
plt.ylim(0,6)
```

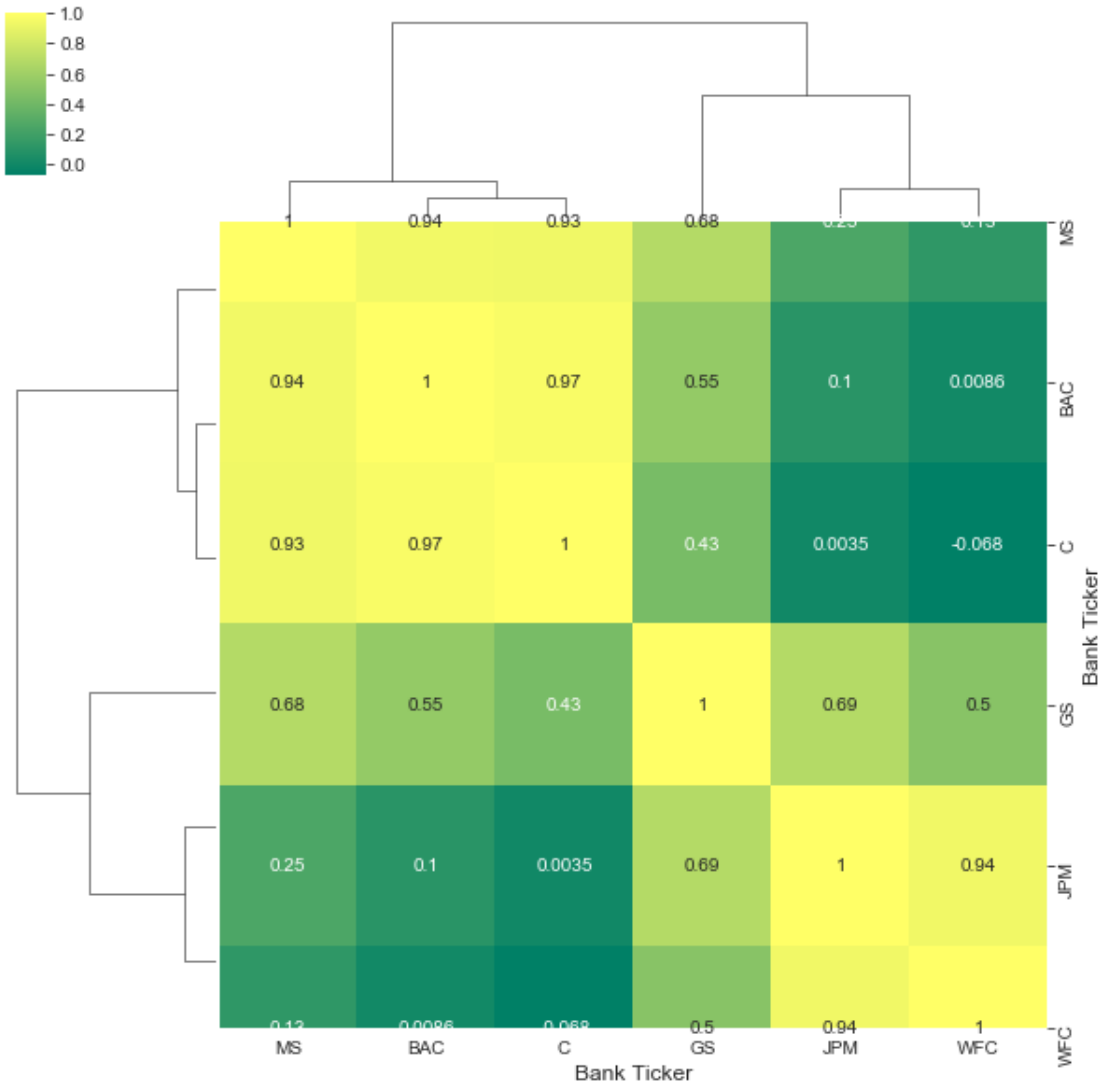
[26]: (0, 6)



** Use seaborn's clustermap to cluster the correlations together:**

[27]:
```
sns.clustermap(bank_stocks.xs(key='Close', axis=1, level='Stock Info').corr(),␣
 ↪cmap='summer', annot=True)
```

[27]: <seaborn.matrix.ClusterGrid at 0x1c20d61090>



[28]:
```
bank_stocks.xs(key='Close', axis=1, level='Stock Info').corr().
↪iplot(kind='heatmap')
```

5 Technical Analysis Plots

** Use .iplot(kind='candle') to create a candle plot of Bank of America's stock from Jan 1st 2015 to Jan 1st 2016.**

[29]:
```
bank_stocks['BAC'].loc['2015-01-01':'2016-01-01'].iplot(kind='candle')
```

** Use .ta_plot(study='sma') to create a Simple Moving Averages plot of Morgan Stanley for the year 2015.**

[30]:
```
bank_stocks['MS']['Close'].loc['2015-01-01':'2016-01-01'].ta_plot(study='sma',␣
↪period = (13,21,55))
```

Use .ta_plot(study='boll') to create a Bollinger Band Plot for Bank of America for the year 2015.

[31]:
```
bank_stocks['BAC']['Close'].ta_plot(study='boll')
```