Stock Price Analysis

February 12, 2018

1 Stock Price Analysis

In this data project we will focus on exploratory data analysis of stock prices. We'll focus on bank stocks and see how they progressed throughout the financial crisis all the way to early 2016.

1.1 Get the Data

We are directly reading data from Google finance using pandas!

```
In [2]: from pandas_datareader import data, wb
        import pandas as pd
        import numpy as np
        import datetime
        %matplotlib inline
```

1.2 Data

We are using pandas datareader to get the data. We will get stock information for the following banks: * Bank of America * CitiGroup * Goldman Sachs * JPMorgan Chase * Morgan Stanley * Wells Fargo

```
In [3]: start = datetime.datetime(2006, 1, 1)
        end = datetime.datetime(2016, 1, 1)
```

```
In [7]: # Bank of America
        BAC = data.DataReader('BAC', 'google', start, end)

        #Citigroup
        C = data.DataReader('C', 'google', start, end)

        #Goldman Sachs
        GS = data.DataReader('GS', 'google', start, end)

        #JPMorgan Chase
        JPM = data.DataReader('JPM', 'google', start, end)

        #Morgan Stanley
```

```
MS = data.DataReader('MS', 'google', start, end)

#Wells Fargo
WFC = data.DataReader('WFC', 'google', start, end)
```

```
c:\python3\lib\site-packages\pandas_datareader\google\daily.py:40: UnstableAPIWarning:
The Google Finance API has not been stable since late 2017. Requests seem
to fail at random. Failure is especially common when bulk downloading.

  warnings.warn(UNSTABLE_WARNING, UnstableAPIWarning)
```

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

```
In [8]: tickers = ['BAC', 'C', 'GS', 'JPM', 'MS', 'WFC']
```

** Use pd.concat to concatenate the bank dataframes together to a single data frame called bank_stocks. Set the keys argument equal to the tickers list. Also pay attention to what axis you concatenate on.**

```
In [10]: bank_stocks = pd.concat([BAC, C, GS, JPM, MS, WFC], axis=1, keys=tickers)
```

```
In [12]: bank_stocks.columns.names = ['Bank Ticker','Stock Info']
```

```
In [14]: bank_stocks.head()
```

Out[14]:

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

Bank Ticker				
Stock Info	High	Low	Close	Volume
Date				
2006-01-03	31.98	31.20	31.90	11016400
2006-01-04	31.82	31.36	31.53	10871000

```
2006-01-05   31.56  31.31  31.50  10158000
2006-01-06   31.78  31.38  31.68   8403800
2006-01-09   31.82  31.56  31.68   5619600

[5 rows x 30 columns]
```

2 Exploratory Data Analysis

Let's explore the data a bit!

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

```
In [17]: bank_stocks.xs(key='Close', axis=1, level='Stock Info').max()

Out[17]: Bank Ticker
         BAC     54.90
         C      564.10
         GS     247.92
         JPM     70.08
         MS      89.30
         WFC     58.52
         dtype: float64
```

** To check returns for each bank's stock**

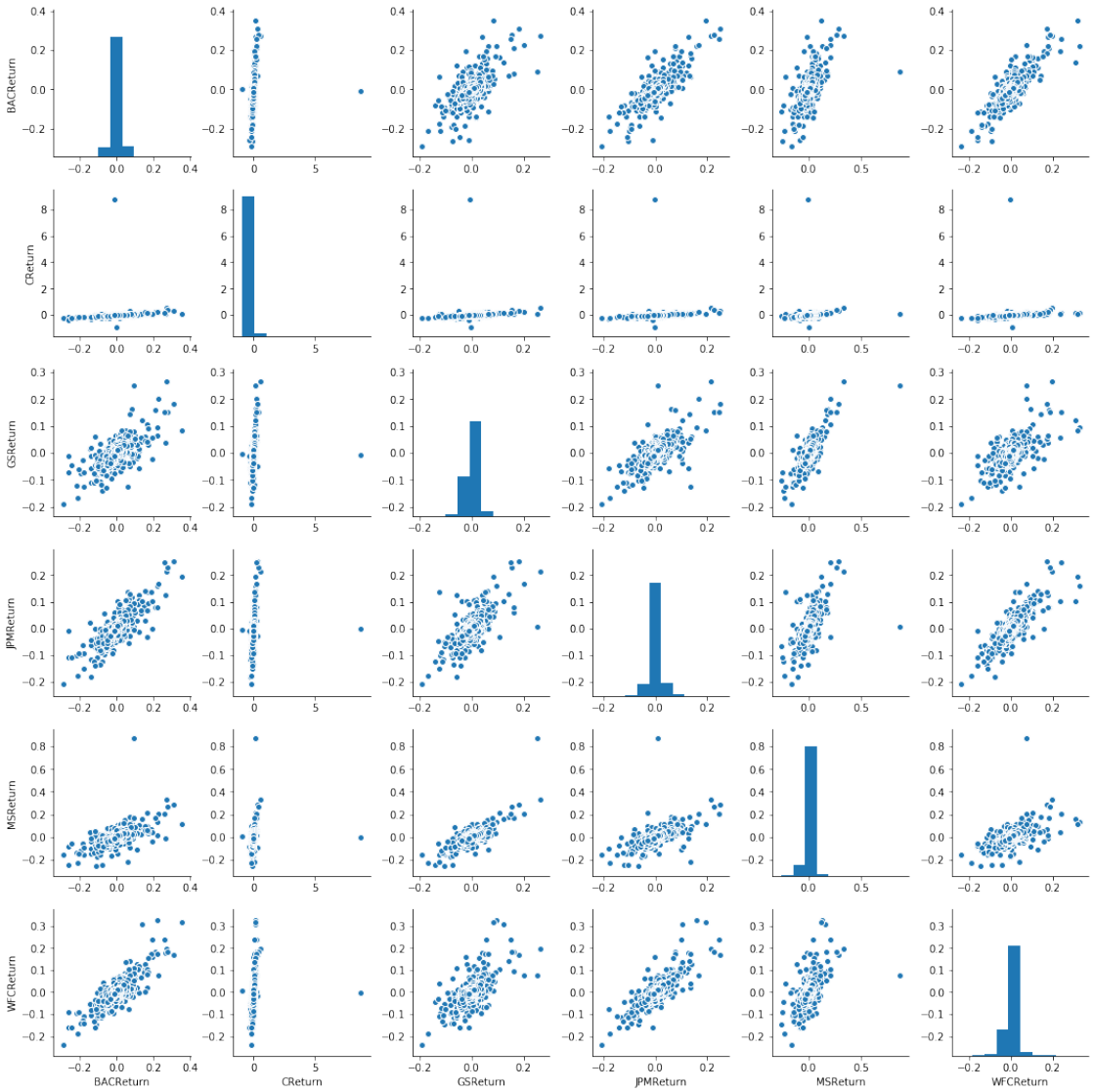
```
In [18]: returns = pd.DataFrame()
```

** We can check for the percent change in the close price **

```
In [21]: for tick in tickers:
             returns[tick + 'Return'] = bank_stocks[tick]['Close'].pct_change()
         returns.head()

Out[21]:             BACReturn   CReturn  GSReturn  JPMReturn  MSReturn  WFCReturn
         Date
         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
```

** Pairplot for the returns data **

```
In [24]: import seaborn as sns
         sns.pairplot(returns[1:])

Out[24]: <seaborn.axisgrid.PairGrid at 0xe66adb0>
```



** We'll check the worst and the best day single day returns for each bank stock **

```
In [28]: returns.idxmin()
```

```
Out[28]: BACReturn   2009-01-20
         CReturn     2011-05-06
         GSReturn    2009-01-20
         JPMReturn   2009-01-20
         MSReturn    2008-10-09
         WFCReturn   2009-01-20
         dtype: datetime64[ns]
```

```
In [29]: returns.idxmax()
```

```
Out[29]: BACReturn   2009-04-09
         CReturn     2011-05-09
         GSReturn    2008-11-24
         JPMReturn   2009-01-21
         MSReturn    2008-10-13
         WFCReturn   2008-07-16
         dtype: datetime64[ns]
```

** We can see that citigroup has minimum and maximum returns days very close to each other. This is because the citi stock had plit 1:10 after its huge drop. **

** We'll check which stock is riskiest over the time period. Turns out Citigroup is the riskiest **

```
In [30]: returns.std()
```

```
Out[30]: BACReturn    0.036650
         CReturn      0.179969
         GSReturn     0.025346
         JPMReturn    0.027656
         MSReturn     0.037820
         WFCReturn    0.030233
         dtype: float64
```

** We'll check which stock is riskiest for a particular year. Below we can see Morgan Stanley was the riskiest stock to purchase in 2015 **

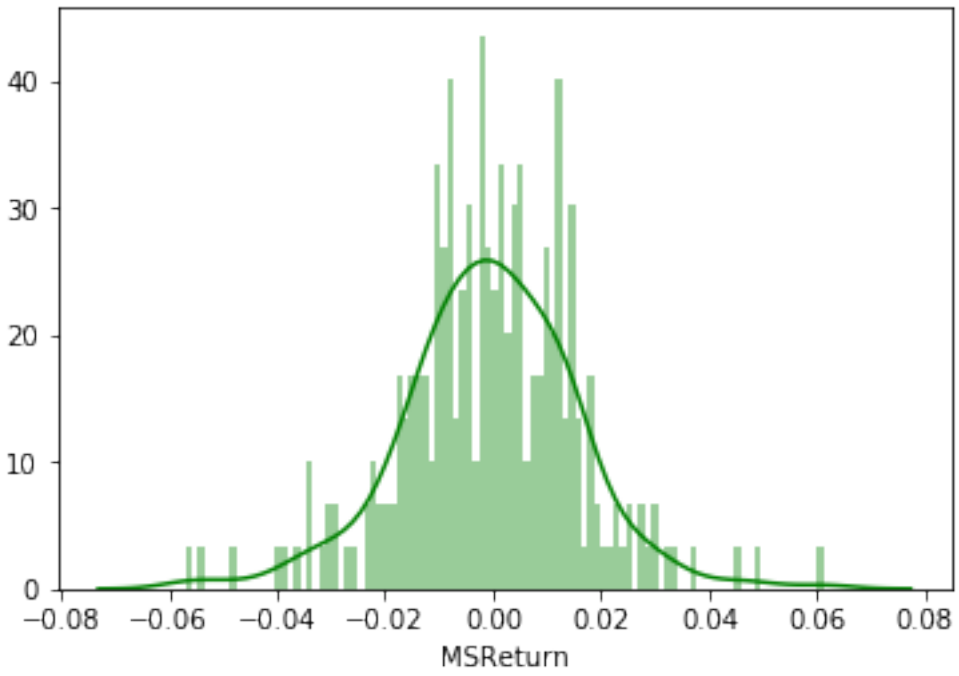
```
In [33]: returns.loc['2015-01-01':'2015-12-31'].std()
```

```
Out[33]: BACReturn    0.016163
         CReturn      0.015289
         GSReturn     0.014046
         JPMReturn    0.014017
         MSReturn     0.016249
         WFCReturn    0.012591
         dtype: float64
```

** distplot using seaborn of the 2015 returns for Morgan Stanley **

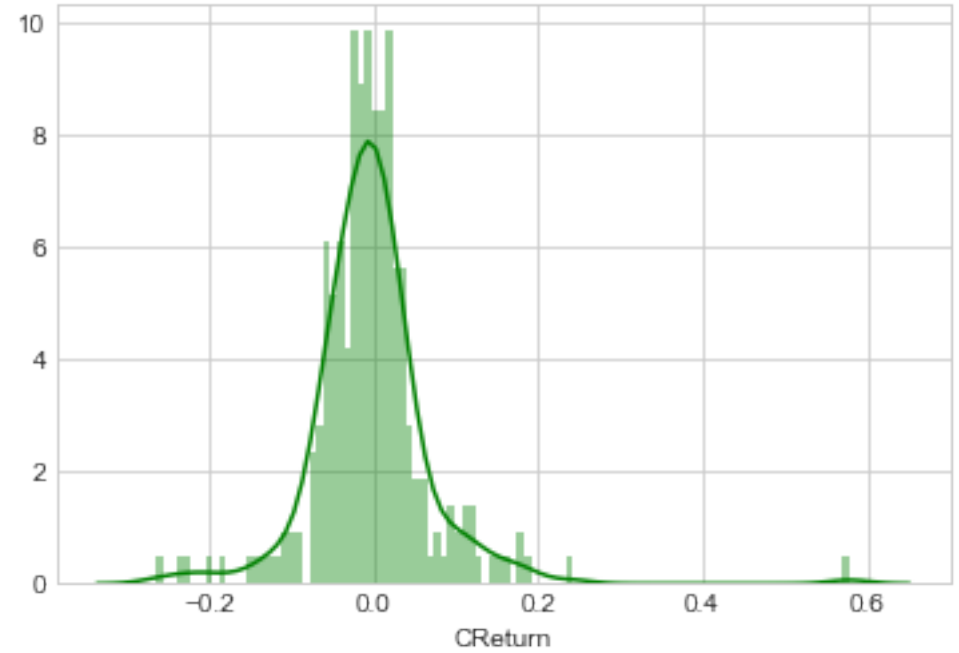
```
In [44]: sns.distplot(returns.loc['2015-01-01':'2015-12-31']['MSReturn'], color = 'green', bins
```

```
Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x81d4e10>
```



** distplot using seaborn of the 2008 returns for CitiGroup **

```
In [49]: sns.distplot(returns.loc['2008-01-01':'2008-12-31']['CReturn'], color = 'green', bins
         sns.set_style('whitegrid')
```

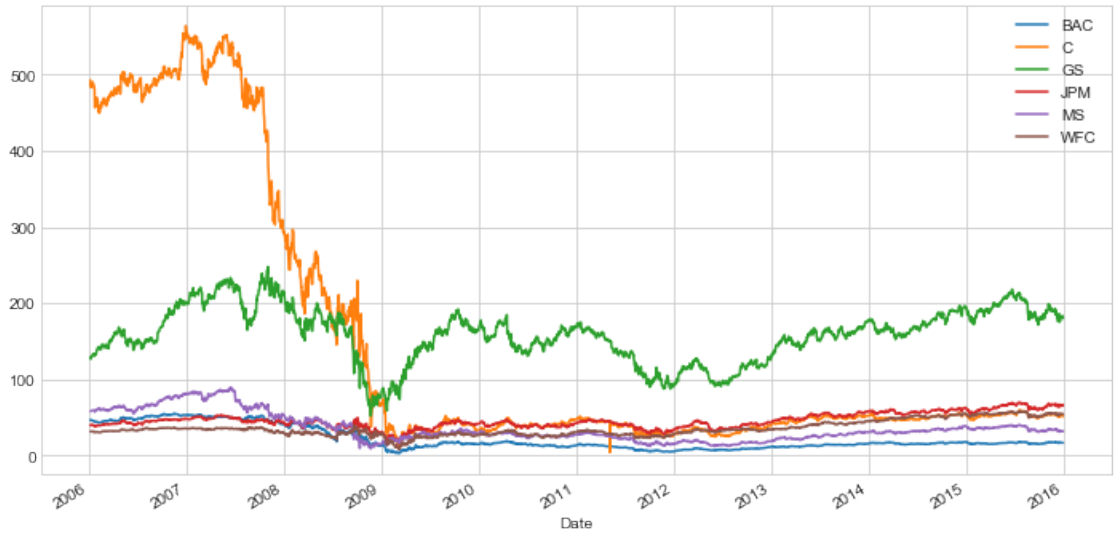


```
In [47]: import matplotlib.pyplot as plt
         import seaborn as sns
         sns.set_style('whitegrid')
         %matplotlib inline

         # Optional Plotly Method Imports
         import plotly
         import cufflinks as cf
         cf.go_offline()
```

** Close price for stock for each bank for entire index of time **

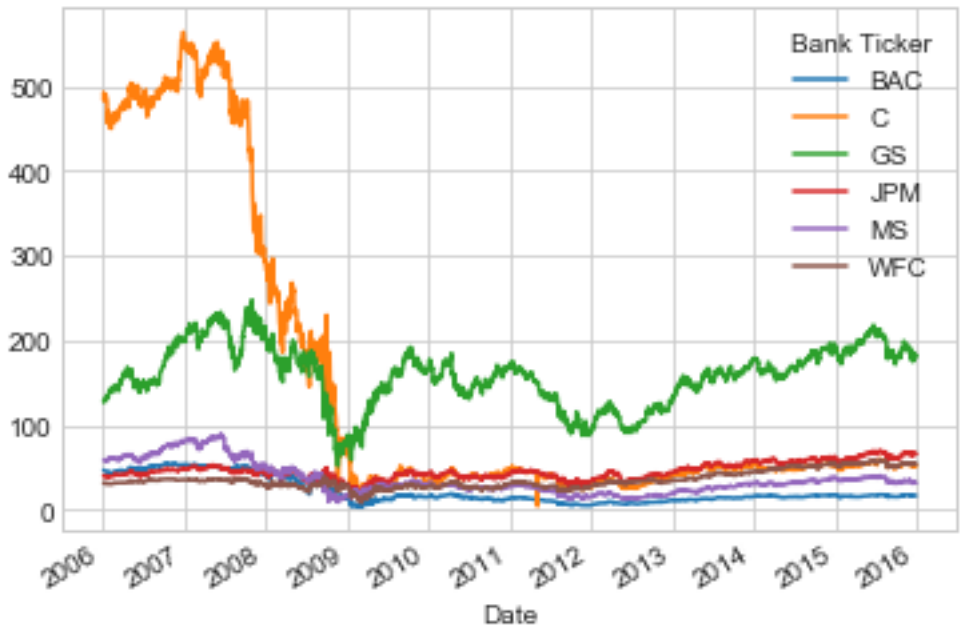
```
In [48]: for i in tickers:
             bank_stocks[i]['Close'].plot(figsize=(12,6), label=i)
         plt.legend()
```

```
Out[48]: <matplotlib.legend.Legend at 0xd9f0cf0>
```




```
In [50]: bank_stocks.xs(key='Close', axis=1, level='Stock Info').plot()
```

```
Out[50]: <matplotlib.axes._subplots.AxesSubplot at 0x81decd0>
```



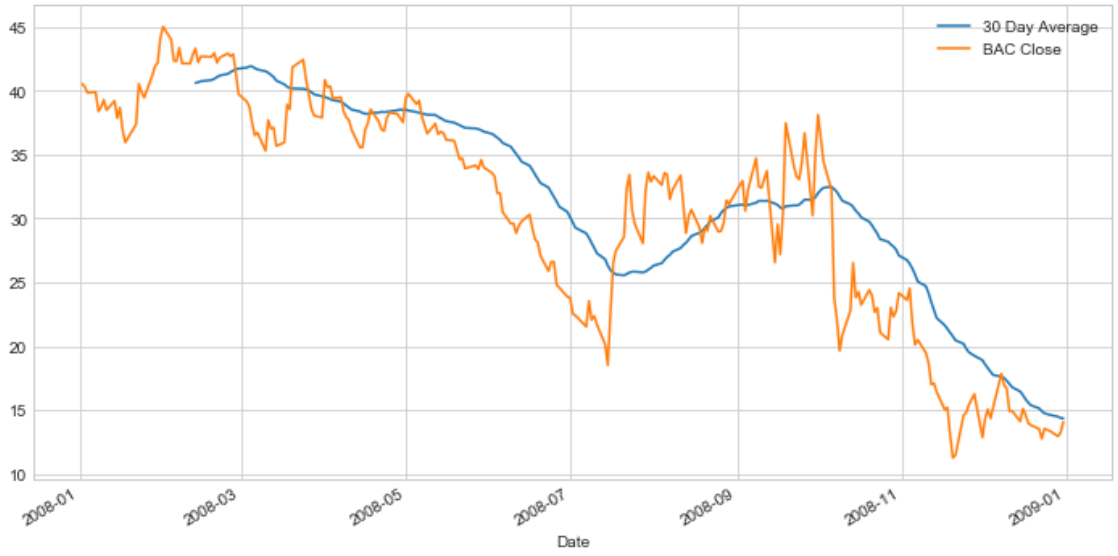

```
In [51]: bank_stocks.xs(key='Close', axis=1, level='Stock Info').iplot()
```

2.1 Moving Averages

Let's analyze the moving averages for these stocks in a year .

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

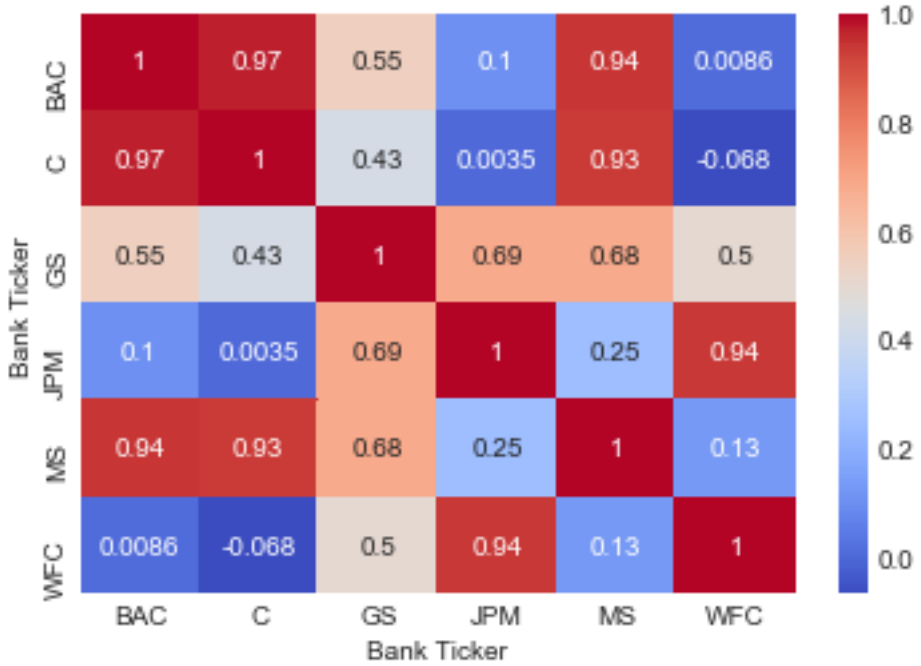
```
In [60]: plt.figure(figsize=(12,6))
         BAC['Close'].loc['2008-01-01':'2008-12-31'].rolling(window=30).mean().plot(label='30 I
         BAC['Close'].loc['2008-01-01':'2008-12-31'].plot(label='BAC Close')
         plt.legend()

Out[60]: <matplotlib.legend.Legend at 0x14740fb0>
```



** Heatmap of the correlation between the stocks Close Price.**

```
In [63]: sns.heatmap(bank_stocks.xs(key='Close', axis = 1, level='Stock Info').corr(), annot=T
```

```
Out[63]: <matplotlib.axes._subplots.AxesSubplot at 0x14730410>
```




```
In [65]: close = bank_stocks.xs(key = 'Close', axis=1, level='Stock Info').corr()
         close.iplot(kind='heatmap', colorscale='rdylbu')
```

** In this second part of the project we will rely on the cufflinks library to create some Technical Analysis plots. **

** candle plot of Bank of America's stock from Jan 1st 2015 to Jan 1st 2016.**

```
In [70]: BAC[['Open', 'High', 'Low', 'Close']].loc['2015-01-01':'2016-01-01'].iplot(kind='candl
```

** Simple Moving Averages plot of Morgan Stanley for the year 2015.**

```
In [71]: MS['Close'].loc['2015-01-01':'2015-12-31'].ta_plot(study='sma', periods=[13,21,55], ti
```

** Bollinger Band Plot for Bank of America for the year 2015.**

```
In [72]: MS['Close'].loc['2015-01-01':'2015-12-31'].ta_plot(study='boll', periods=[13,21,55], t
```