# Playing with Plotly for time series

November 12, 2016

#### 0.0.1 PLOTLY SAMPLES

```
In [22]: # https://plot.ly/ipython-notebooks/cufflinks/
         import datetime
         from IPython.display import display  #from sqlalchemy import create_engine
         import plotly.plotly as py # interactive graphing
         from plotly.graph_objs import * #Bar, Scatter, Marker, Layout, Data, Line,
         import plotly.graph_objs as go
         import plotly.plotly as py
         #from plotly.graph_objs import *
         #py.sign_in('username', 'api_key')
         #import pandas.io.data as web
         from pandas_datareader import data, wb
         import pandas_datareader.data as web
         #pdr.get_data_yahoo('AAPL')

         import pandas as pd
         import numpy as np
         ## Plotting distribution of shopping trips made each day of the week
         import matplotlib.pyplot as plt
         from plotly.graph_objs import Data, Line
         import cufflinks as cf
         print cf.__version__

         import plotly.tools as tls  # load online plot
         tls.embed('https://plot.ly/~cufflinks/8')



         df = cf.datagen.lines()
         df.head()

         py.iplot([{
             'x': df.index,
             'y': df[col],
             'name': col
         }  for col in df.columns], filename='cufflinks-simple-line')
```

```
Out[22]: <plotly.tools.PlotlyDisplay object>

0.8.2


Out[22]:                WYT.KV    VBX.FG    SFC.TL    KRN.RQ    MRW.PX
          2015-01-01 -0.546407 -0.345418 -0.139482 -0.482782 -0.877292
          2015-01-02 -1.440201  0.600512 -1.009671 -0.318871 -1.698958
          2015-01-03 -1.266050  0.349589  0.136959 -2.192919 -2.169261
          2015-01-04 -1.688169  0.132127  0.885331 -4.582110 -1.134107
          2015-01-05 -1.443156  1.054130  0.775650 -5.505717 -2.291436

Out[22]: <plotly.tools.PlotlyDisplay object>

In [7]: def to_unix_time(dt):
            epoch =  datetime.datetime.utcfromtimestamp(0)
            return (dt - epoch).total_seconds() * 1000

        x = [datetime.datetime(year=2013, month=10, day=04),
            datetime.datetime(year=2013, month=11, day=05),
            datetime.datetime(year=2013, month=12, day=06)]
        data = [go.Scatter(
                    x=x,
                    y=[1, 3, 6])]

        layout = go.Layout(xaxis = dict(
                           range = [to_unix_time(datetime.datetime(2013, 10, 17)),
                                    to_unix_time(datetime.datetime(2013, 11, 20))]
            ))

        fig = go.Figure(data = data, layout = layout)
        py.iplot(fig)

Out[7]: <plotly.tools.PlotlyDisplay object>

In [8]: df = web.DataReader("aapl", 'yahoo',
                            datetime.datetime(2007, 10, 1),
                            datetime.datetime(2009, 4, 1))
        df.head()
        trace = go.Scatter(x=df.index,
                           y=df.High)

        data = [trace]
        layout = dict(
            title='Time series with range slider and selectors',
            xaxis=dict(
                rangeselector=dict(
                    buttons=list([
                        dict(count=1,
                             label='1m',
                             step='month',
                             stepmode='backward'),
                        dict(count=6,
                             label='6m',
                             step='month',
                             stepmode='backward'),
                        dict(count=1,
                            label='YTD',
                            step='year',
                            stepmode='todate'),
                        dict(count=1,
                            label='1y',
                            step='year',
                            stepmode='backward'),
                        dict(step='all')
                    ])
                ),
                rangeslider=dict(),
                type='date'
            )
        )

        fig = dict(data=data, layout=layout)
        py.iplot(fig)
```

```
Out[8]:                    Open        High         Low       Close     Volume  \
        Date
        2007-10-01  154.630001  157.409998  152.930002  156.340000  209267100
        2007-10-02  156.550005  158.589998  155.890003  158.449995  198017400
        2007-10-03  157.780006  159.179998  157.010002  157.919996  173129600
        2007-10-04  157.999998  158.080000  153.500002  156.239998  164239600
        2007-10-05  158.370007  161.580000  157.700005  161.449997  235867800

                    Adj Close
        Date
        2007-10-01  20.343159
        2007-10-02  20.617714
        2007-10-03  20.548750
        2007-10-04  20.330146
        2007-10-05  21.008078
```

```
Out[8]: <plotly.tools.PlotlyDisplay object>
```

```
In [9]: df.T.head()
```

```
Out[9]: Date      2007-10-01    2007-10-02    2007-10-03    2007-10-04    2007-10-0
        Open    1.546300e+02  1.565500e+02  1.577800e+02  1.580000e+02  1.583700e+0
        High    1.574100e+02  1.585900e+02  1.591800e+02  1.580800e+02  1.615800e+0
```

```
Low      1.529300e+02  1.558900e+02  1.570100e+02  1.535000e+02  1.577000e+0
Close    1.563400e+02  1.584500e+02  1.579200e+02  1.562400e+02  1.614500e+0
Volume   2.092671e+08  1.980174e+08  1.731296e+08  1.642396e+08  2.358678e+0

Date       2007-10-08    2007-10-09    2007-10-10    2007-10-11    2007-10-1
Open     1.634900e+02  1.702000e+02  1.675500e+02  1.694900e+02  1.630100e+0
High     1.679100e+02  1.711100e+02  1.678800e+02  1.718800e+02  1.672800e+0
Low      1.629700e+02  1.666800e+02  1.656000e+02  1.532100e+02  1.618000e+0
Close    1.679100e+02  1.678600e+02  1.667900e+02  1.622300e+02  1.672500e+0
Volume   2.089822e+08  2.760716e+08  1.668975e+08  4.109980e+08  2.470440e+0

Date         ...         2009-03-19    2009-03-20    2009-03-23    2009-03-2
Open         ...       1.018500e+02  1.020900e+02  1.027100e+02  1.063600e+0
High         ...       1.032000e+02  1.031100e+02  1.081600e+02  1.094400e+0
Low          ...       1.002500e+02  1.005700e+02  1.017500e+02  1.053900e+0
Close        ...       1.016200e+02  1.015900e+02  1.076600e+02  1.065000e+0
Volume       ...       1.250452e+08  1.738968e+08  1.665993e+08  1.601530e+0

Date       2009-03-25    2009-03-26    2009-03-27    2009-03-30    2009-03-3
Open     1.075800e+02  1.078300e+02  1.082300e+02  1.045100e+02  1.054500e+0
High     1.083600e+02  1.099800e+02  1.085300e+02  1.050100e+02  1.074500e+0
Low      1.038600e+02  1.075800e+02  1.064000e+02  1.026100e+02  1.050000e+0
Close    1.064900e+02  1.098700e+02  1.068500e+02  1.044900e+02  1.051200e+0
Volume   1.616545e+08  1.540630e+08  1.232182e+08  1.256990e+08  1.425200e+0

Date       2009-04-01
Open     1.040900e+02
High     1.090000e+02
Low      1.038900e+02
Close    1.086900e+02
Volume   1.473430e+08

[5 rows x 379 columns]
```

```
In [10]: def to_unix_time(dt):
             epoch =  datetime.datetime.utcfromtimestamp(0)
             return (dt - epoch).total_seconds() * 1000

          x = [datetime.datetime(year=2013, month=10, day=04),
             datetime.datetime(year=2013, month=11, day=05),
             datetime.datetime(year=2013, month=12, day=06)]
          data = [go.Scatter(
                      x=x,
                      y=[1, 3, 6])]

          layout = go.Layout(xaxis = dict(
                             range = [to_unix_time(datetime.datetime(2013, 10, 17)),
                                      to_unix_time(datetime.datetime(2013, 11, 20))]
```

```
    ))

fig = go.Figure(data = data, layout = layout)
py.iplot(fig)
```

Out[10]: <plotly.tools.PlotlyDisplay object>

In [12]:
```
# Get this figure: fig = py.get_figure("https://plot.ly/~Dreamshot/8235/")
# Get this figure's data: data = py.get_figure("https://plot.ly/~Dreamshot
# Add data to this figure: py.plot(Data([Scatter(x=[1, 2], y=[2, 3])]), fi
# Get y data of first trace: y1 = py.get_figure("https://plot.ly/~Dreamsho

# Get figure documentation: https://plot.ly/python/get-requests/
# Add data documentation: https://plot.ly/python/file-options/

# If you're using unicode in your file, you may need to specify the encodi
# You can reproduce this figure in Python with the following code!

# Learn about API authentication here: https://plot.ly/python/getting-star
# Find your api_key here: https://plot.ly/settings/api


trace1 = Scatter(
    x=['', '', '2004-01-01', '2004-02-01', '2004-03-01', '2004-04-01', '20
    y=['', '', 28, 25, 27, 30, 30, 32, 33, 33, 31, 31, 30, 40, 31, 32, 32,
    line=Line(
        color='rgb(109, 158, 235)'
    ),
    name='Relevance',
    uid='2132ed'
)
trace2 = Scatter(
    x=['2004-01-01', '2004-04-03 18:07:20.816', '2004-07-06 13:14:41.632',
    y=[19.13701054331718, 20.20915869657822, 21.281306849839257, 22.353455
    hoverinfo='none',
    line=Line(
        color='rgb(109, 158, 235)',
        dash='dot',
        width=0.5
    ),
    name='Best Fit',
    opacity=1,
    uid='c143ce',
    xaxis='x',
    yaxis='y'
)
data = Data([trace1, trace2])
layout = Layout(
    annotations=Annotations([
        Annotation(
            x=1379998884375,
            y=76.23095238095237,
            ax=-79,
            ay=-30,
            text='"Happy" released by<br>Pharrell Williams'
        )
    ]),
    autosize=False,
    height=600,
    showlegend=False,
    title='"Happy"',
    width=800,
    xaxis=XAxis(
        autorange=True,
        range=[1072868400000, 1469966400000],
        showgrid=False,
        title='Source: Google Trends',
        type='date'
    ),
    yaxis=YAxis(
        autorange=False,
        range=[-2, 101],
        title='Google Search Relevance',
        type='linear'
    )
)
fig = Figure(data=data, layout=layout)
plot_url = py.plot(fig)
```

```
    ------------------------------------------------------------------------------

    NameError                                 Traceback (most recent call last)

    <ipython-input-12-18aa55c7cc3f> in <module>()
     17     x=['', '', '2004-01-01', '2004-02-01', '2004-03-01', '2004-04-01',
     18     y=['', '', 28, 25, 27, 30, 30, 32, 33, 33, 31, 31, 30, 40, 31, 32,
---> 19     line=Line(
     20         color='rgb(109, 158, 235)'
     21     ),


    NameError: name 'Line' is not defined
```

```
In [13]: def oneTailedTTest(data):
             true_mu = data.quote_rate.mean()
```

```
          print "Quote rate mean and std dev:: ", true_mu, "  ", data.quote_rate
          onesample_results = scipy.stats.ttest_1samp(df_quote_rate, true_mu)

          matrix_onesample = [
              ['', 'Test Statistic', 'p-value'],
              ['Sample Data', onesample_results[0], onesample_results[1]]
          ]

          onesample_table = FF.create_table(matrix_onesample, index=True)
          py.iplot(onesample_table, filename='onesample-table')

          #Since our p-value is greater than our Test-Statistic, we have good ev

In [24]: # This code generates the plot online on :  https://plot.ly/~NataliaDiazRo
        trace1 = Scatter(
            x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
            y=[0.6250887811924892, 0.06785768246991553, 0.31840935466629805, 0.781
            fill='tonexty',
            fillcolor='rgba(255, 153, 51, 0.3)',
            line=Line(
                color='rgba(255, 153, 51, 1.0)',
                width='1.3'
            ),
            mode='lines',
            name='a'
        )
        trace2 = Scatter(
            x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
            y=[1.40370681075778, 0.5188087660032598, 0.8803242969698729, 1.1412432
            fill='tonexty',
            fillcolor='rgba(55, 128, 191, 0.3)',
            line=Line(
                color='rgba(55, 128, 191, 1.0)',
                width='1.3'
            ),
            mode='lines',
            name='b'
        )
        trace3 = Scatter(
            x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
            y=[1.6581069418787613, 0.7916528456960947, 1.0154945674308113, 1.63390
            fill='tonexty',
            fillcolor='rgba(50, 171, 96, 0.3)',
            line=Line(
                color='rgba(50, 171, 96, 1.0)',
                width='1.3'
            ),
            mode='lines',
            name='c'
        )
        trace4 = Scatter(
            x=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
            y=[1.7234649739018388, 1.6887047058595843, 1.3468455263500552, 1.9972
            fill='tonexty',
            fillcolor='rgba(128, 0, 128, 0.3)',
            line=Line(
                color='rgba(128, 0, 128, 1.0)',
                width='1.3'
            ),
            mode='lines',
            name='d'
        )
        data = Data([trace1, trace2, trace3, trace4])
        layout = Layout(
            legend=Legend(
                bgcolor='#F5F6F9',
                font=Font(
                    color='#4D5663'
                )
            ),
            paper_bgcolor='#F5F6F9',
            plot_bgcolor='#F5F6F9',
            xaxis1=XAxis(
                gridcolor='#E1E5ED',
                tickfont=dict(
                    color='#4D5663'
                ),
                title='',
                titlefont=dict(
                    color='#4D5663'
                ),
                zerolinecolor='#E1E5ED'
            ),
            yaxis1=YAxis(
                gridcolor='#E1E5ED',
                tickfont=dict(
                    color='#4D5663'
                ),
                title='',
                titlefont=dict(
                    color='#4D5663'
                ),
                zeroline=False,
                zerolinecolor='#E1E5ED'
            )
        )
```

```
fig = Figure(data=data, layout=layout)
#Saves and plots the image in a public url
plot_url = py.plot(fig)
```

```
In [ ]:
```