```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct  5 04:25:36 2018

@author: lappy
"""
from matplotlib.dates import date2num
import matplotlib.pyplot as plt
import seaborn; seaborn.set()
import requests
from bs4 import BeautifulSoup
import numpy as np
import scipy
import pandas as pd
import lxml
import datetime
from datetime import date
from nsepy import get_history

startdate = pd.to_datetime("2015-1-1").date()
enddate = pd.to_datetime("2015-12-31").date()



data = get_history(symbol= "INFY", start=date(2015,1,1), end=date(2015,12,31))

data.loc[startdate:enddate]

tcs = get_history(symbol= "TCS", start=date(2015,1,1), end=date(2015,12,31))
tcs.loc[startdate:enddate]



from matplotlib.dates import DateFormatter, WeekdayLocator,\
    DayLocator, MONDAY
from matplotlib.finance import candlestick_ohlc

def pandas_candlestick_ohlc(dat, stick = "day", otherseries = None):
    """
    :param dat: pandas DataFrame object with datetime64 index, and float columns "Open", "High", "L
    :param stick: A string or number indicating the period of time covered by a single candlestick.
    :param otherseries: An iterable that will be coerced into a list, containing the columns of dat

    This will show a Japanese candlestick plot for stock data stored in dat, also plotting other se
    """
    mondays = WeekdayLocator(MONDAY)        # major ticks on the mondays
    alldays = DayLocator()              # minor ticks on the days
    dayFormatter = DateFormatter('%d')      # e.g., 12

    # Create a new DataFrame which includes OHLC data for each period specified by stick input
    transdat = dat.loc[:,["Open", "High", "Low", "Close"]]
    if (type(stick) == str):
        if stick == "day":
            plotdat = transdat
            stick = 1 # Used for plotting
        elif stick in ["week", "month", "year"]:
            if stick == "week":
```

```
            transdat["week"] = pd.to_datetime(transdat.index).map(lambda x: x.isocalendar()[1])
        elif stick == "month":
            transdat["month"] = pd.to_datetime(transdat.index).map(lambda x: x.month) # Identif
        transdat["year"] = pd.to_datetime(transdat.index).map(lambda x: x.isocalendar()[0]) # I
        grouped = transdat.groupby(list(set(["year",stick]))) # Group by year and other appropr
        plotdat = pd.DataFrame({"Open": [], "High": [], "Low": [], "Close": []}) # Create empty
        for name, group in grouped:
            plotdat = plotdat.append(pd.DataFrame({"Open": group.iloc[0,0],
                                       "High": max(group.High),
                                       "Low": min(group.Low),
                                       "Close": group.iloc[-1,3]},
                                      index = [group.index[0]]))
        if stick == "week": stick = 5
        elif stick == "month": stick = 30
        elif stick == "year": stick = 365

elif (type(stick) == int and stick >= 1):
    transdat["stick"] = [np.floor(i / stick) for i in range(len(transdat.index))]
    grouped = transdat.groupby("stick")
    plotdat = pd.DataFrame({"Open": [], "High": [], "Low": [], "Close": []}) # Create empty dat
    for name, group in grouped:
        plotdat = plotdat.append(pd.DataFrame({"Open": group.iloc[0,0],
                                   "High": max(group.High),
                                   "Low": min(group.Low),
                                   "Close": group.iloc[-1,3]},
                                  index = [group.index[0]]))

else:
    raise ValueError('Valid inputs to argument "stick" include the strings "day", "week", "mont


# Set plot parameters, including the axis object ax used for plotting
fig, ax = plt.subplots()
fig.subplots_adjust(bottom=0.2)
if plotdat.index[-1] - plotdat.index[0] < pd.Timedelta('730 days'):
    weekFormatter = DateFormatter('%b %d')  # e.g., Jan 12
    ax.xaxis.set_major_locator(mondays)
    ax.xaxis.set_minor_locator(alldays)
else:
    weekFormatter = DateFormatter('%b %d, %Y')
ax.xaxis.set_major_formatter(weekFormatter)

ax.grid(True)

# Create the candelstick chart
candlestick_ohlc(ax, list(zip(list(date2num(plotdat.index.tolist())), plotdat["Open"].tolist(),
                  plotdat["Low"].tolist(), plotdat["Close"].tolist())),
                  colorup = "black", colordown = "red", width = stick * .4)

# Plot other series (such as moving averages) as lines
if otherseries != None:
    if type(otherseries) != list:
        otherseries = [otherseries]
    dat.loc[:,otherseries].plot(ax = ax, lw = 1.3, grid = True)

ax.xaxis_date()
ax.autoscale_view()
```

```
    plt.setp(plt.gca().get_xticklabels(), rotation=75, horizontalalignment='right')

    plt.show()

data1=data
data1["28d"] = np.round(data["Close"].rolling(window = 28, center = False).mean(), 2)
pandas_candlestick_ohlc(data.loc[startdate:enddate], otherseries = "28d")



tcs["28d"] = np.round(tcs["Close"].rolling(window = 28, center = False).mean(), 2)
pandas_candlestick_ohlc(tcs.loc[startdate:enddate], otherseries = "28d")


nifty = get_history(symbol="NIFTY IT",
                            start=date(2015,1,1),
                            end=date(2015,12,31),index=True)
nifty1=nifty
mix = get_history("TCS", "INFY", start=date(2015,1,1), end=date(2015,12,31))
mix["4w"] = np.round(apple["Close"].rolling(window = 28, center = False).mean(), 2)

pandas_candlestick_ohlc(mix.loc['2015-01-01':'2015-12-31',:], otherseries = "4w")

nifty.head()
data.head()

data2 = data['Close']
data2= data2.astype(float)


def check(Volume, val):

    # traverse in the list
    for x in Volume:

        # compare with all the values
        # with val
        if val>= 10%x+x or val<x-10%x:
            return False
    return True


dummies = pd.get_dummies(data['Volume']).rename(columns=lambda x: 'Volume_' + str(x))
df = pd.concat([data, dummies], axis=1)
df = df.drop(['Volume'], inplace=True, axis=1)
```