import component

import techindicator as ti

import datetime as dt #in-built module

import pandas as pd   #

from pandas\_datareader import data

import yfinance as yf

import matplotlib.pyplot as plt

from matplotlib import style

import mplfinance as mpf

import matplotlib.dates as mdates

from backtesting import Strategy, Backtest

from backtesting.lib import crossover

def getrawdata ():  ##using pandas

    yf.pdr\_override()

    style.use('ggplot')

    start = dt.datetime(2009, 1, 1)

    end = dt.datetime.now()

    df = data.get\_data\_yahoo('V', start, end)

    df.to\_csv(r'G:\VScode\_project\Stock\_raw\pddatareader\_mpf.csv', index = True, header=True)

    #print(df.shape)

    #print(df.columns)

    #print(df)

    #print(df.info())

    #print(df.describe)

    #df[['Adj Close','100ma','150ma']].plot() #Spyder可用此法秀chartcls

    return df

def rawdata2pd():

    return pd.read\_csv("pddatareader\_mpf.csv",index\_col=0,header='infer', parse\_dates= True)

def strategytest(df):

        df.loc[(df['bbp'] <0 & df['rsi\_14']<30),['signal']] = 1

        df.loc[(df['bbp'] >0 & df['rsi\_14']>70),['signal']] = 0

        print (df.signal)

def plotting(df):

    style.use('ggplot')

    ax1 = plt.subplot2grid((6,1), (0,0), rowspan=5, colspan=1)      #\*\*\*\*\*\*refer to https://pythonprogramming.net/subplot2grid-add\_subplot-matplotlib-tutorial/

    ax2 = plt.subplot2grid((6,1), (5,0), rowspan=1, colspan=1,sharex=ax1)

    ax3 = plt.subplot2grid((6,1), (5,0), rowspan=1, colspan=1,sharex=ax1)

    ax1.plot(df.index, df['Adj Close'])

    ax1.plot(df.index, df['100ma'])

    ax1.plot(df.index, df['150ma'])

    #ax1.plot(df.index,df.loc['up','low'],color='black')

    #ax1.plot(df.index,df['low'],color='black')

    ax2.bar(df.index, df['Volume'])

    #ax3.plot(df.index,df['rsi\_14'])

    plt.show()

def plotting2(df): #using mpf to ontain raw data

    mpf.plot(df, type='candle', style='charles',

            title='Test',

            ylabel='Price ($)',

            ylabel\_lower='Shares \nTraded',

            volume=True,

            mav=(20,100,150),

            )

def SMA(value,n):

    return (pd.Series(value).rolling(n).mean())

#def savefile(data):

#    path= "report/"+ str(data) + ".txt"

#    text\_file= open(path,"w+")

#    data = data.to\_string()

#    text\_file.write(data)

#    text\_file.close()

class SmaCross(Strategy):

    # Define the two MA lags as \*class variables\*

    # for later optimization

    n1 = 1

    n2 = 20

    def init(self):

        # Precompute two moving averages

        self.sma1 = self.I(SMA, self.data.Close, self.n1)

        self.sma2 = self.I(SMA, self.data.Close, self.n2)

    def next(self):

        # If sma1 crosses above sma2, buy the asset

        if crossover(self.sma1, self.sma2):

            self.buy()

        # Else, if sma1 crosses below sma2, sell it

        elif crossover(self.sma2, self.sma1):

            self.sell()

#stock\_in = input('Please input the stock ID:')

df=getrawdata() #df is the rawdata with date being index, OHLC and Volume

#print (df.head())

df=rawdata2pd()

#print (df.index)

###########################################

###########################################

df['20ma'] = df['Adj Close'].rolling(window=100,min\_periods=0).mean()

df['50ma'] = df['Adj Close'].rolling(window=100,min\_periods=0).mean()

df['100ma'] = df['Adj Close'].rolling(window=100,min\_periods=0).mean()

df['150ma'] = df['Adj Close'].rolling(window=150,min\_periods=0).mean()

df['200ma'] = df['Adj Close'].rolling(window=200,min\_periods=0).mean()

df = ti.rsi\_14(df)

df = ti.Bollinger(df)

#df.to\_csv(r'G:\VScode\_project\Stock\_raw\TechIndi\_Test.csv', index = True, header=True)

#print (df.rsi\_14)

############################################

#plotting(df)

#plotting2(df)

############################################

#df = strategytest(df)

df.to\_csv(r'G:\VScode\_project\Stock\_raw\export\_dataframe.csv', index = True, header=True)

bt = Backtest(df, SmaCross, cash=10000, commission=.002)

stat\_raw = bt.run()

bt.plot()

############# TO TXT    #####################

text\_file= open(r"report\Original\_Report.txt","w+")

stat\_raw = stat\_raw.to\_string()

text\_file.write(stat\_raw)

text\_file.close()

#############################################

stats\_optimized = bt.optimize(n1=range(5,20,1),

                   n2=range(10,50,1),

                   maximize='Equity Final [$]',

                   constraint=lambda p:p.n1<p.n2

                   )

stat\_optimuied= stats\_optimized

bt.plot()

############# TO TXT    #####################

text\_file= open(r"report\Optimized\_Report.txt","w+")

stat\_optimuied = stat\_optimuied.to\_string()

n = text\_file.write(stat\_optimuied)

text\_file.close()

###############################################