紫金量化交易策略代码编写帮助说明

1. 策略策略开头

```
#coding: UTF-8
# 结合KDJ贴标签，五分类
```

说明：

#coding: UTF-8 方便在代码中加入中文汉字。

#注释说明代码方法及名称

2. 导入策略代码中需要的库函数

```
from __future__ import division
import time
import numpy as np
import pandas as pd
from numpy import *
from pandas import Series,DataFrame
from sklearn.externals import joblib
from sklearn import preprocessing    # 数据预处理工具
import tarfile
import talib
from sklearn.externals import joblib
import datetime
```

说明：

(导入的基础库主要包括 numpy,pandas 等,技术指标库包括 talib 等,若需要用到预测模型文件则需要导入 tarfile，joblib 等用于解压缩文件，另外基于预测模型可能需要加载数据预处理库，例如：sklearn 的 preprocessing 等)。

注意：此处库函数，应尽量使用 os,sys 等库防止系统出错。

3. 具体策略代码编写说明

3.1 策略中会使用哪些回调函数。

这些函数可以只实现部分，不需要的使用的 pass 略过。

on_init, 定义变量,并初始化。

需要在函数外部使用的变量都要用 context.var.XXX 来定义。

```
context.var.barList = []
context.var.high = []
context.var.low = []
context.var.open = []
context.var.volume = []
context.var.num_die = context.var.num_zhang = context.var.num_ping = c
context.var.num_Rdie = context.var.num_Rzhang = context.var.num_Rping
context.var.num_Mdie = context.var.num_Mzhang = context.var.num_Mping
context.var.num_ying = []       #止盈差价
context.var.num_sun = []       # 止损差价
context.var.num_predict = []   # 预测平仓差价
context.var.x_test1 = []        #用来存储测试数据
context.var.real_label1 = []    #用来存储真实标签
context.var.lab_answers = []    #存储预测标签
context.var.num_all  = 0
context.var.testPos = 0        # 记录持仓
```

on_start, 策略启动时调用，一般除写日志，外不做别的操作。

```
context.function.log(u'nAdaboost_KDJ策略启动')
```

on_stop，策略停止时调用，一般除写日志，外不做别的操作。

```
context.function.log(u'nAdaboost_KDJ策略停止')
```

on_tick，使用 TICK 数据（高频数据）时在此函数中使用，tick 是由平台维护。

若需使用，首先获取 tick `tick = context.var.tick`

再调用
```
context.function.buy(tick.price)
context.function.sell(tick.price)
```
获取 tick 时刻的价格。

on_bar，使用分钟数据，该函数使用中，bar 由平台维护。（本平台推荐使用 bar 数据预测，平台上绝大部分逻辑代码都在此函数中实现。）

首先获取 bar `bar = context.var.bar`

存分钟数据的 CHLOV（bar.close,bar.high,bar.low,bar.open,bar.volume）

```
context.var.barList.append(bar.close)    # 存分钟价格
context.var.high.append(bar.high)       # 一分钟内价格最高价
context.var.low.append(bar.low)        #  一分钟内价格最高价
context.var.open.append(bar.open)        # 每分钟的开盘价
context.var.volume.append(bar.volume)     #
```

若需要调用，使用金融技术分析指标（调用 talib）则需要存数据列表，并转换为 numpy.array 格式。

```
arrClose    = np.array(context.var.barList, dtype=np.float)
arrHigh     = np.array(context.var.high, dtype=np.float)
arrLow      = np.array(context.var.low, dtype=np.float)
arrOpen     = np.array(context.var.open, dtype=np.float)
arrVolume   = np.array(context.var.volume, dtype=np.float)
```

例如使用 talib 计算 KDJ

```
context.var.raw_data = DataFrame()
```

```
context.var.real_data['KDJ_K'] = talib.STOCH(arrHigh, arrLow, arrClose)[0]
context.var.real_data['KDJ_D'] = talib.STOCH(arrHigh, arrLow, arrClose)[1]
```

使用 talib 计算并将计算数据存到 Pandas dataFrame 中。

若需要用预测模型，还需要加载模型文件

```
clf = joblib.load('Adaboost_IF_M3_KDJ.pkl')
```

调用模型预测之前还需要将数据规整化处理

```
context.var.min_max_scaler =preprocessing.MinMaxScaler()
context.var.x_test1 =context.var.min_max_scaler.fit_transform(context.var.raw_data[40:])
```

开始进行预测

```
context.var.answer = clf.predict(context.var.x_test1)   # # 进行预测
```

根据预测，止盈止损条件等等逻辑判断，进行相应买卖操作。一般伴随买卖操作也会写相应的 log 日志。

```
context.function.cover(context.var.barList[-1])
context.function.log(u'平空仓: '+str(context.var.barList[-1])+' '+str(bar.datetime))
```

on_order，用于实盘模拟及真实交易的订单预定，由平台维护。一般不在其中定义实现，只简单的 pass。

on_newday 每天的 24 点后平台会调用此函数。系统平台内部使用，一般不在其中定义实现，只简单的 pass。

3.2 函数的输入项均为 context,

context 是一个封装类，包含定义变量使用的对象 context.var，调用函数使用的对象 context.function。

其中 context.function 对象可调用的方法函数有：

context.function.buy(float price)（以 price 价格买多）

context.function.sell(float price)（以 price 价格卖多）

context.function.short(float price)（以 price 价格卖空）

context.function.cover(float price)（以 price 价格买空）

Context.function.log(String loginfo) （向策略日志中写入内容）