【数据】【自动化交易】Python编写策略模拟股票交易

> 这节我就用上节提到的pyalgotrade来编写回测策略程序，模拟股票交易。本篇文章里用的是SMA均线策略。

数据

数据我使用的是 **大恒科技(600288.SH)** 2010年到2016年的day级数据，我将其变换成了pyalgotrade教程的格式:

```
Adj. Close,Adj. High,Adj. Low,Adj. Open,Adj. Volume,Close,Date,Ex-Dividend,
0,13.53,13.45,12.65,12.62,51597046.82,11.16,2010-01-04,572108102.0,11.37,10
1,13.64,13.72,12.91,12.92,48822968.82,11.27,2010-01-05,549279336.0,11.64,10
2,13.38,13.56,13.05,13.07,41360141.82,11.01,2010-01-06,461521697.0,11.48,11
... ...
```

简单交易程序

简单的策略程序：

```
# -*-coding:utf-8-*-
from __future__ import print_function
from pyalgotrade import strategy
from pyalgotrade.barfeed import quandlfeed
from pyalgotrade.technical import ma


class MyStrategy(strategy.BacktestingStrategy):
    def __init__(self, feed, instrument, smaPeriod):
        super(MyStrategy, self).__init__(feed, 1000)
        self.__position = None
        self.__instrument = instrument
        # We'll use adjusted close values instead of regular close values.
        self.setUseAdjustedValues(True)
        self.__sma = ma.SMA(feed[instrument].getPriceDataSeries(), smaPerio

    #---- BUY ----
    def onEnterOk(self, position):
        execInfo = position.getEntryOrder().getExecutionInfo()
        #self.info("BUY at $%.2f" % (execInfo.getPrice()))
        self.info("在价格 ￥%.2f 时买入" % (execInfo.getPrice()));

    #---- NO BUY ----
    def onEnterCanceled(self, position):
        self.__position = None

    #---- SELL ----
    def onExitOk(self, position):
        execInfo = position.getExitOrder().getExecutionInfo()
        #self.info("SELL at $%.2f" % (execInfo.getPrice()))
        self.info("在价格 ￥%.2f 时抛出" % (execInfo.getPrice()));
        self.__position = None

    #---- NO SELL ----
    def onExitCanceled(self, position):
        # If the exit was canceled, re-submit it.
        self.__position.exitMarket()

    def onBars(self, bars):
        # Wait for enough bars to be available to calculate a SMA.
        if self.__sma[-1] is None:
            return

        bar = bars[self.__instrument]
        # If a position was not opened, check if we should enter a long pos
        if self.__position is None:
            if bar.getPrice() > self.__sma[-1]:
                # Enter a buy market order for 10 shares. The order is good
                self.__position = self.enterLong(self.__instrument, 10, True
        # Check if we have to exit the position.
        elif bar.getPrice() < self.__sma[-1] and not self.__position.exitAc
            self.__position.exitMarket()


def run_strategy(smaPeriod):
    # Load the bar feed from the CSV file
    feed = quandlfeed.Feed()
    feed.addBarsFromCSV("orcl", "600288SH.csv")
```

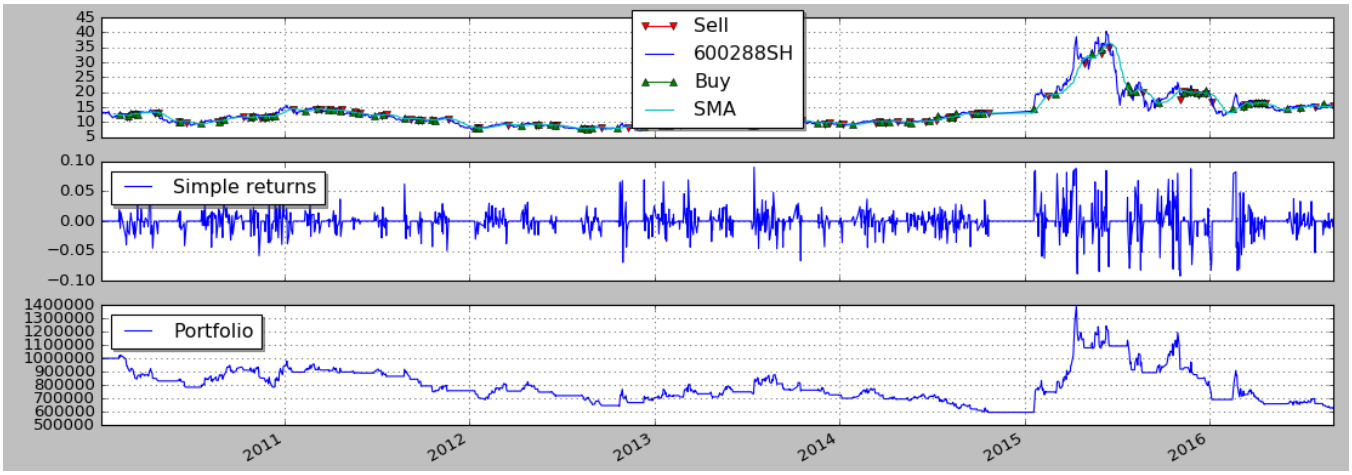
```
    # Evaluate the strategy with the feed.
    myStrategy = MyStrategy(feed, "orcl", smaPeriod)
    myStrategy.run()
    #print("Final portfolio value: $%.2f" % myStrategy.getBroker().getEquity
    print("最终盈亏情况： ￥ %.2f" % myStrategy.getBroker().getEquity())

run_strategy(15);
```

输出结果：

```
... ...
2016-06-22 00:00:00 strategy [INFO] 在价格 ￥14.62 时抛出
2016-06-23 00:00:00 strategy [INFO] 在价格 ￥14.94 时买入
2016-06-27 00:00:00 strategy [INFO] 在价格 ￥14.68 时抛出
2016-06-28 00:00:00 strategy [INFO] 在价格 ￥14.95 时买入
2016-07-19 00:00:00 strategy [INFO] 在价格 ￥15.35 时抛出
2016-07-20 00:00:00 strategy [INFO] 在价格 ￥15.34 时买入
2016-07-28 00:00:00 strategy [INFO] 在价格 ￥15.12 时抛出
2016-08-11 00:00:00 strategy [INFO] 在价格 ￥15.88 时买入
2016-08-26 00:00:00 strategy [INFO] 在价格 ￥15.58 时抛出
最终盈亏情况： ￥ 925.78
```

策略和绘制曲线程序：




```
# -*-coding:utf-8-*-
from pyalgotrade import strategy
from pyalgotrade.technical import ma
from pyalgotrade.technical import cross
from pyalgotrade import plotter
from pyalgotrade.barfeed import quandlfeed
from pyalgotrade.stratanalyzer import returns

class SMACrossOver(strategy.BacktestingStrategy):
    def __init__(self, feed, instrument, smaPeriod):
        super(SMACrossOver, self).__init__(feed)
        self.__instrument = instrument
        self.__position   = None
        # We'll use adjusted close values instead of regular close values.
        self.setUseAdjustedValues(True)
```

```
        self.__prices     = feed[instrument].getPriceDataSeries()
        self.__sma        = ma.SMA(self.__prices, smaPeriod)

    def getSMA(self):
        return self.__sma

    def onEnterCanceled(self, position):
        self.__position = None

    def onExitOk(self, position):
        self.__position = None

    def onExitCanceled(self, position):
        # If the exit was canceled, re-submit it.
        self.__position.exitMarket()

    def onBars(self, bars):
        # If a position was not opened, check if we should enter a long pos
        if self.__position is None:
            if cross.cross_above(self.__prices, self.__sma) > 0:
                shares = int(self.getBroker().getCash() * 0.9 / bars[self._
                # Enter a buy market order. The order is good till canceled
                self.__position = self.enterLong(self.__instrument, shares,
        # Check if we have to exit the position.
        elif not self.__position.exitActive() and cross.cross_below(self.__
            self.__position.exitMarket()

# Load the bar feed from the CSV file
feed = quandlfeed.Feed()
#feed.addBarsFromCSV("orcl", "WIKI-ORCL-2000-quandl.csv")
feed.addBarsFromCSV("600288SH", "600288SH.csv")

# Evaluate the strategy with the feed's bars.
myStrategy = sma_crossover.SMACrossOver(feed, "600288SH", 20)

# Attach a returns analyzers to the strategy.
returnsAnalyzer = returns.Returns()
myStrategy.attachAnalyzer(returnsAnalyzer)

# Attach the plotter to the strategy.
plt = plotter.StrategyPlotter(myStrategy)
# Include the SMA in the instrument's subplot to get it displayed along with
plt.getInstrumentSubplot("600288SH").addDataSeries("SMA", myStrategy.getSMA
# Plot the simple returns on each bar.
plt.getOrCreateSubplot("returns").addDataSeries("Simple returns", returnsAn

# Run the strategy.
myStrategy.run()
myStrategy.info("Final portfolio value: $%.2f" % myStrategy.getResult())

# Plot the strategy.
plt.plot()
```