bt即为backtrader框架本身

Bt接口即为backtrader所能提供的接口

# dataFeed

整个数据加载过程如下：

数据源：本地数据——》格式化：bt接口——》载入

## 数据源

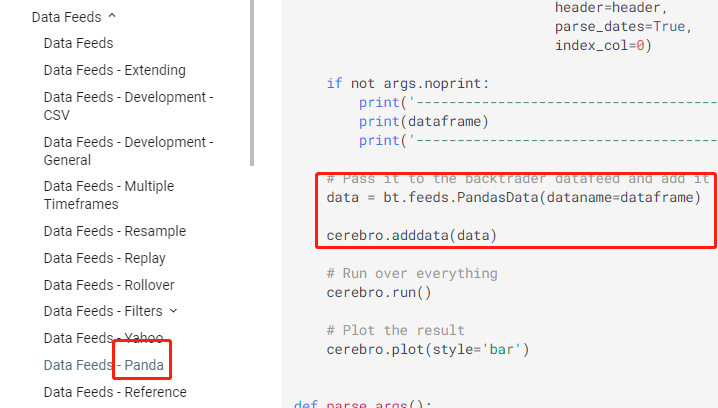
指各种途径获取的本地数据，可以是第三方tushare，本地数据库，本地csv，无关格式，有数据就行

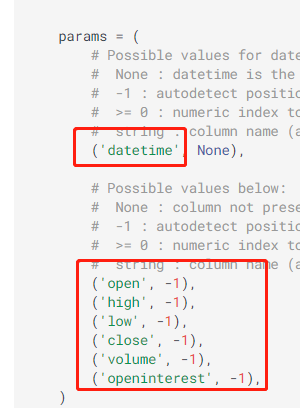
## 格式化

1找到跟自己数据源匹配的接口.

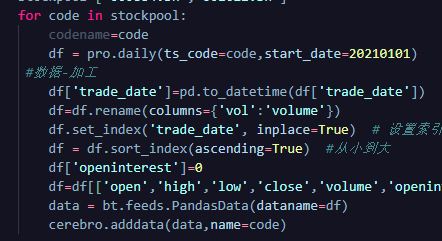
比如tushare用的是pandas

俗称数据加工，为了把本地数据跟bt里的datafeeds接口对接，对接接口如下：



2按bt参数要求，进行数据格式处理

比如：pandas的是按以上6个字段进行格式化



多数据载入，用name进行区分。

## 调用及生命周期

cerebro导入后，只表示在self.datas里存在多个数据集合，在next函数里买入时，仍旧需要通过data=self.getdatabyname(stock)来获取数据集

### 调用方案1

在datafeed被载入后，可使用self.datas来访问加载数据

Self.datas[0]或者self.data0表示第一支股票

Self.datas[1]表示第二支股票

Self.data=self.data0=self.data[0]

规范化，建议用self.data0表第一个股票，self.data1表第二支股票

### 调用方案2

载入adddata时，如果用了指定了股票名称，使用self.getdatabyname(“name”)获取数据

# **TestStrategy\_next使用**

# 操作

重要参数：

Data：股票/期货种类，默认cerebro加载的第一个数据

Size：交易数量

**self.buy()**

**Self.sell()**

**Self.close()**默认data跟size是空，平仓当前data的数据

**Self.cancel()**传入的参数是order,self.cancel(order)会把这个order给平掉，如果这个order还没有成交的话。

**order\_target\_size：按仓位市值成交**

参数：

Data：数据对象

Target：int，目标值，比如50000

效果：使用后，当前bar上的value就是target

**order\_target\_percent：按仓位百分比成交**

参数：

Data：数据对象

Target：小数点，目标值，比如0.5

效果：使用后，市值为target\*self.broker.get\_value()

# 框架解析

Init：self的属性

Start策略开始运行，主要用于调试用

prenext指标生效的最小周期bar的值，在bar满足后，进入next函数

next策略主要部分

stop策略结束打印的输出

notify\_cashvalue 获取cash和value变动

notify\_order获取订单变动，有套自己的固定打印模板即可

notify\_trade打印交易的变动信息

# **函数**

执行成本

order.executed.value，只对buy有效，sell不准

获取一个data的持仓

self.getposition(self.datas[0]).size

self.getposition().size

order.executed.size

获取当前市值。self.broker.get\_value()

获取当前现金。self.broker.get\_cash()

* 成员变量Open, High, Low, Close, Volume, OpenInterest
* 先获取线变量集合，再从集合里按时间序列获取当日值

获取所有收盘价格 self.datas[0].close

# 数据源

JData聚宽试用版本，每天100w条

Tbquant交易开拓者，同上相互验证

万矿，wind线上平台，研究推介

Tushare，推荐

Quantopian国外基于开源框架的量化平台zipline

掘金量化