量化交易平台开发手册

闫涛
阿尔山金融科技有限公司
北京
{yt7589}@qq.com

Abstract

基于 tushare.org 开放数据集，构建本地量化交易开发平台。

1 概述

1.1 环境搭建

创建开源项目，项目网址：https://github.com/yt7589/aqp ，本地环境为：d:/awork/aftdc/incubate/aqp，虚拟环境激活：d:/aadesk/dev/python/quant/Script/activate ，这个是通过 python -m venv quant 来创建的虚拟环境。

1.2 整体架构

app_main.py：程序主入口；
app_registry.py：管理程序中所有全局性配置和变量；
controller 目录：所有业务逻辑实现类；
model 目录：所有数据库操作类；

1.3 数据服务商

我们采用的是 tushare.pro 提供的数据服务：https://tushare.pro 。

2 数据处理

数据处理包括从 tushare.org 网站获取数据，将其转化为量化平台所需的数据格式。

2.1 获取沪深市场所有挂牌股票

获取在沪深两市挂牌的所有股票的基本信息。

2.1.1 接口定义

获取股票基本信息接口为 stock_basic，其参数为：

Table 1: stock_basic 接口输入参数说明

名称	类型	必选	描述
is_hs	str	N	是否沪深港通标的，N 否 H 沪股通 S 深股通
list_status	str	N	上市状态：L 上市 D 退市 P 暂停上市
exchange	str	N	交易所：SSE 上交所 SZSE 深交所 HKEX 港交所

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

返回值为：

Table 2: stock_basic 接口返回结果说明

名称	类型	描述
ts_code	str	TS 代码
symbol	str	股票代码
name	str	股票名称
area	str	所在地域
industry	str	所属行业
fullname	str	股票全称
enname	str	英文全称
market	str	市场类型（主板/中小板/创业板）
exchange	str	交易所代码
curr_type	str	交易货币
list_status	str	上市状态：L 上市 D 退市 P 暂停上市
list_date	str	上市日期
delist_date	str	退市日期
is_hs	str	是否沪深港通标的，N 否 H 沪股通 S 深股通

调用格式为：

```
import tushare as ts
pro = ts.pro_api()
data = pro.stock_basic(exchange='', list_status='L',
                       fields='ts_code,symbol,name,area,industry,list_date')
data = pro.query('stock_basic', exchange='', list_status='L',
                 fields='ts_code,symbol,name,area,industry,list_date')
```

Listing 1: 获取股票基本信息

见5所示结果示例：

```
    ts_code  symbol    name  area industry  list_date
0  000001.SZ  000001  平安银行  深圳      银行   19910403
1  000002.SZ  000002   万科A  深圳    全国地产   19910129
2  000004.SZ  000004  国农科技  深圳    生物制药   19910114
3  000005.SZ  000005  世纪星源  深圳    房产服务   19901210
4  000006.SZ  000006  深振业A  深圳    区域地产   19920427
5  000007.SZ  000007   全新好  深圳    酒店餐饮   19920413
```

Listing 2: 获取股票基本信息结果示例

2.1.2 区域信息

如代码2所示，地区是以字符串形式返回的。我们可能需要按地区来统计股票表现，因此需要将地区统计出来，放到单独的一个表中进行管理。

数据库设计 数据库结构表结构如下所示：

```
create table t_area(
    area_id int primary key auto_increment,
    area_name varchar(200)
);
```

Listing 3: 地区表数据结构

信息处理 当我们读到返回结果的一行时，我们取出地区信息，然后查询 t_area 表中是否包含该地区，如果包含则返回对应的 area_id，否则将该地区添加到 t_area 表中，并返回其 area_id。

2.1.3 行业信息

数据库设计

信息处理

2.1.4 股票信息

接口定义

获取并处理数据

2.2 获取日线行情数据

3 量化模型

3.1 时间序列分析

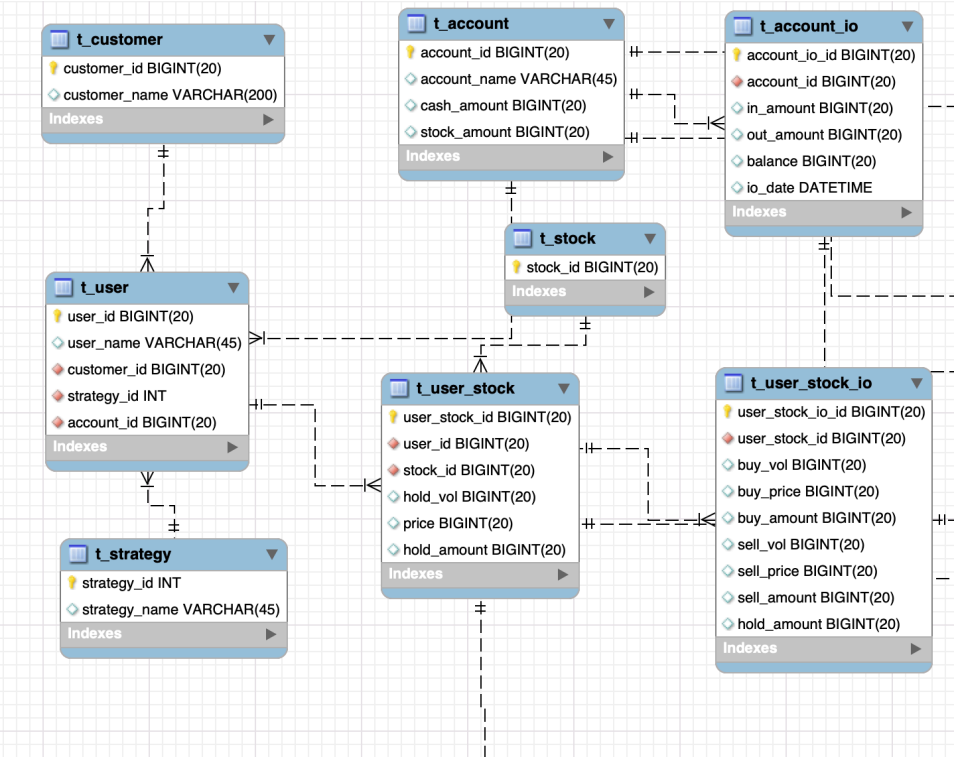
4 回测系统

4.1 数据库系统设计

客户是自然人，用 t_customer 表示。客户加上账户再加上量化策略，形成我们系统的用户，用 t_user 来表示。账户中具有现金资产和股票资产。账户具有资金的入和出，有股票的买入和卖出。用户持有一些股票，可以对股票进行买卖。用户可以买入和卖出指定数量股票，与账户资金变动相关联股票参数表：买入费率、印花税；卖出费率、印花税等，计入交易成本中。

4.1.1 ER 图

Figure 1: 数据库表 ER 图



4.1.2 客户表

表示自然人客户，结构如下所示：

Table 3: 客户表 (t_custome)

字段	名称	类型	描述
customer_id	客户编号	bigint	主键且自增长
customer_name	客户姓名	varchar	真实姓名

4.1.3 策略表

主要包括股票交易策略：包括 SVM、XGBoost、LSTM、ARIMA、GARCH 等，客户可以订购某个策略而成为我们的用户。

Table 4: 策略表 (t_strategy)

字段	名称	类型	描述
strategy_id	策略编号	bigint	主键且自增长
strategy_name	策略名称	varchar	真实姓名

4.1.4 账户表

客户拥有账户，每个客户对应的用户具有唯一的账户。账户中有现金和股票资产。其有对应

Table 5: 账户表 (t_account)

字段	名称	类型	描述
account_id	账户编号	bigint	主键且自增长
account_name	账户名称	varchar	易于记忆的名称
cash_amount	现金资产	bigint	以分为单位
stock_amount	股票资产	bigint	以分为单位

的历史表 t_account_hist，除上述字段外，还加上 hist_date 字段，用于记录每一天的资产。

4.1.5 账户流水表

显示用户现金账户资金进出情况，与股票流水表主键相同，用于表示股票买卖过程中资金的变化情况。

4.1.6 用户股票表

用于表示用户当前拥有的股票。有对应的历史表，在上述字段基础上添加 hist_date 字段，用于记录每一天的资产。

4.1.7 用户股票流水表

记录用户股票买卖情况，主键与 t_account_id 相同，股票买卖是账户流水的一个子类。对应有历史表，除上述字段外，还有 hist_date 字段。

4.2 准备实验数据

创建一个新客户：

Listing 4: 创建新客户

股票预测：https://medium.com/m/global-identity?redirectUrl=https

Table 6: 账户流水表 (t_account_io)

字段	名称	类型	描述
account_io_id	账户流水编号	bigint	主键且自增长
account_id	账户编号	bigint	t_account 表外键
in_amount	转入金额	bigint	以分为单位
out_amount	转出金额	bigint	以分为单位
balance	余额	bigint	以分为单位
io_date	发生日期	datetime	

Table 7: 用户股票表 (t_user_stock)

字段	名称	类型	描述
user_stock_id	账户流水编号	bigint	主键且自增长
user_id	用户编号	bigint	t_account 表外键
stock_id	股票编号	bigint	以分为单位
hold_vol	持有量	bigint	
price	价格	bigint	
hold_amount	转出金额	bigint	以分为单位
balance	余额	bigint	以分为单位
io_date	发生日期	datetime	

5 深度学习入门

5.1 Tensorflow 底层技术

5.1.1 依赖控制

由于 tensorflow 相当于一门语言，我们在定义计算图时，相当于进行编程，这些程序只有在 session 中才会执行，而 session 中是会根据依赖关系来决定执行哪些代码，所以我们必须通过依赖关系来告诉 tensorflow 执行哪些代码，如下所示：

```
\end{lstinputlisting

:-1: No account for team "9CVMN4UZK4". Add a new account in the Accounts preference pane or verify that your accounts have valid credentials. (in target 'ServiceExtension')

Showing Recent Messages
:-1: No profiles for 'cn.rongcloud.im.shareextension' were found: Xcode couldn't find any iOS App Development provisioning profiles matching 'cn.rongcloud.im.shareextension'. (in target 'SealTalkShareExtension')
```

```
\subsection{线性回归}
线性回归的定义：假设有一个问题，观察到的样本为$\boldsymbol{x} \in R^{n}$且共有$m$个训练样本，同时每个
样本$\boldsymbol{x}_{i}$对应一个数值$y_{i}$，并且我们假设其对应关系为：
$y=\boldsymbol{w} \cdot \boldsymbol{x}+b$，整个问题可以表示为：
\begin{equation}
    \begin{aligned}
```

Table 8: 用户股票表 (t_user_stock_io)

字段	名称	类型	描述
user_stock_io_id	账户流水编号	bigint	主键且自增长
user_stock_id	用户编号	bigint	t_account 表外键
buy_vol	买入量	bigint	手数
buy_price	买入价格	bigint	
buy_cost	买入成本	bigint	
buy_amount	买入金额	bigint	包括印花税、手续费、所得税等成本
sell_vol	卖出量	bigint	以分为单位
sell_price	卖出价格	bigint	以分为单位
sell_cost	卖出成本	bigint	包括印花税、手续费、所得税等成本
sell_amount	卖出金额	bigint	
hold_vol	持仓量	bigint	
hold_price	收盘价	bigint	
hold_amount	市值	bigint	

```
        \boldsymbol{y} = X \cdot \boldsymbol{w} + \boldsymbol{b}
    \end{aligned}
    \label{e000001}
\end{equation}
其中矩阵$X \in R^{m \times n}$, 每一行代表一个样本。\newline
我们的代价函数定义为最小平方误差函数:
\begin{equation}
    \begin{aligned}
        \mathcal{L}=\frac{1}{n}\sum_{i=1}^{m}(y_{i} - \boldsymbol{w} \
    boldsymbol{x}_{i} + b_{i})
    \end{aligned}
    \label{e000002}
\end{equation}



这时我们的任务就变为:
\begin{equation}
    \begin{aligned}
        \mathop{\arg \min}_{\boldsymbol{w}, \boldsymbol{b}} \frac{1}{n}\
    sum_{i=1}^{m}(y_{i} - \boldsymbol{w} \boldsymbol{x}_{i} + b_{i})
    \end{aligned}
    \label{e000003}
\end{equation}
由于这个问题比较简单, 解这个问题有两种方法: 解析法和迭代法。其中解析法的
    解为:
\begin{equation}
    \begin{aligned}
        \bar{\boldsymbol{x}}=\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_{i
    } \\
        \bar{y}=\frac{1}{n} \sum_{i=1}^{n} y_{i}
    \end{aligned}
    \label{e000004}
\end{equation}
其解析解为:
\begin{equation}
    \begin{aligned}
        \hat{\boldsymbol{w}}=\frac{\sum_{i=1}^{n}(\boldsymbol{x} - \bar{\
    boldsymbol{x}})(y_{i}-\bar{y})}{\sum_{i=1}^{n}(\boldsymbol{x}_{i} - \
    bar{\boldsymbol{x}})^{2}} \\
        \hat{\boldsymbol{b}}=\bar{y}-\boldsymbol{w} \bar{\boldsymbol{x}}
    \end{aligned}
```

```
    \label{e000005}
\end{equation}
如果是迭代法则为梯度下降算法，参数更新公式为：
\begin{equation}
    \begin{aligned}
        \boldsymbol{w}_{t+1}=\boldsymbol{w}_{t} - \alpha \frac{\partial {\mathcal{L}}}{\partial{\boldsymbol{w}_{t}}} \\
        \boldsymbol{b}_{t+1} = \boldsymbol{b}_{t} - \alpha \frac{\partial {\mathcal{L}}}{\partial{\boldsymbol{b}_{t}}}
    \end{aligned}
    \label{e000006}
\end{equation}


在讲解具体的代码之前，我们先来讲解一下Python中的\_\_call\_\_函数。我们知道，在Python中，类实例也是可调用对象，
只需要在类定义中添加\_\_call\_\_函数定义，即使用类实例加括号的形式进行调用了。我们利用这一特性定义线性回归类，如
下所示：
\lstset{language=PYTHON, caption={线性回归类定义}, label={c000005}}
\begin{lstlisting}
    class LinearRegression(object):
    def __init__(self):
        self.w = tf.get_variable('w',
                    dtype=tf.float32, shape=[],
                    initializer=tf.zeros_initializer()
        )
        self.b = tf.get_variable(
            'b', dtype=tf.float32, shape=[],
            initializer=tf.zeros_initializer()
        )

    def __call(self, x):
        return self.w * x + self.b
```

Listing 5: 获取股票基本信息

参考文献：I.MLearning [1999]—A.NikolaosAI [1999]—Bakry et al. [2015]

References

Andreas Nikolaos A.NikolaosAI. *A Book He Wrote*. His Publisher, Erewhon, NC, 1999.

Amr Bakry, Mohamed Elhoseiny, Tarek El-Gaaly, and Ahmed M. Elgammal. Digging deep into the layers of cnns: In search of how cnns achieve view invariance. *CoRR*, abs/1508.01983, 2015. URL http://arxiv.org/abs/1508.01983.

Ivan Marc I.MLearning. Some related article I wrote. *Some Fine Journal*, 99(7):1–100, January 1999.