
quant
量化交易策略

周杰，上海交通大学计算机系本科，复旦大学管理学硕士，杭州道进资产管理公司创始人之一。从事量化交易十余年，投资业绩斐然。目前主要研究高频交易与量化研究方法论。

CONTENT

1 量化研究方法论

我们如何设计一个策略

2 有色金属套利策略

从零开始的实践

3 策略陷阱

自上而下vs自下而上

免责声明：
本文内容不可视为投资意见。
市场有风险，入市需谨慎。

01

量化研究方法论

我们如何设计一个策略

想法

- 基于学术研究的理论
- 在交易过程中的新思路

模型雏形

- 用python或其他工具的简单验证
- 通常会进行小规模的数据分析

历史数据测试

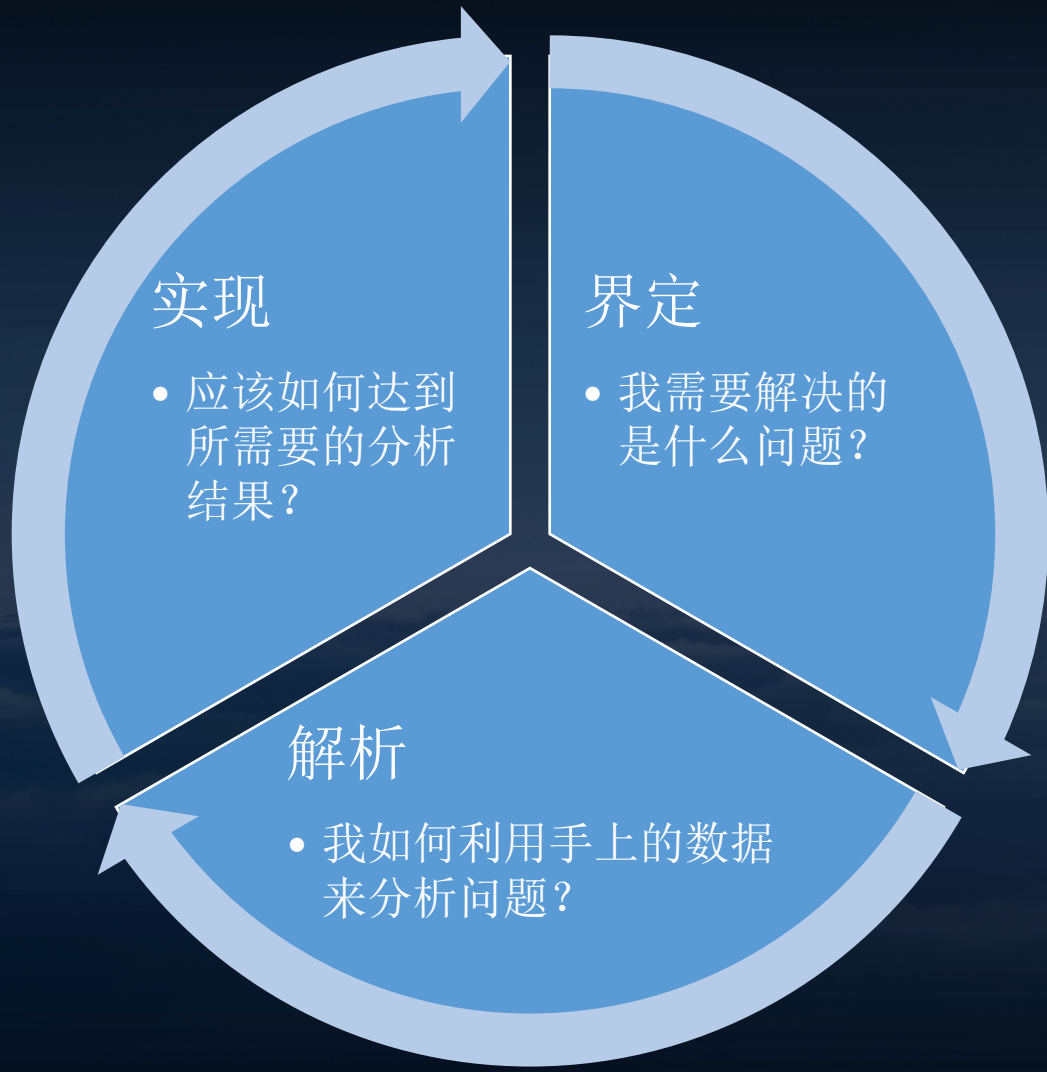
- 用C++进一步完善策略模型
- 使用大规模全部相关数据，很少抽样
- 取得数据检验的评估指标

细节修正

- 解决实盘可能遇到的问题
- 将策略抽象化，策略精简
- 写出统一的策略接口，以用于实盘测试

实盘测试

- 以测试为目的
- 严格监控新模型的风险
- 表现评估，确定实盘交易指标，是否有效


界定
• 我需要解决的是什么问题？
解析
• 我如何利用手上的数据来分析问题？
实现
• 应该如何达到所需要的分析结果？

定价体系：
BSM期权定价、基于基本面的股票定价等等；

· 因子体系：
来源于珐玛的CAPM理论，通过将信号元做线性回归来提供信息量。

· 产品体系：
最常见的是FOF，或者MOM，以及曾经备受争议的ABS等等；

· 套利体系：
自协整概念推广以来，统计套利在过去的三十年里进入了一个爆发期；

· 固收体系：
基于收益率衍生概念对货币、外汇、债券市场的交易；

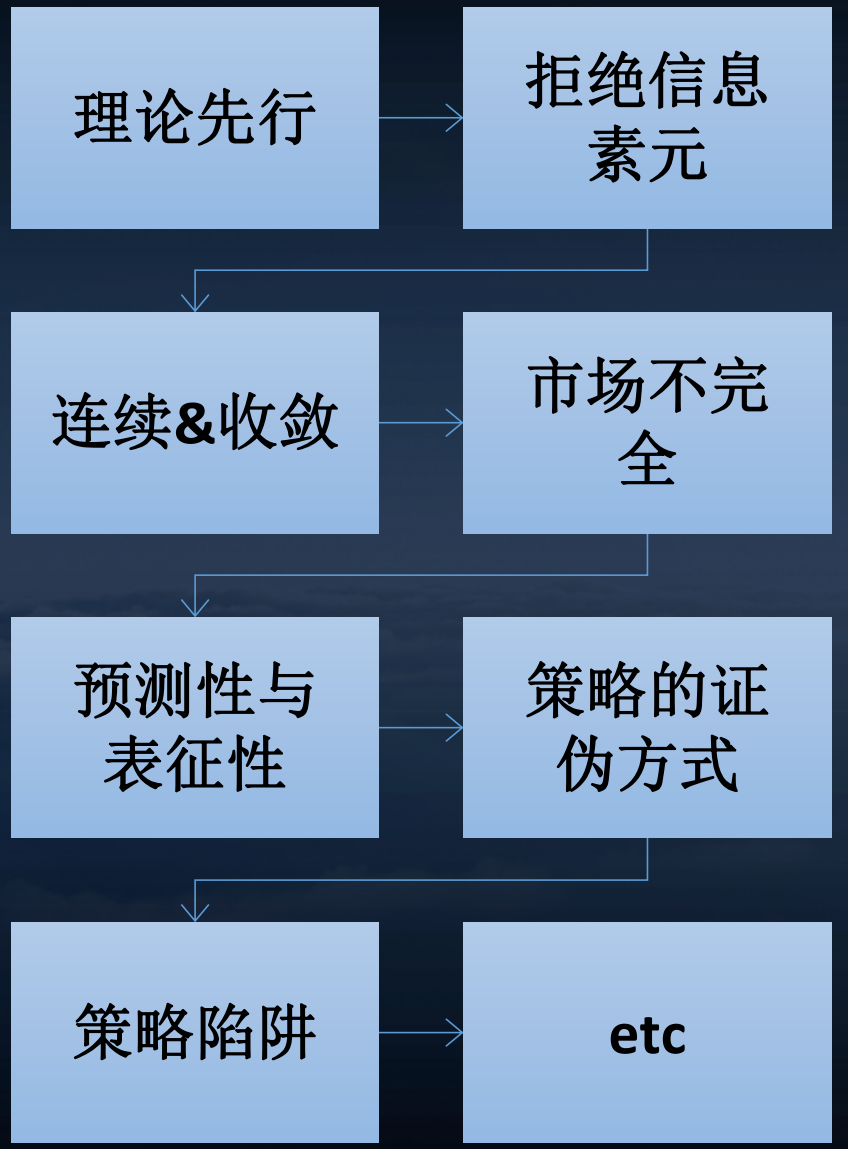
· 高频体系：
基于市场微观结构的验证；

为什么深度学习体系不在其列？

因为我始终没把深度学习当作一个单独的策略系统，而是一种研究方法。

当我讨论如何制定一个策略是，我们究竟在说啥？

信息元的发掘与使用。



量化交易研究方法论

一切皆套路。

方法论意味着你的工作步骤与要求，你可以理解为一个操作手册，
在任一体系下你每时每刻应该干什么。

02

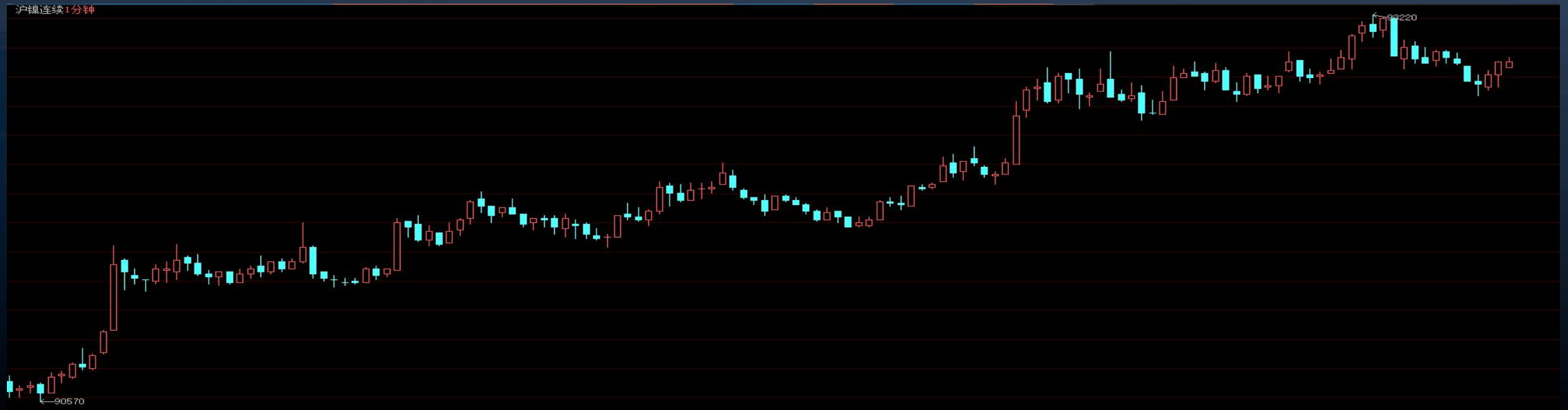
有色金属套利策略

从零开始的实践

几个基础概念：

套利：买入某种交易对象的同时，卖出相关的另一种交易对象，当两者的价差收缩或扩大到一定程度的时候，平仓了结头寸。


沪铜连续1分钟
39750
38990


沪镍连续1分钟
93220
90570

简单线性回归模型

该模型也称作一元一次回归方程，模型中：
y：因变量
x：自变量
a：常数项（回归直线在y轴上的截距）
b：回归系数（回归直线的斜率）
e：随机误差（随机因素对因变量所产生的影响）

e的平方和也称为残差，残差是判断线性回归拟合好坏的重要指标之一

从简单线性回归模型可以知道，简单线性回归是研究一个因变量与一个自变量间线性关系的方法。

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
import statsmodels.api as sm
X = data['cu_close'].values.reshape(-1,1)
y = data['ni_close'].values.reshape(-1,1)
reg = LinearRegression()
reg.fit(X, y)
print("The linear model is: Y = {:.5} + {:.5}X".format(reg.intercept_[0], reg.coef_[0][0]))
```

The linear model is: Y = 3.1188e+04 + 1.872X

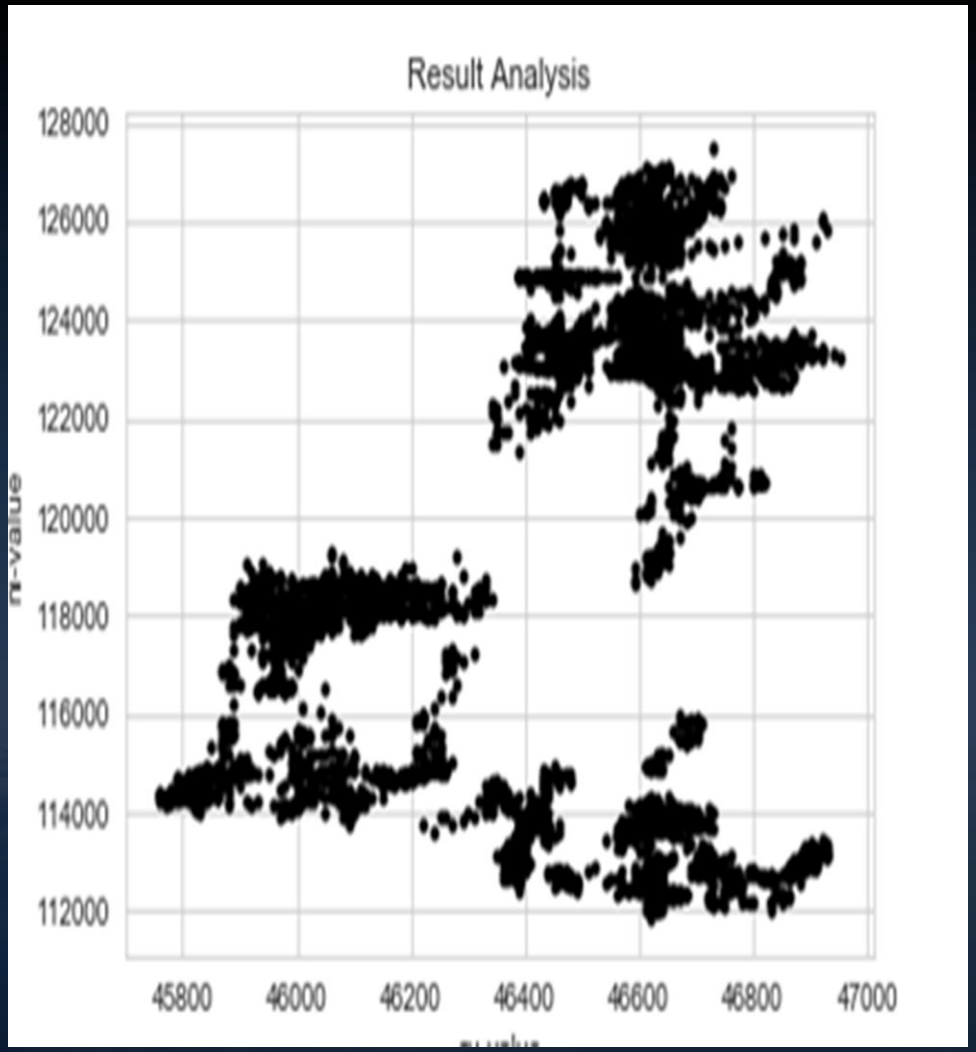
计算cu和ni的历史价格线性关系；

给定一个显著性水平，通常选取α=5%

假设其残差服从正态分布，并计算置信区间；

置信区间的上下限作为交易开仓的阀值；

突破置信区间上限开仓做空；突破区间下限开仓做多。



```
// 价差计算系数
Factor0 = Lots0*Data0.ContractUnit*Data0.BigPointValue;
Factor1 = Lots1*Data1.ContractUnit*Data1.BigPointValue;

// 计算价差并输出价差K线
CC = Data0.Close*Factor0 - Data1.Close*Factor1;
OO = Data0.Open*Factor0 - Data1.Open*Factor1;
HH = Max(OO,CC);
LL = Min(OO,CC);

// 计算价差的周期通道上下轨
UpperLine = Highest(HH[1],Length);
LowerLine = Lowest(LL[1],Length);
PlotNumeric("UpperLine", UpperLine);
PlotNumeric("LowerLine", LowerLine);

// 开仓
 If(Data0.Marketposition == 0 And CC[1] <= LowerLine[1] And Data0.Vol > 0 And Data1.Vol > 0)
{
                Data0.SellShort(Lots0, data0.Open);
                Data1.Buy(Lots1, Data1.Open);
}

// 反向平仓
If(Data0.Marketposition == -1 And CC[1] >= UpperLine[1] And Data0.Vol > 0 And Data1.Vol > 0)
{
                Data0.BuyToCover(0, Data0.Open);
                Data1.Sell(0, Data1.Open);
}
```

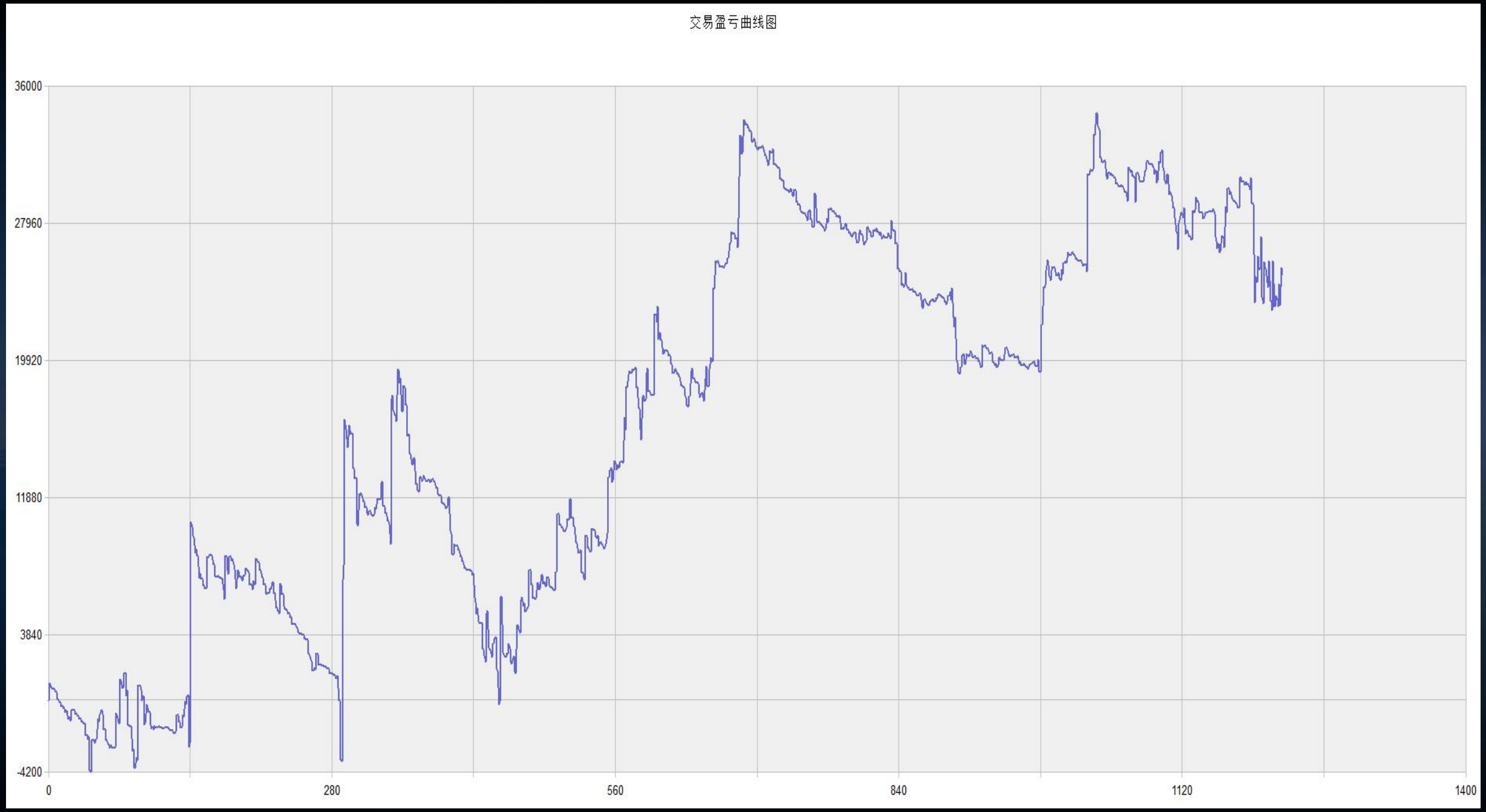
```
// 止损
Stopline = Highest(HH[1], StopLen);
If(Data0.MarketPosition == -1 And Data0.BarsSinceEntry > 0 And Data0.Vol > 0 And Data1.Vol >
0)
{
                If(CC[1] >= StopLine[1])
                {
                                Data0.BuyToCover(0,Data0.Open);
                                Data1.Sell(0,Data1.Open);
                }
}

// 开仓
If(Data0.Marketposition == 0 And CC[1] >= UpperLine[1] And Data0.Vol > 0 And Data1.Vol > 0)
{
                Data0.Buy(Lots0, data0.Open);
                Data1.SellShort(Lots1, Data1.Open);
}

// 反向平仓
If(Data0.Marketposition == 1 And CC[1] <= LowerLine[1] And Data0.Vol > 0 And Data1.Vol > 0)
{
                Data0.Sell(0, Data0.Open);
                Data1.BuyToCover(0, Data1.Open);
}

// 止损
Stopline = Lowest(LL[1], StopLen);
If(Data0.MarketPosition == 1 And Data0.BarsSinceEntry > 0 And Data0.Vol > 0 And Data1.Vol >
0)
{
                If(CC[1] <= StopLine[1])
                {
                                Data0.Sell(0,Data0.Open);
                                Data1.BuyToCover(0,Data1.Open);
                }
}
```


134390
[3]
[1]
[2]
[1]
[1]
[1]
[3]
[3]
[3]
[3]
131230
Close -158370.00 UpperLine -156700.00 LowerLine -164900.00

交易理念形成之后，需要使用历史数据对其进行侧试。这个测试过程就是人们常说的历史数据检验，或者说，复盘。
历史数据检验有两个目的。首先，历史数据检验可以在交易模型用于实际资金之前用大量历史数据验证其绩效。其次，回顾性测试还可以反映战略准确性，以捕捉利润机会，并显示战略是否可以改善，以实现更高的回报。

理想情况下，交易策略本身是由较小的历史数据集构成的。这些样品的策略性能称为“样品内性能”。

使用全新的历史数据集进行回顾测试称为“样本外推断"。为了对手头上的策略模型进行统计上显着的推断，有必要使用未在模型开发中使用的数据进行验证，这通常称之为实盘仿真。


交易盈亏曲线图
36000
27960
19920
11880
3840
-4200
0
280
560
840
1120
1400

性能概要

统计指标	全部交易	多头	空头	
净利润	24930.00	3040.00	21890.00	
总盈利	256170.00	117550.00	138620.00	
总亏损	(231240.00)	(114510.00)	(116730.00)	
总盈利/总亏损		1.11	1.03	1.19
交易手数	2440	1492	948	
盈利比率	34.39%	33.04%	36.50%	
盈利手数	839	493	346	
亏损手数	1556	957	599	
持平手数	45	42	3	
平均利润	10.22	2.04	23.09	
平均盈利	305.33	238.44	400.64	
平均亏损	(148.61)	(119.66)	(194.87)	
平均盈利/平均亏损		2.05	1.99	2.06
最大盈利	12960.00	10620.00	12960.00	
最大亏损	(3540.00)	(3450.00)	(3540.00)	
最大盈利/总盈利		0.05	0.09	0.09
最大亏损/总亏损		0.02	0.03	0.03
净利润/最大亏损		7.04	0.88	6.18
最大连续盈利手数		9	15	12
最大连续亏损手数		40	45	18
平均持仓周期		10	5	18
平均盈利周期		15	6	28
平均亏损周期		7	4	12
平均持平周期		3	1	37

最大使用资金	67905.00	44670.00	43746.00
最大持仓手数	4.00	3.00	3.00
交易成本合计	24400.00	14920.00	9480.00

收益率	24.93%	
年化收益率	53.82%	
有效收益率	36.71%	
月度平均盈利		3053.67
收益曲线斜率		0.0047
收益曲线截距		1.06
收益曲线R平方值		0.8448
夏普比率	1.4547	
总交易时间	249天	
持仓时间比率		9.64%
持仓时间	24天	
最大空仓时间		11天
持仓周期	6322	
资产最大升水		38890.00
发生时间	2020/01/22 09:09	
最大升水/前期低点		40.57%
单日最大资产回撤比率		5.37%

最大资产回撤值(按Bar收盘计算)

回撤值	(21180.00)
发生时间	2019/10/15 00:08
回撤值/前期高点	17.54%
净利润/回撤值	117.71%

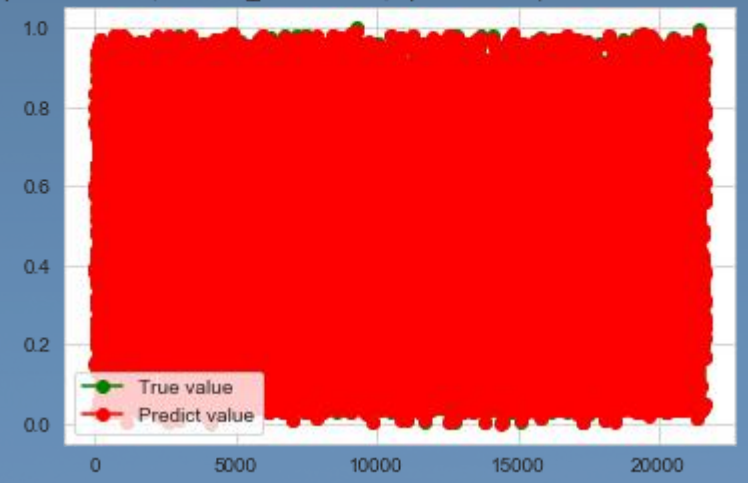
最大资产回撤值比率(按Bar收盘计算)

回撤值	(21180.00)
发生时间	2019/10/15 00:08
回撤值/前期高点	17.54%

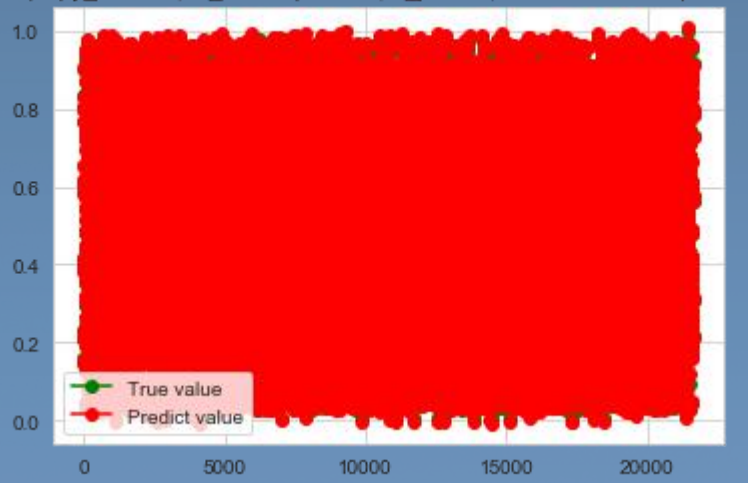
```
from sklearn import tree
from sklearn.linear_model import LinearRegression
from sklearn import svm
from sklearn import neighbors
from sklearn import ensemble
from sklearn.tree import ExtraTreeRegressor
from sklearn.model_selection import train_test_split

X = data[['cu_scaled', 'cu_change']]
y = data[['ni_scaled']]

train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=0.33, random_state=42)

# 回归部分
def try_different_method(model):
    model.fit(train_X, train_y)
    score = model.score(test_X, test_y)
    result = model.predict(test_X)
    plt.figure()
    plt.plot(np.arange(len(result)), test_y, "go-", label="True value")
    plt.plot(np.arange(len(result)), result, "ro-", label="Predict value")
    plt.title(f"model:{model}---score:{score}")
    plt.legend(loc="best")
    plt.show()


# 方法选择
# 1.决策树回归
model_decision_tree_regression = tree.DecisionTreeRegressor()

# 2.线性回归
model_linear_regression = LinearRegression()

# 3.SVM回归
model_svm = svm.SVR()
```

```
# 4.kNN回归
model_k_neighbor = neighbors.KNeighborsRegressor()

# 5.随机森林回归
model_random_forest_regressor = ensemble.RandomForestRegressor(n_estimators=200)  # 使用20个决策树

# 6.Adaboost回归
model_adaboost_regressor = ensemble.AdaBoostRegressor(n_estimators=500)  # 这里使用50个决策树

# 7.GBRT回归
model_gradient_boosting_regressor = ensemble.GradientBoostingRegressor(n_estimators=1000)  # 这里使用100个决策树

# 8.Bagging回归
model_bagging_regressor = ensemble.BaggingRegressor()

# 9.ExtraTree极端随机数回归
model_extra_tree_regressor = ExtraTreeRegressor()

models = [model_decision_tree_regression,
          model_linear_regression,
          model_svm,
          model_k_neighbor,
          model_random_forest_regressor,
          model_adaboost_regressor,
          model_gradient_boosting_regressor,
          model_bagging_regressor,
          model_extra_tree_regressor]

for model in models:
    try_different_method(model)
```

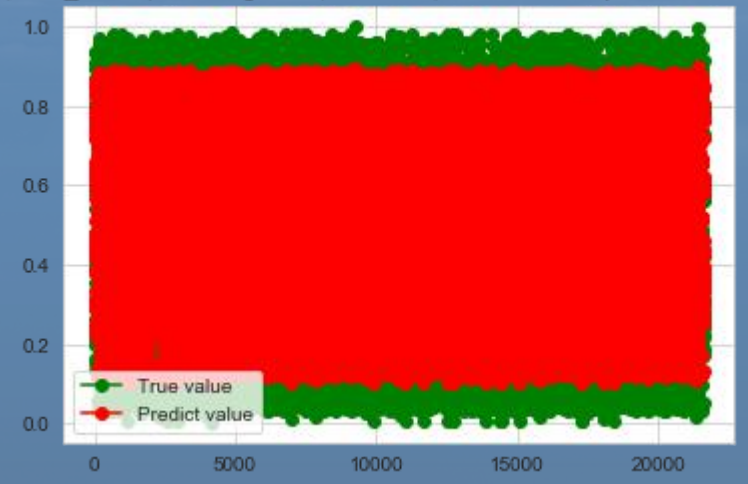
model:DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1,
min_samples_split=2, min_weight_fraction_leaf=0.0,
presort=False, random_state=None, splitter='best')---score:0.9999062592517386



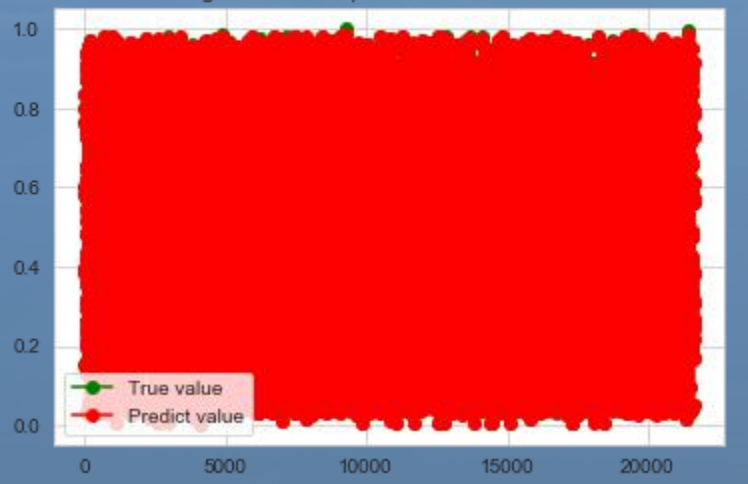
model:LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)---score:0.9996950769330426



model:SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='auto',
kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)---score:0.942212678587787

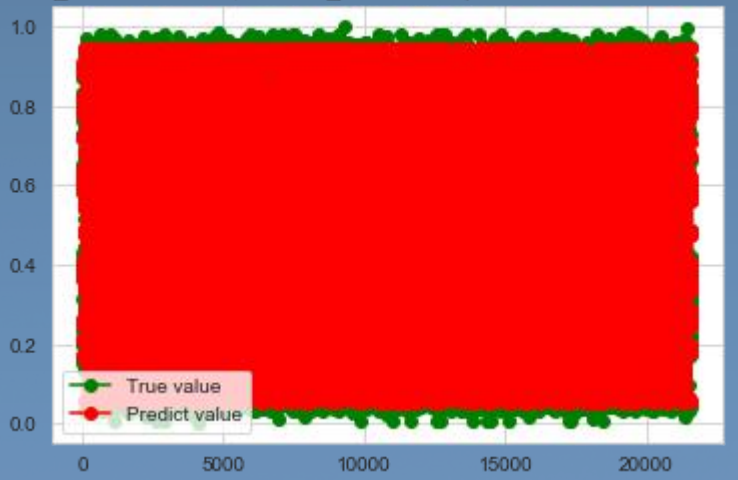


model:KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
metric_params=None, n_jobs=1, n_neighbors=5, p=2,
weights='uniform')---score:0.9999467728592004

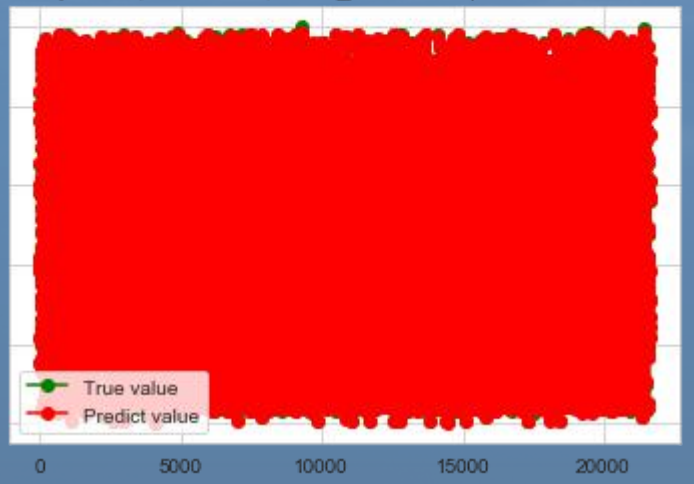


model:RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
max_features='auto', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=200, n_jobs=1,
oob_score=False, random_state=None, verbose=0, warm_start=False)---score:0.9999366291743532

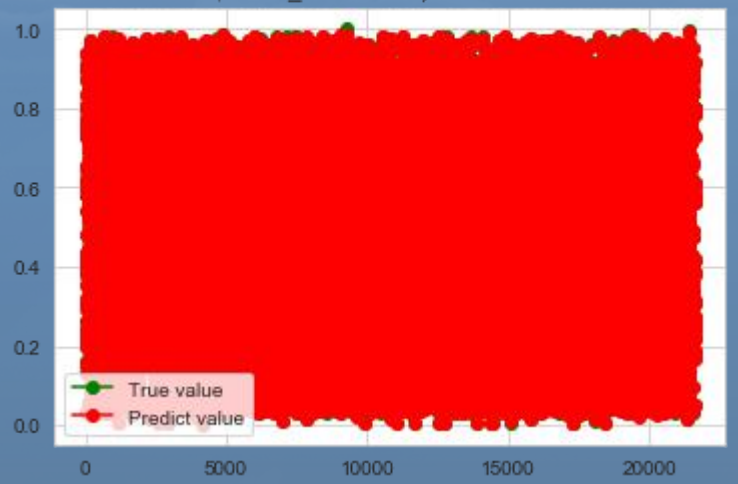
model:AdaBoostRegressor(base_estimator=None, learning_rate=1.0, loss='linear',
n_estimators=500, random_state=None)---score:0.9900202842395989



model:GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse', init=None,
learning_rate=0.1, loss='ls', max_depth=3, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1,
min_samples_split=2, min_weight_fraction_leaf=0.0,
n_estimators=1000, presort='auto', random_state=None,
subsample=1.0, verbose=0, warm_start=False)---score:0.9998829742835166



model:BaggingRegressor(base_estimator=None, bootstrap=True,
bootstrap_features=False, max_features=1.0, max_samples=1.0,
n_estimators=10, n_jobs=1, oob_score=False, random_state=None,
verbose=0, warm_start=False)---score:0.999935710848478



滑点

在实战交易中，往往最终成交价和预期价格有一定偏差。尤其是一些商品期货的人气比较低的时候，买一和卖一存在很大差距。

流动性

非主力合约的期货合约，往往成交量很低、流动性很差。简单说就是有价无市，有可能你按一个价格挂一天委托都成交不了，就算成交也不能全部成交。

瘸腿

单边成交，造成了风险暴露，对套利而言面临着非常危险的处境。

止损

哪怕是套利，也要有严格的风险控制制度，黑天鹅事件经常性出现，世间没有百分百的事。

老司机的几个建议：

03

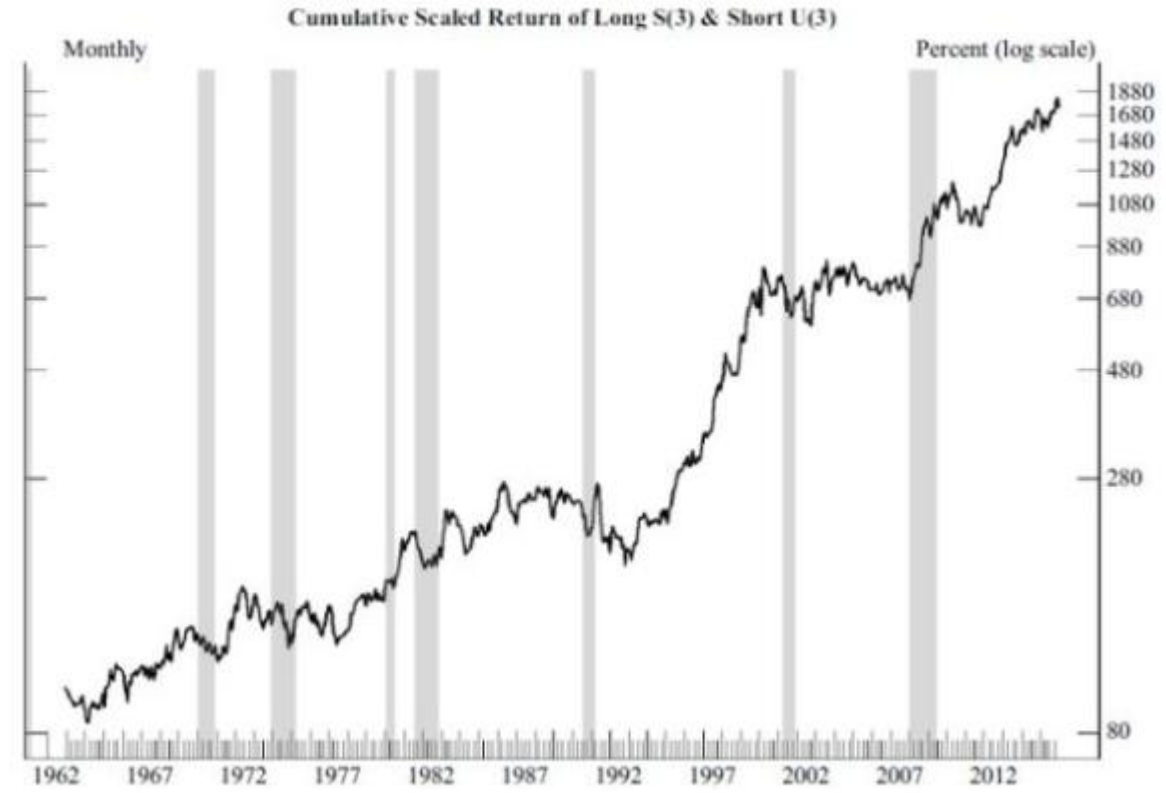
量化陷阱

自下而上vs自上而下

该策略的构建完全没有使用任何基本面或者交易数据，而仅仅依赖美股上市公司股票代码上的字母。比如苹果公司的股票代码是 AAPL，该代码上的第 1 至 4 位上的字母分别为 A、A、P 以及 L。该因子的构建方法是做多股票代码第三位字母为 S 的股票、做空股票代码第三位字母为 U 的股票（记为 S(3) – U(3)）。

EXHIBIT 1

Long–Short Market-Neutral Strategy Based on NYSE Stocks, January 1963 to December 2015



Notes: Gray areas denote NBER recessions. Strategy returns scaled to match S&P 500 T-bill volatility during this period.

Source: Campbell Harvey, using data from CRSP.

A clear economic foundation for any model。

对于从事量化交易的人来说，特别容易掉入“数字游戏”的圈子之中，提出一个数字上看似美好的模型而无法真正应用于实际交易之中。如果我以某个金融学或经济学原理为先验，构建了一个策略并测试有效，那么它大概是真有效；然而，如果我两眼一抹黑试了 100 个策略，然后只挑出了最好的那一个，那么这个策略很可能只是个 luck。很多人都会认为：量化交易是计算机视角下的数字结果；而事实恰恰相反，量化交易扎根于理论之中。

形成你自己的哲学基石

04

CQF介绍

国际一流量化专家团队

国际一流量化投资专家 - 科研

大部分是博士/教授，国际知名大学授课，经常是量化投资领域的经典教科书的作者，期刊的主编

Quantitative Finance 是很多国内外金融工程硕士学生的教科书。Paul 还是量化投资期刊的主编。

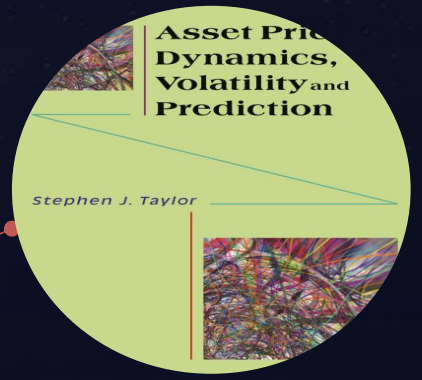
Dr.Paul Wilmott

Tony Guida

Tony Guida是Big Data and Machine Learning的作者，也是机器学习期刊的主编。

Peter Jaeckel博士是畅销教科书Monte Carlo Methods in Finance (2002)的作者 。

Dr. Peter Jaeckel



Professor Stephen Taylor

著名经济学家，随机波动和期权定价权威专家，国际一级期刊主编。是教科书Asset Price Dynamics, Volatility and Prediction的作者。

国际一流量化投资专家 - 实务

大都为知名量化投资团队负责人，有实际量化投资经验；CQF作为独立结构，不强求论文，侧重实务

Dr. Espen Gaarder Haug

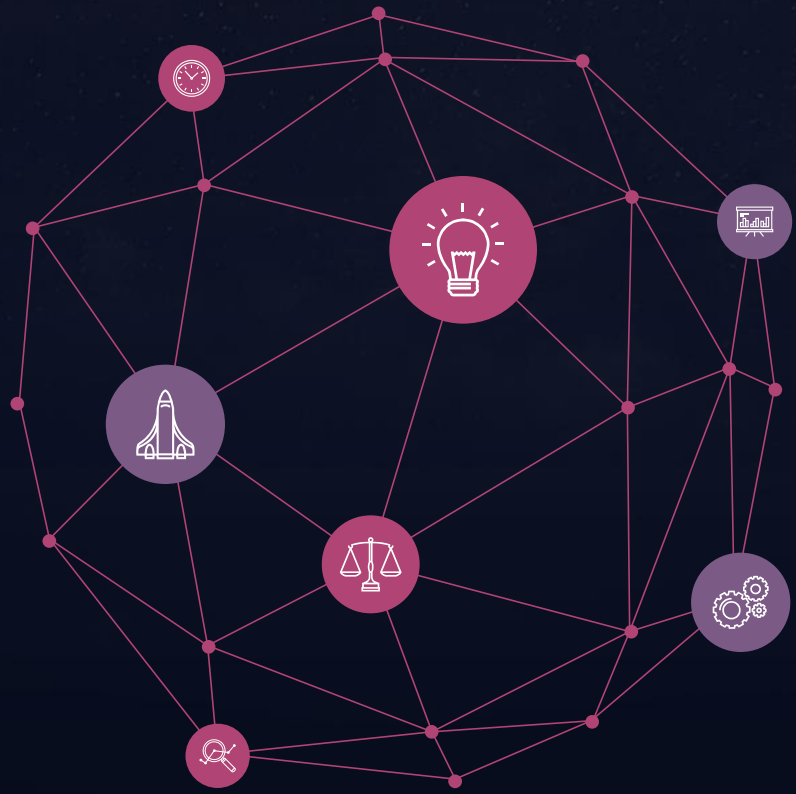
20年衍生品交易和研究经验，分别从业于J.P. Morgan和大型对冲基金Amaranth。

Dr. Jon Gregory

交易对手风险管理专家，曾就职与巴克莱，巴黎银行，和花旗银行。现在是Solum Financial Partners 的高级风险管理顾问。

Dr. Richard Vladimir Diamond

私募基金投资副总监。曾设计，编码并且实盘交易VIX futures。也曾直接管理上百万美金的交易账户。



Tony Guida

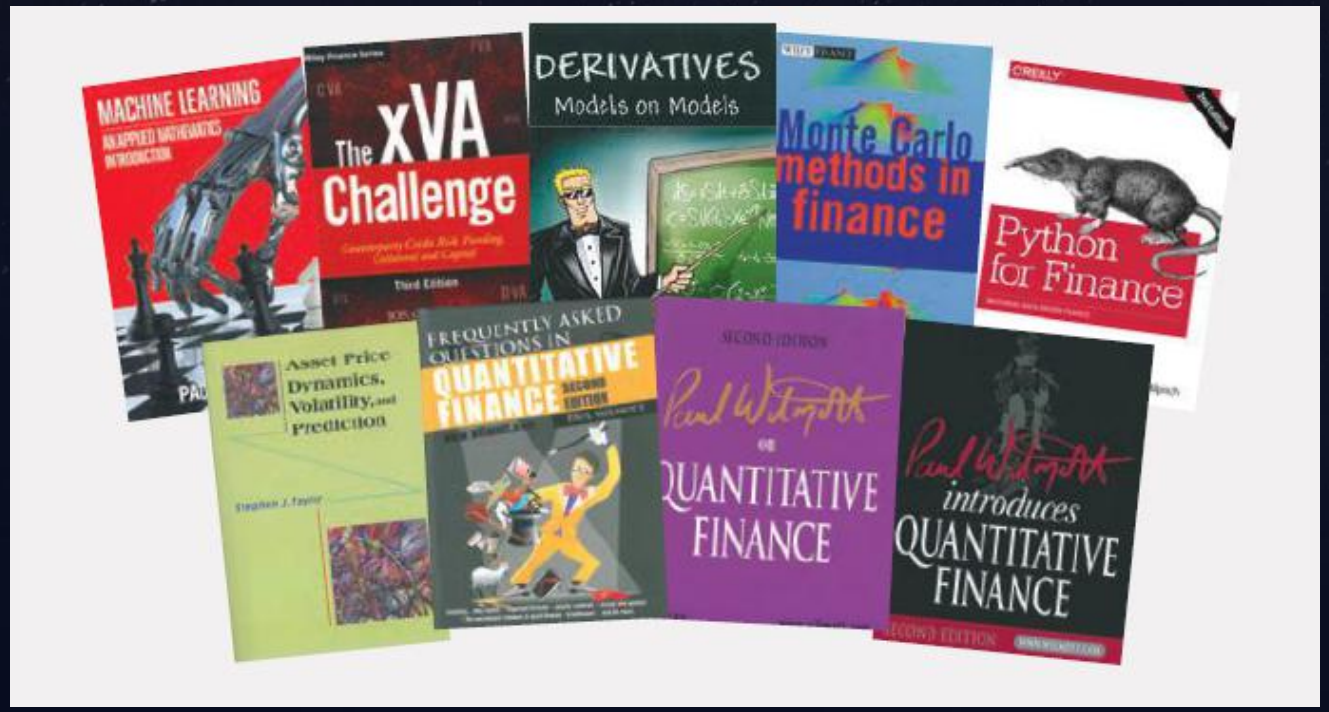
高级量化投资组合经理，参与管理60亿英镑的英国养老基金的投资组合。

Dr. Peter Jaeckel

原荷兰排行前三的银行的信用和衍生品分析部门主管。后来自己创立衍生品分析公司OTC Analytics。

Dr. Miquel Noguer Alonso

20年资产管理经验，曾就职于瑞士银行，Andbank，和毕马威。大数据和人工智能公司 Global AI 的 CDO.



知名量化教材合集

协会给所有报名的学员邮寄左边的量化投资领域教科书。这些教科书都是CQF的老师所编写，且正在被国际上各大学校的金融工程学院的硕士研究生使用。


THANKS!