人工智慧投資分析

AI for Investment Analysis

投資組合最佳化與程式交易

(Portfolio Optimization and Algorithmic Trading)

1082AIIA10

MBA, IMTKU (M2399) (8409) (Spring 2020)

Wed 3, 4 (10:10-12:00) (B206)

Min-Yuh Day

戴敏育

Associate Professor

副教授

Dept. of Information Management, Tamkang University

淡江大學 資訊管理學系

http://mail. tku.edu.tw/myday/

2020-06-10

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2020/03/04	人工智慧投資分析課程介紹 (Course Orientation on AI for Investment Analysis)
2	2020/03/11	AI 金融科技: 金融服務創新應用 (AI in FinTech: Financial Services Innovation and Application)
3	2020/03/18	機器人理財顧問與AI交談機器人 (Robo-Advisors and AI Chatbots)
4	2020/03/25	投資心理學與行為財務學 (Investing Psychology and Behavioral Finance)
5	2020/04/01	財務金融事件研究法 (Event Studies in Finance)
6	2020/04/08	人工智慧投資分析個案研究 I (Case Study on AI for Investment Analysis I)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
7	2020/04/15	Python AI投資分析基礎 (Foundations of AI Investment Analysis in Python)
8	2020/04/22	Python Pandas 量化投資分析 (Quantitative Investing with Pandas in Python)
9	2020/04/29	期中報告 (Midterm Project Report)
10	2020/05/06	Python Scikit-Learn 機器學習投資分析 (Machine Learning for Investment Analysis with Scikit-Learn in Python)
11	2020/05/13	TensorFlow 深度學習投資分析 I (Deep Learning for Investment Analysis with TensorFlow I)
12	2020/05/20	TensorFlow 深度學習投資分析 II (Deep Learning for Investment Analysis with TensorFlow II)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
13	2020/05/27	人工智慧投資分析個案研究 II (Case Study on Artificial Intelligence for Investment Analysis II)
14	2020/06/03	TensorFlow 深度學習投資分析 III (Deep Learning for Investment Analysis with TensorFlow III)
15	2020/06/10	投資組合最佳化與程式交易 (Portfolio Optimization and Algorithmic Trading)
16	2020/06/17	期末報告 I (Final Project Presentation I)
17	2020/06/24	期末報告 II (Final Project Presentation II)
18	2020/07/01	教師彈性補充教學

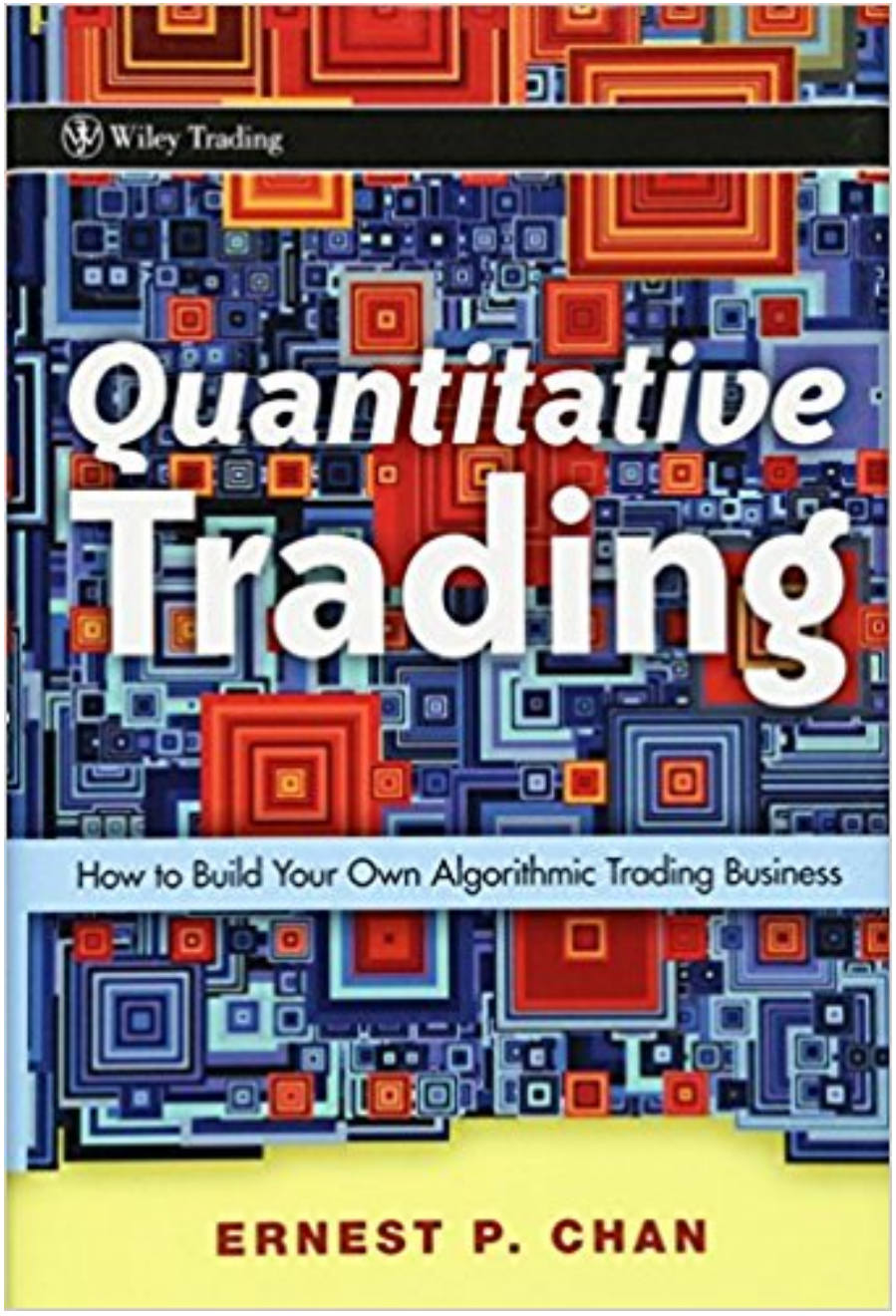
Portfolio Optimization and Algorithmic Trading

Outline

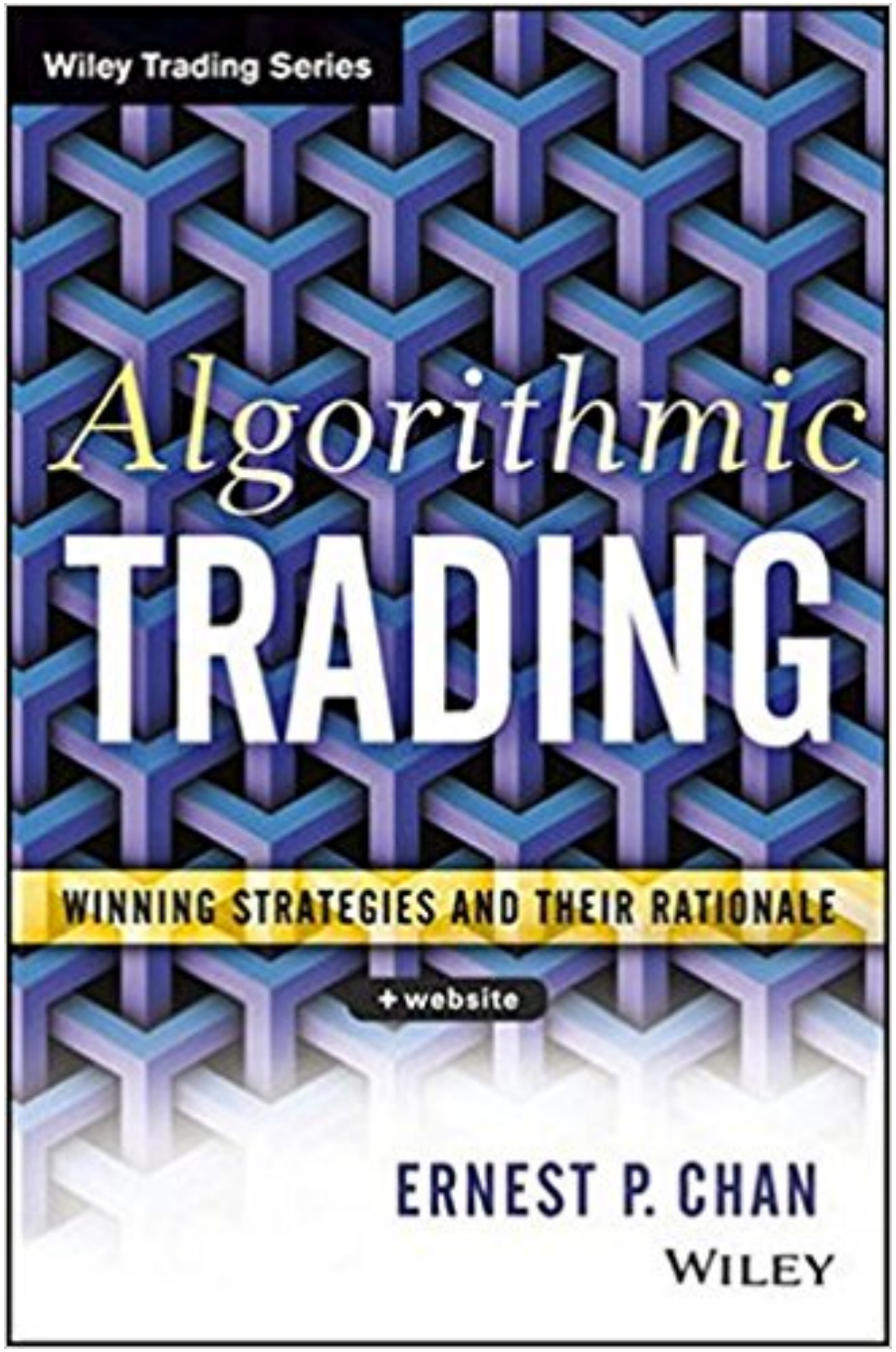
- Portfolio Optimization
- Algorithmic Trading

Portfolio Optimization

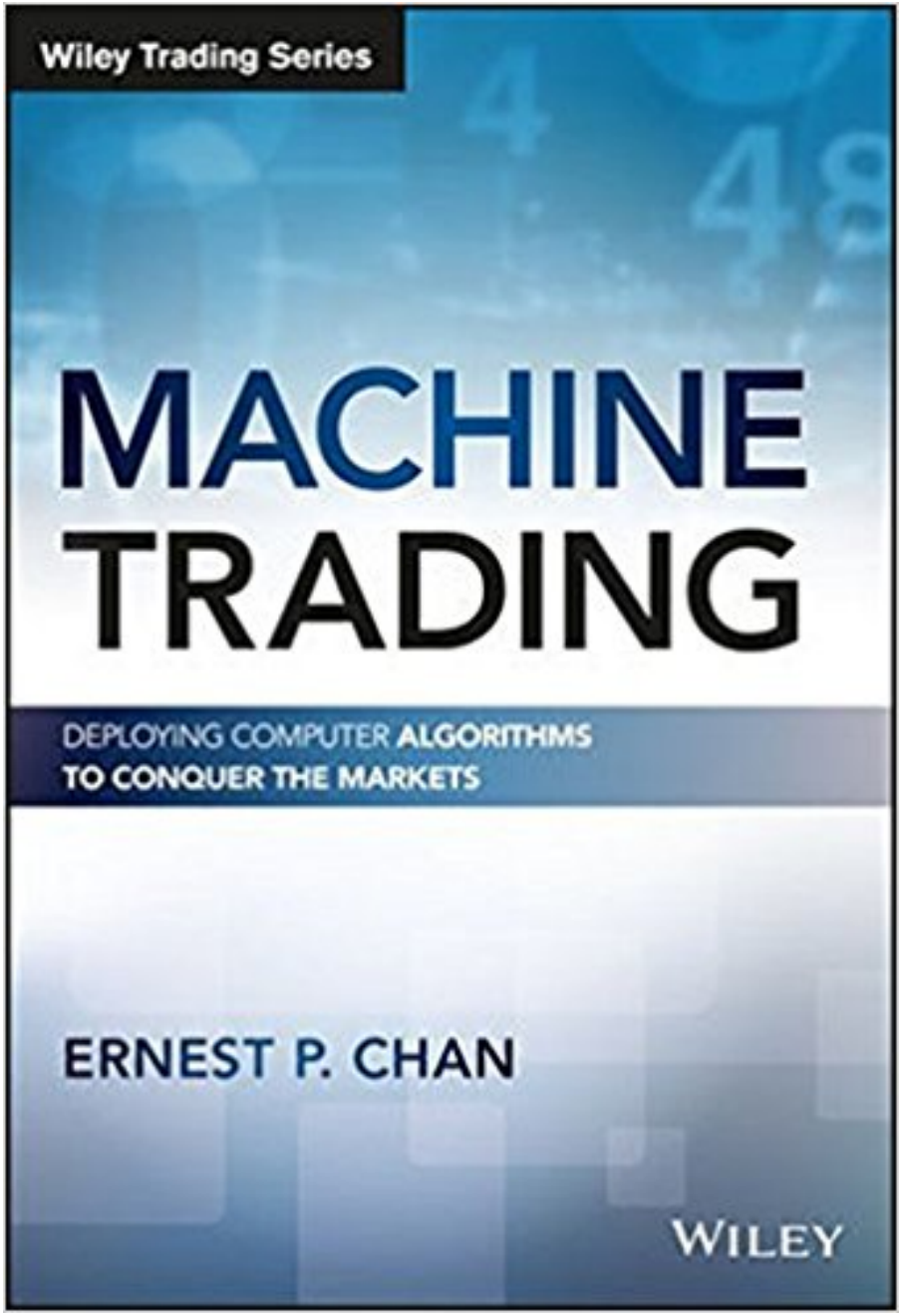
Algorithmic Trading



Source: Ernie Chan (2008), “Quantitative Trading: How to Build Your Own Algorithmic Trading Business”, Wiley



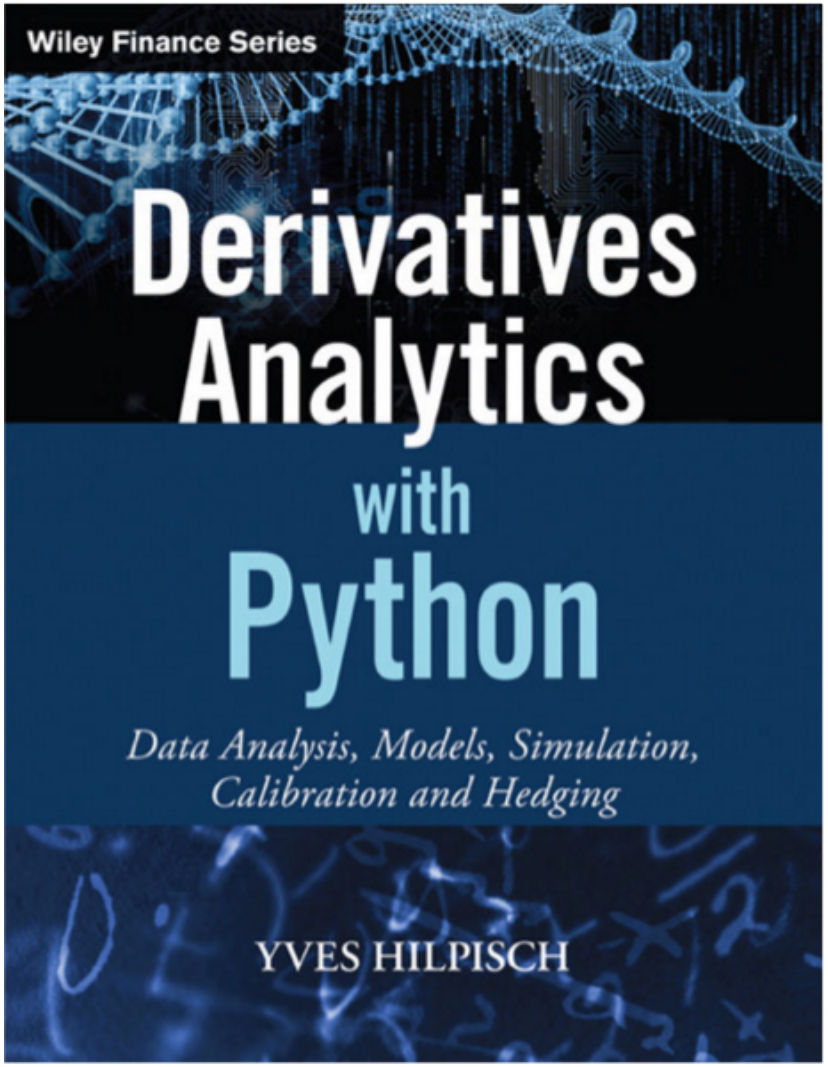
Source: Ernie Chan (2013), “Algorithmic Trading: Winning Strategies and Their Rationale”, Wiley



Source: Ernest P. Chan (2017), “Machine Trading: Deploying Computer Algorithms to Conquer the Markets”, Wiley

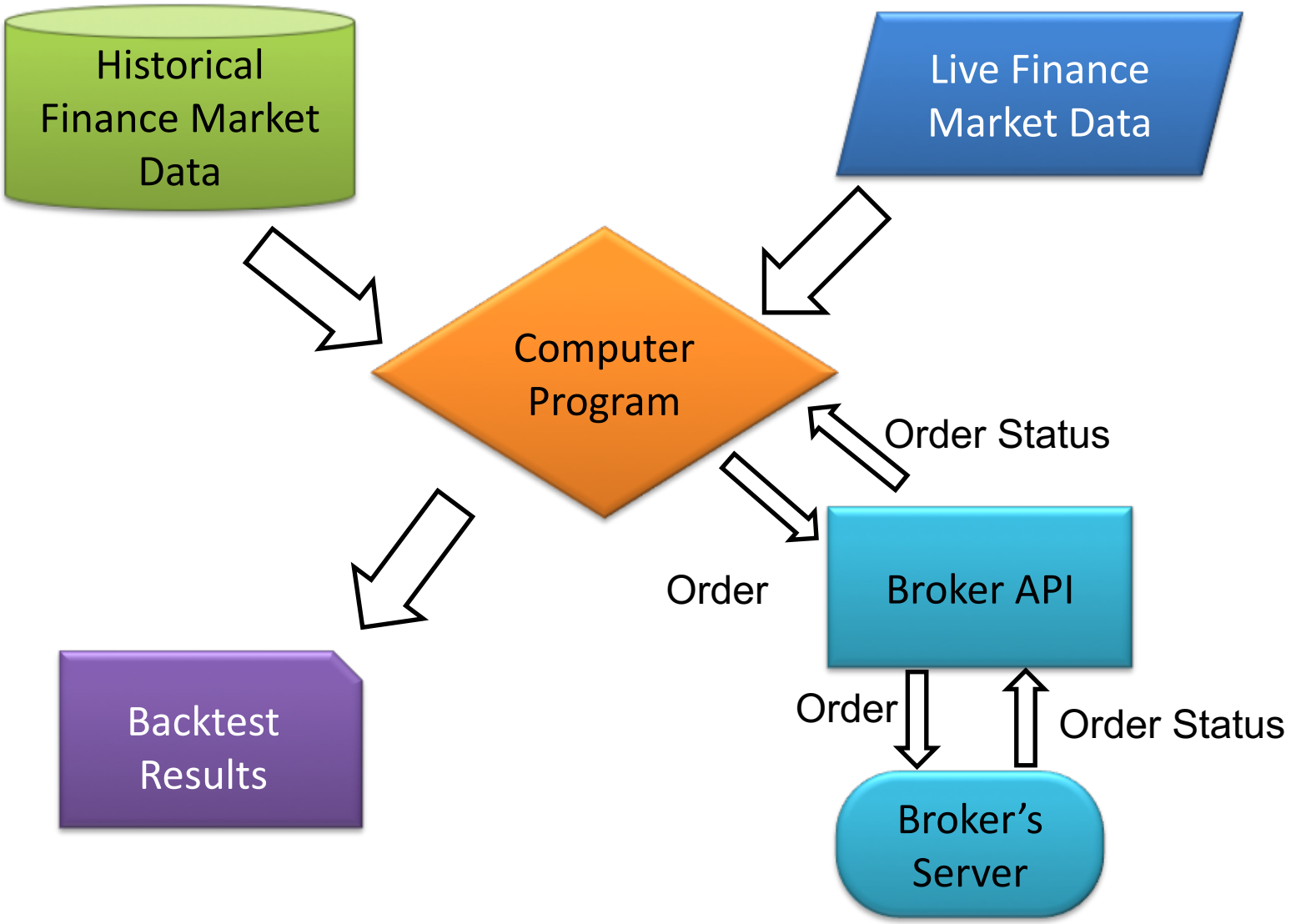
Yves Hilpisch (2015), Derivatives Analytics with Python:

Data Analysis, Models, Simulation, Calibration and Hedging, Wiley



Source: http://www.amazon.com/Derivatives-Analytics-Python-Simulation-Calibration/dp/1119037999/

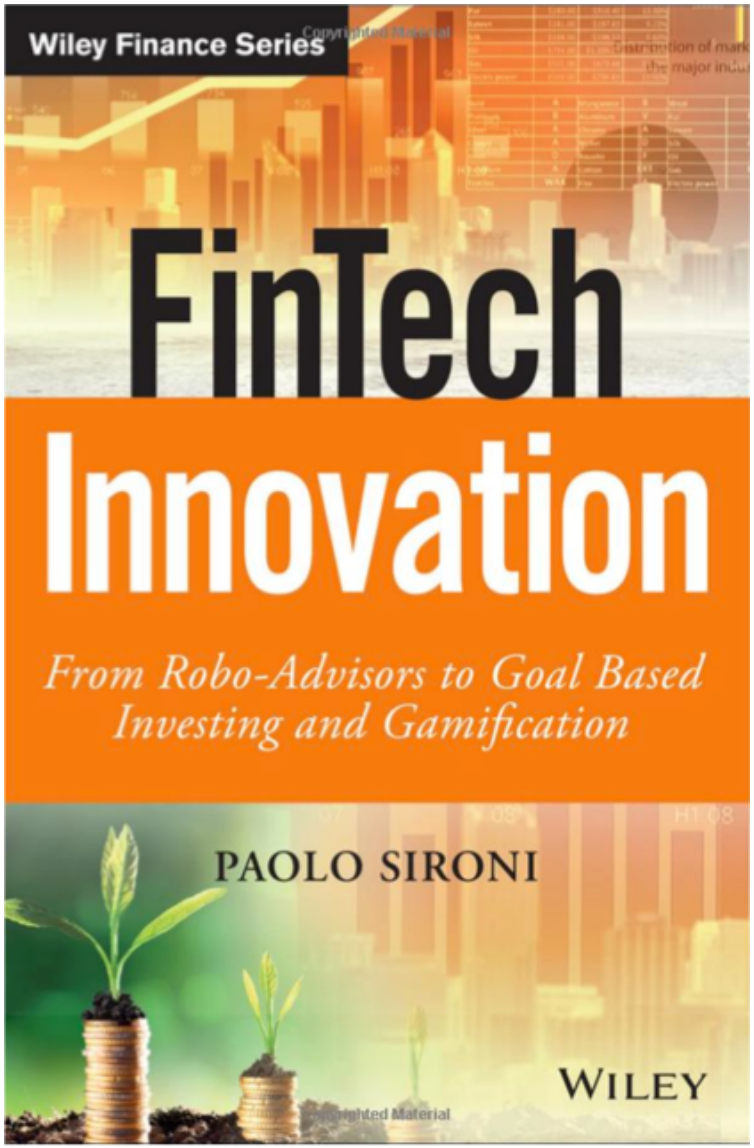
Algorithmic Trading



Source: Ernest P. Chan (2017), Machine Trading: Deploying Computer Algorithms to Conquer the Markets, Wiley

FinTech Innovation:
From Robo-Advisors to Goal Based Investing and Gamification,

Paolo Sironi, Wiley, 2016



Source: https://www.amazon.com/FinTech-Innovation-Robo-Advisors-Investing-Gamification/dp/1119226988

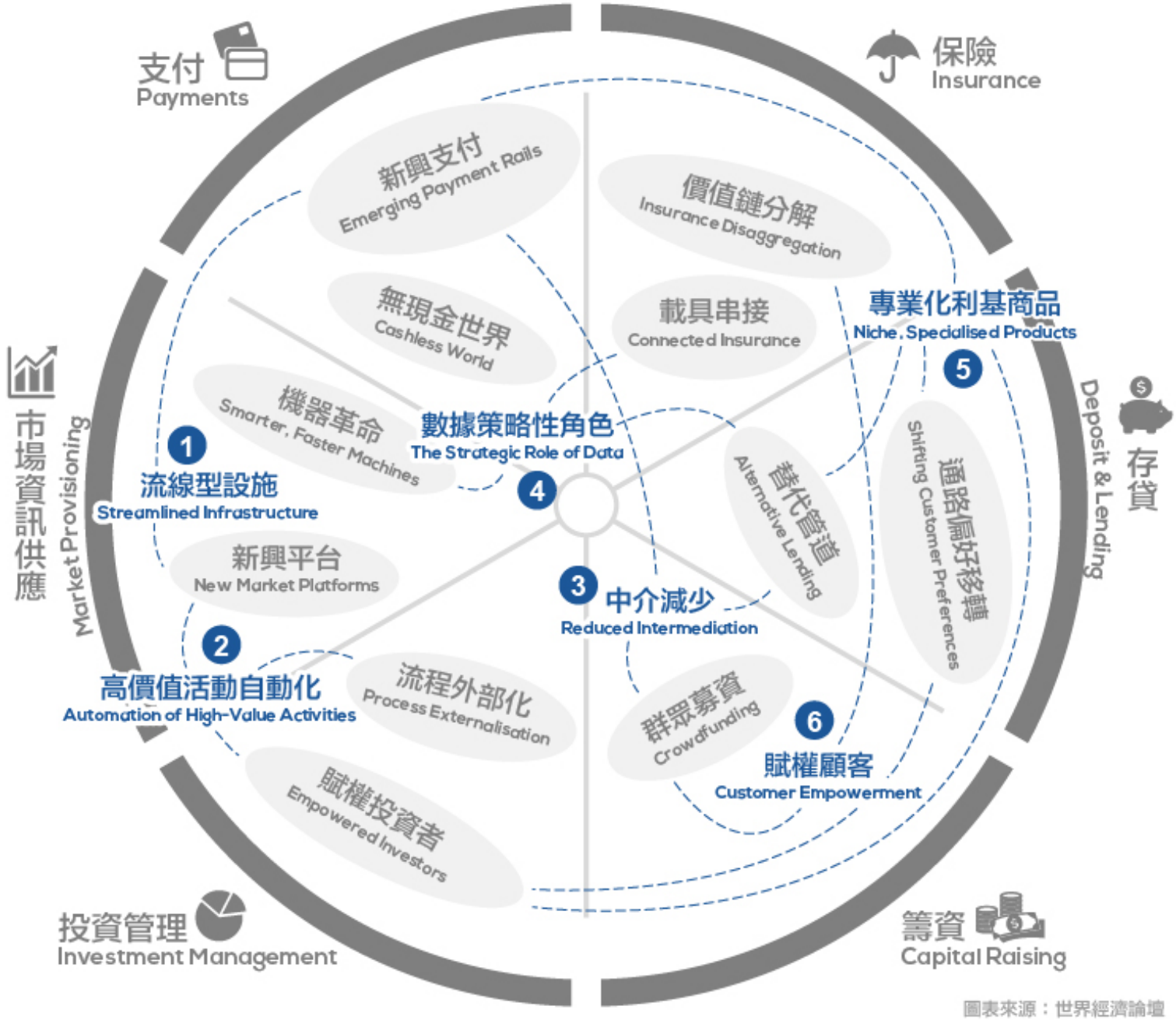
FinTech: Financial Services Innovation

Source: http://www3.weforum.org/docs/WEF_The_future__of_financial_services.pdf

FinTech:
Financial Services Innovation

1. Payments
2. Insurance
3. Deposits & Lending
4. Capital Raising
5. Investment Management
6. Market Provisioning

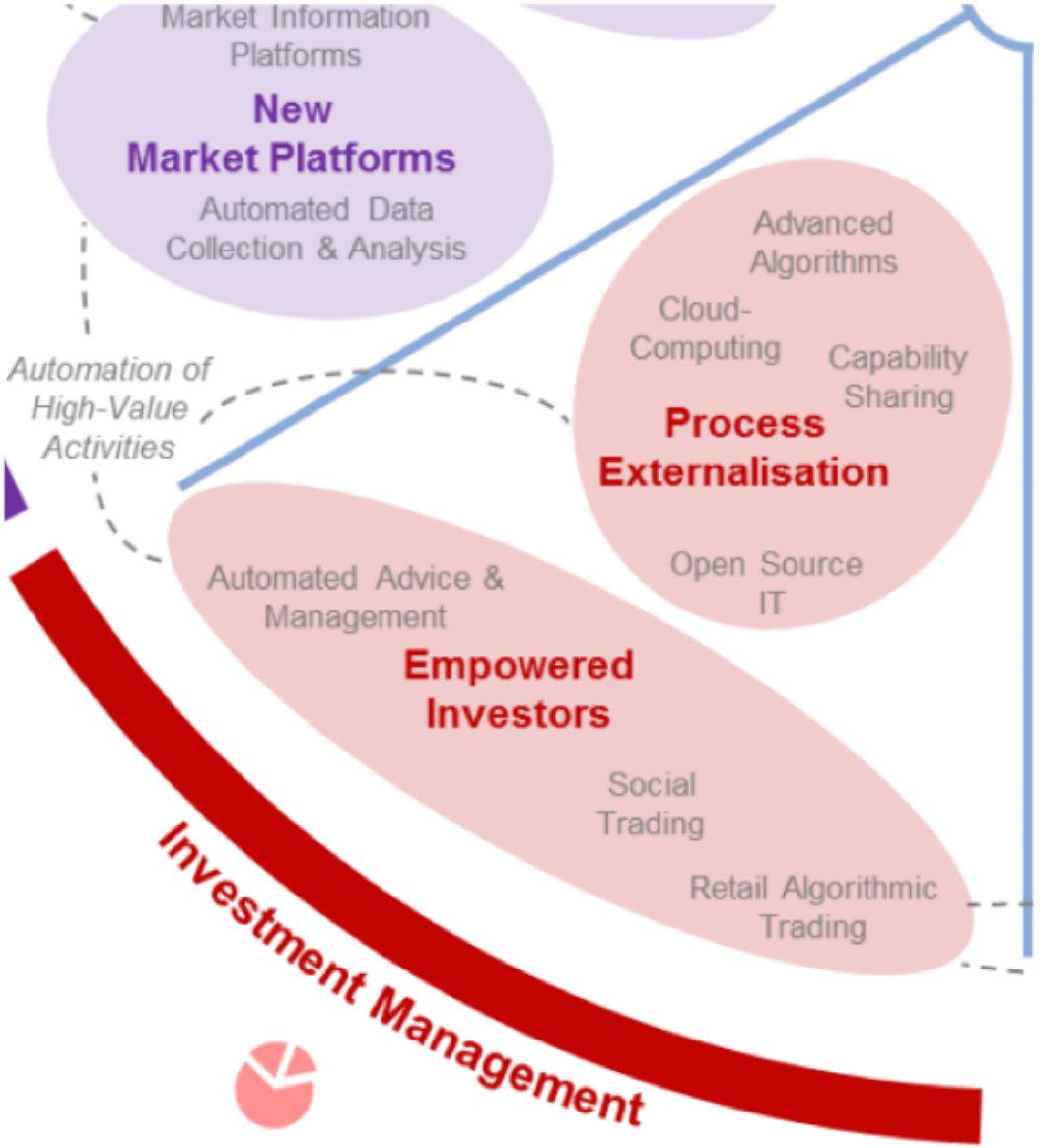
Source: http://www3.weforum.org/docs/WEF_The_future__of_financial_services.pdf



圖表來源：世界經濟論壇

Source: https://www.stockfeel.com.tw/2015年世界經濟論壇－未來的金融服務/

5 FinTech: Investment Management



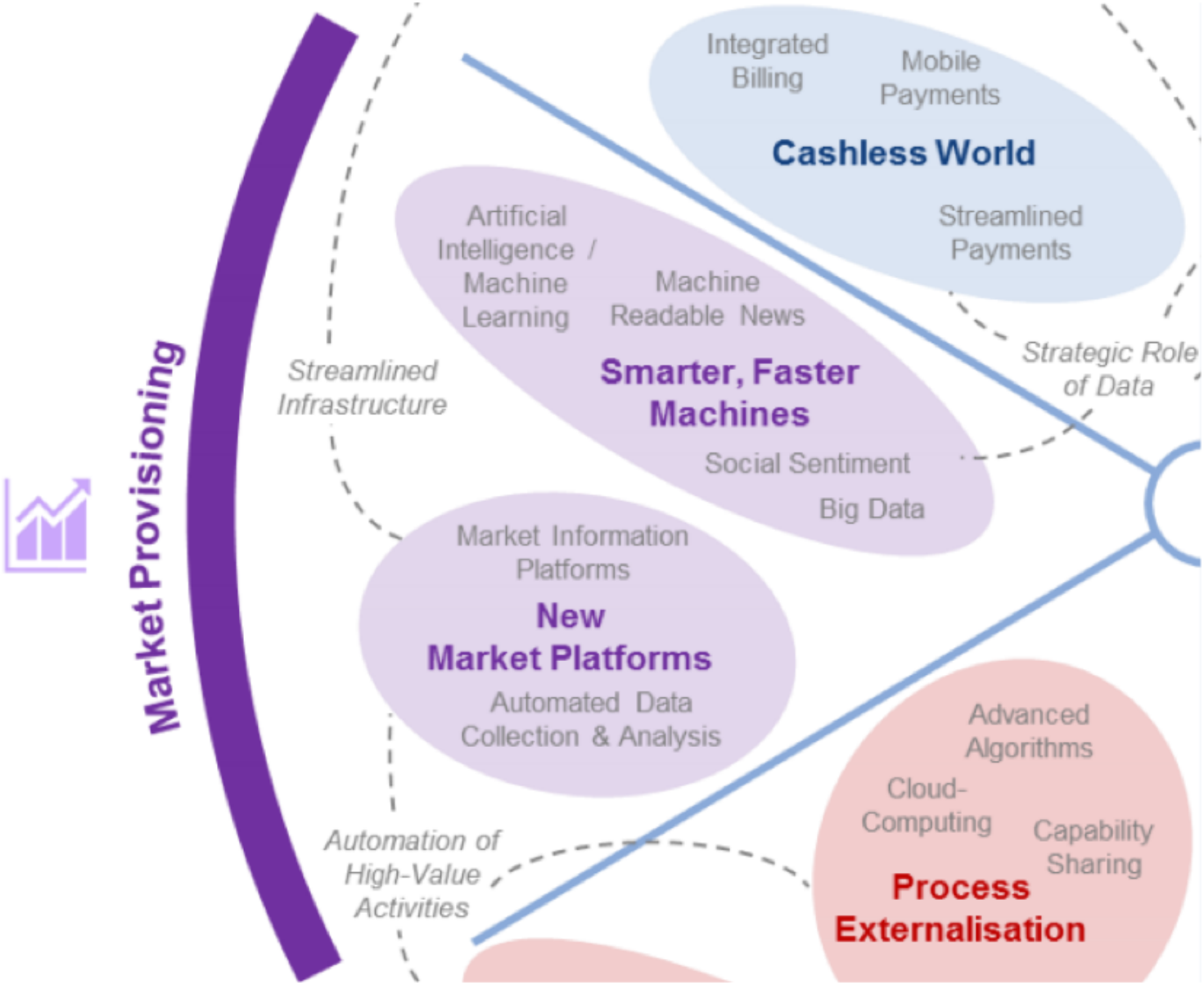
Source: http://www3.weforum.org/docs/WEF_The_future__of_financial_services.pdf

5 FinTech: Investment Management Empowered Investors Process Externalization

圖表來源：Fugle團隊整理

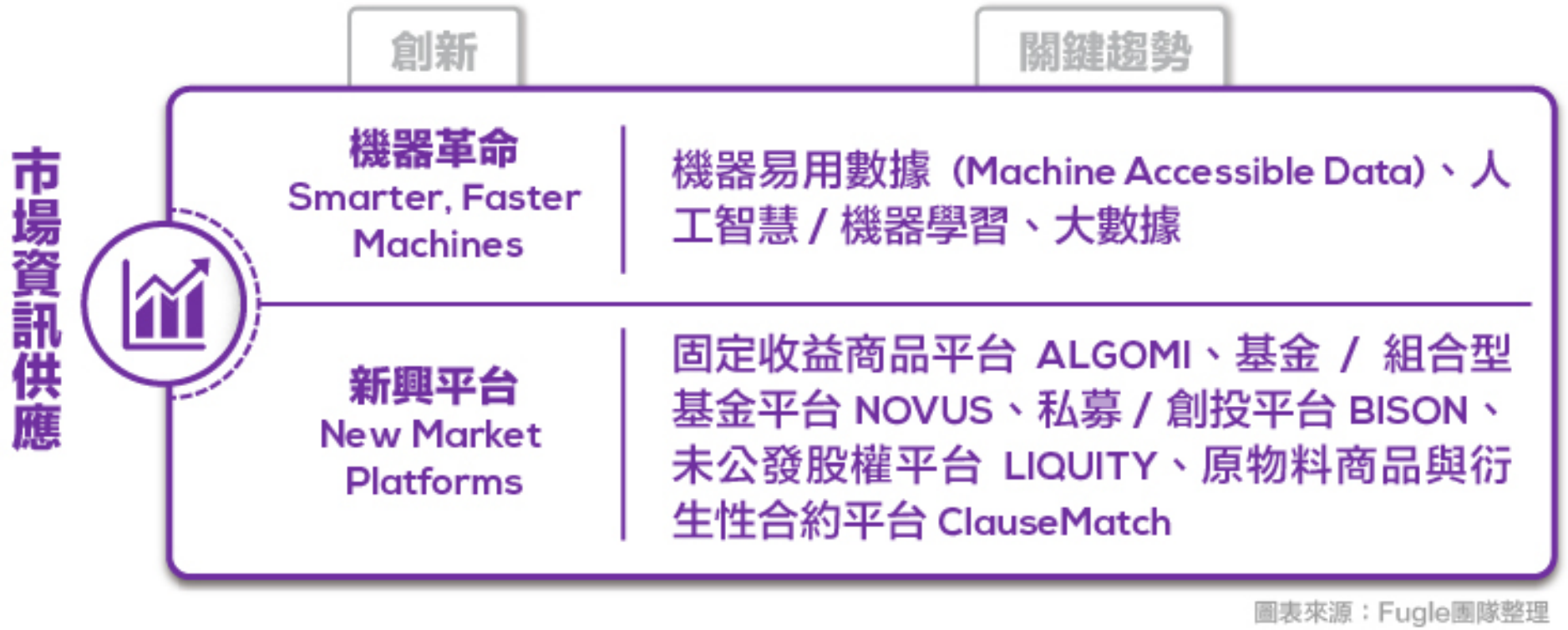
Source: https://www.stockfeel.com.tw/2015年世界經濟論壇－未來的金融服務/

6 FinTech: Market Provisioning



Source: http://www3.weforum.org/docs/WEF_The_future__of_financial_services.pdf

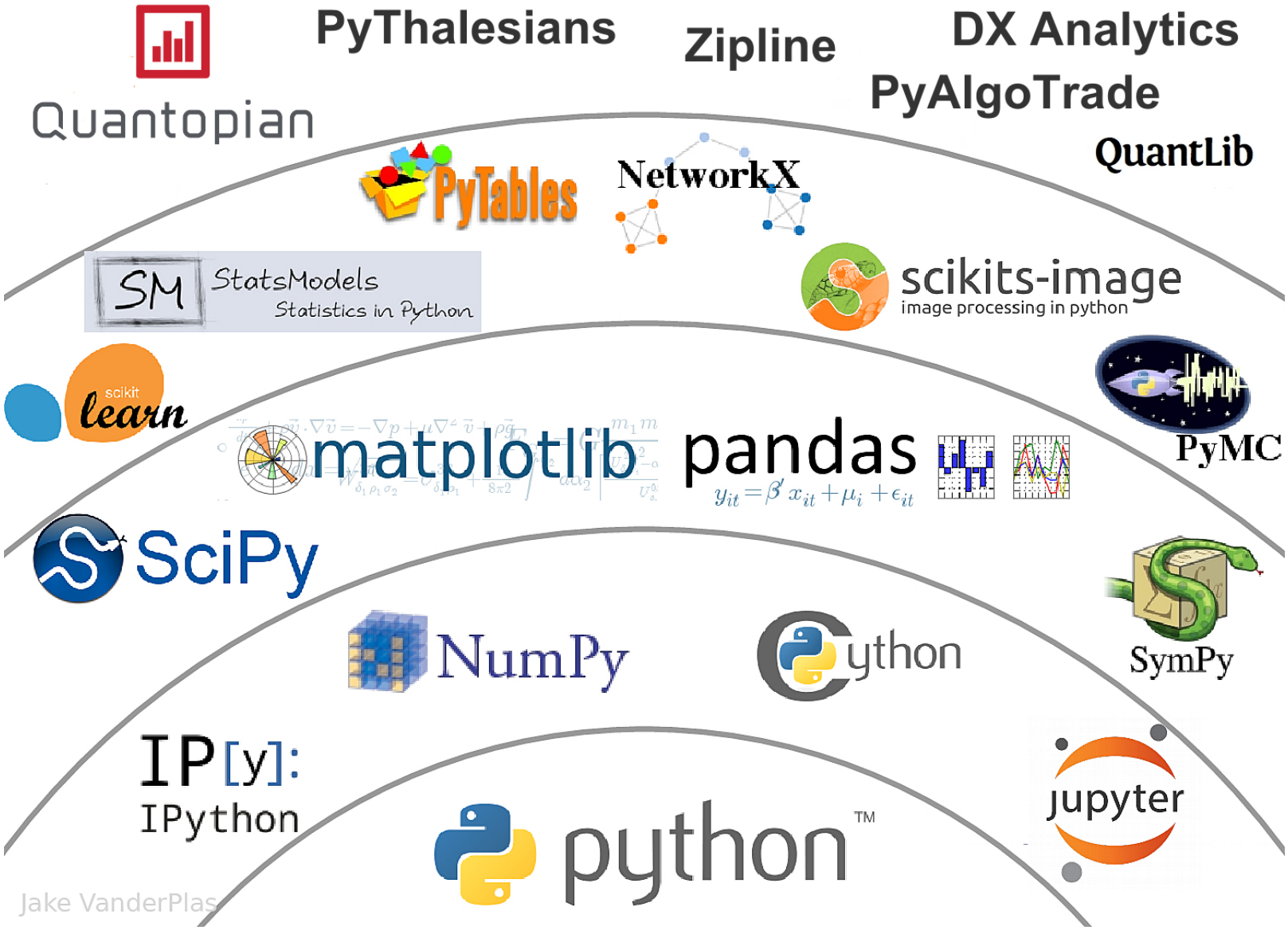
6 FinTech: Market Provisioning Smarter, Faster Machines New Market Platforms



圖表來源：Fugle團隊整理

Source: https://www.stockfeel.com.tw/2015年世界經濟論壇－未來的金融服務/

The Quant Finance PyData Stack



Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5

Zipline

a Pythonic

Algorithmic Trading Library

http://www.zipline.io/

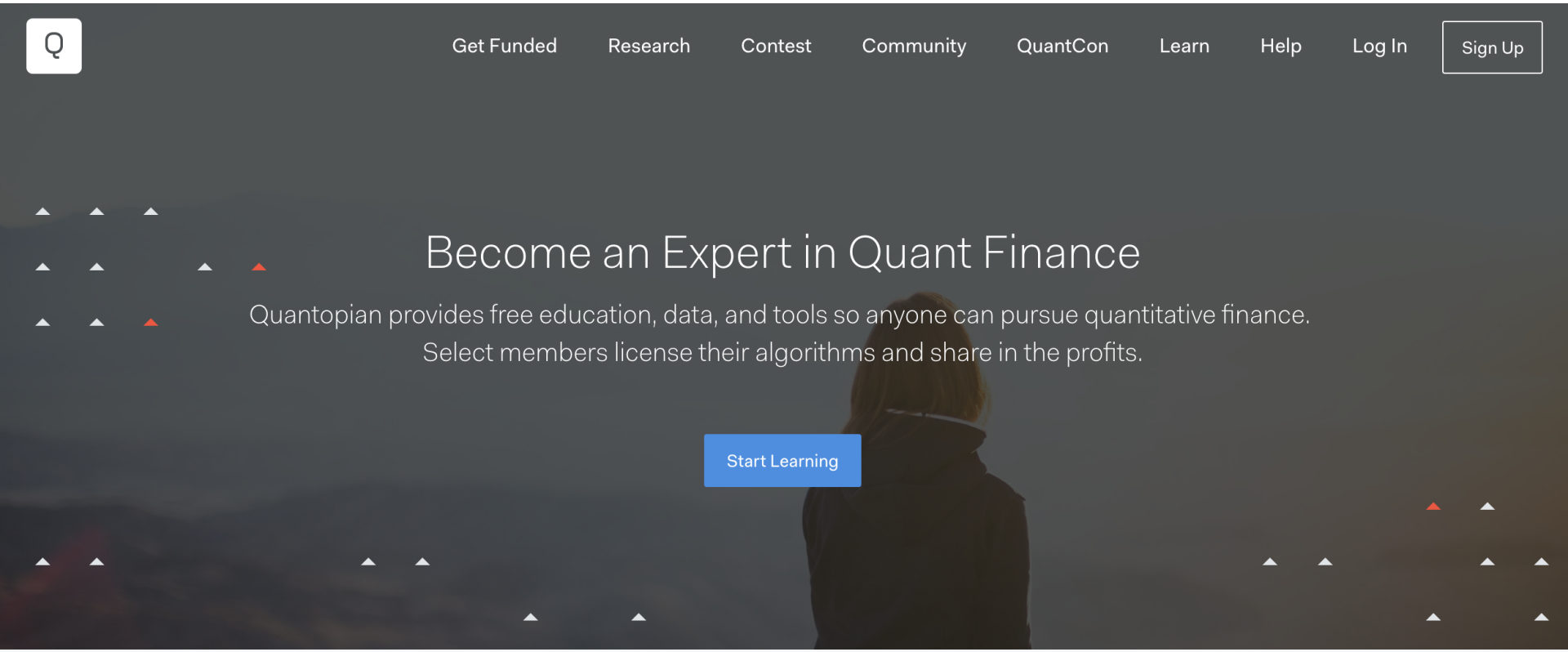
Source: https://github.com/quantopian/zipline

Zipline

- Zipline: Pythonic algorithmic trading library.
- Event-driven system
  - supports both backtesting and live-trading.
- Zipline is currently used in production as the backtesting and live-trading engine powering Quantopian
  - a free, community-centered, hosted platform for building and executing trading strategies.

Source: http://www.zipline.io/

Quantopian

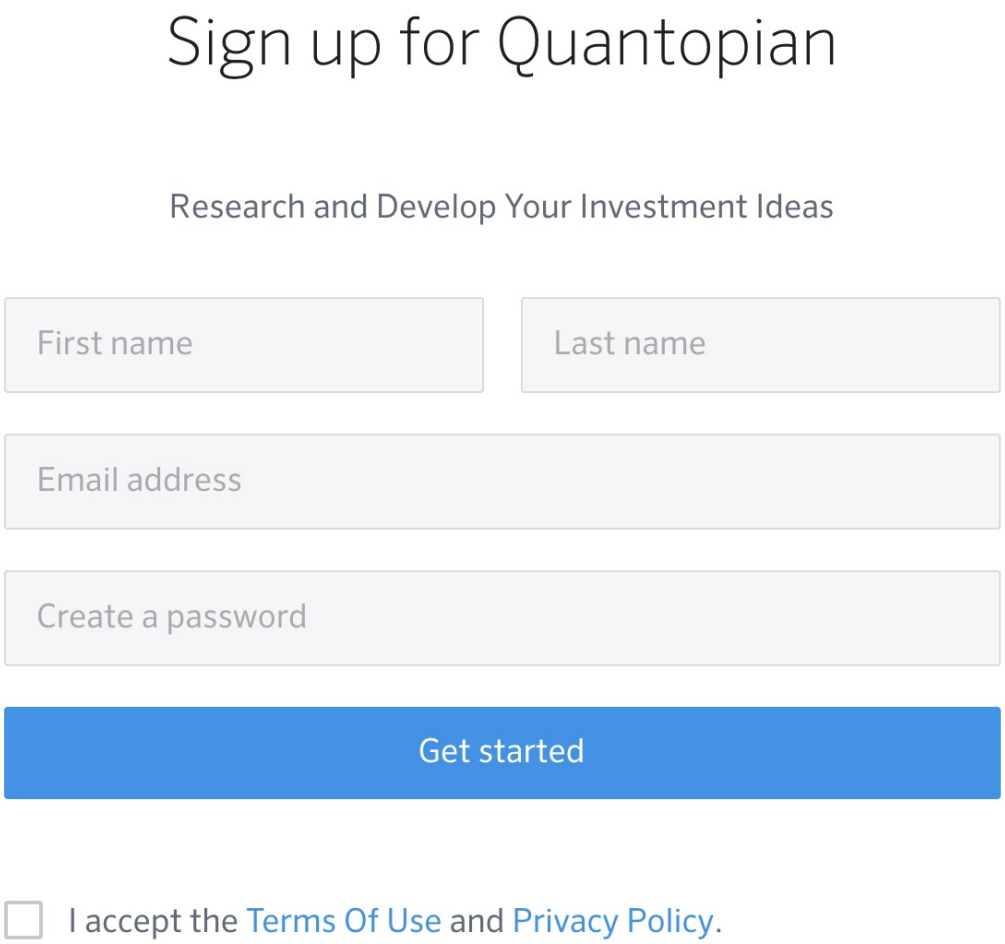


Community Achievements

All numbers are as of June 1, 2018

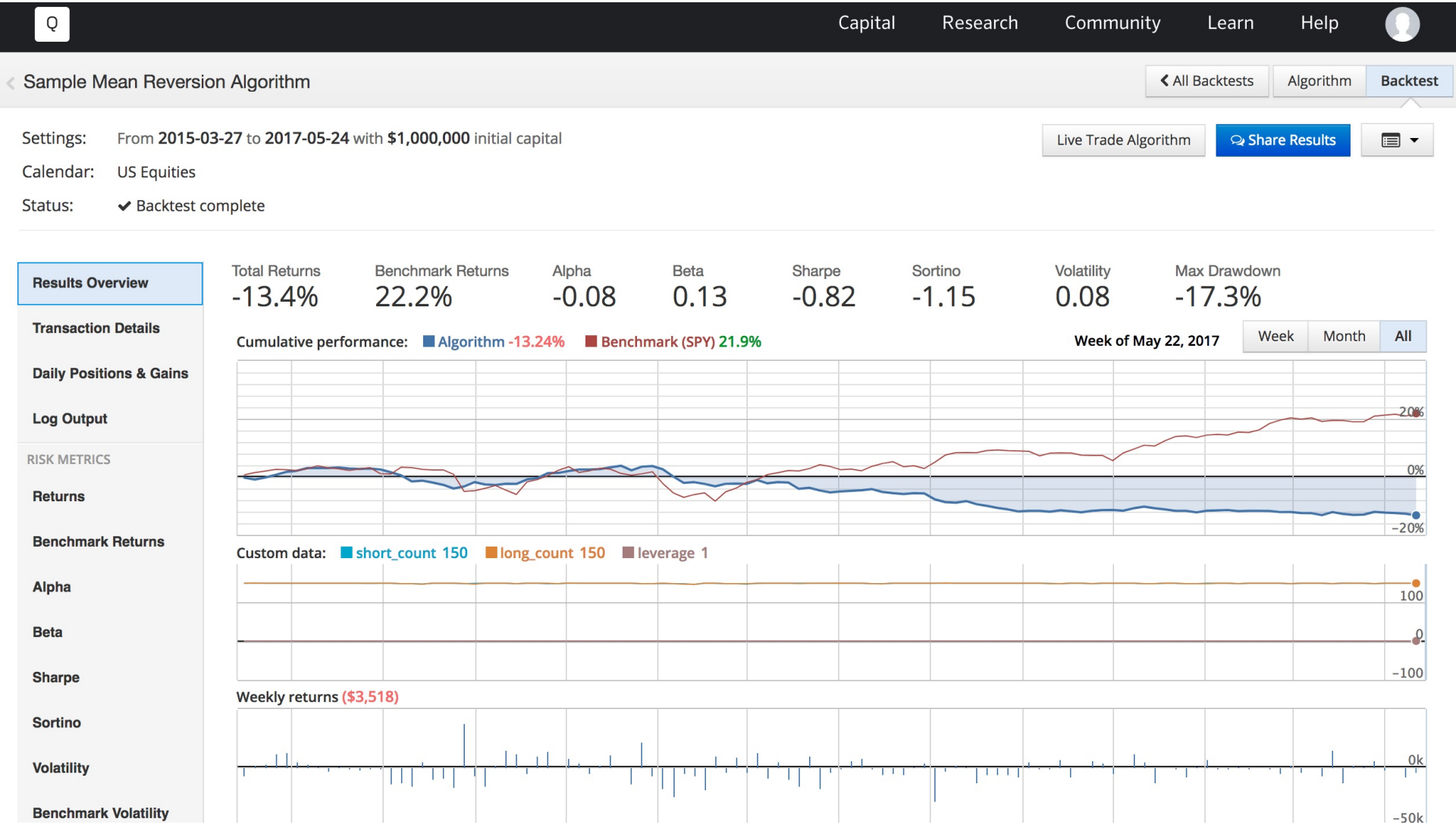
https://www.quantopian.com/

Sign up for Quantopian



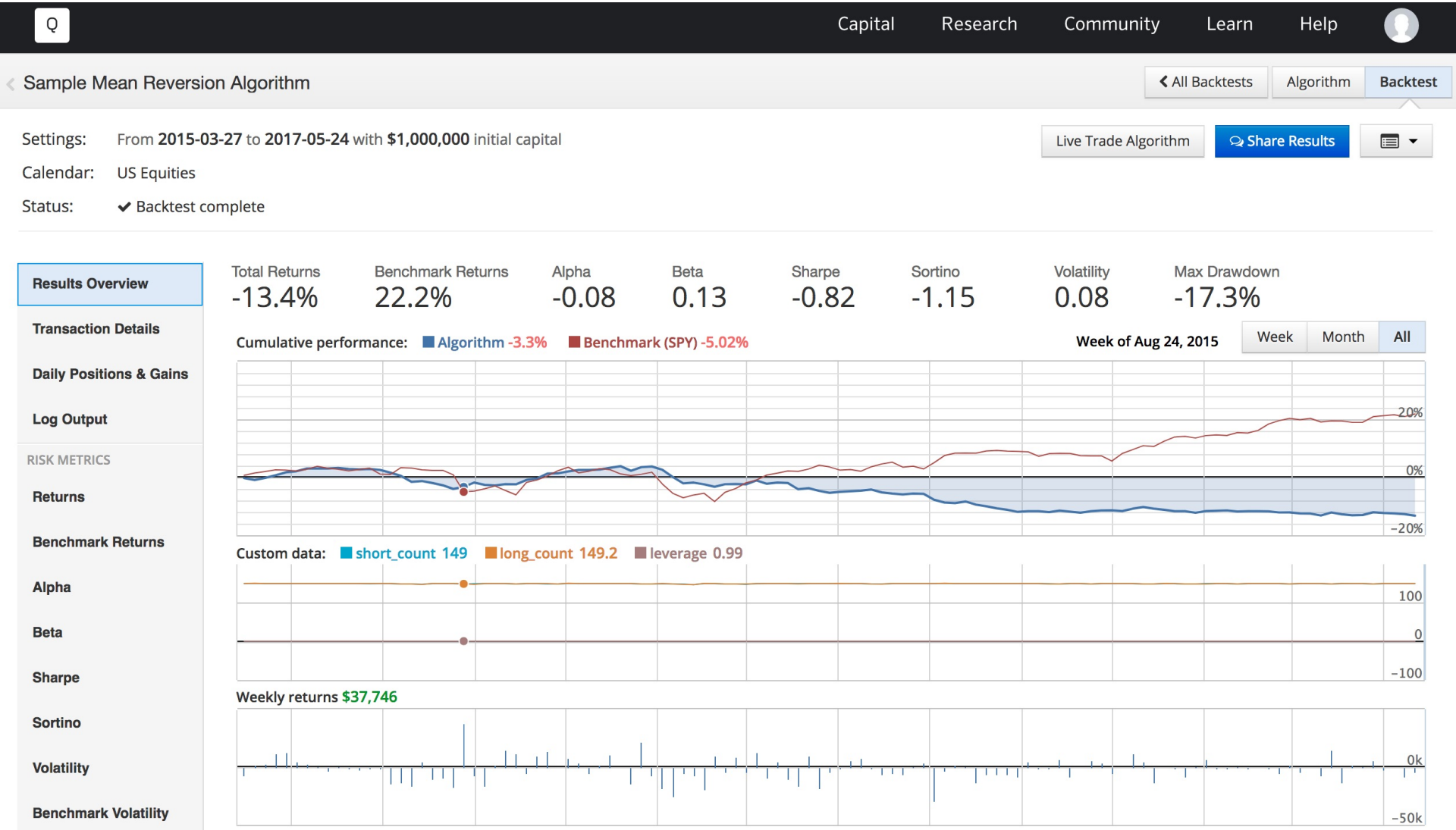
https://www.quantopian.com/users/sign_up

Quantopian
Sample Mean Reversion Algorithm



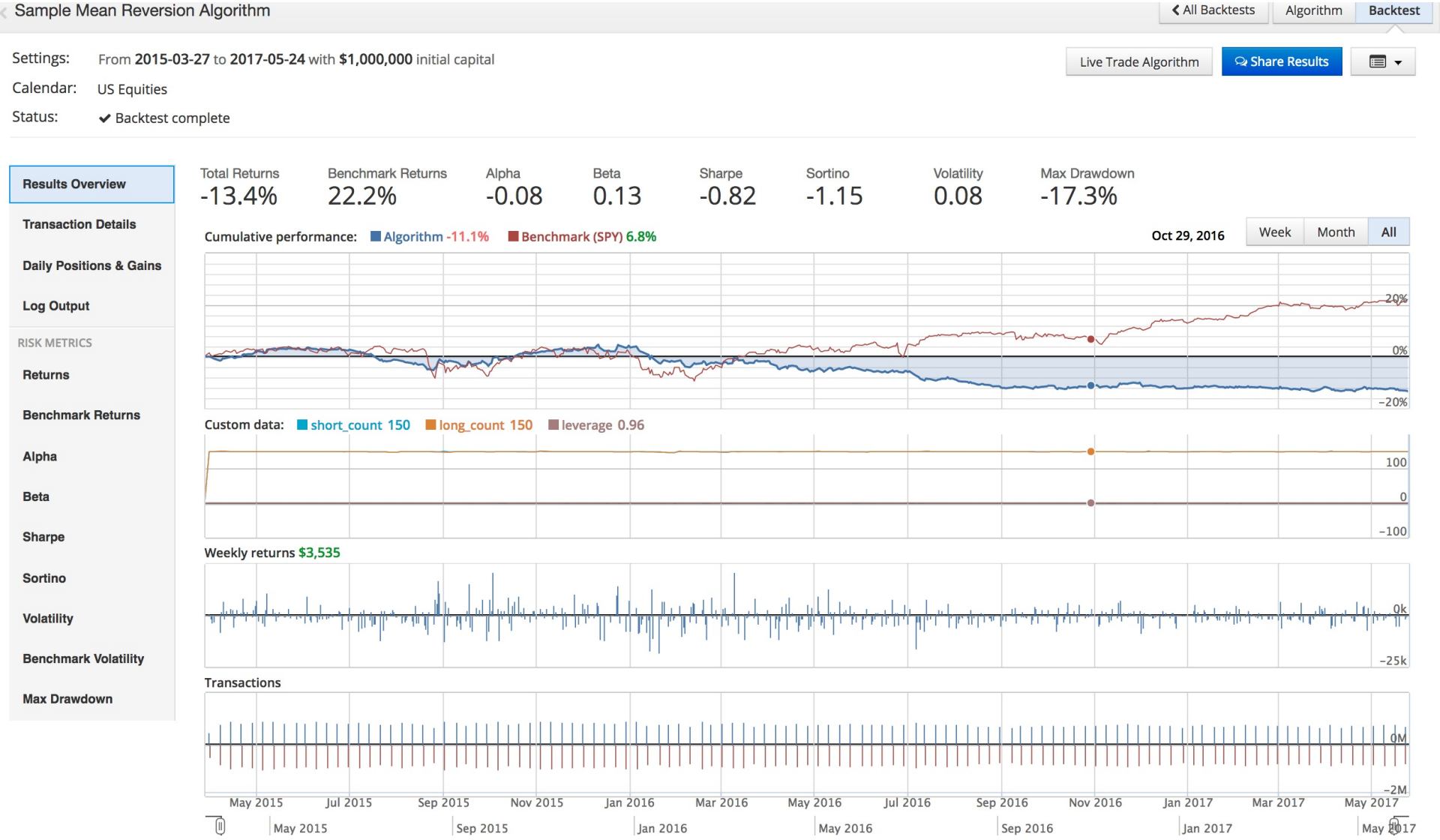
Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

Quantopian
Sample Mean Reversion Algorithm



Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

Quantopian
Sample Mean Reversion Algorithm



Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

Quantopian
Sample Mean Reversion Algorithm

Sample Mean Reversion Algorithm

All Backtests | Algorithm | Backtest

Settings: From **2015-03-27** to **2017-05-24** with **$1,000,000** initial capital

Calendar: US Equities

Status: ✔ Backtest complete

Live Trade Algorithm | Share Results

- Results Overview
- Transaction Details
- Daily Positions & Gains
- Log Output

RISK METRICS

- Returns
- Benchmark Returns
- Alpha
- Beta
- Sharpe
- Sortino
- Volatility
- Benchmark Volatility
- Max Drawdown

Transaction Details · Expand All · Collapse All

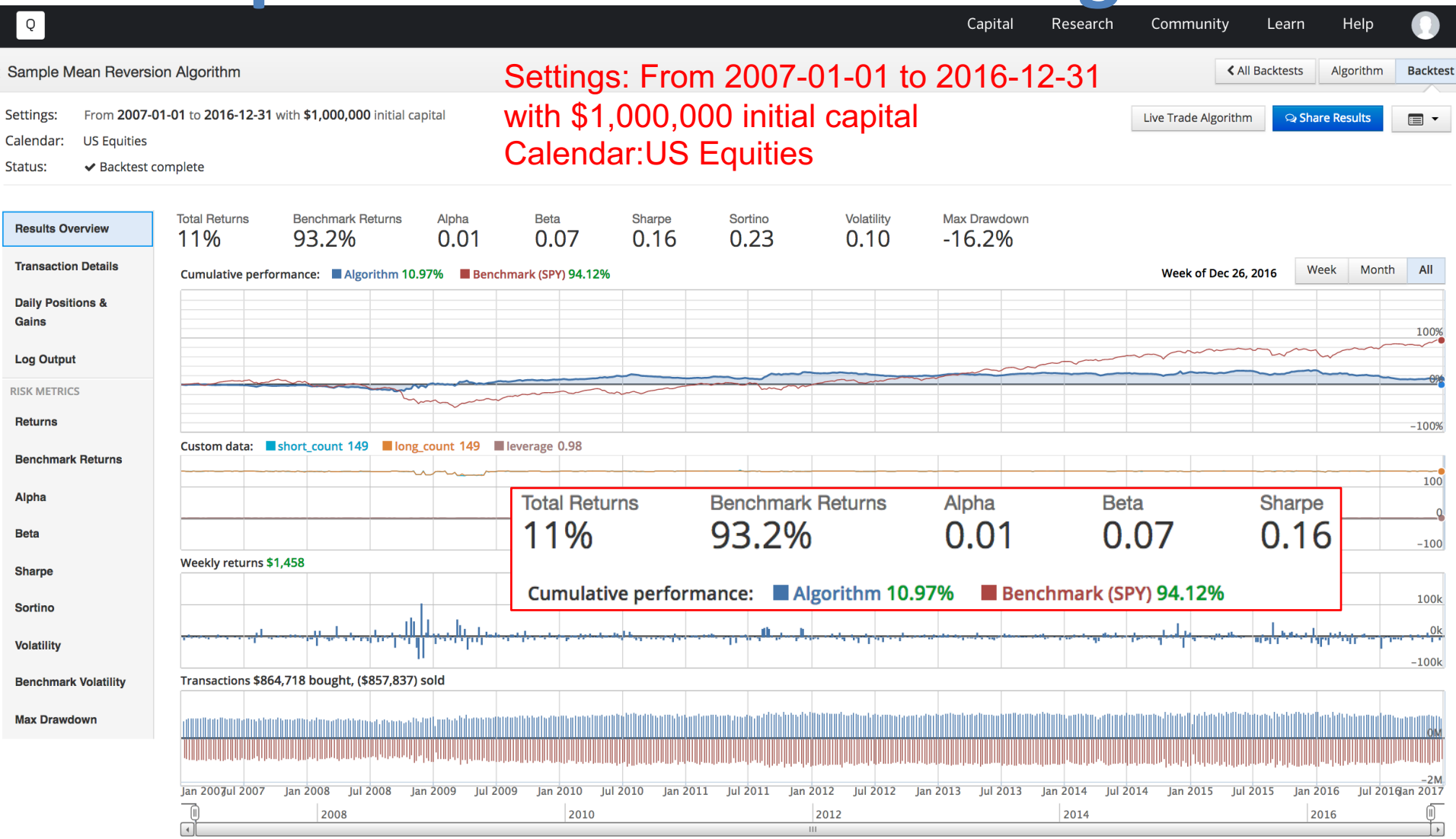
Group by day

Date	Asset	Transaction	Unit Price	Quantity	Position Value
2017-05-15 - 11:07 PM	PRAA	SELL	$37.32	-2	($74.65)
2017-05-15 - 11:07 PM	PRTA	SELL	$55.83	-31	($1,730.64)
2017-05-15 - 11:07 PM	PSTG	BUY	$11.68	44	$513.96
2017-05-15 - 11:07 PM	PTCT	SELL	$13.31	-10	($133.09)
2017-05-15 - 11:07 PM	QLYS	BUY	$43.50	10	$435.03
2017-05-15 - 11:07 PM	RGR	SELL	$64.07	-2	($128.14)
2017-05-15 - 11:07 PM	RRD	BUY	$13.30	62	$824.60
2017-05-15 - 11:07 PM	RXN	BUY	$23.45	9	$211.05
2017-05-15 - 11:07 PM	SUPN	SELL	$33.50	-12	($401.98)
2017-05-15 - 11:07 PM	TCO	SELL	$59.08	-7	($413.54)
2017-05-15 - 11:07 PM	TIVO	BUY	$17.15	2	$34.30
2017-05-15 - 11:07 PM	TLRD	BUY	$12.11	52	$629.77
2017-05-15 - 11:07 PM	TPC	SELL	$28.20	-5	($140.99)
2017-05-15 - 11:07 PM	TROX	SELL	$19.21	-60	($1,152.54)
2017-05-15 - 11:07 PM	TWNK	BUY	$15.69	17	$266.75

Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/592756885310d86a415cb0e1#backtest

Quantopian

Sample Mean Reversion Algorithm



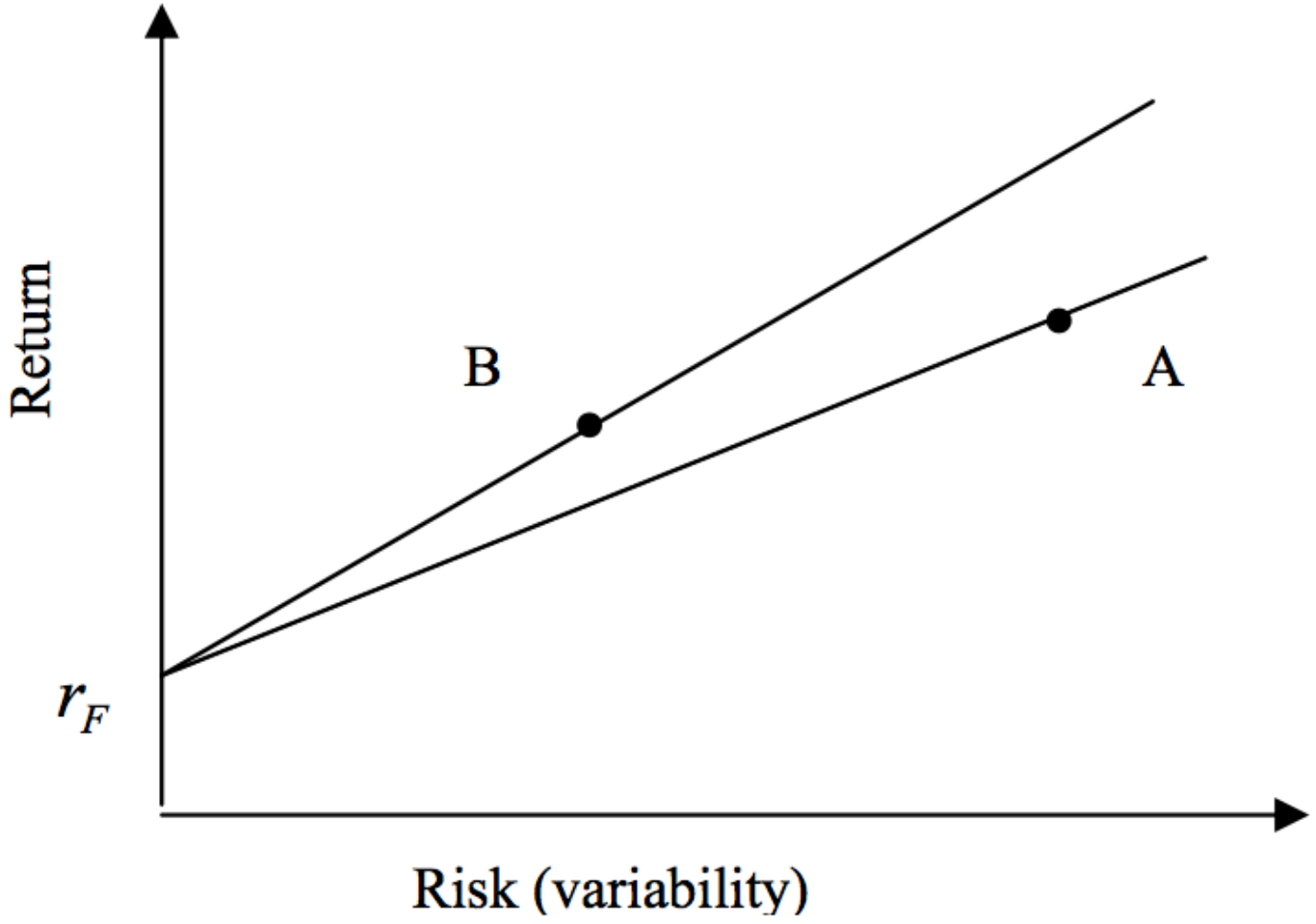
Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/592756885310d86a415cb0e1#backtest

Quantopian
Sample Mean Reversion Algorithm

Total Returns	Benchmark Returns	Alpha	Beta	Sharpe
11%	93.2%	0.01	0.07	0.16

Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/592756885310d86a415cb0e1#backtest

Risk and Return



Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sharpe Ratio

$$\textbf{Sharpe Ratio} = \frac{Portofolio\ Return - Risk\ Free\ Return}{Portofolio\ Risk}$$

Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sharpe Ratio

$$\textbf{Sharpe Ratio } SR = \frac{r_P - r_F}{\sigma_P}$$

Where

r_P = portfolio return

r_F = risk free rate

σ_P = portfolio risk (variability, standard deviation of return)

Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sortino Ratio

$$\textbf{Sortino Ratio} = \frac{r_P - r_T}{\sigma_D}$$

Where

r_P = portfolio return

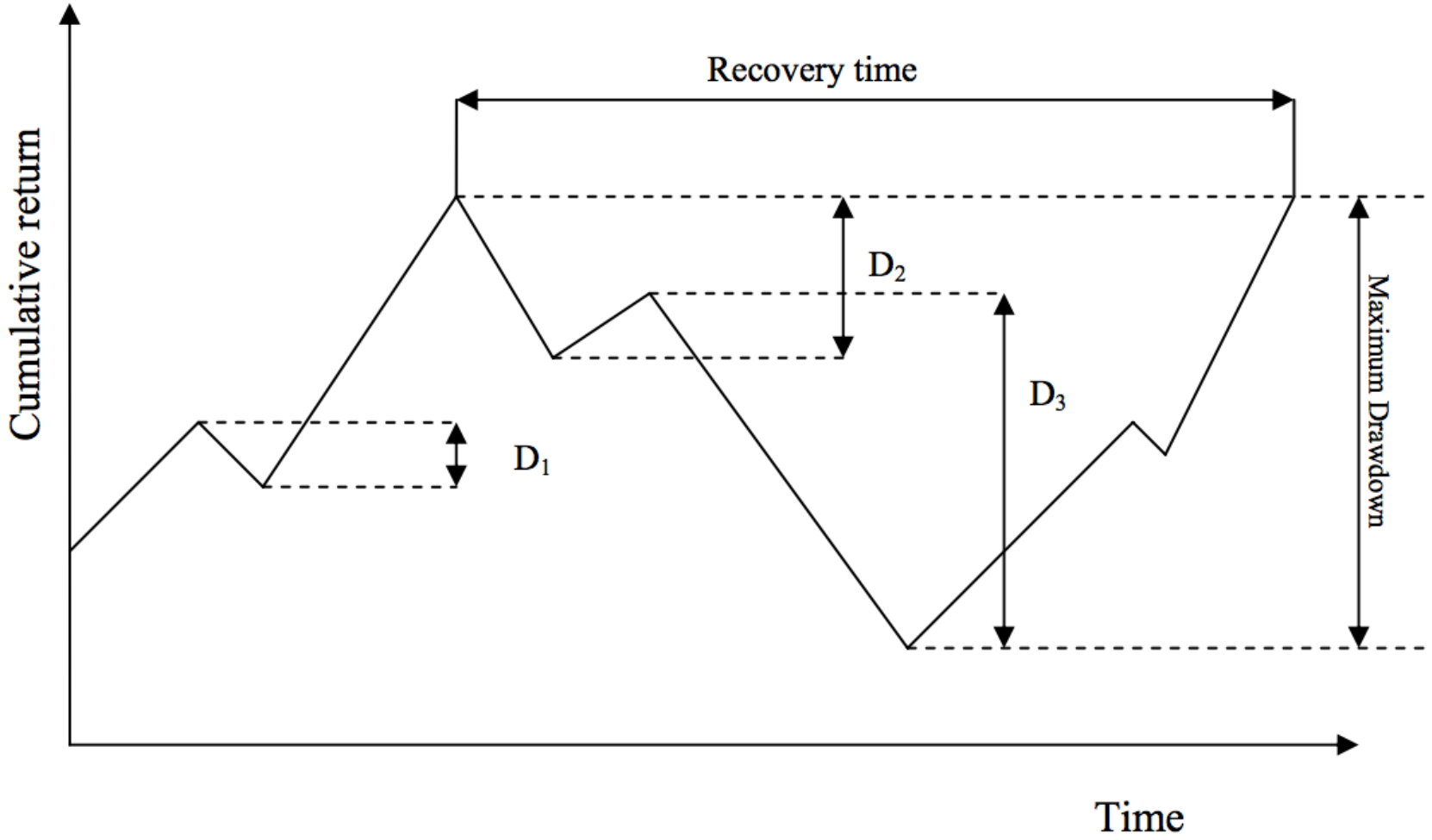
r_T = Minimum Target Return

σ_D = Downside Risk

$$\textbf{Downside Risk } \sigma_D = \sqrt{\sum_{i=1}^{n} \frac{\min[(r_i - rT), 0]^2}{n}}$$

Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Max Drawdown



Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Quantopian

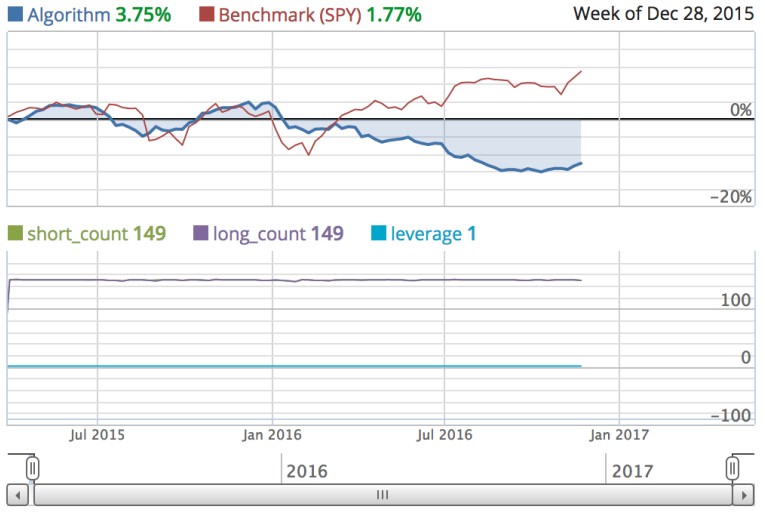
Sample Mean Reversion Algorithm

Save | Build Algorithm | Enter Contest | Collaborate | API Reference | Exit Fullscreen

```
"""
This is a sample mean-reversion algorithm on Quantopian for you to test and adapt.
This example uses a dynamic stock selector, pipeline, to select stocks to trade.
It orders stocks from the top 1% of the previous day's dollar-volume (liquid
stocks).

Algorithm investment thesis:
Top-performing stocks from last week will do worse this week, and vice-versa.

Every Monday, we rank high dollar-volume stocks based on their previous 5 day returns.
We long the bottom 10% of stocks with the WORST returns over the past 5 days.
We short the top 10% of stocks with the BEST returns over the past 5 days.

This type of algorithm may be used in live trading and in the Quantopian Open.
"""

# Import the libraries we will use here.
from quantopian.algorithm import attach_pipeline, pipeline_output
from quantopian.pipeline import Pipeline
from quantopian.pipeline.data.builtin import USEquityPricing
from quantopian.pipeline.factors import Returns
from quantopian.pipeline.filters.morningstar import Q1500US


def initialize(context):
    """
    Called once at the start of the program. Any one-time
    startup logic goes here.
    """
    # Define context variables that can be accessed in other methods of
    # the algorithm.
    context.long_leverage = 0.5
    context.short_leverage = -0.5
    context.returns_lookback = 5

    # Rebalance on the first trading day of each week at 11AM.
    schedule_function(rebalance,
                      date_rules.week_start(days_offset=0),
                      time_rules.market_open(hours=1, minutes=30))

    # Record tracking variables at the end of each day.
    schedule_function(record_vars,
                      date_rules.every_day()
```

Backtesting 77.6% Cancel

RETURNS	ALPHA	BETA	SHARPE	DRAWDOWN
-10%	-0.07	0.12	-0.72	-16.4%



Logs | Runtime Errors | More

```
2015-03-30 23:00 rebalance:146 INFO This week's longs: ARCB, ACXM, ARW, AVT, AXP,
BBY, CAMP, CDE, CMCS_A, CY, ATGE, S, HL, HMSY, HPQ, IDCC, BIIB, IDTI, IMGN, IONS,
JCP, KLAC, LRCX, MAT, MBI, MGM, MTG, REGN, RGLD, SMTC, SONC, TDS, TDW, TER, TRMB,
WDC, WERN, WOR, YRCW, ZBRA, CREE, RIG, GMCR, HST, BCRX, KNX, SSYS, FOXA, VECO, MYGN,
SNDK, NVAX, SCCO, ANDE, HUBG, CRR, NBIX, GWR, OCN, CIEN, TIVO, POWI, WAC, BRCM, AMKR,
NVDA, BRCD, LPNT, JNPR, VIAV, FNSR, ON, SGMO, CYH, ARNA, KERX, MDCO, ARRY, UTIW,
EXAS, SGMS, BTU, JOY, GME, LCI, BDSI, STX, CNX, APOL, ARRS, XPER, WLL, NRG, AGO,
ACAD, GNW, ALNY, DRH, HPY, IHS, CROX, ZIOP, SFLY, LDOS, ACHN, GSAT, AFSI, TWC, SMCI,
CZZ, DFT, CATM, SEM, PEB, PPC, FNGN, TSLA, SWFT, ZG, ZLTQ, GEVA, IMPV, HLSS, NSM,
PBYI, FRGI, WAGE, BLMN, RGLS, FLTX, ENTA, CSTM, PTCT, GOGO, AGIO, COMM, NMBL, RARE,
TRUE, ANET, TMST, ECR, KITE, MIK, SYF, MBLY, KEYS, VA, LC, QRVO
```

Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

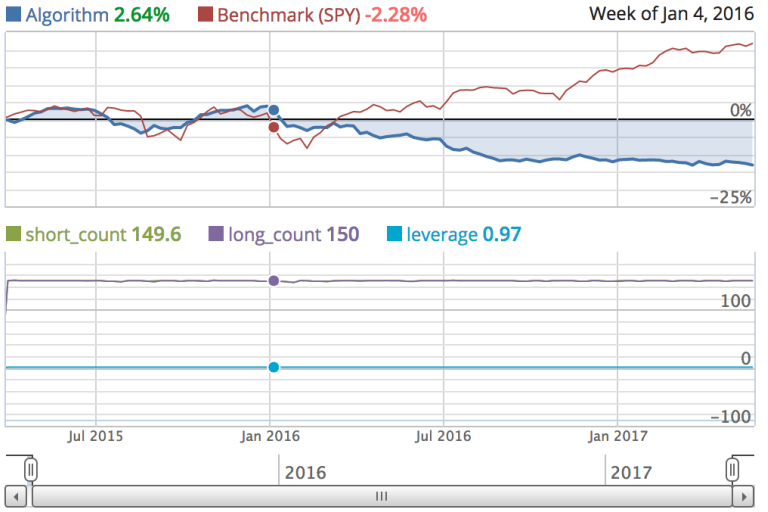
Quantopian
Sample Mean Reversion Algorithm

Save | Build Algorithm | Enter Contest | Collaborate | API Reference | Exit Fullscreen

```
"""
This is a sample mean-reversion algorithm on Quantopian for you to test and adapt.
This example uses a dynamic stock selector, pipeline, to select stocks to trade.
It orders stocks from the top 1% of the previous day's dollar-volume (liquid
stocks).

Algorithm investment thesis:
Top-performing stocks from last week will do worse this week, and vice-versa.

Every Monday, we rank high dollar-volume stocks based on their previous 5 day returns.
We long the bottom 10% of stocks with the WORST returns over the past 5 days.
We short the top 10% of stocks with the BEST returns over the past 5 days.

This type of algorithm may be used in live trading and in the Quantopian Open.
"""

# Import the libraries we will use here.
from quantopian.algorithm import attach_pipeline, pipeline_output
from quantopian.pipeline import Pipeline
from quantopian.pipeline.data.builtin import USEquityPricing
from quantopian.pipeline.factors import Returns
from quantopian.pipeline.filters.morningstar import Q1500US


def initialize(context):
    """
    Called once at the start of the program. Any one-time
    startup logic goes here.
    """
    # Define context variables that can be accessed in other methods of
    # the algorithm.
    context.long_leverage = 0.5
    context.short_leverage = -0.5
    context.returns_lookback = 5

    # Rebalance on the first trading day of each week at 11AM.
    schedule_function(rebalance,
                      date_rules.week_start(days_offset=0),
                      time_rules.market_open(hours=1, minutes=30))

    # Record tracking variables at the end of each day.
    schedule_function(record_vars,
```

03/27/2015 to 05/23/2017 | $ 1000000 | US Equities | Run Full Backtest

RETURNS	ALPHA	BETA	SHARPE	DRAWDOWN
-13.2%	-0.08	0.13	-0.81	-17.3%



Logs | Runtime Errors | More

```
2015-03-30 23:00 rebalance:146 INFO This week's longs: ARCB, ACXM, ARW, AVT, AXP,
BBY, CAMP, CDE, CMCS_A, CY, ATGE, S, HL, HMSY, HPQ, IDCC, BIIB, IDTI, IMGN, IONS,
JCP, KLAC, LRCX, MAT, MBI, MGM, MTG, REGN, RGLD, SMTC, SONC, TDS, TDW, TER, TRMB,
WDC, WERN, WOR, YRCW, ZBRA, CREE, RIG, GMCR, HST, BCRX, KNX, SSYS, FOXA, VECO, MYGN,
SNDK, NVAX, SCCO, ANDE, HUBG, CRR, NBIX, GWR, OCN, CIEN, TIVO, POWI, WAC, BRCM, AMKR,
NVDA, BRCD, LPNT, JNPR, VIAV, FNSR, ON, SGMO, CYH, ARNA, KERX, MDCO, ARRY, UTIW,
EXAS, SGMS, BTU, JOY, GME, LCI, BDSI, STX, CNX, APOL, ARRS, XPER, WLL, NRG, AGO,
ACAD, GNW, ALNY, DRH, HPY, IHS, CROX, ZIOP, SFLY, LDOS, ACHN, GSAT, AFSI, TWC, SMCI,
CZZ, DFT, CATM, SEM, PEB, PPC, FNGN, TSLA, SWFT, ZG, ZLTQ, GEVA, IMPV, HLSS, NSM,
PBYI, FRGI, WAGE, BLMN, RGLS, FLTX, ENTA, CSTM, PTCT, GOGO, AGIO, COMM, NMBL, RARE,
TRUE, ANET, TMST, ECR, KITE, MIK, SYF, MBLY, KEYS, VA, LC, QRVO
```

Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

Writing and Backtesting an Algorithm on Quantopian

Source: https://www.quantopian.com/tutorials/getting-started

What is a Trading Algorithm?

On Quantopian,
a trading algorithm
is a Python program
that defines two special functions:
initialize() and handle_data()

Source: https://www.quantopian.com/tutorials/getting-started

An example of an algorithm that allocates 100% of its portfolio in AAPL

```
def initialize(context):
    # Reference to AAPL
    context.aapl = sid(24)

def handle_data(context, data):
    # Position 100% of our portfolio to be long in AAPL
    order_target_percent(context.aapl, 1.00)
```

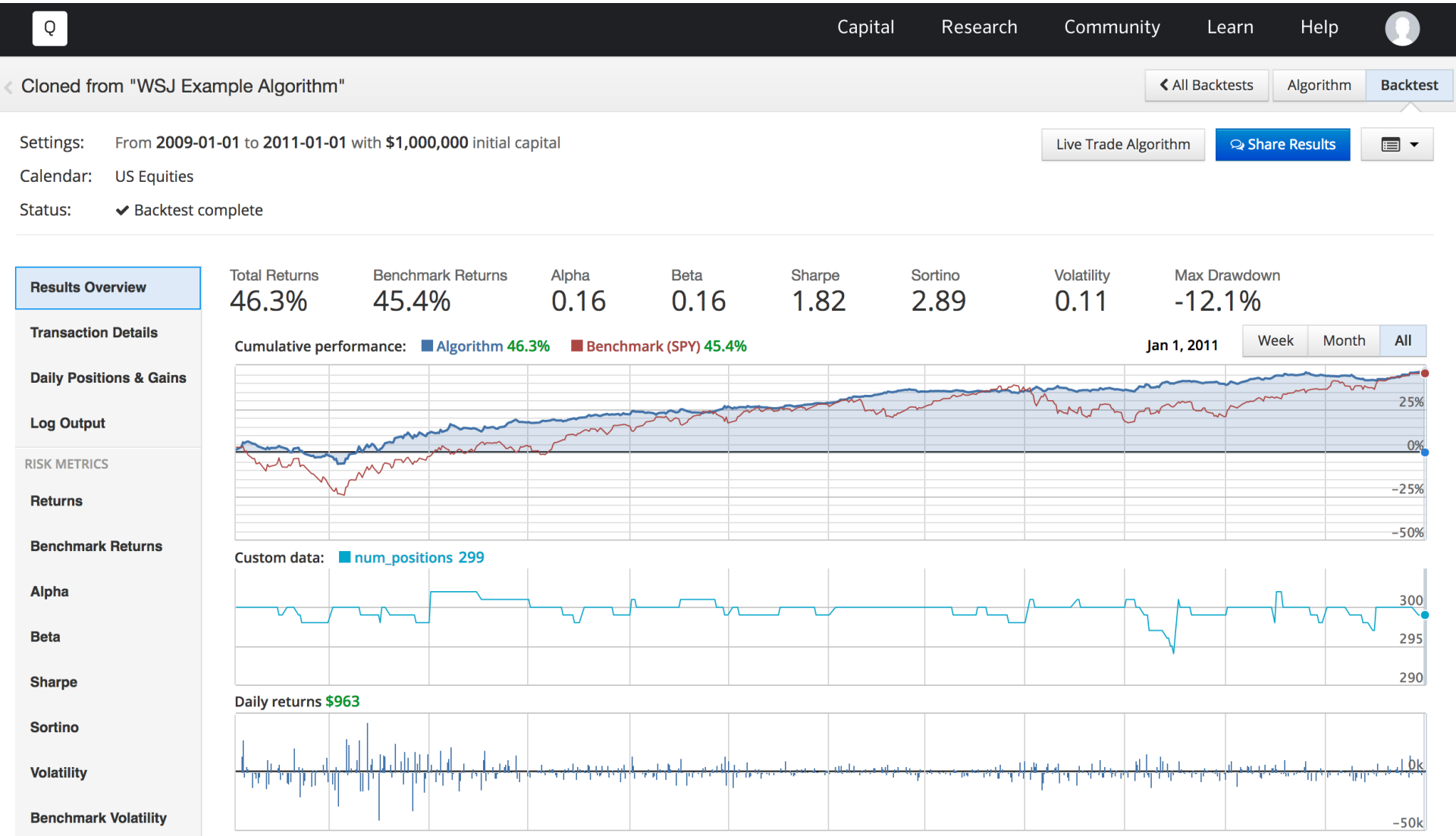
Source: https://www.quantopian.com/tutorials/getting-started

Moving Average

```
def initialize(context):
    context.security = symbol('AAPL')
    schedule_function(myfunc, date_rules.every_day(),  time_rules.market_open(minutes = 15))

def handle_data(context, data):
    MovingAvg1 = data[context.security].mavg(20)
    MovingAvg2 = data[context.security].mavg(60)

    current_positions = context.portfolio.positions[symbol('AAPL')].amount

    if (MovingAvg1 > MovingAvg2) and current_positions == 0:
        order_target_percent(context.security, 0.25)

    elif (MovingAvg1 < MovingAvg2) and current_positions != 0:
        order_target(context.security, 0)
```

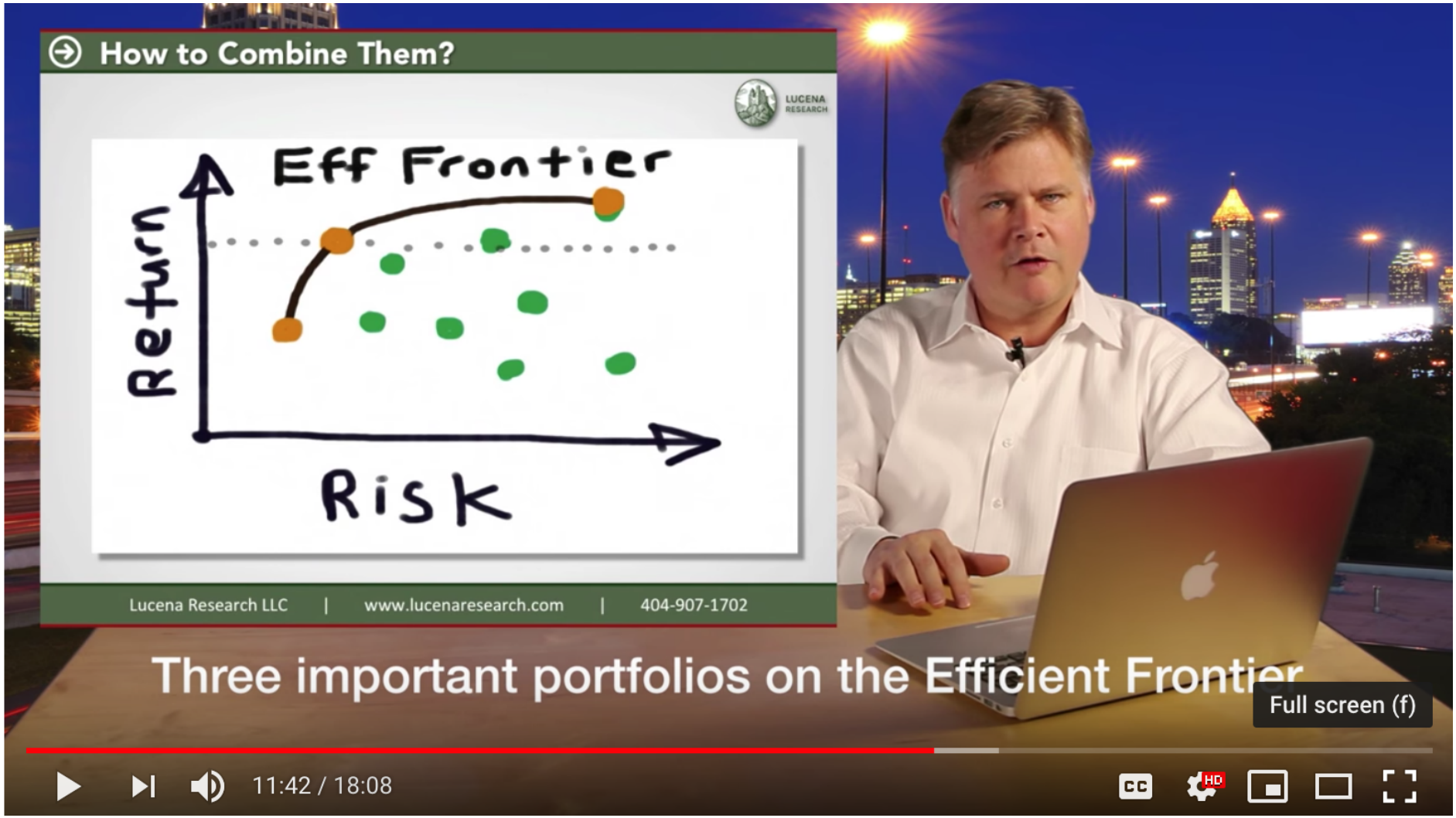
Source: https://www.quantopian.com/tutorials/getting-started

Quantopian
WSJ Example Algorithm



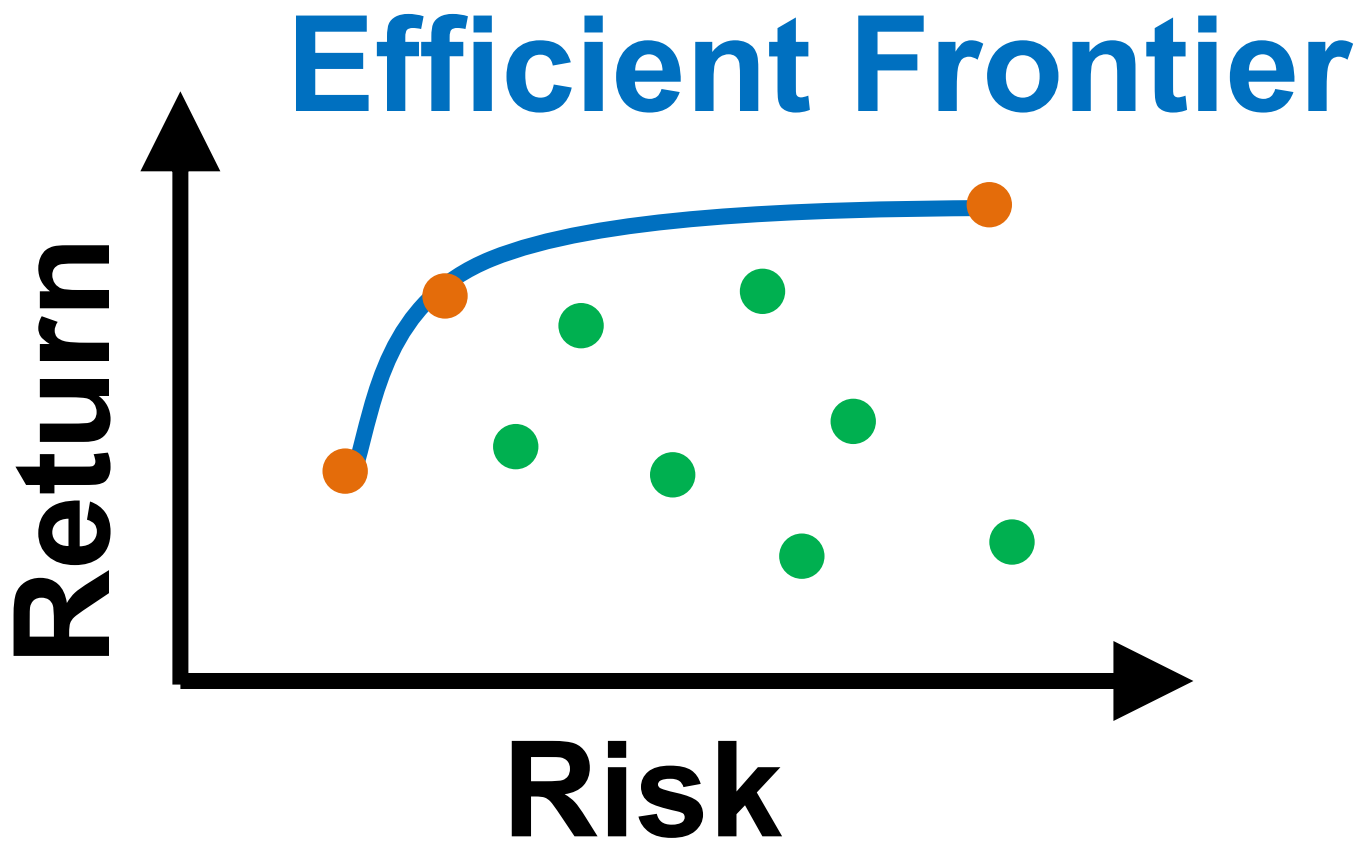
Source: https://www.quantopian.com/algorithms/59274d707875d1000d41937d/5927506f3cf7da6fec5951b6#backtest

Investment Science: Portfolio Optimization



Source: Tucker Balch (2012), Investment Science: Portfolio Optimization, https://www.youtube.com/watch?v=5qbMhXXq0vI

Portfolio Optimization
Efficient Frontier



Source: Tucker Balch (2012), Investment Science: Portfolio Optimization, https://www.youtube.com/watch?v=5qbMhXXq0vI

Portfolio Optimization and Algorithmic Trading

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

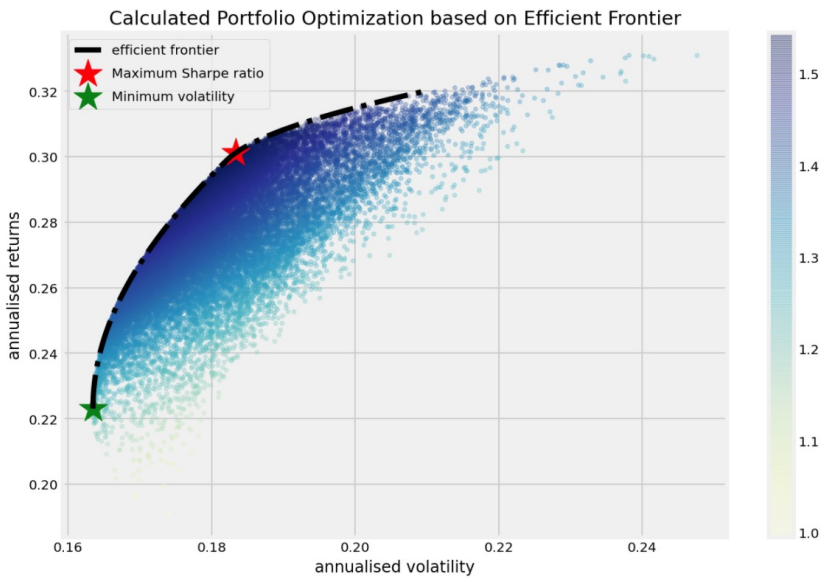
Comment Share A

Table of contents

+ Code + Text

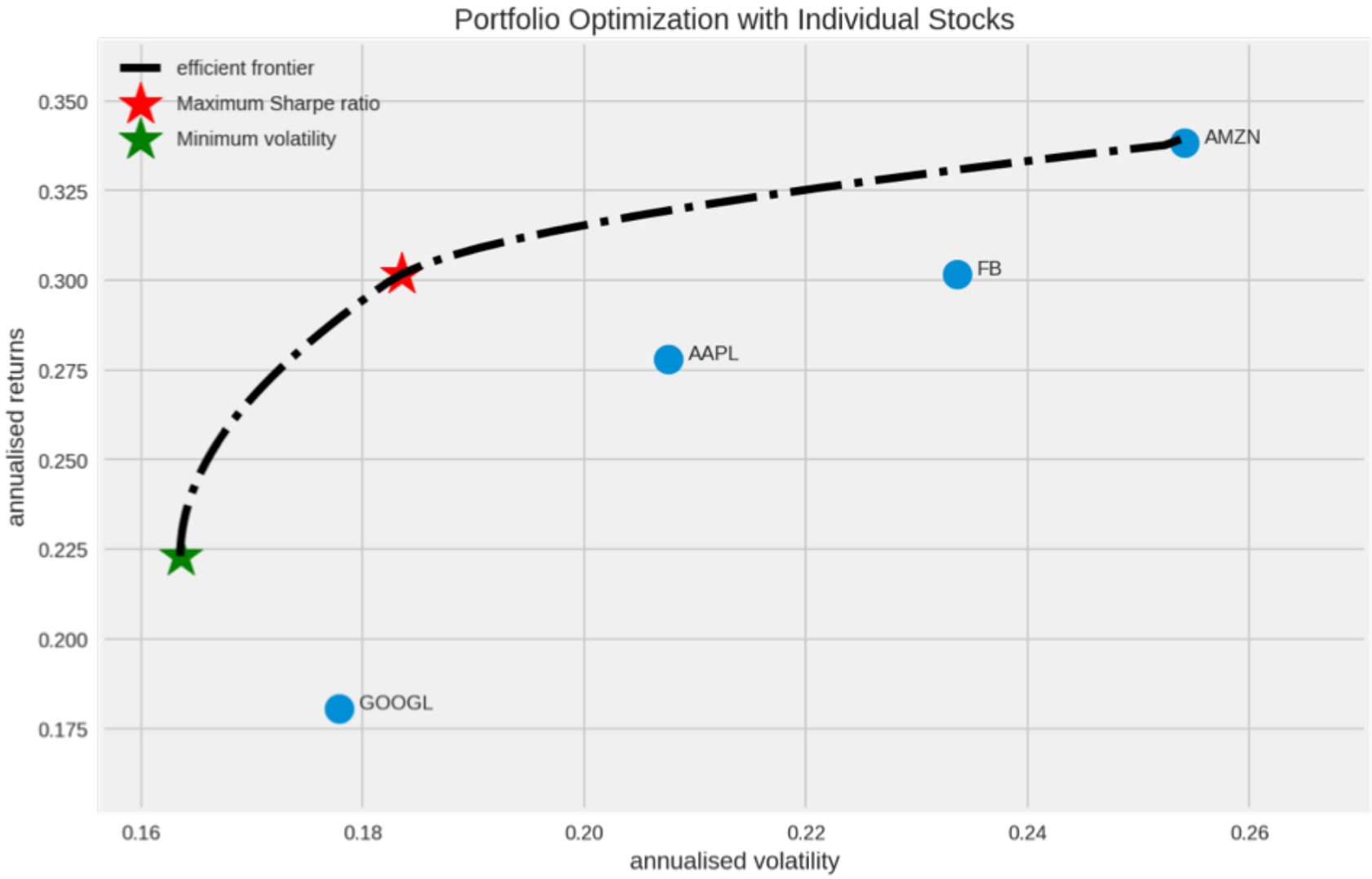
RAM Disk Editing

```
Annualised Volatility: 0.18


              AAPL    AMZN     FB  GOOGL
allocation   44.67   29.05  26.28    0.0
--------------------------------------------------------------------------------
Minimum Volatility Portfolio Allocation

Annualised Return: 0.22
Annualised Volatility: 0.16


              AAPL   AMZN    FB  GOOGL
allocation   34.02   0.73  6.98  58.26
```



https://tinyurl.com/imtkupython101

Portfolio Optimization Efficient Frontier



UQER

Source: https://uqer.io/home/

UQER

海量金融大数据

高质量的海量金融数据支撑，轻松实现大数据时代的交易策略

云端平台，高效研究，极速回测

稳定、安全、高可扩展的云平台，零门槛获得华尔街专业级别量化研究装备

Source: https://uqer.io/home/

UQER

模拟交易，赢取基金管理权

一键实盘模拟 ，云端托管，更有机会赢取500万实盘资金管理收益

Source: https://uqer.io/home/

JoinQuant

JoinQuant 聚宽 | 了解企业版

首页 | 策略擂台 HOT | 投资研究 | 我的策略 | 我的交易 | 数据 | 帮助 | 量化课堂 | 社区 | 登录 | 注册

十行代码，玩转聚宽

JOINQUANT

与7万+宽客一起

玩转量化投资

免费试用

立即登录

策略广场

赢率季胜季 模拟实盘	基于SVM的机器学习策略 策略回测	稳增高爆组合 模拟实盘
鏖戰先覺者	走得很慢的海龟	阴吹思婷

Source: https://www.joinquant.com/

JoinQuant

策略广场

赢率季胜季
模拟实盘


鏖戰先覺者

■ 策略收益 ■ 基准收益

年化收益	最大回撤	初始资金
305.71%	13.31%	¥ 50000

已有540人订阅 [免费订阅]

基于SVM的机器学习策略
策略回测


走得很慢的海龟

■ 策略收益 ■ 基准收益

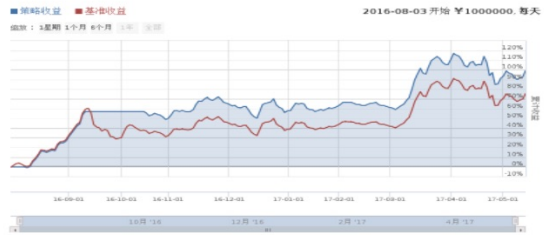
年化收益	最大回撤	初始资金
10.05%	20.49%	¥ 1000000

已有633人获取源码 [获取源码]

稳增高爆组合
模拟实盘


阴吹思婷

■ 策略收益 ■ 基准收益



年化收益	最大回撤	初始资金
142.52%	15.88%	¥ 1000000

已有585人订阅 [免费订阅]

银行日内
模拟实盘


囚徒之爱

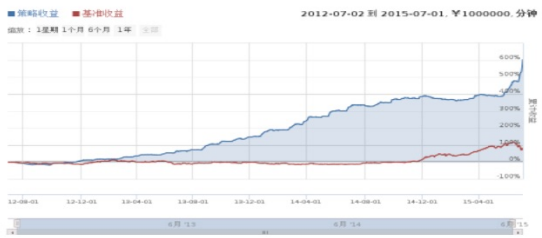
■ 策略收益 ■ 基准收益

年化收益	最大回撤	初始资金
78.64%	4.51%	¥ 30000

【量化课堂】股指期货跨期套...
策略回测


JoinQuant量化课堂

■ 策略收益 ■ 基准收益



年化收益	最大回撤	初始资金
56.11%	17.13%	¥ 1000000

分级A轮动策略
策略回测


囚徒

■ 策略收益 ■ 基准收益

年化收益	最大回撤	初始资金
18.58%	1.08%	¥ 300000

Source: https://www.joinquant.com/

RiceQuant

策略研究

历史回测

强大、易用的量化接口API，易于编写交易策略
免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据
极速、精准的回测体验，快速开发和验证投资策略

Source: https://www.ricequant.com/

RiceQuant

策略研究

免费提供IPython Notebook研究平台以及强大的金融、数学等工具库
免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据
灵活的文本编辑和绘图功能，提供无与伦比的交互式体验

Source: https://www.ricequant.com/

RiceQuant

历史回测

强大、易用的量化接口API，易于编写交易策略
免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据
极速、精准的回测体验，快速开发和验证投资策略

Source: https://www.ricequant.com/

RiceQuant

实时模拟交易

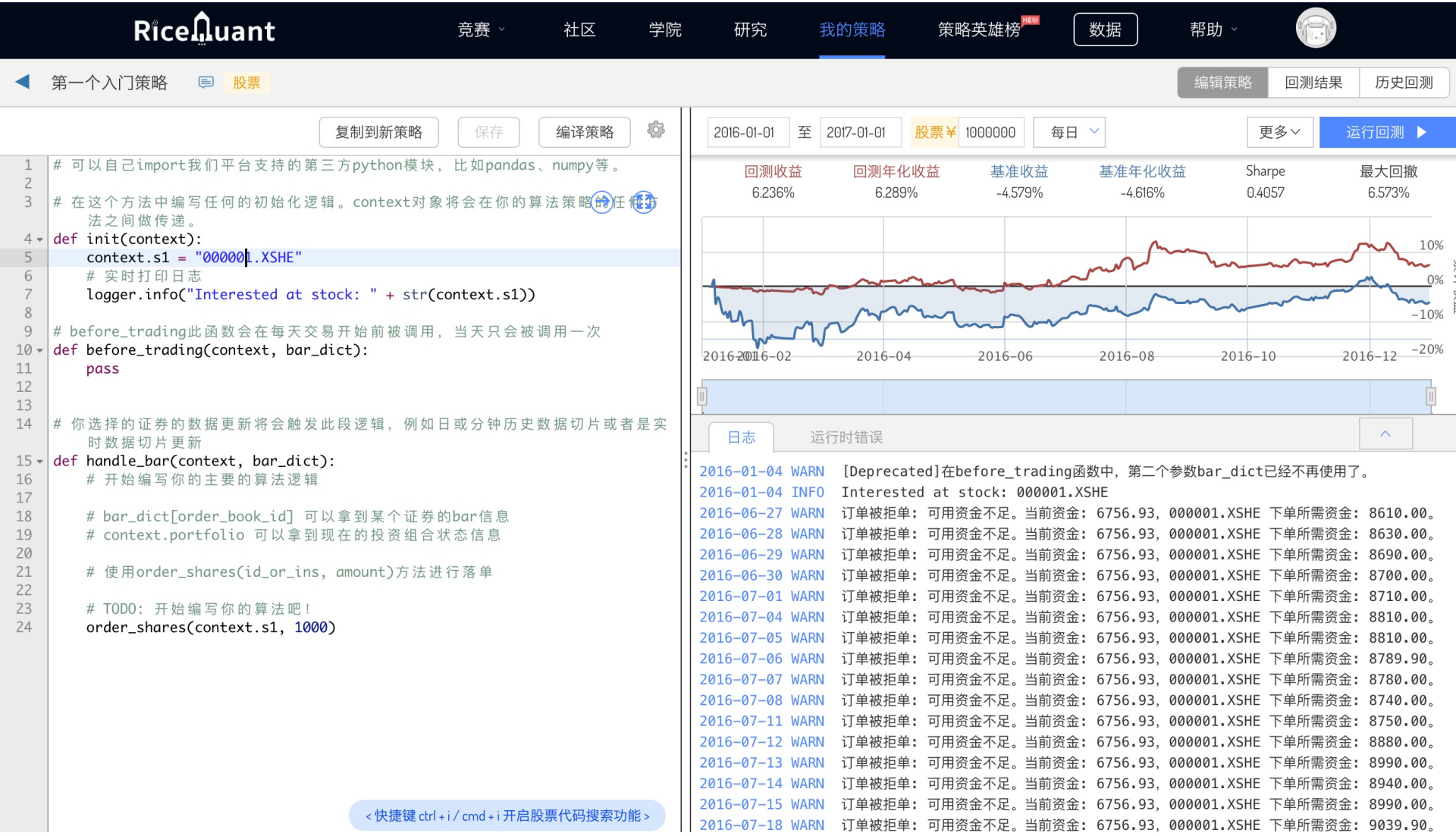
一键部署，云端永久运行

微秒级别实时数据推送计算

将会提供微信、邮件等交易信号推送

Source: https://www.ricequant.com/

RiceQuant



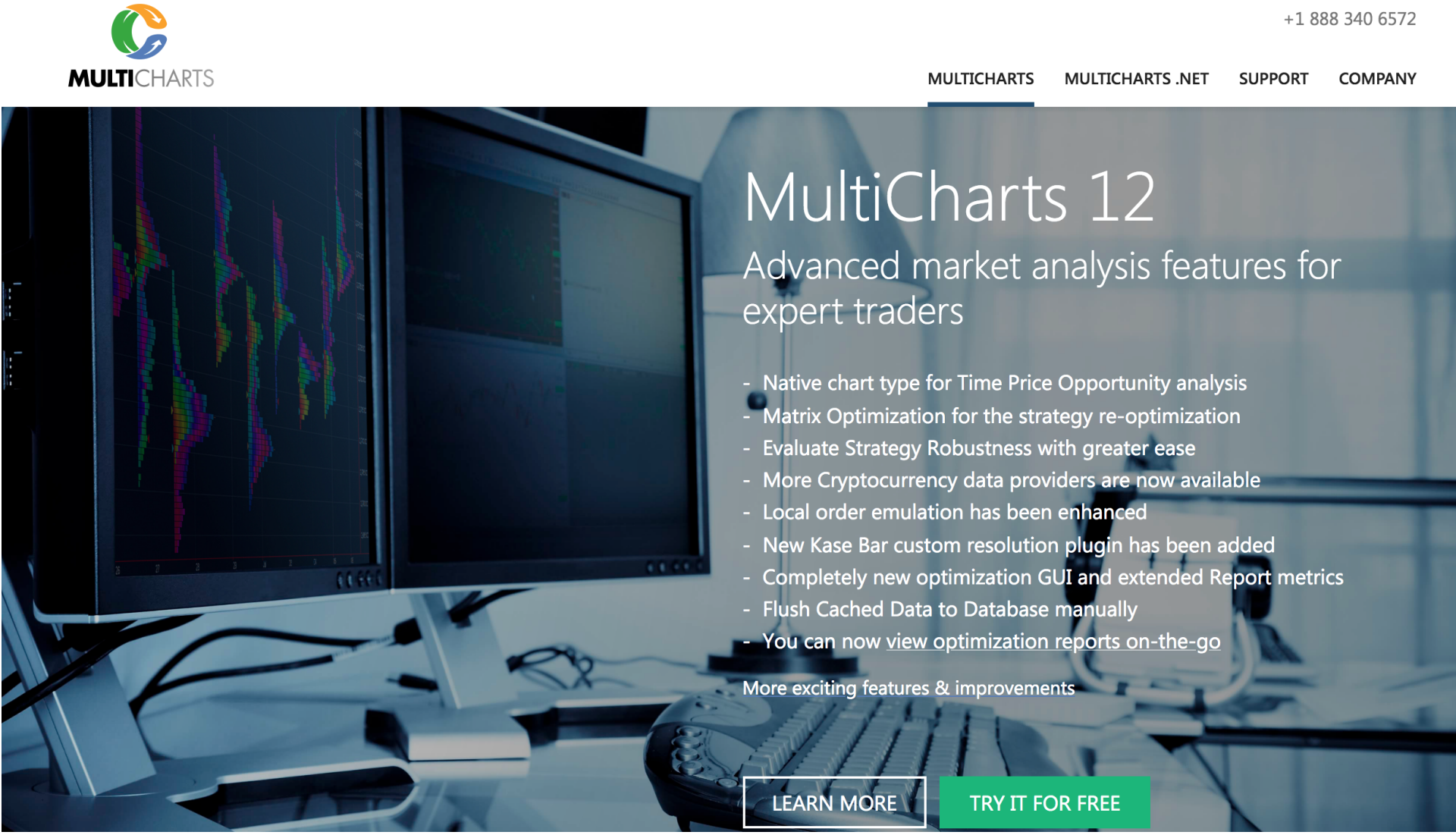

```
# 可以自己import我们平台支持的第三方python模块，比如pandas、numpy等。

# 在这个方法中编写任何的初始化逻辑。context对象将会在你的算法策略的任何方法之间做传递。
def init(context):
    context.s1 = "000001.XSHE"
    # 实时打印日志
    logger.info("Interested at stock: " + str(context.s1))

# before_trading此函数会在每天交易开始前被调用，当天只会被调用一次
def before_trading(context, bar_dict):
    pass


# 你选择的证券的数据更新将会触发此段逻辑，例如日或分钟历史数据切片或者是实时数据切片更新
def handle_bar(context, bar_dict):
    # 开始编写你的主要的算法逻辑

    # bar_dict[order_book_id] 可以拿到某个证券的bar信息
    # context.portfolio 可以拿到现在的投资组合状态信息

    # 使用order_shares(id_or_ins, amount)方法进行落单

    # TODO: 开始编写你的算法吧！
    order_shares(context.s1, 1000)
```

RiceQuant


```
# 在这个方法中编写任何的初始化逻辑。context对象将会在你的算法策略的任何方法之间做传递。
def init(context):
    #沪深300指数、中证500指数和国债指数
    context.stocks = ["000300.XSHG","000905.XSHG","000012.XSHG"]
# before_trading此函数会在每天交易开始前被调用，当天只会被调用一次
# 你选择的证券的数据更新将会触发此段逻辑，例如日或分钟历史数据切片或者是实时数据切片更新
def handle_bar(context, bar_dict):
    # 开始编写你的主要的算法逻辑
    hs300 = history_bars(context.stocks[0],20,"1d","close")
    zz500 = history_bars(context.stocks[1],20,"1d","close")
    hsIncrease = hs300[19] - hs300[0]
    zzIncrease = zz500[19] - zz500[0]
    p = context.portfolio.positions
    hsQuality = p[context.stocks[0]].quantity
    zzQuality = p[context.stocks[1]].quantity
    gzQuality = p[context.stocks[2]].quantity
    if hsIncrease < 0 and zzIncrease < 0:
        if hsQuality > 0:
            order_target_percent(context.stocks[0],0)
            logger.info("卖出沪深300")
        if zzQuality > 0:
            order_target_percent(context.stocks[1],0)
            logger.info("卖出中证500")
        if gzQuality <= 0.001:
            order_target_percent(context.stocks[2],1)
            logger.info("买入国债")
    elif hsIncrease < zzIncrease:
        if hsQuality > 0:
            order_target_percent(context.stocks[0],0)
            logger.info("卖出沪深300")
        if gzQuality > 0:
            order_target_percent(context.stocks[2],0)
            logger.info("卖出国债")
        if zzQuality <= 0.001:
```

Source: https://www.ricequant.com/algorithm/523092

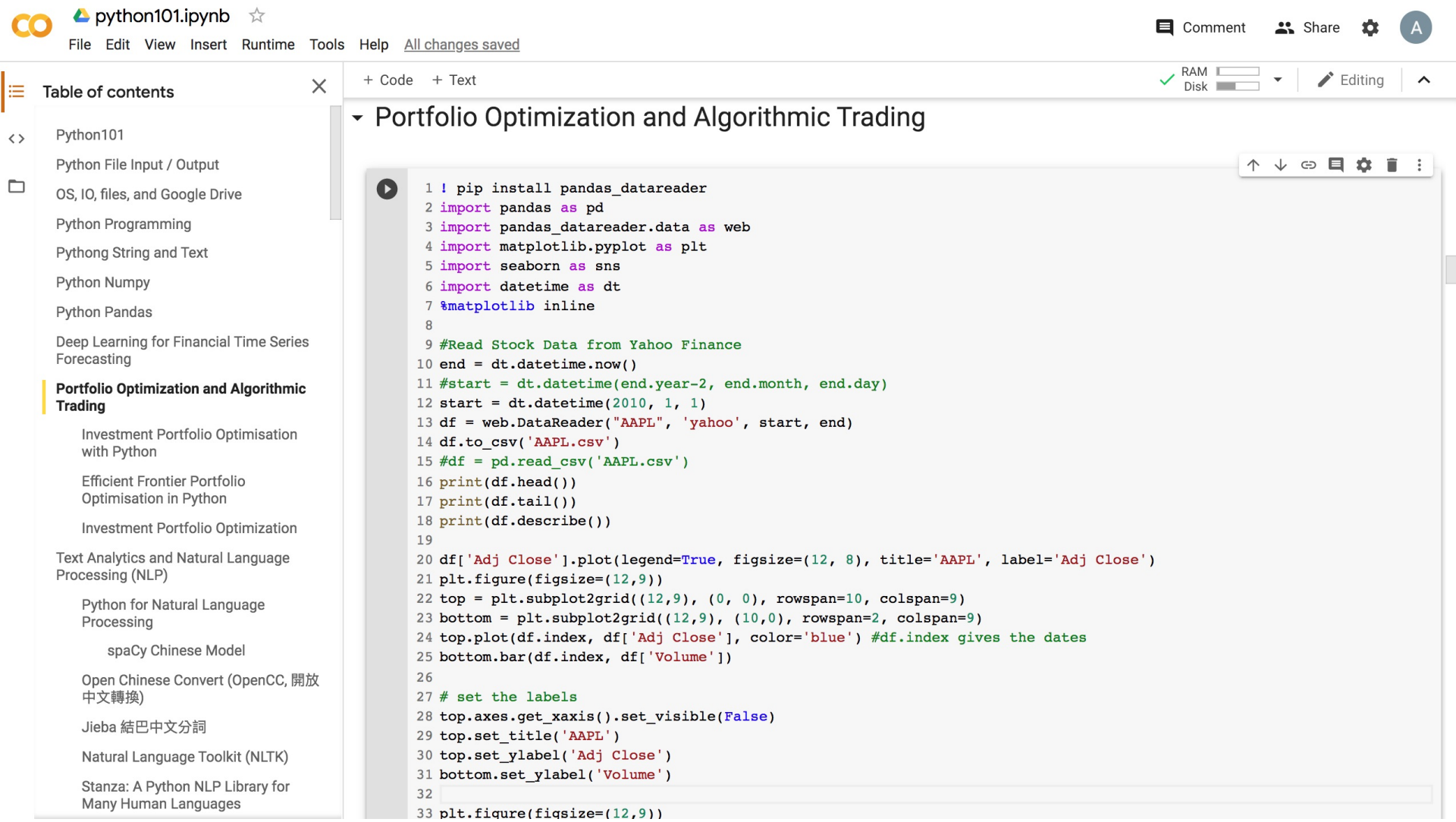
MultiCharts



https://www.multicharts.com/

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT



https://tinyurl.com/imtkupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

Table of contents


```
!pip install plotly
import plotly.graph_objects as go

import pandas as pd
from datetime import datetime
df = pd.read_csv('AAPL.csv')
fig = go.Figure(data=[go.Candlestick(x=df['Date'],
                open=df['Open'],
                high=df['High'],
                low=df['Low'],
                close=df['Close'])])

fig.show()
```

```
Requirement already satisfied: plotly in /usr/local/lib/python3.6/dist-packages (4.4.1)
Requirement already satisfied: retrying>=1.3.3 in /usr/local/lib/python3.6/dist-packages (from plotly) (1.3.3)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from plotly) (1.12.0)
```

https://tinyurl.com/imtkupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

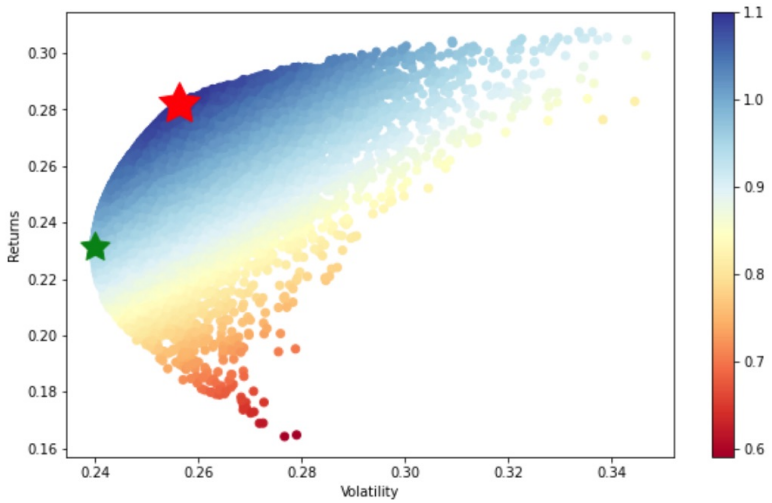
+ Code + Text

RAM Disk Editing

Table of contents


```
51 max_sharpe_port = results_frame.iloc[results_frame['sharpe'].idxmax()]
52 #locate positon of portfolio with minimum standard deviation
53 min_vol_port = results_frame.iloc[results_frame['stdev'].idxmin()]
54
55 #create scatter plot coloured by Sharpe Ratio
56 plt.figure(figsize=(10,6))
57 plt.scatter(results_frame.stdev,results_frame.ret,c=results_frame.sharpe,cmap='RdYlBu')
58 plt.xlabel('Volatility')
59 plt.ylabel('Returns')
60 plt.colorbar()
61 #plot red star to highlight position of portfolio with highest Sharpe Ratio
62 plt.scatter(max_sharpe_port[1],max_sharpe_port[0],marker=(5,1,0),color='r',s=1000)
63 #plot green star to highlight position of minimum variance portfolio
64 plt.scatter(min_vol_port[1],min_vol_port[0],marker=(5,1,0),color='g',s=500)
```

```
<matplotlib.collections.PathCollection at 0x7f13132a01d0>
```



https://tinyurl.com/imtkupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb

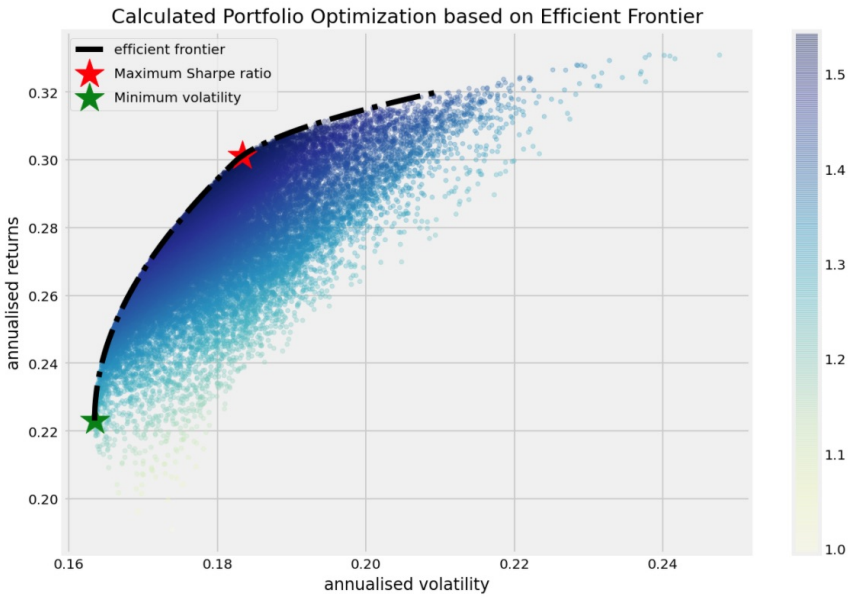
File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

+ Code + Text RAM Disk Editing

Table of contents


```
Annualised Volatility: 0.18


             AAPL   AMZN     FB  GOOGL
allocation  44.67  29.05  26.28    0.0
--------------------------------------------------------------------------------
Minimum Volatility Portfolio Allocation

Annualised Return: 0.22
Annualised Volatility: 0.16


             AAPL  AMZN    FB  GOOGL
allocation  34.02  0.73  6.98  58.26
```



https://tinyurl.com/imtkupython101

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb ☆

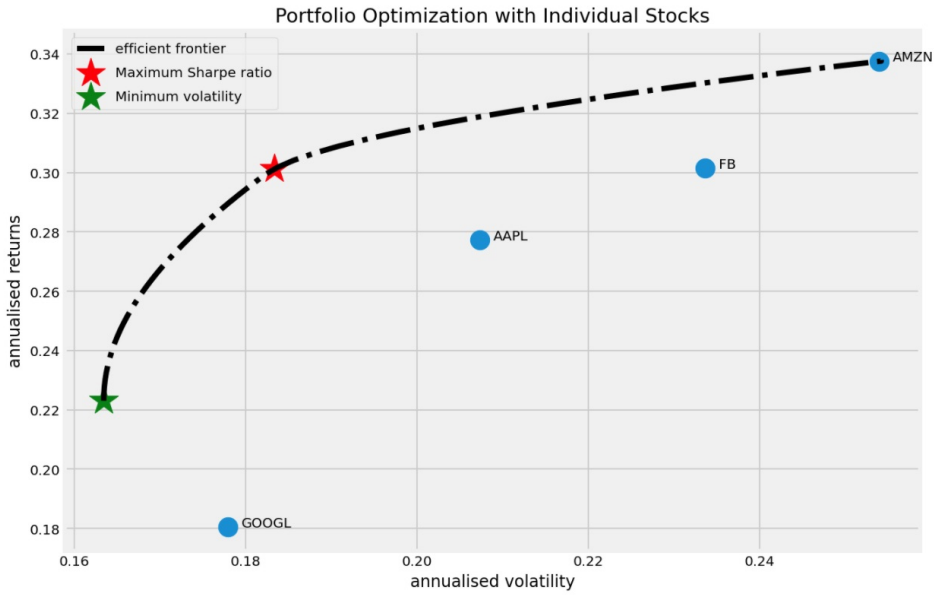
File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

+ Code + Text RAM Disk Editing

```
Annualised Return: 0.22
Annualised Volatility: 0.16


              AAPL  AMZN    FB  GOOGL
allocation   34.02  0.73  6.98  58.26
--------------------------------------------------------------------------------
Individual Stock Returns and Volatility

AAPL : annuaised return 0.28 , annualised volatility: 0.21
AMZN : annuaised return 0.34 , annualised volatility: 0.25
FB : annuaised return 0.3 , annualised volatility: 0.23
GOOGL : annuaised return 0.18 , annualised volatility: 0.18
--------------------------------------------------------------------------------
```



https://tinyurl.com/imtkupython101

Summary

- Portfolio Optimization
- Algorithmic Trading

References

- Paolo Sironi (2016), “FinTech Innovation: From Robo-Advisors to Goal Based Investing and Gamification”, Wiley.
- Ernie Chan (2008), “Quantitative Trading: How to Build Your Own Algorithmic Trading Business”, Wiley
- Ernie Chan (2013), “Algorithmic Trading: Winning Strategies and Their Rationale”, Wiley
- Ernest P. Chan (2017), “Machine Trading: Deploying Computer Algorithms to Conquer the Markets”, Wiley
- Aurélien Géron (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition, O’Reilly Media, 2019, https://github.com/ageron/handson-ml2
- Yves Hilpisch (2018), "Python for Finance: Mastering Data-Driven Finance", 2nd Edition, O'Reilly Media, https://github.com/yhilpisch/py4fi2nd
- Michael Heydt (2015) , Mastering Pandas for Finance, Packt Publishing
- Tucker Balch (2012), Investment Science: Portfolio Optimization, https://www.youtube.com/watch?v=5qbMhXXq0vI
- Quantopian, https://www.quantopian.com/
- Zipline, https://github.com/quantopian/zipline
- Pyfolio, https://github.com/quantopian/pyfolio
- UQER, https://uqer.io/
- Joinquant, https://www.joinquant.com/
- Ricequant, https://www.ricequant.com/
- MultiCharts, https://www.multicharts.com/