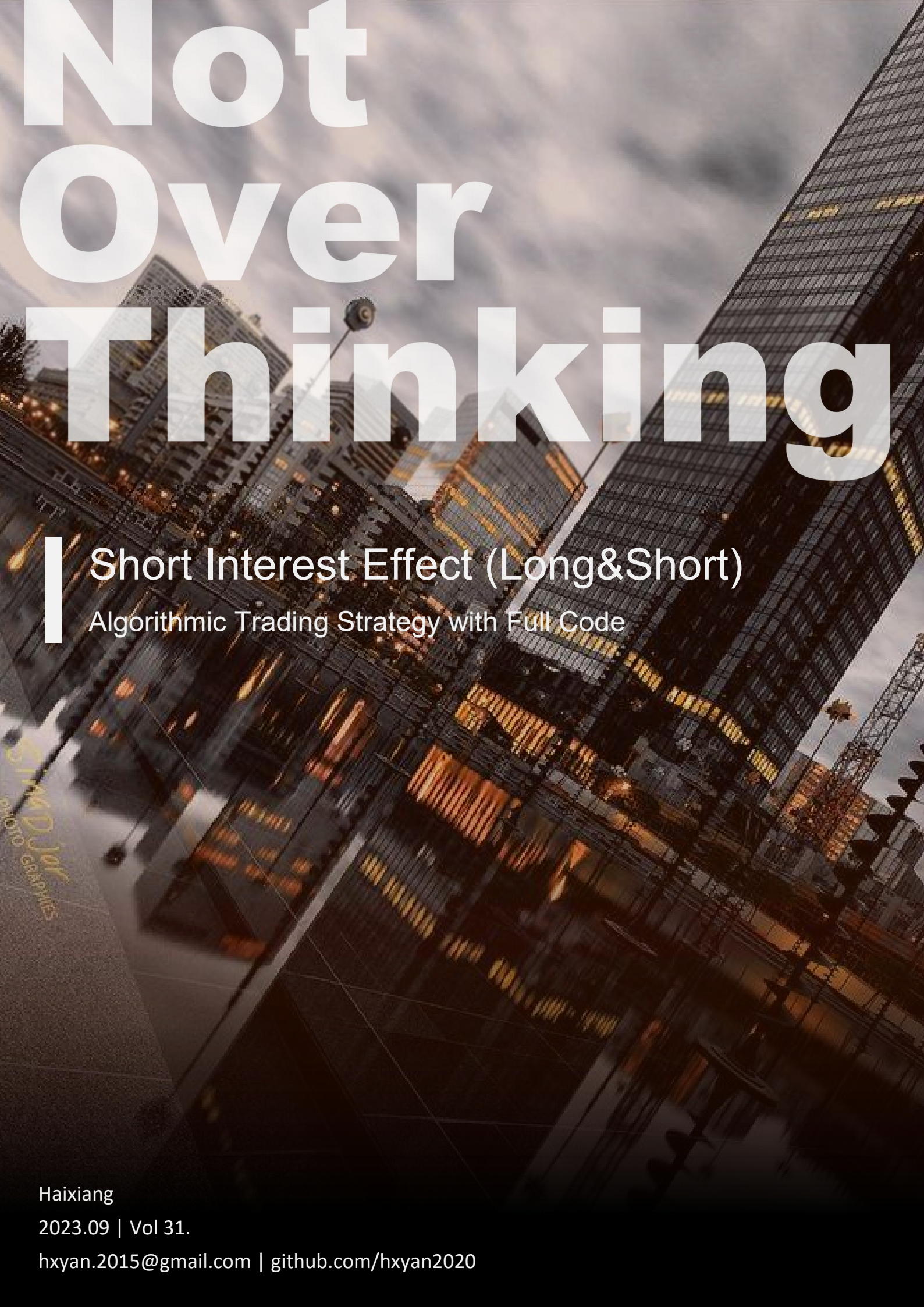

Short Interest Effect (Long&Short)

Algorithmic Trading Strategy with Full Code

Haixiang
2023.09 | Vol 31.
hxyan.2015@gmail.com | github.com/hxyan2020

STRATEGY & ECONOMIC RATIONALE

All stocks from NYSE, AMEX, and NASDAQ are part of the investment universe. Stocks are then sorted each month into short-interest deciles based on the ratio of short interest to shares outstanding. The investor then goes long on the decile with the lowest short ratio and short on the decile with the highest short ratio. The portfolio is rebalanced monthly, and stocks in the portfolio are weighted equally.

BUY	SELL
the decile with the lowest short ratio	the decile with the highest short ratio

PARAMETER & VARIABLES

PARAMETER	VALUE
MARKETS TRADED	Equity
FINANCIAL INSTRUMENTS	Stocks
REGION	United States
PERIOD OF REBALANCING	Monthly
NO. OF TRADED INSTRUMENTS	1000
WEIGHTING	Equal weighting
LOOKBACK PERIODS	Monthly
LONG/SHORT	Long & Short

ALGORITHM

```
from AlgorithmImports import *

class ShortInterestEffect(QCAlgorithm):

    def Initialize(self):
        self.SetStartDate(2010, 1, 1)
        self.SetCash(100000)

        # NOTE: We use only s&p 100 stocks so it's possible to fetch short interest data from quandl.
        self.symbols = [
            'AAPL','MSFT','AMZN','FB','GOOGL','GOOG','JPM','JNJ','V','PG','XOM','UNH','BAC','MA','T','DIS','INTC','HD','VZ','MRK','PFE',
            'CVX','KO','CMCSA','CSCO','PEP','WFC','C','BA','ADBE','WMT','CRM','MCD','MDT','BMY','ABT','NVDA','NFLX','AMGN','PM','PYPL','TMO',
            'COST','ABBV','ACN','HON','NKE','UNP','UTX','NEE','IBM','TXN','AVGO','LLY','ORCL','LIN','SBUX','AMT','LMT','GE','MMM','DHR','QCOM',
            'CVS','MO','LOW','FIS','AXP','BKNG','UPS','GILD','CHTR','CAT','MDLZ','GS','USB','CI','ANTM','BDX','TJX','ADP','TFC','CME','SPGI',
            'COP','INTU','ISRG','CB','SO','D','FISV','PNC','DUK','SYK','ZTS','MS','RTN','AGN','BLK'
            ]

        for symbol in self.symbols:
```

```
            data = self.AddEquity(symbol, Resolution.Daily)
            data.SetFeeModel(CustomFeeModel())
            data.SetLeverage(5)

            self.AddData(QuandlFINRA_ShortVolume, 'FINRA/FNSQ_' + symbol, Resolution.Daily)

        self.recent_month = -1

    def OnData(self, data):
        if self.recent_month == self.Time.month:
            return
        self.recent_month = self.Time.month

        short_interest = {}
        for symbol in self.symbols:
            sym = 'FINRA/FNSQ_' + symbol
            if sym in data and data[sym] and symbol in data and data[symbol]:
                short_vol = data[sym].GetProperty("SHORTVOLUME")
                total_vol = data[sym].GetProperty("TOTALVOLUME")

                short_interest[symbol] = short_vol / total_vol

        long = []
        short = []
        if len(short_interest) >= 10:
            sorted_by_short_interest = sorted(short_interest.items(), key = lambda x: x[1],
reverse = True)
            decile = int(len(sorted_by_short_interest) / 10)
            long = [x[0] for x in sorted_by_short_interest[-decile:]]
            short = [x[0] for x in sorted_by_short_interest[:decile]]

        # trade execution
        stocks_invested = [x.Key.Value for x in self.Portfolio if x.Value.Invested]
        for symbol in stocks_invested:
            if symbol not in long + short:
                self.Liquidate(symbol)

        for symbol in long:
            if symbol in data and data[symbol]:
                self.SetHoldings(symbol, 1 / len(long))
        for symbol in short:
            if symbol in data and data[symbol]:
                self.SetHoldings(symbol, -1 / len(short))

class QuandlFINRA_ShortVolume(PythonQuandl):
    def __init__(self):
        self.ValueColumnName = 'SHORTVOLUME'    # also 'TOTALVOLUME' is accesible

# Custom fee model.
class CustomFeeModel(FeeModel):
    def GetOrderFee(self, parameters):
        fee = parameters.Security.Price * parameters.Order.AbsoluteQuantity * 0.00005
        return OrderFee(CashAmount(fee, "USD"))
```

BACKTESTING PERFORMANCE

Fig 1. Overall Performance

PSR	0.001%	Sharpe Ratio	-0.006
Total Trades	3237	Average Win	0.57%
Average Loss	-0.59%	Compounding Annual Return	-0.383%
Drawdown	30.200%	Expectancy	0.003
Net Profit	-4.938%	Loss Rate	49%
Win Rate	51%	Profit-Loss Ratio	0.98
Alpha	0.002	Beta	-0.021
Annual Standard Deviation	0.067	Annual Variance	0.004
Information Ratio	-0.572	Tracking Error	0.164
Treynor Ratio	0.019	Total Fees	$1667.38
Estimated Strategy Capacity	$72000000.00	Lowest Capacity Asset	BDX R735QTJ8XC9X

Fig 2. Performance Metrics

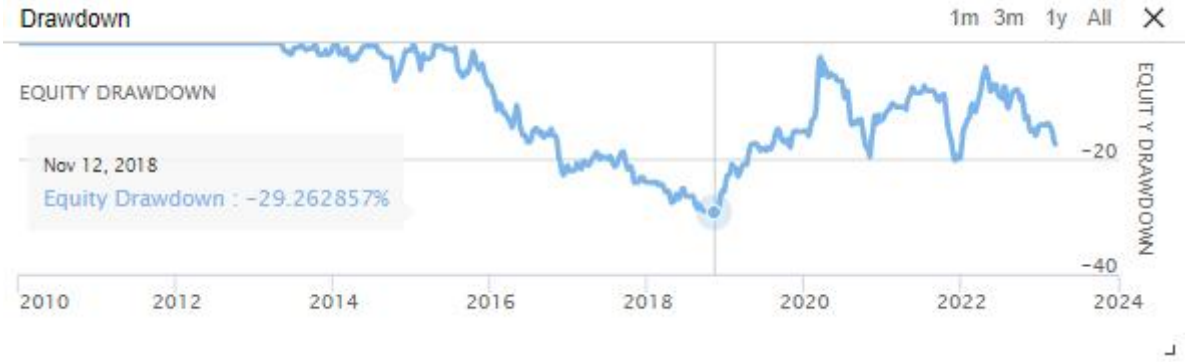

Fig 3. Drawdown

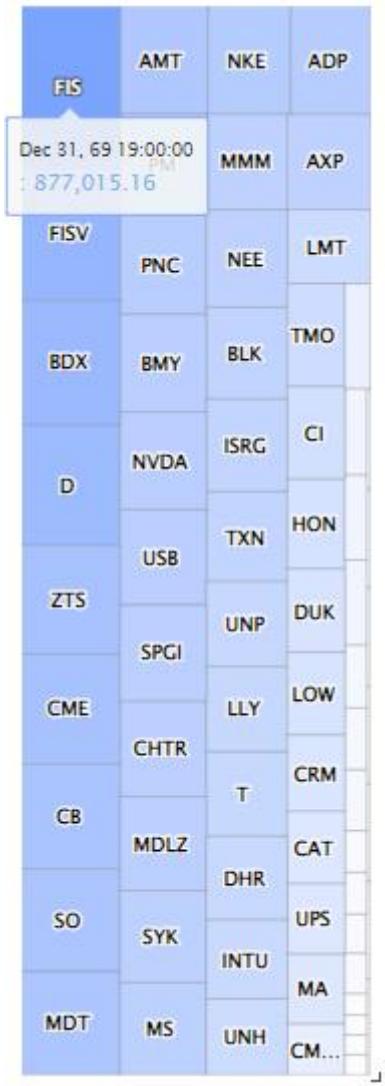

Fig 4. Assets Sales Volume