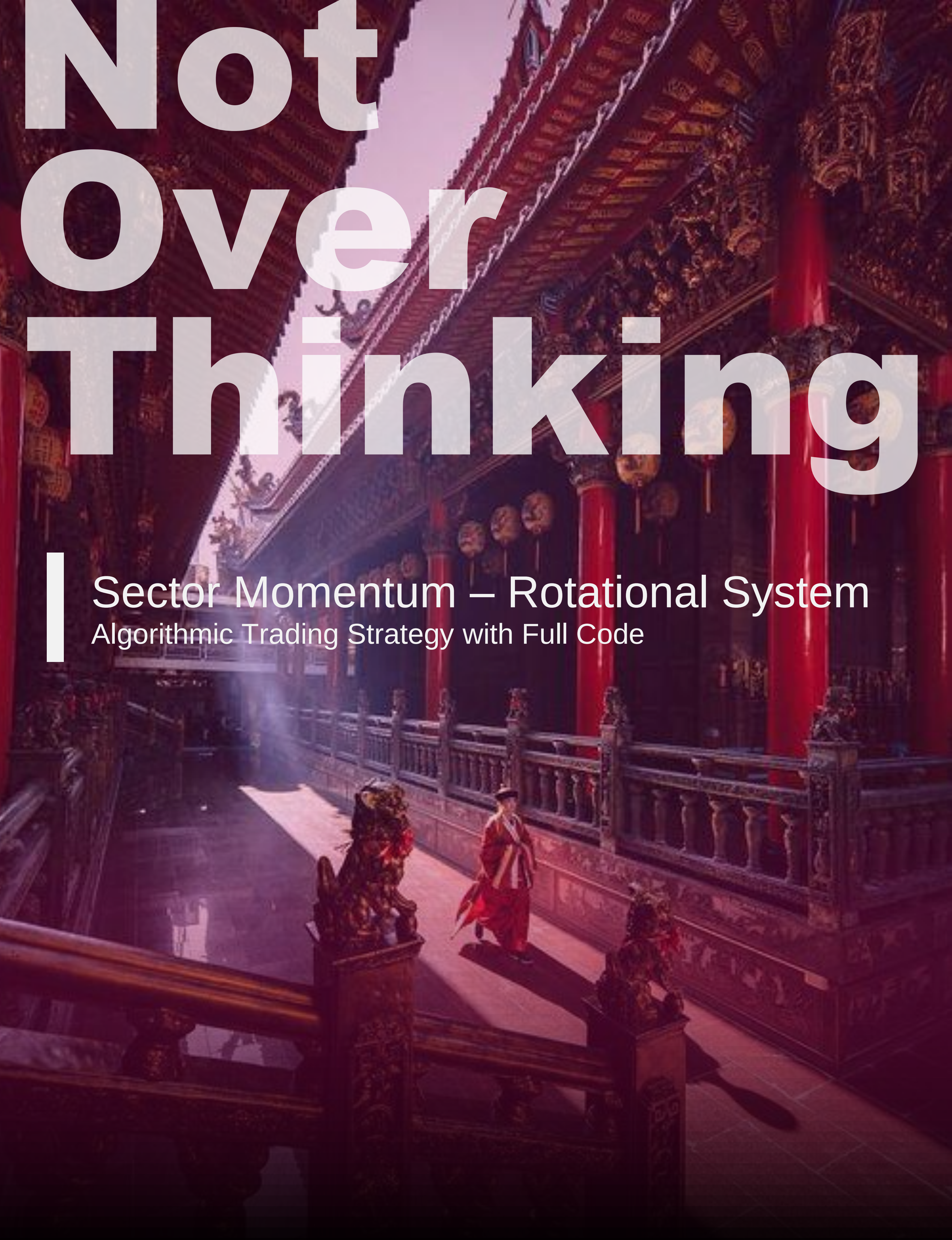


Not Over Thinking

Sector Momentum – Rotational System

Algorithmic Trading Strategy with Full Code

Haixiang

2023.03 | Vol 14.

Hxyan.2015@gmail.com | github.com/hxyan2020

STRATEGY & ECONOMIC RATIONALE

Use ten sector ETFs. Pick 3 ETFs with the strongest 12-month momentum into your portfolio and weight them equally. Hold them for one month and then rebalance.

BUY	SELL
Pick 3 ETFs with the strongest 12-month momentum into portfolio. Hold them for one month and then rebalance.	When the ETF falls out of top 3 based on momentum.

PARAMETER & VARIABLES

PARAMETER	VALUE
MARKETS TRADED	Equities
FINANCIAL INSTRUMENTS	ETFs, funds, stocks
REGION	United States
NO. OF TRADED INSTRUMENTS	10
WEIGHTING	Equal Weighting
LOOKBACK PERIODS	12 months
HOLDING PERIODS	1 month
LONG/SHORT	Long Only

ALGORITHM

```python
class SectorMomentumAlgorithm(QCAlgorithm):

    def Initialize(self):
        self.SetStartDate(2000, 1, 1)
        self.SetCash(100000)

        # Daily ROC data.
        self.data = {}

        self.period = 12 * 21
        self.SetWarmUp(self.period)

        self.symbols = [
                        "VNQ",  # Vanguard Real Estate Index Fund
                        "XLK",  # Technology Select Sector SPDR Fund
                        "XLE",  # Energy Select Sector SPDR Fund
                        "XLV",  # Health Care Select Sector SPDR Fund
                        "XLF",  # Financial Select Sector SPDR Fund
                        "XLI",  # Industrials Select Sector SPDR Fund
                        "XLB",  # Materials Select Sector SPDR Fund
                        "XLY",  # Consumer Discretionary Select Sector SPDR Fund
                        "XLP",  # Consumer Staples Select Sector SPDR Fund
                        "XLU"   # Utilities Select Sector SPDR Fund
                        ]
        for symbol in self.symbols:
            data = self.AddEquity(symbol, Resolution.Daily)
            data.SetFeeModel(CustomFeeModel())
            data.SetLeverage(5)

            self.data[symbol] = self.ROC(symbol, self.period, Resolution.Daily)
            ## daily rate of changes and extend to the entire period

        self.data[self.symbols[0]].Updated += self.OnROCUpdated
        ## update each of the 10 symbols with the output of the OnROCUpdated function

        self.recent_month = -1
        self.rebalance_flag = False

    def OnROCUpdated(self, sender, updated):
        # set rebalance flag
        if self.recent_month != self.Time.month:
            self.recent_month = self.Time.month
            self.rebalance_flag = True

    def OnData(self, data):
        if self.IsWarmingUp: return
        ## .IsWarmingUp is an attribute under self object > no need to specify the indicator
that has been warmed up

        # rebalance once a month
        if self.rebalance_flag:
            self.rebalance_flag = False

            sorted_by_momentum = sorted([x for x in self.data.items() if x[1].IsReady and x[0]
in data and data[x[0]]], key = lambda x: x[1].Current.Value, reverse = True)
            long = [x[0] for x in sorted_by_momentum[:3]]  ## only buy in the top 3

            # Trade execution.
            invested = [x.Key for x in self.Portfolio if x.Value.Invested]
            for symbol in invested:
                if symbol not in long:
                    self.Liquidate(symbol)

            for symbol in long:
                self.SetHoldings(symbol, 1 / len(long))

# 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

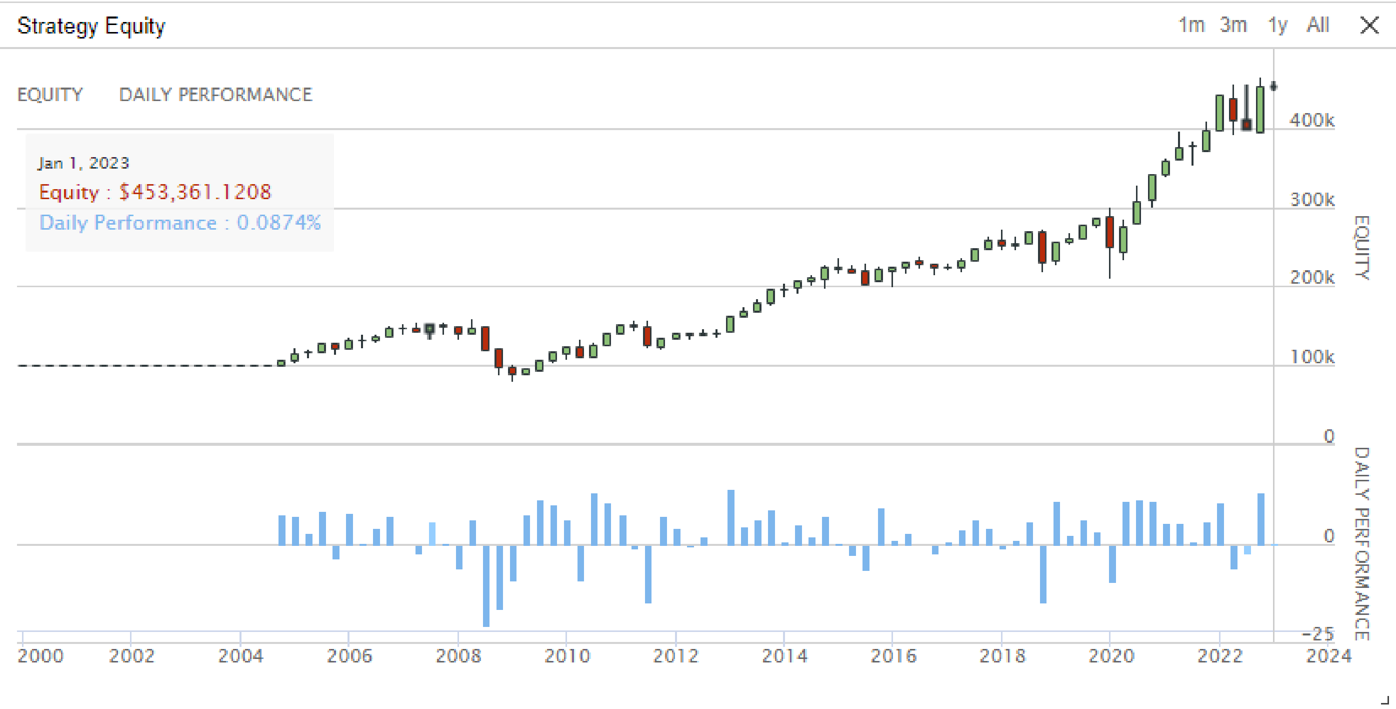
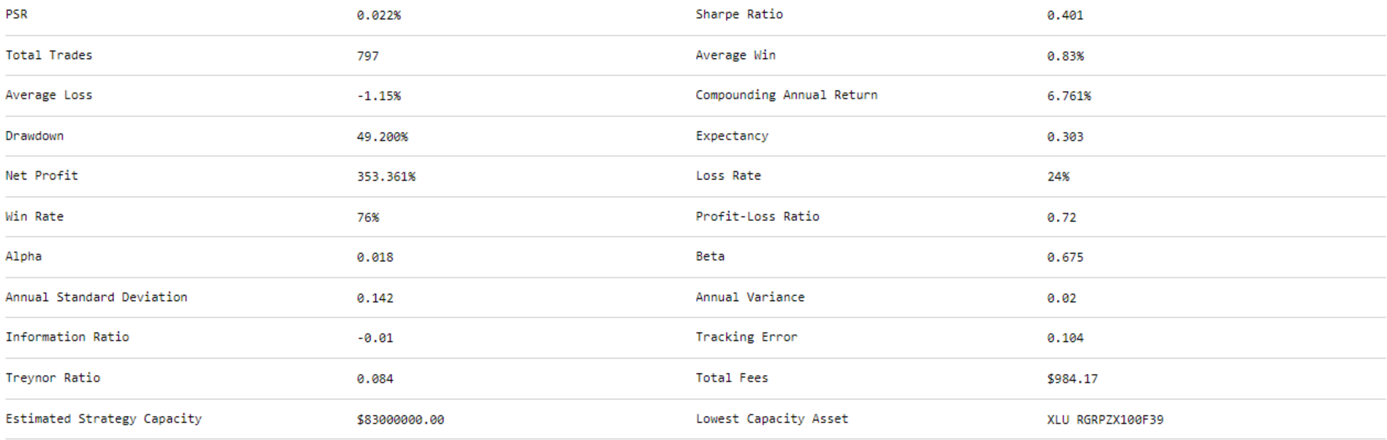


Fig 1. Overall Performance



PSR	0.022%	Sharpe Ratio	0.401
Total Trades	797	Average Win	0.83%
Average Loss	-1.15%	Compounding Annual Return	6.761%
Drawdown	49.200%	Expectancy	0.303
Net Profit	353.361%	Loss Rate	24%
Win Rate	76%	Profit-Loss Ratio	0.72
Alpha	0.018	Beta	0.675
Annual Standard Deviation	0.142	Annual Variance	0.02
Information Ratio	-0.01	Tracking Error	0.104
Treynor Ratio	0.084	Total Fees	$984.17
Estimated Strategy Capacity	$83000000.00	Lowest Capacity Asset	XLU RGRPZX100F39

Fig 2. Performance Metrics

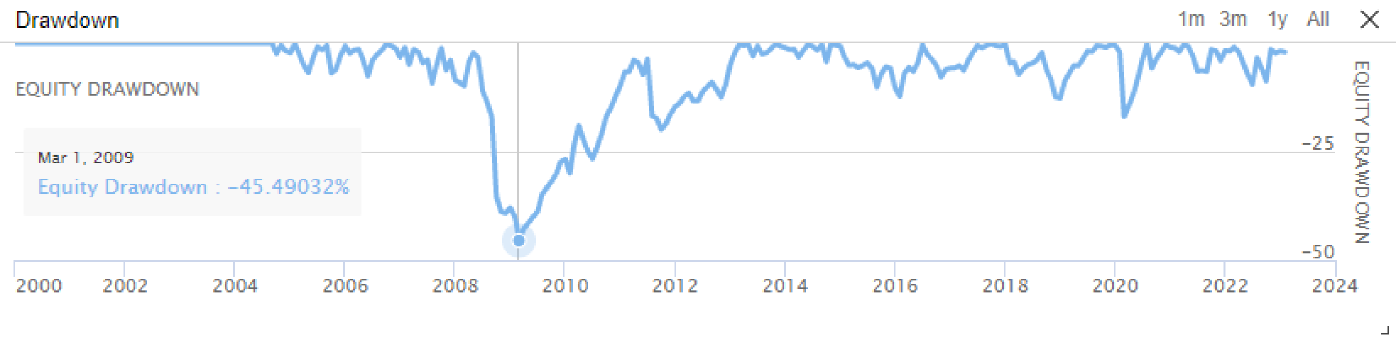


Fig 3. Drawdown

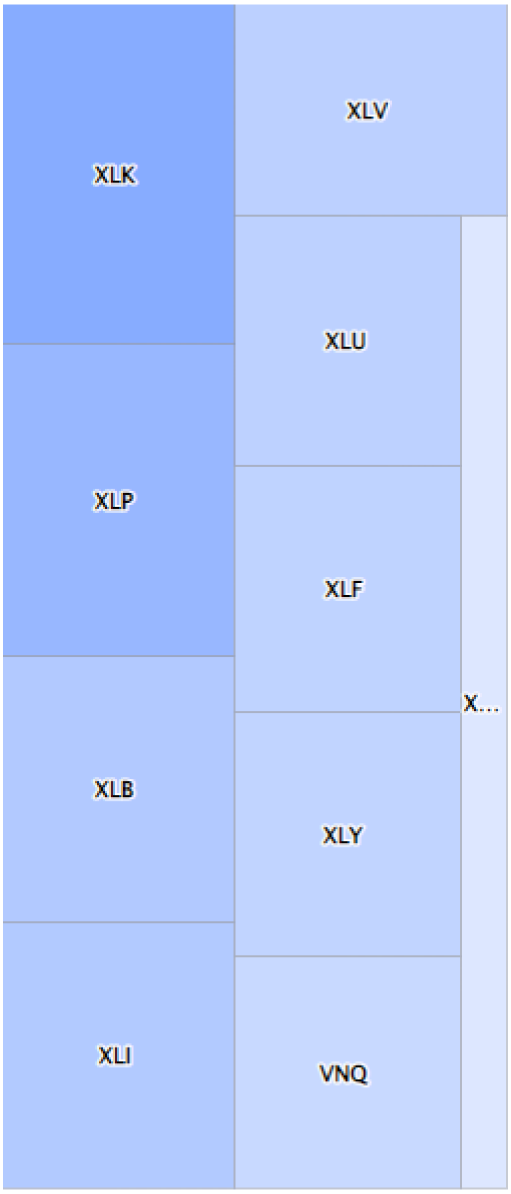


Fig 4. Assets Sales Volume