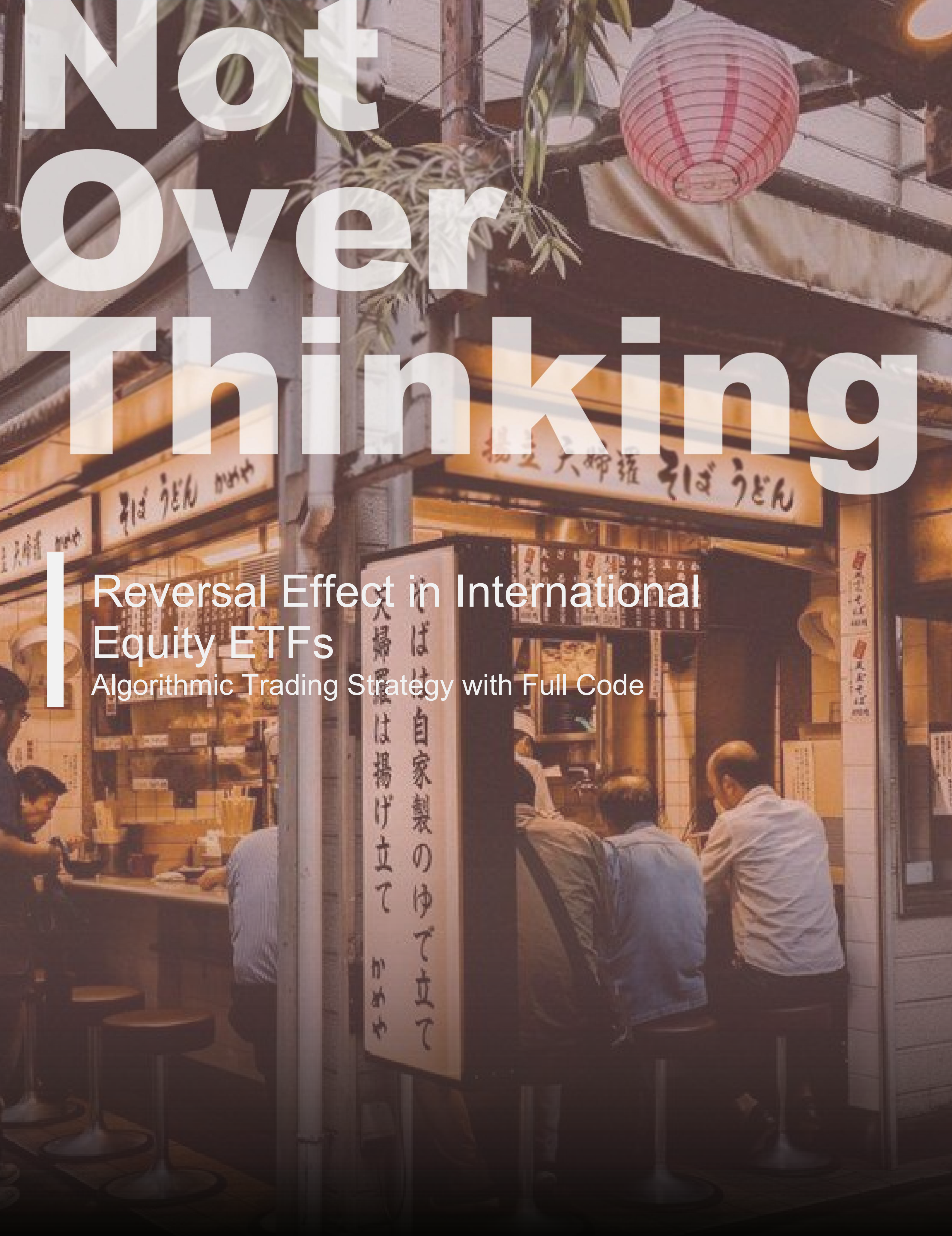

Not Over Thinking

Reversal Effect in International Equity ETFs

Algorithmic Trading Strategy with Full Code

Haixiang
2023.08 | Vol 23.
hxyan.2015@gmail.com | github.com/hxyan2020

STRATEGY & ECONOMIC RATIONALE

The investment universe consists of 16 ETFs (funds) that invest in individual countries' equity indexes. Go long on the bottom four countries with the worst 36 – month return and go short on the top 4 countries with the best 36-month return. Rebalance every three years.

BUY	SELL
the bottom four countries with the worst 36 – month return	the top 4 countries with the best 36-month return

PARAMETER & VARIABLES

PARAMETER	VALUE
MARKETS TRADED	Equity
FINANCIAL INSTRUMENTS	ETFs, funds
REGION	United States
PERIOD OF REBALANCING	3 years
NO. OF TRADED INSTRUMENTS	8
WEIGHTING	Equal Weighting
LOOKBACK PERIODS	36 months
LONG/SHORT	Long & Short

ALGORITHM

```python
from AlgorithmImports import *

class ReversalEffectinInternationalEquityETFs(QCAlgorithm):

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

        # Daily ROC data.
        self.perf = {}

        self.period = 36 * 21
        self.SetWarmUp(self.period)

        self.symbols = [
                        "EWA",  # iShares MSCI Australia Index ETF
                        "EWO",  # iShares MSCI Austria Investable Mkt Index ETF
                        "EWK",  # iShares MSCI Belgium Investable Market Index ETF
                        "EWZ",  # iShares MSCI Brazil Index ETF
                        "EWC",  # iShares MSCI Canada Index ETF
                        "FXI",  # iShares China Large-Cap ETF
                        "EWQ",  # iShares MSCI France Index ETF
                        "EWG",  # iShares MSCI Germany ETF
                        "EWH",  # iShares MSCI Hong Kong Index ETF
                        "EWI",  # iShares MSCI Italy Index ETF
```

```
                        "EWJ",  # iShares MSCI Japan Index ETF
                        "EWM",  # iShares MSCI Malaysia Index ETF
                        "EWW",  # iShares MSCI Mexico Inv. Mt. Idx
                        "EWN",  # iShares MSCI Netherlands Index ETF
                        "EWS",  # iShares MSCI Singapore Index ETF
                        "EZA",  # iShares MSCI South Africe Index ETF
                        "EWY",  # iShares MSCI South Korea ETF
                        "EWP",  # iShares MSCI Spain Index ETF
                        "EWD",  # iShares MSCI Sweden Index ETF
                        "EWL",  # iShares MSCI Switzerland Index ETF
                        "EWT",  # iShares MSCI Taiwan Index ETF
                        "THD",  # iShares MSCI Thailand Index ETF
                        "EWU",  # iShares MSCI United Kingdom Index ETF
                        "SPY",  # SPDR S&P 500 ETF
                        ]

        for symbol in self.symbols:
            data = self.AddEquity(symbol, Resolution.Daily)
            data.SetFeeModel(CustomFeeModel())
            data.SetLeverage(15)

            self.perf[symbol] = self.ROC(symbol, self.period, Resolution.Daily)

        # rebalance month
        self.month = 36
        self.recent_month = -1

    def OnData(self, data):
        if self.IsWarmingUp:
            return

        if self.Time.month == self.recent_month:
            return
        self.recent_month = self.Time.month

        self.month += 1
        if self.month > 36:
           self.month = 1
        else:
            return

        sorted_by_momentum = sorted([x for x in self.perf.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[-4:]]
        short = [x[0] for x in sorted_by_momentum[:4]]

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

        for symbol in long:
```

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

# 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	0.000%	Sharpe Ratio	-0.122
Total Trades	85	Average Win	15.72%
Average Loss	-10.87%	Compounding Annual Return	-4.247%
Drawdown	80.200%	Expectancy	0.041
Net Profit	-63.490%	Loss Rate	57%
Win Rate	43%	Profit-Loss Ratio	1.45
Alpha	-0.014	Beta	-0.074
Annual Standard Deviation	0.151	Annual Variance	0.023
Information Ratio	-0.326	Tracking Error	0.23
Treynor Ratio	0.249	Total Fees	$77.24
Estimated Strategy Capacity	$73000.00	Lowest Capacity Asset	EWO R735QTJ8XC9X

Fig 2. Performance Metrics

Fig 3. Drawdown

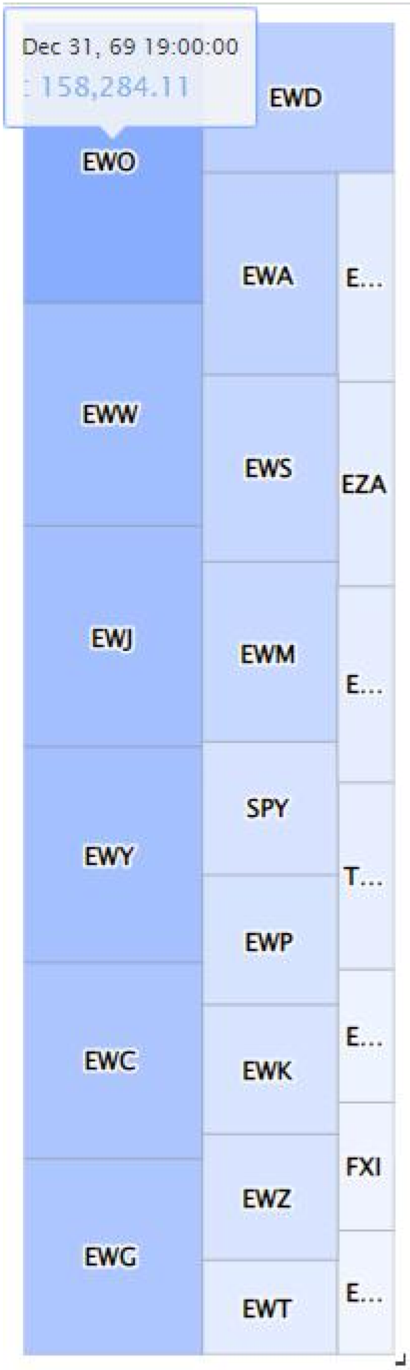

Fig 4. Assets Sales Volume