MS&E 448 Cluster-based Strategy

Anran Lu Huanzhong Xu Atharva Parulekar

Stanford University

June 5, 2018

Summary

- Background

Summary

- Background
- Trading Algorithm

Summary

- Background
- Trading Algorithm
- Simulation and Back-testing

Summary

- Background
- Trading Algorithm
- Simulation and Back-testing
- Result and Discussion

Summary

- Background
- Trading Algorithm
- Simulation and Back-testing
- Result and Discussion
- Problems and Further Improvement

Background: Review of Statistical Arbitrage

Figure: Stock chart of United Airline, Delta Airline and XTN

Statistical Arbitrage employs the price inefficiency between mean-reverting pairs or buckets of stocks.

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

- "Pair": two stocks in the same industry or with similar characteristics

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

- "Pair": two stocks in the same industry or with similar characteristics
  - α: negligible compared to dX_t
  - β: determined by regression
  - X_t: key part of the portfolio

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

- "Pair": two stocks in the same industry or with similar characteristics
  - α: negligible compared to dX_t
  - β: determined by regression
  - X_t: key part of the portfolio
- "Generalized Pair": a stock and an ETF

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

- Trading idea:

Background: Review of Statistical Arbitrage

Basic model of pairs trading:

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t$$

- Trading idea:
  - Open long: Long one share of stock & short β shares of ETF
  - Open short: Short one share of stock & and long β shares of ETF

Background: Cluster-based Trading Strategy

Traditional clustering method:

- Cluster based on empirical returns
- Unstable & risky

Background: Cluster-based Trading Strategy

Traditional clustering method:

- Cluster based on empirical returns
- Unstable & risky

Our cluster motivation:

- Pairs with different "qualities"
- Dynamically adjusted pairs based on changing factors
- Non-stationary time series and cointegration

Background: Cluster-based Trading Strategy

Basic idea: We run pairs trading on each stock-ETF pair by calculating key factors for a stock with respect to all 6 ETFs. We then cluster these stocks based on these key factors.

Background: Cluster-based Trading Strategy

Basic idea: We run pairs trading on each stock-ETF pair by calculating key factors for a stock with respect to all 6 ETFs. We then cluster these stocks based on these key factors.

Key factors for pair:

- Residual s_t. (Non decorrelated $s_t = X_t - m$)
- Mean Reversion Time $1/\kappa$
- Cointegration factors

Background: Cluster-based Trading Strategy

Problem: This strategy is extremely expensive as operations are difficult to vectorize and needs the use of GPUs and multi threading

Background: Cluster-based Trading Strategy

Problem: This strategy is extremely expensive as operations are difficult to vectorize and needs the use of GPUs and multi threading

Better Strategy:

- Calculate a cointegration metric for each stock-ETF pair
- Calculate key factors only for top 20 stocks.
- Create portfolios for only upto 20 stocks.

Trading Algorithm: Cointegration

Cointegration: u_t is the stationary residual obtained after regressing non-stationary x_t and y_t which have a unit root. We try both Engel Granger and Johansen test to determine cointegration.

$$y_t - \beta x_t - \alpha = u_t$$

Trading Algorithm: Cointegration

Johansen's test:

- Cointegrate on price instead of return
- More applicable than Engel Granger test
- Most often used in recent literature

Trading Algorithm: Cointegration

Johansen's test:

- Cointegrate on price instead of return
- More applicable than Engel Granger test
- Most often used in recent literature

The Johansen's test assumes that there are r amount of co-integrating vectors in the data. Thus for a test between two timeseries we get two values for the statistic. One for $r = 0$ and another for $r = 1$. We call them co0 and co1 values through out this presentation.

Trading Algorithm: Cointegration

Co-integration metric

$$\eta = \mathbb{1}\{stat_{r=0} > \hat{c}_{r=0}\}\left(\frac{1}{stat_{r=1}}\right)$$

Trading Algorithm: Cointegration

Co-integration metric

$$\eta = \mathbb{1}\{stat_{r=0} > \hat{c}_{r=0}\}\left(\frac{1}{stat_{r=1}}\right)$$

This metric is designed by us in such a way that when the test passes for r=0 we set η to 0. If the test fails for $r = 0$ we judge based on which stock has lesser statistic values for $r = 1$. Lesser the statistic values for the $r = 1$ case, better the cointegration.

Trading Algorithm: Clustering

Clustering

- We apply K means, and Mixture of Gaussian on mean reversion values and cointegration metric
- The metric values are normalized.
- Mean reversion values are also normalized.

Trading Algorithm: Clustering

Clustering

- We apply K means, and Mixture of Gaussian on mean reversion values and cointegration metric
- The metric values are normalized.
- Mean reversion values are also normalized.

Why it fails:

- Clustering cannot intelligently assign weights to each factor.
- Stocks behave erratically when clustered against multiple ETFs as cointeg matrix is sparce.
- Buckets don't usually make sense.

Results: Clustering

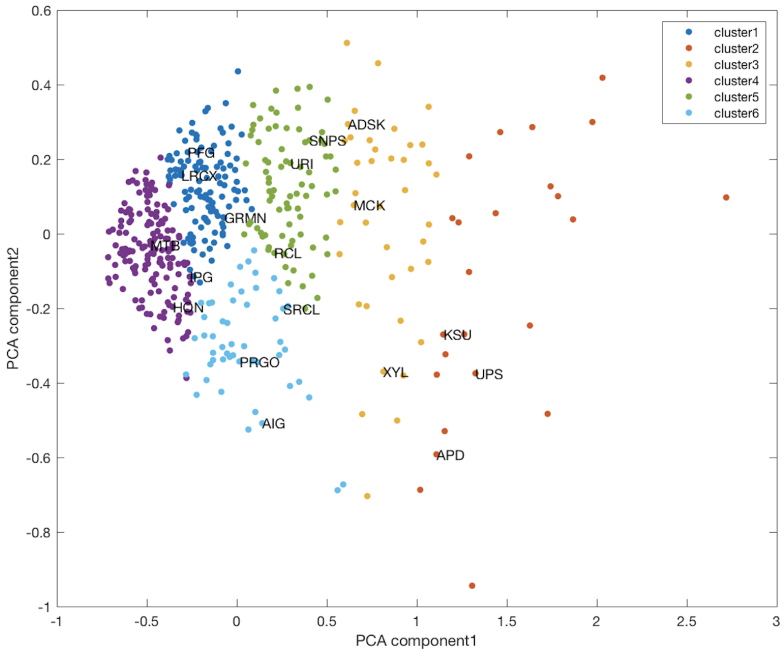
Clustering

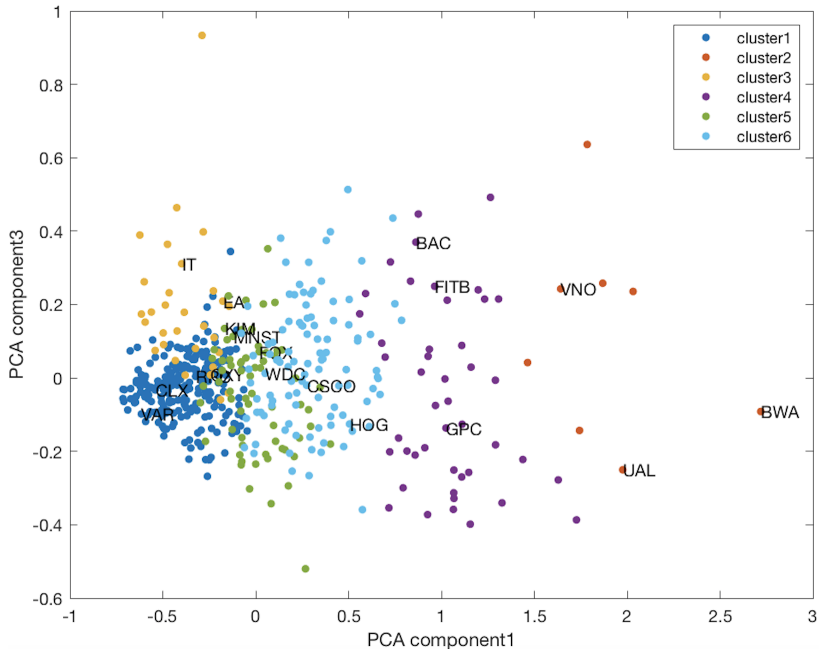
We infer that clustering based on multiple factors cannot automatically or effectively determine the importance given to each factor.

Results: Clustering

Clustering

We infer that clustering based on multiple factors cannot automatically or effectively determine the importance given to each factor.





Trading Algorithm: Cointegration

Select based on cointegration metric:

- For each 60-day window, calculate a cointegration score matrix for each stock ETF pair.
- Sort according to cointegration score.
- Select the top 20 stocks to perform further trading.

Example:
The top 7 cointegrating stocks with the XLF ETF are "Express Scripts", "Nielsen Holdings", "U.S. Bancorp", "TJX Companies Inc.", "Fiserv Inc" "Raymond James Financial", "CME Group Inc". Four of which are financial companies.

Trading Algorithm: Predict Residual

- Base action on $r_t = -m$
  - Here m is the mean of the OU process which models the residual in a 60 day window at time t.
  - m is given by the formulae $m = \frac{a}{1-b}$
  - Here we take actions based on current value of m/σ_{eq}
- Predict r_{t+1} by Long Short Term Memory networks (LSTMs)
  - Pro: Avoid gradient explosion
  - Con: Risk of over-fitting
    Take a long time to train

Trading Algorithm: LSTM

Data Selection

We use an input size of 12 for each time-step for our LSTM. We choose 11 optimal stock returns from the data at time and choose our residual values using that data.

Trading Algorithm: LSTM

Data Selection

We use an input size of 12 for each time-step for our LSTM. We choose 11 optimal stock returns from the data at time and choose our residual values using that data.

Data Prepping

The returns are chosen based on highest cointegration metric with the stock in question and the resudual for each stock is calculated. Thus we have and input for the LSTM. The output at time t-1 is the residual value at t of the stock in question.

Trading Algorithm: LSTM

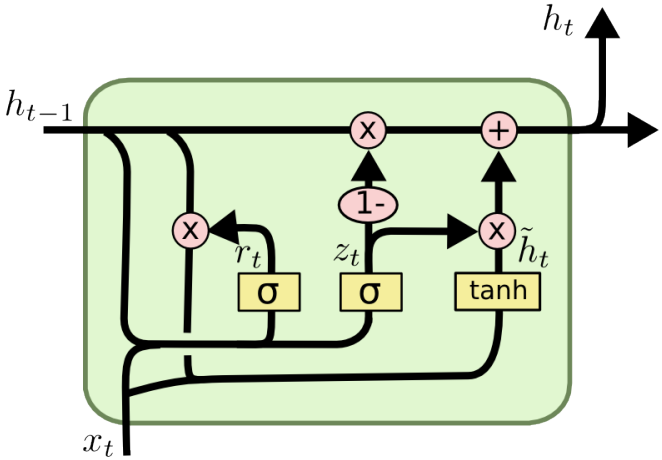
What is an LSTM?

LSTM is a recurrent neural network with sigmoid gates and internal matrix dot products to prevent gradient explosion or vanishing. They can typically train better on longer temporal data lengths.

Trading Algorithm: LSTM

What is an LSTM?

LSTM is a recurrent neural network with sigmoid gates and internal matrix dot products to prevent gradient explosion or vanishing. They can typically train better on longer temporal data lengths.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure: Internal workings of LSTM

Trading Algorithm: LSTM

Training

We train the LSTM using an ADAM optimizer with learning rate 0.01 and the temporal layer as a regression layer. The loss metric is Squared error loss.

Trading Algorithm: LSTM

Training

We train the LSTM using an ADAM optimizer with learning rate 0.01 and the temporal layer as a regression layer. The loss metric is Squared error loss.

The network achieves an RMS error of 0.012 on the test set when trained for 500 epochs. The elapsed time for each stock is 20 mins.

Trading Algorithm: LSTM

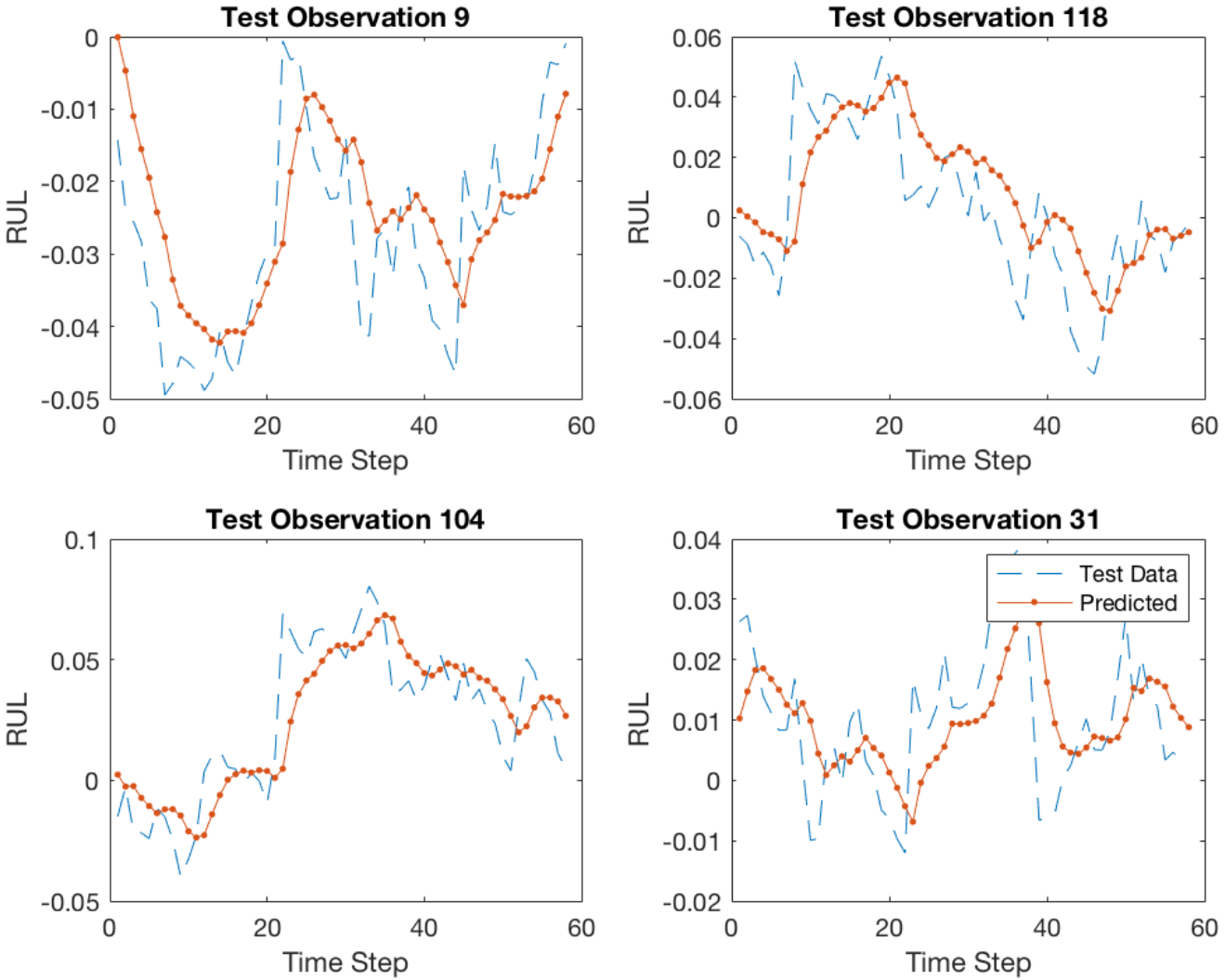


Figure: LSTMs predicted result compared to test data

Trading Algorithm: Predict Residual

Ornstein — Uhlenbeck modelling of X_t

$$dX_t = \kappa(\mu - X_t)dt + \sigma dW_t$$

- some implicit assumptions
- κ: mean reversion
- $X_t \sim \mathcal{N}(\mu, \frac{\sigma^2}{2\kappa})$ at equilibrium
- $s = \frac{X_t - \mathbf{E}\, X_t}{\mathbf{Var}\, X_t}$ determines long/short position

Trading Algorithm: Generate Trading Signal

Our trading strategy is determined by s:

$$s_{long,enter} = -1.25 \; s_{long,exit} = -0.5$$
$$s_{short,enter} = 1.25 \; s_{short,exit} = 0.75.$$

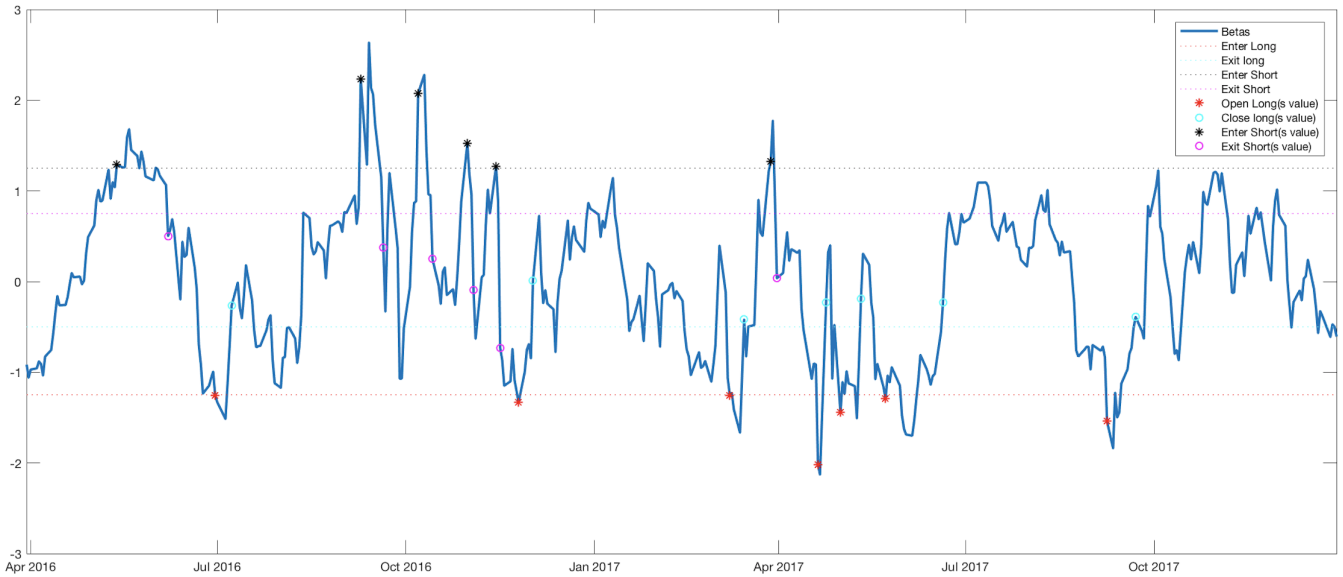


Figure: Generate signals according to our strategy.

Trading Algorithm: Determine Portfolio

$$\begin{bmatrix} SPY & XLF & XLI & XLE & XLU & XLK \\ -1 & 1 & 0 & 0 & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 1 & -1 & -1 \end{bmatrix}$$

State Matrix:The state matrix is a 470×6 matrix representing the stock ETF replationship in the trading algorithm. A value of 1 indicates that the stock should go long for a pairs trade with the corresponding ETF and a 0 indicated its not included while a -1 indicates a short.

Trading Algorithm: Determine Portfolio

SPY	*XLF*	*XLI*	*XLE*	*XLU*	*XLK*
0.01	0.12	0	0	0	0.065
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...
0	0	0	0.0375	0.334	0.141

Determine the weight matrix W for the portfolio:

1. Get mean-reversion parameter κ for the 20 stocks with corresponding ETF.
2. Normalized $\kappa_{ij} = \hat{\kappa}_{ij}$ for each ETF.
3. $W_{ij} = \hat{\kappa}$ if ith stock is in jth ETF list; otherwise, $W_{ij} = 0$.

Simulation and Back-testing

Simplest Modelling:

$$Value(t) = \sum_i h_i(t) \frac{p_i(t+1) - p_i(t)}{p_i(t)}$$

Simulation and Back-testing

Simplest Modelling:

$$Value(t) = \sum_i h_i(t) \frac{p_i(t+1) - p_i(t)}{p_i(t)}$$

$$CumulativeValue(t) = \sum_t Value(t)$$

Simulation and Back-testing

- Transaction Cost, slippage, commission
- Prices to use in reality

Results and Discussion: Simulation

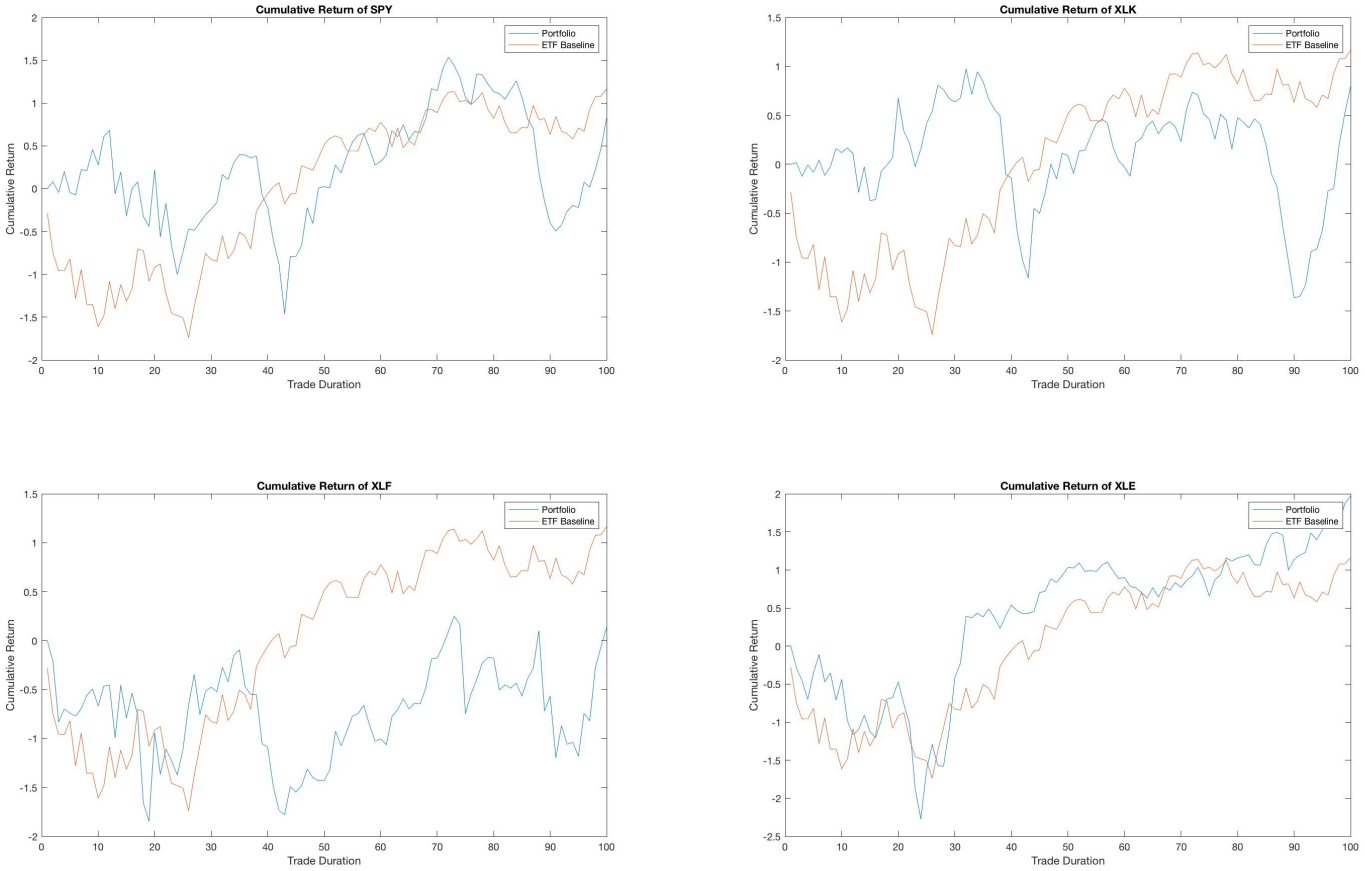


Figure: Cumulative Returns of different ETFs

Results and Discussion: Simulation

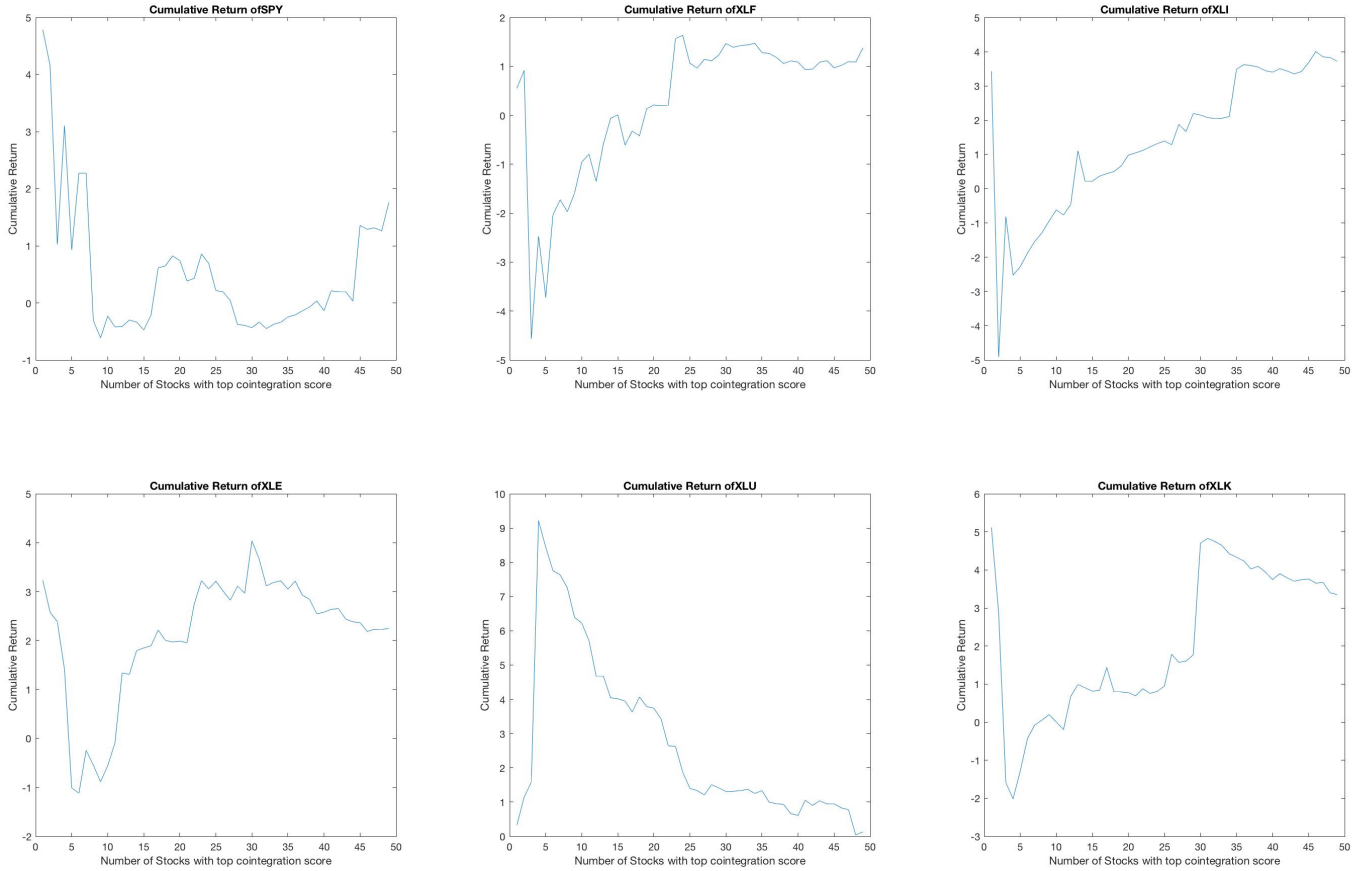


Figure: Impact of Number of Stocks Picked in the Portfolio

Problems

- Data inconsistency between Yahoo Finance and Quantopian
- LSTM requires a huge amount of time to run over the entire universe
- Face the risk of over-fitting
- Lack of test in real environment

References

- Marco Avellaneda and Jeong-Hyun Lee. "Statistical arbitrage in the US equities market". In: *Quantitative Finance* 10.7 (Jan. 2010), pp. 761–782. URL: http://dx.doi.org/10.1080/14697680903124632.
- Sepp Hochreiter and Jurgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.
- Soren Johansen. "Estimation and Hypothesis Testing of Cointegration Vectors in Gaussian Vector Autoregressive Models". In: *Econometrica* 59.6 (1991), pp. 1551–1580.

Thank you!

Q&A?