HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

GRADUATION RESEARCH 1 REPORT

Statistical Arbitrage by Pair Trading using Clustering and Machine Learning

NGUYEN KIM TUYEN
tuyen.nk205196@sis.hust.edu.vn

NGUYEN KHANH TRUNG
trung.nk2051338@sis.hust.edu.vn

DO TUAN MINH
minh.dt200390@sis.hust.edu.vn

Supervisor:	Professor: Tran Viet Trung	______________
		Signature
Department:	Computer Science	
School:	School of Information and Communications Technology	

HANOI, 02/2023

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. ABSTRACT

Pair trading is one of the most popular trading strategies since 1980 for finding statistical arbitrage opportunities in the stock market. The logic behind pairs trading is to trade pairs of stocks belonging to the same industry or having similar characteristics, such that their historical returns move together and are expected to continue to do so in the future. In this project, we investigate the application of machine learning methods to find statistical arbitrage opportunities in the stock market using pair trading strategy. Overall, the project demonstrates the potential of machine learning techniques for improving the performance of pair trading strategies and provides a valuable contribution to the field of financial engineering.

CHAPTER 2. INTRODUCTION

Pairs trading is an investment strategy of constructing a portfolio with matching stocks in terms of market risk factors with a long/buy position in the stock we are optimistic and a short/sell position in the stock we are pessimistic. The approach is designed to eliminate market risk and exploit temporary discrepancies in the relative returns of stocks.

The first step in constructing the portfolio is to identify pairs of stocks based on fundamental analysis, i.e. having common factors such as being in the same industry and having similar market capitalization. The second step is to track return spread between each pair of stocks. A pair of cointegrated assets is selected, that are known to historically move close to each other and are expected to continue to do so. Under the assumption that the spread, defined as the difference in price between the paired assets, is mean-reverting, deviations from the mean can be exploited.

When the spread is abnormally wide and deviation from the mean reaches some predetermined threshold, the outperforming stock is sold short, and the under-performing stock is purchased. As soon as the spread converges back to its mean, the investor liquidates both positions, resulting in a profit. Although this may sound intuitive and trivial, sophisticated machine learning techniques can be used at every step of the pairs trading process. The key to successful pairs trading is the ability to detect patterns in spreads and correctly identify when a spread has become abnormally large and is likely to converge back to it's mean.

In this project, we focus on the first part of the pair trading strategy mentioned above. Our approach will be to use DBSCAN clustering algorithm on a large set of features for clustering stocks. Afterwards, we use cointegration test to extract all possible combinations of stocks in each cluster that are within 5% significance level.

CHAPTER 3. METHOD AND APPROACH

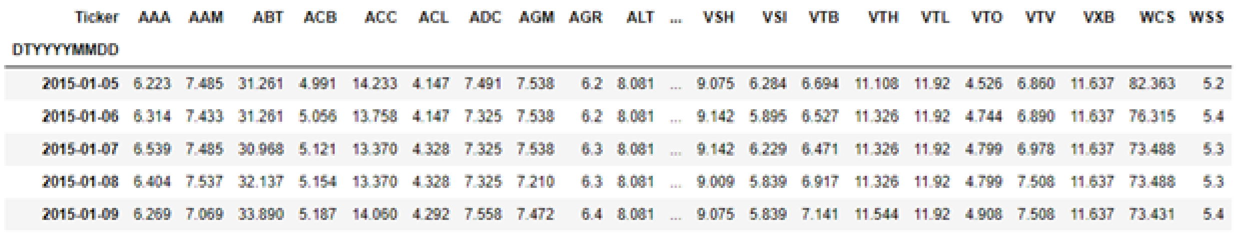
For constructing a Pairs Trading strategy, the first task is to find valid, eligible pairs which exhibit unconditional mean-reverting behavior. One simple approach is to iterate through all stock pairs in the universe of stocks. Then, we incorporate feature engineering for each stock by combining the price movement features processed above with the features from economic prior knowledge. Afterwards, we apply DBSCAN clustering algorithm to cluster the stocks into different groups, and stock pairs with $p - values$ less than 5% are selected from each of the groups.

4.1 Pair Discovering

We start by specifying the stock universe constrained to the HNX and $HOSE$, which are considered to be large and liquid enough. Next, we combine historical daily pricing data with fundamental and industry/sector data in an effort to classify stocks into clusters, after which co-integration test is done on stocks within clusters with the goal of finding pairs with strong mean-reverting relationships.

4.2 Data Collection

To illustrate the pair selection process, we consider daily data with time horizon, early $January$ 2015 to late $December$ 2018, splitting into interval of 3 $months$. In the stock clustering process, we take daily prices as features, which results in approximately 60 $dimensions$.



Ticker	AAA	AAM	ABT	ACB	ACC	ACL	ADC	AGM	AGR	ALT	...	VSH	VSI	VTB	VTH	VTL	VTO	VTV	VXB	WCS	WSS
DTYYYYMMDD																					
2015-01-05	6.223	7.485	31.261	4.991	14.233	4.147	7.491	7.538	6.2	8.081	...	9.075	6.284	6.694	11.108	11.92	4.526	6.860	11.637	82.363	5.2
2015-01-06	6.314	7.433	31.261	5.056	13.758	4.147	7.325	7.538	6.2	8.081	...	9.142	5.895	6.527	11.326	11.92	4.744	6.890	11.637	76.315	5.4
2015-01-07	6.539	7.485	30.968	5.121	13.370	4.328	7.325	7.538	6.3	8.081	...	9.142	6.229	6.471	11.326	11.92	4.799	6.978	11.637	73.488	5.3
2015-01-08	6.404	7.537	32.137	5.154	13.370	4.328	7.325	7.210	6.3	8.081	...	9.009	5.839	6.917	11.326	11.92	4.799	7.508	11.637	73.488	5.3
2015-01-09	6.269	7.069	33.890	5.187	14.060	4.292	7.558	7.472	6.4	8.081	...	9.075	5.839	7.141	11.544	11.92	4.908	7.508	11.637	73.431	5.4

Figure 4.1: Example of daily prices dataframe of the first interval

4.3 Data Preprocessing

4.3.1 Additional Features

Mean Returns

A return is the change in price of an asset, investment, or project over time, which may be represented in terms of price change or percentage change.

Returns are often annualized for comparison purposes, while a holding period return calculates the gain or loss during the entire period an investment was held.

$$Returns_{daily} = \frac{close_i}{close_j} - 1$$

$$MeanReturns = Mean(Returns_{daily}) * T$$

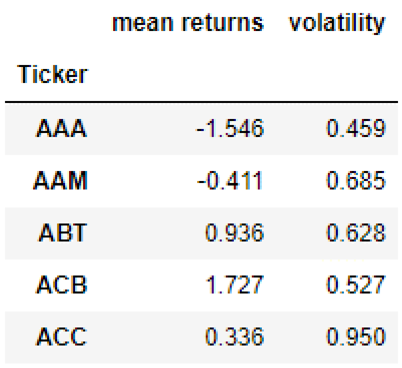
Volatility

One way to measure an asset's variation is to quantify the daily returns (percent move on a daily basis) of the asset. Historical volatility is based on historical prices and represents the degree of variability in the returns of an asset. This number is without a unit and is expressed as a percentage.

While variance captures the dispersion of returns around the mean of an asset in general, volatility is a measure of that variance bounded by a specific period of time. Thus, we can report daily volatility, weekly, monthly, or annualized volatility. It is, therefore, useful to think of volatility as the annualized standard deviation.

$$vol = \sigma\sqrt{T}$$

where:
vol = volatility over some interval time.
σ = standard deviation of returns.
T = number of periods in the time horizon.

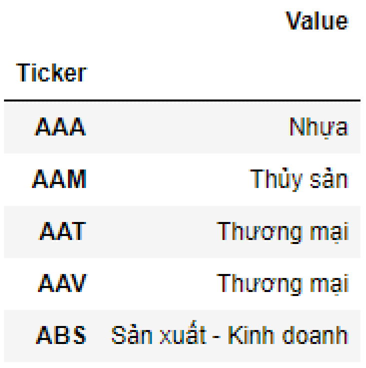


	mean returns	volatility
Ticker		
AAA	-1.546	0.459
AAM	-0.411	0.685
ABT	0.936	0.628
ACB	1.727	0.527
ACC	0.336	0.950

Figure 4.2: The mean returns and volatility dataframe example of the first interval

Industry

Business analysts often classify stocks into industry groups primarily based on similarity in revenue lines. Stocks of similar industries should be related in the future. Our industry data are crawled from VietstockFinance.



Ticker	Value
AAA	Nhựa
AAM	Thủy sản
AAT	Thương mại
AAV	Thương mại
ABS	Sản xuất - Kinh doanh

Figure 4.3: The industry dataframe

Most Machine Learning algorithms cannot work with categorical data and needs to be converted into numerical data. Our approach to this is using one-hot encoding, a technique used to represent categorical variables as numerical values in a machine learning model.

4.3.2 Data Scaling

As we know, most of the machine learning models learn from the data by the time the learning model maps the data points from input to output. And the distribution of the data points can be different for every feature of the data. Larger differences between the data points of input variables increase the uncertainty in the results of the model.

The machine learning models provide weights to the input variables according to their data points and inferences for output. In that case, if the difference between the data points is so high, the model will need to provide the larger weight to the points and in final results, the model with a large weight value is often unstable. This means the model can produce poor results or can perform poorly during learning. So if the data in any conditions has data points far from each other, scaling is a technique to make them closer to each other or in simpler words, we can say that the scaling is used for making data points generalized so that the distance between them will be lower.

Normalization and Standardization are the two main methods for the scaling of the data. Which are widely used in the algorithms where scaling is required. In our case, $sklearn\ StandardScaler$ is prefered as it is more useful in classification problem and when the data have negative values.

4.4 DBSCAN

Density-based spatial clustering of applications with noise ($DBSCAN$) is a data clustering algorithm proposed by Martin Ester, Hans-Peter Kriegel, Jorg Sander and Xiaowei Xu in 1996 (Ester, 1996). It is a density-based clustering non-parametric algorithm which can be described in the following steps:

1. Find the points in the ε neighborhood of every point, and identify the core points with more than min P_{ts} neighbors, where $min\ P_{ts}$ is a parameter to be tuned.
2. Find the connected components of core points on the neighbor graph, ignoring all non-core points.
3. Assign each non-core point to a nearby cluster if the cluster is an ε neighbor, otherwise assign it to noise.

Compared with $K - Means$, $DBSCAN$ has advantages in our use case. Specifically, $DBSCAN$ does not cluster all stocks,i.e. it leaves out stocks which do not neatly fit into a cluster, and the number of clusters does not need to be specified.

Once the data is clustered, we need a way to visualize the high dimensional data and its clusters into a two-dimensional graph. One approach to this visualization is using the nonlinear dimensionality technique known as *t-Distributed Stochastic Neighbor Embedding* $(t - SNE)$ (Hakon Andersen, 2018).

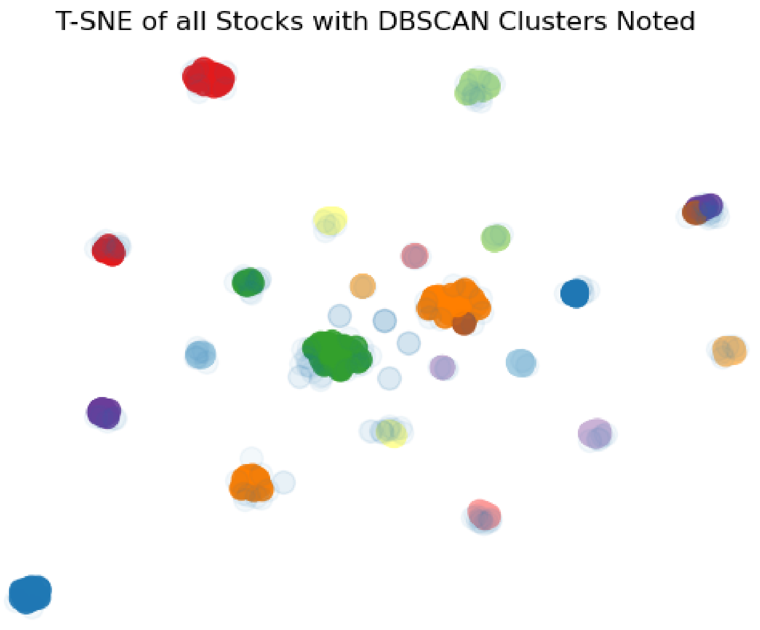


Figure 4.4: Visualization of DBSCAN clusters of the first interval

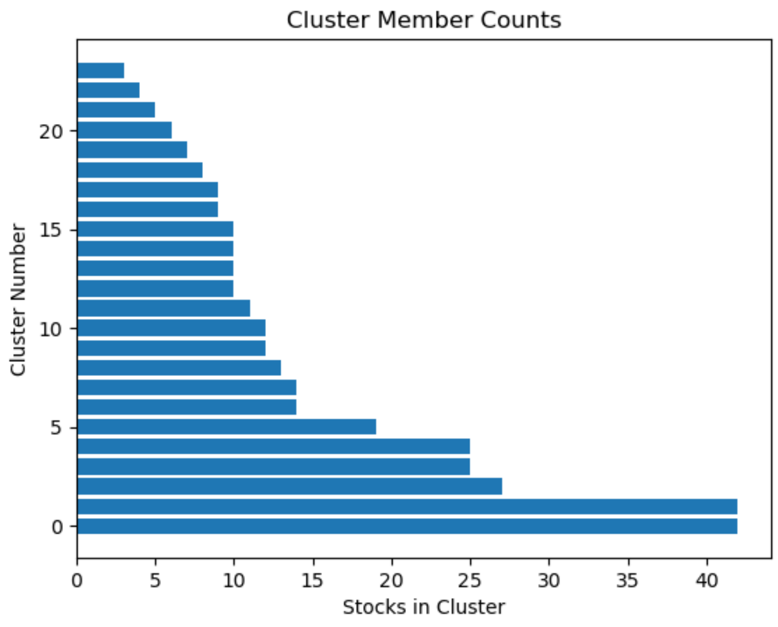


Figure 4.5: Stock number counted of DBSCAN clusters of the first interval

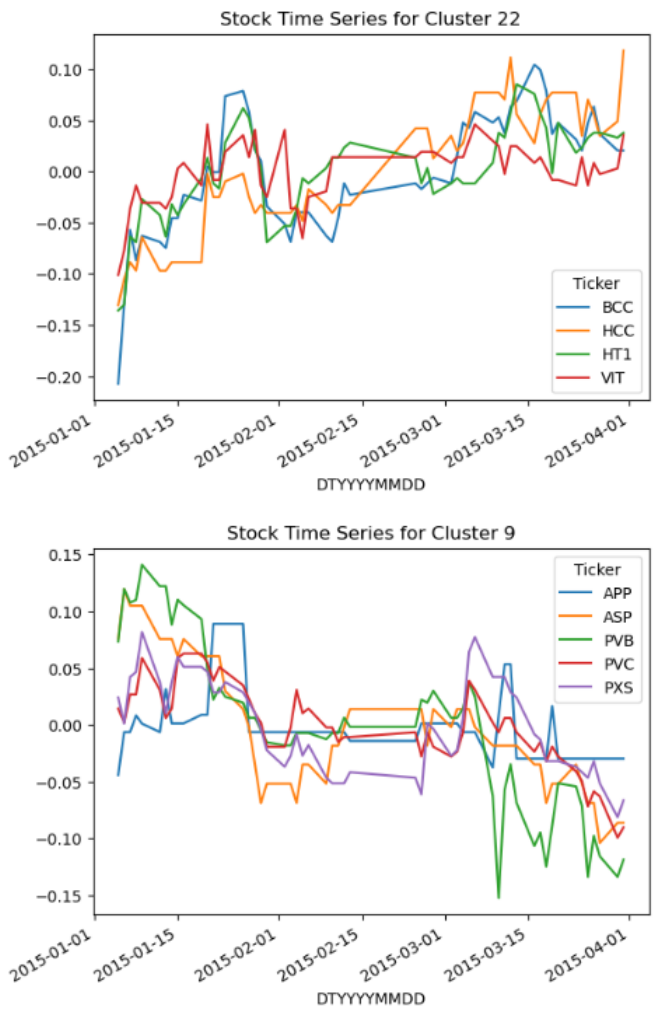


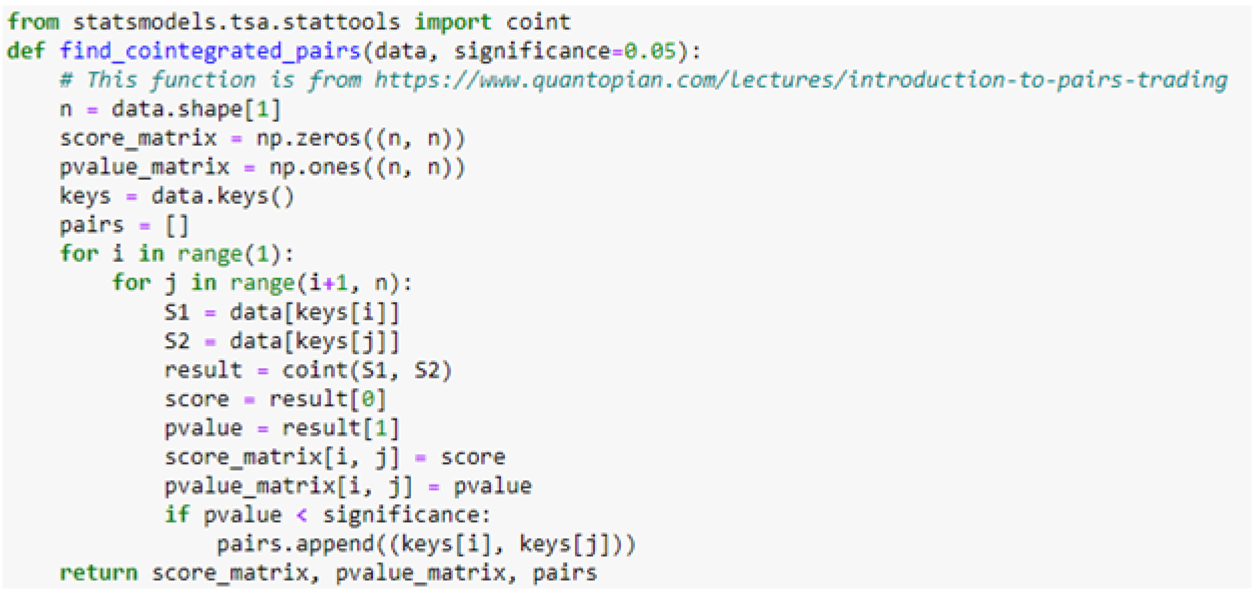
Figure 4.6: Time Series of some clusters noted

4.5 CO-INTEGRATION TEST

A time series is stationary when the parameters of the underlying data do not change over time (Wooldridge, 2009). While cointegration implies that 2 time series $Y(t)$ and $X(t)$ share similar stochastic trends, and they never diverge too far from each other. The cointegrated variables exhibits a longterm equilibrium relationship defined by $Y(t) = \alpha + \beta X(t) + S(t)$, where $S(t)$ is the equilibrium error, which represents short-term deviations from the long-term relationship (Wooldridge, 2009).

For pairs trading, the intuition is that if we find two stocks $Y(t)$ and $X(t)$ whose prices are cointegrated, then any short-term deviations from the spread mean, S, can be an opportunity to place trades accordingly, as we bet on the relationship to be mean reverting. Pairs are deemed as cointegrated when they aren't stationary and tend to move together.

The statsmodels.tsa.stattools module in Python comes with $coint(S_1, S_2)$, a handy function to verify cointegration between 2 time series S_1, S_2 , which implements the Augmented-Dickey-Fuller (ADF) test.



```python
from statsmodels.tsa.stattools import coint
def find_cointegrated_pairs(data, significance=0.05):
    # This function is from https://www.quantopian.com/lectures/introduction-to-pairs-trading
    n = data.shape[1]
    score_matrix = np.zeros((n, n))
    pvalue_matrix = np.ones((n, n))
    keys = data.keys()
    pairs = []
    for i in range(1):
        for j in range(i+1, n):
            S1 = data[keys[i]]
            S2 = data[keys[j]]
            result = coint(S1, S2)
            score = result[0]
            pvalue = result[1]
            score_matrix[i, j] = score
            pvalue_matrix[i, j] = pvalue
            if pvalue < significance:
                pairs.append((keys[i], keys[j]))
    return score_matrix, pvalue_matrix, pairs
```

Figure 4.7: Function to identify cointegration pairs with 95% confidence level

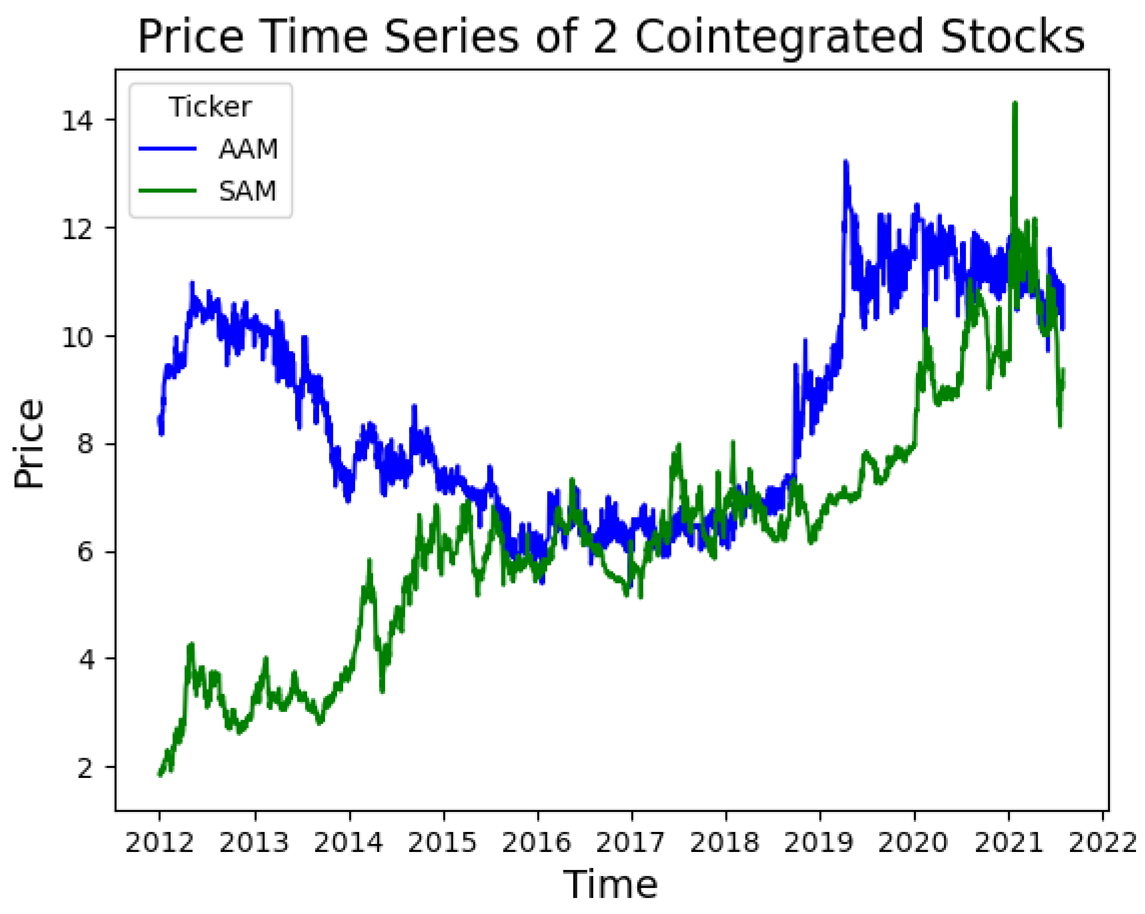


Figure 4.8: An instance of a discovered set of cointegrated pairs

4.6 LINK TO CODE AND DATA

Source to Github Repo: Pair-Trading-Analysis-Repo

CHAPTER 5. CONCLUSIONS

In conclusion, the machine learning pair trading project was successful in achieving its objective of identifying profitable trading pairs for using a pair trading strategy.

However, it's important to note that the strategy had some limitations and risks, including the potential for high transaction costs and the possibility of market disruptions or other unforeseen events. Therefore, further research and analysis would be necessary to determine the long-term viability of the strategy.

Overall, the project demonstrated the potential of using machine learning techniques to improve trading strategies and generate alpha in the financial markets. It also highlighted the importance of rigorous testing and risk management in developing successful trading strategies.