FE520-Introduction to Python for Financial Applications
Project Report
Pair Trading using Data Driven Techniques and Machine Learning

By-

Avirat Belekar Yash Chauhan Shrey Kshatriya

1. **Problem Statement**

   When it comes to trading, managing risk is critical. The market is full of expected and unexpected risk factors. These risks can make it harder for you to profit from trades and minimize risks at the same time. No market on Wall Street is entirely risk-free. This means we need some strategies to help mitigate the risk. In this regard, to help minimize the risk of an unexpected event, we aim to implement the pair trading strategy to protect a particular trading idea.

2. **Introduction**

   a) What is Pair Trading

   A pair trade is trading strategy that involves matching a long position with a short position with two stocks with high cointegration

   b) Underlying principle of Pair Trading

   Let us say you have a pair of securities say A and B, that have some underlying economic link, in our case they are IBM and SIM. You expect the ratio of the difference of the prices of these two remain constant over time. However, this is not the case all the time. There might be divergence in the spread between these two pairs. In this scenario one stock moves up and the other moves down relative to each other. If we expect this divergence to revert to normal with time, we can make pairs trade.

   When there is temporary divergence, the ideal strategy is to buy the underperforming stock and sell the outperforming stock. We are making a bet that the spread between the two stocks will eventually converge and hence pair trading is neutral trading strategy enabling traders to profit from virtually any market condition.

3. **Implementation**

   a) Test of Co-integration

   Cointegration is very similar to correlation, meaning the ratio between two series will vary around the mean. They two series, B and A follow the following:

$$B = \alpha A + e$$

   Where α is the constant ratio and e is the white noise. For pair trading to work between two timeseries, the expected value of the ratio over time must converge to the mean i.e., they must be cointegrated

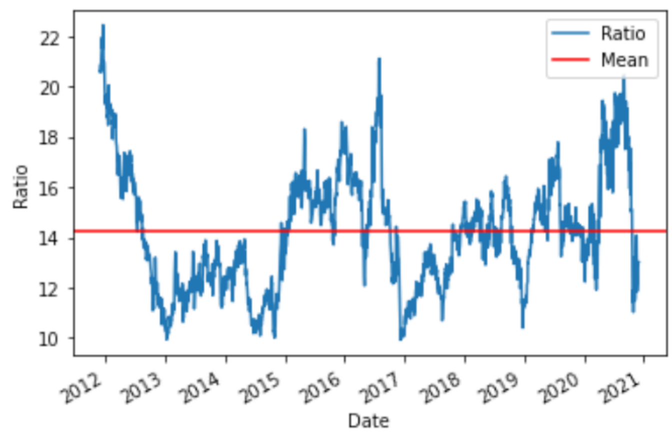

Fig 3.1.1: Ratio between process of two cointegrated stocks and its mean

b) Finding Stocks that behave like Cointegrated Pairs

We have worked with a basket of US Large Cap tech Stocks and included some cryptocurrencies as well. These stocks operate in a similar segment and could have cointegrated prices. We have scanned through of securities and conducted test of cointegration between all the pairs. The function find_cointegrated_pairs returns any pair which has score_matrix, p_value matrix, p_value greater than 0.05

Using the above we have found two securities i.e., IBM and SIM to have the best p value hence become the securities to place our trades on using trading strategy

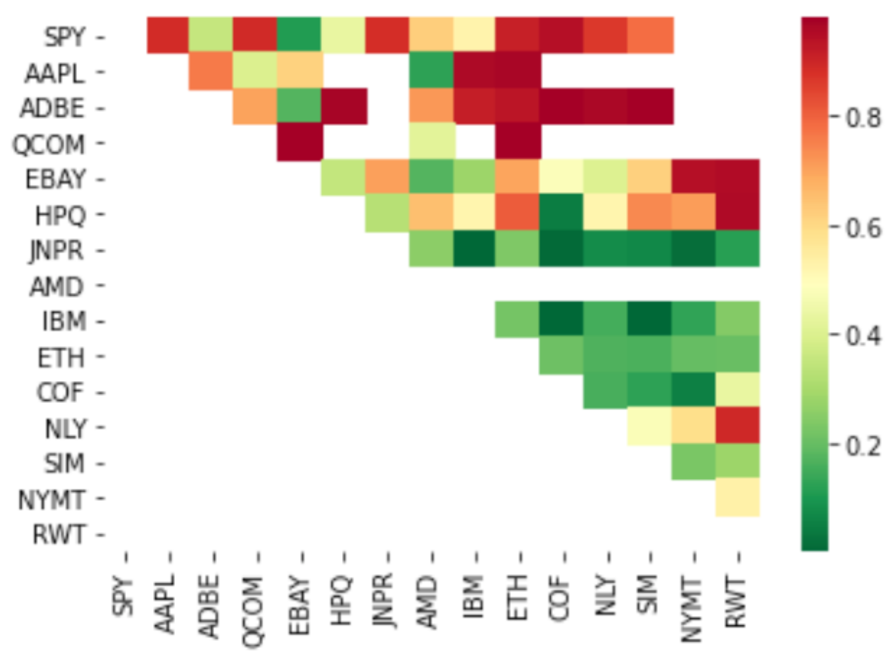

Fig 3.2.1 Seaborn Plot of all securities in S&P500 with p_value > 0.98

c) Developing a trading signal

1) Problem Setup

We have tried to create a signal that tells us if the ratio is a buy or a sell at the next instant of time.
Let Y be out prediction variable
If the Ratio is 1 buy and if ratio is -1, we sell
Based on the above hypothesis we have based our prediction equation

$$Y(t) = Sign(Ratio(t+1) - Ratio(t))$$

Where t is any instant of time

We are not trying to predict actual stock price or even actual value of the ratio in the future, we are just trying to predict the direction of the next move of the ratio

2) Data Collection and Cleaning

We have used Auquan Toolbox to get stock data. Auquan Toolbox is a API that returns stock data if you only mention the trade and the data source. This API also cleans for dividends and stock splits. Hence, we do not need any data cleaning or preprocessing.

From Fig 3.3.b.1 below we can see that the ratio does look like moved around a stable mean. The absolute isn't very useful in statistical terms. It is more helpful to normalize our signal by treating it as a z-score

Z-Score is defined as following:

$$Z\ Score\ (Value) = \frac{(Value - Mean)}{Standard\ Deviation}$$

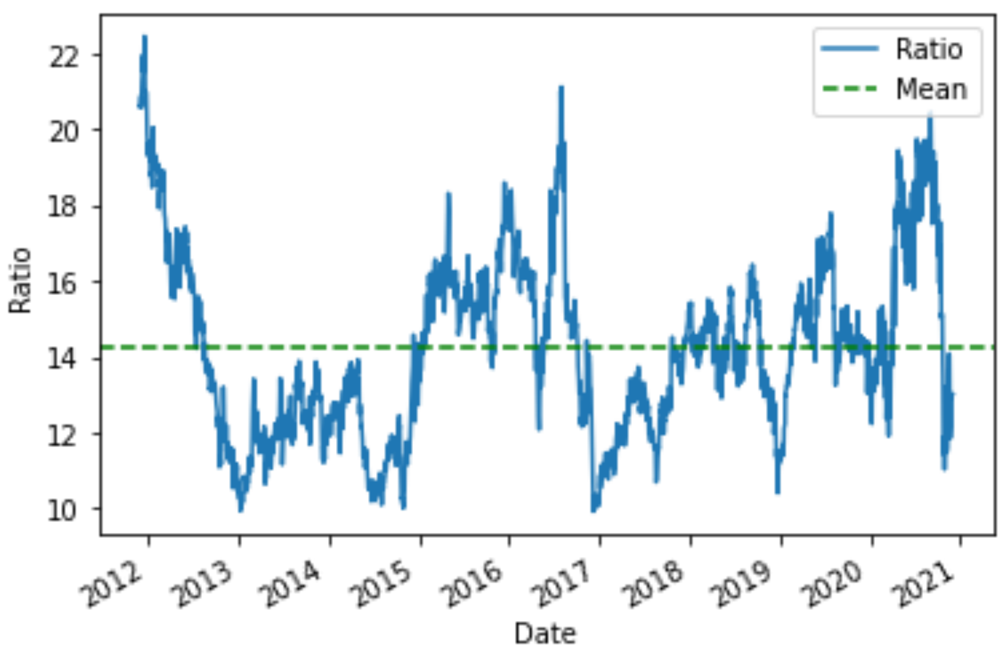

Fig 3.3.b.1 Movement of Ratio around the Mean

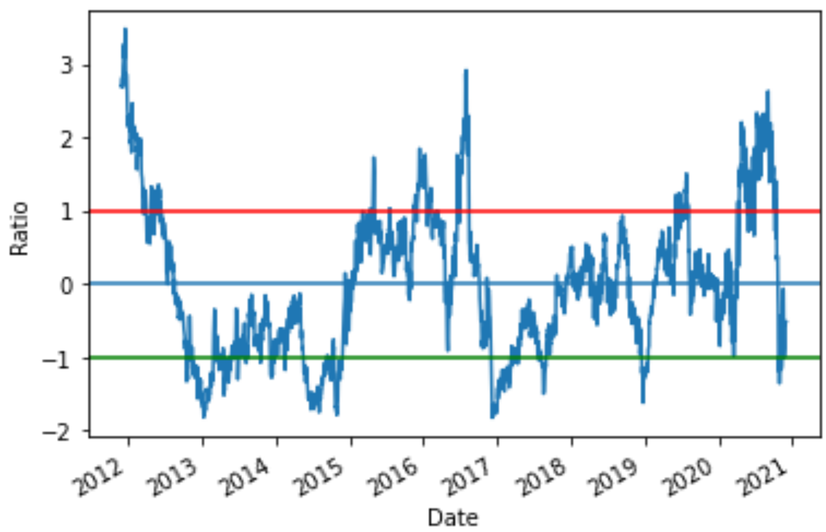

Fig 3.3.b.2 Z Score of Price Ratio between IBM and SIM from 2011-2021

3) Splitting the data into training and testing

We are using data from Yahoo at intervals of trading days over the last 10 years (2264 data points)

Training data consist of 80% of the data which is 2014 data point, and the rest is used for testing

Fig 3.3.c.1 shows the data from Yahoo data source. We have used the cointegration test and to determine which stock pair move around the mean.

	SPY	AAPL	ADBE	QCOM	EBAY	HPQ	JNPR	AMD	I
2011-12-01	103.565163	11.913649	27.139999	41.947304	12.092983	9.802757	19.071377	5.70	135.646
2011-12-02	103.473991	11.968005	27.110001	41.648399	12.060388	9.615181	19.371376	5.65	135.796
2011-12-05	104.601028	12.069658	27.910000	41.763367	12.508577	9.768022	20.022802	5.76	136.641
2011-12-06	104.634201	12.006392	27.959999	42.023960	12.590067	9.788859	19.474232	5.66	138.145
2011-12-07	105.023697	11.949272	27.860001	41.732700	12.606366	9.868756	18.265667	5.72	138.939

Fig 3.3.c.1 Data from Yahoo Data Source

4) Feature Engineering

Since we want to predict the direction of the ratio move and we also know that the two securities are cointegrated, so the ratio tends to move around and revert to the mean. It would be relevant to use features which have certain measures for the mean of the ratio, the divergence of the current value from the mean to be able to generate our trading signal

We have used the following features for generating our trading signal:

- 60 day Moving Average of Ratio: Measure of rolling mean
- 5 day Moving Average of Ratio: Measure of current value of mean
- 60-day Standard Deviation
- z score: (5d MA — 60d MA) /60d SD

Fig 3.3.d.1 shows the 60-day and 5-day moving average of price ratios
Fig 3.3.d.2 shows Z Score of the price ratio and it also brings out the mean reverting nature of the ratios

Fig 3.3.d.1 60-day and 5-day moving average of price ratios

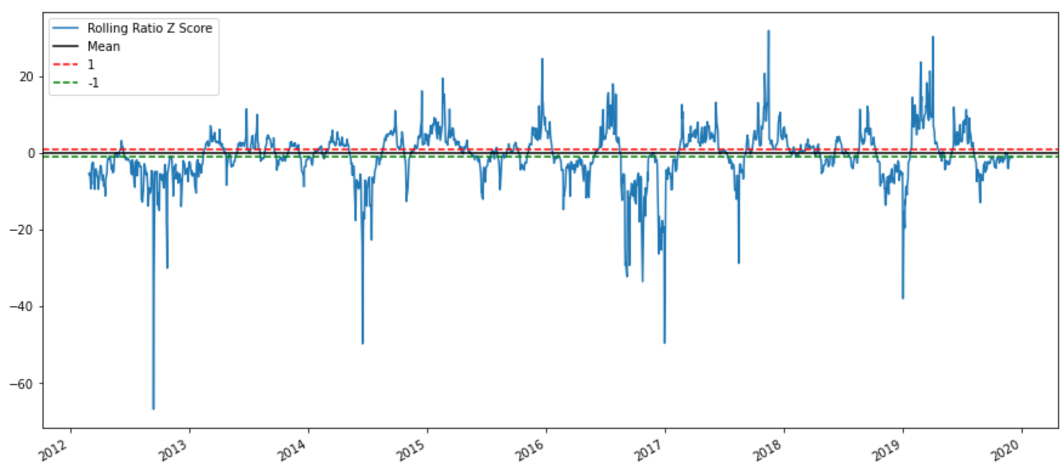

Fig 3.3.d.2 Z score of price ratio

5) Model Selection

We have analyzed the zscore chart and have analyzed that whenever the z-score feature gets too high or too low, it tends to revert.
We have used +1/-1 as our thresholds for too high or too low and have used the following model to generate a trading signal

- Ratio is buying (1) when the zscore is below -1.0 because we expect z score to go back up to 0, hence the ratio increase

- Ratio is selling (-1) when the zscore is above 1.0 because we expect the zscore to go back down to 0, hence the ratio decrease

The graph shows in Fig 3.3.e.1 depicts the buy and sell signal for the IBM and SIM stocks. We tend the sell the ratio (red dots) when it is increasing or buy it back when it low (green dots) and decreasing

Fig 3.3.e.1 Buy and Sell for IBM and SIM stocks

6) **Training, Validation**

To check if our model makes profit, we have made a simple back tester which buys 1 ratio (buy 1 IBM stock and sell ratio * SIM stock) when ratio is low, sell 1 ratio (sell 1 IBM stock and buy ratio * SIM stock) and have calculated the Profit or Loss on these trades

Table 3.3.f.1 shows the profit and loss made on different trading windows

Trading Window	Profit or Loss
30	1070.53
60	1524.38
90	2229.85
120	1529.07

4. Results

To better understand our model performance and accuracy we have evaluated our model against the sharpe ratio

Sharpe ratio is the ratio describes how much excess return you receive for the extra volatility you endure for holding a riskier asset.

$$Sharpe\ Ratio = \sqrt{(N)} * \frac{mean(returns)}{standard\ deviation\ (returns)}$$

For our model we have a sharpe ratio of 7.4 when the trading windows are between 30 and 120 days and 4.5 when trading days are between 60 and 150 days
Thus, we have concluded that it is better to trade during 30 and 120 trading days to get high profits from trades

5. References

- https://www.investopedia.com/articles/07/sharpe_ratio.asp
- https://towardsdatascience.com/machine-learning-for-day-trading-27c08274df54
- https://blog.quantinsti.com/trading-using-machine-learning-python/
- https://www.aptech.com/blog/how-to-interpret-cointegration-test-results/