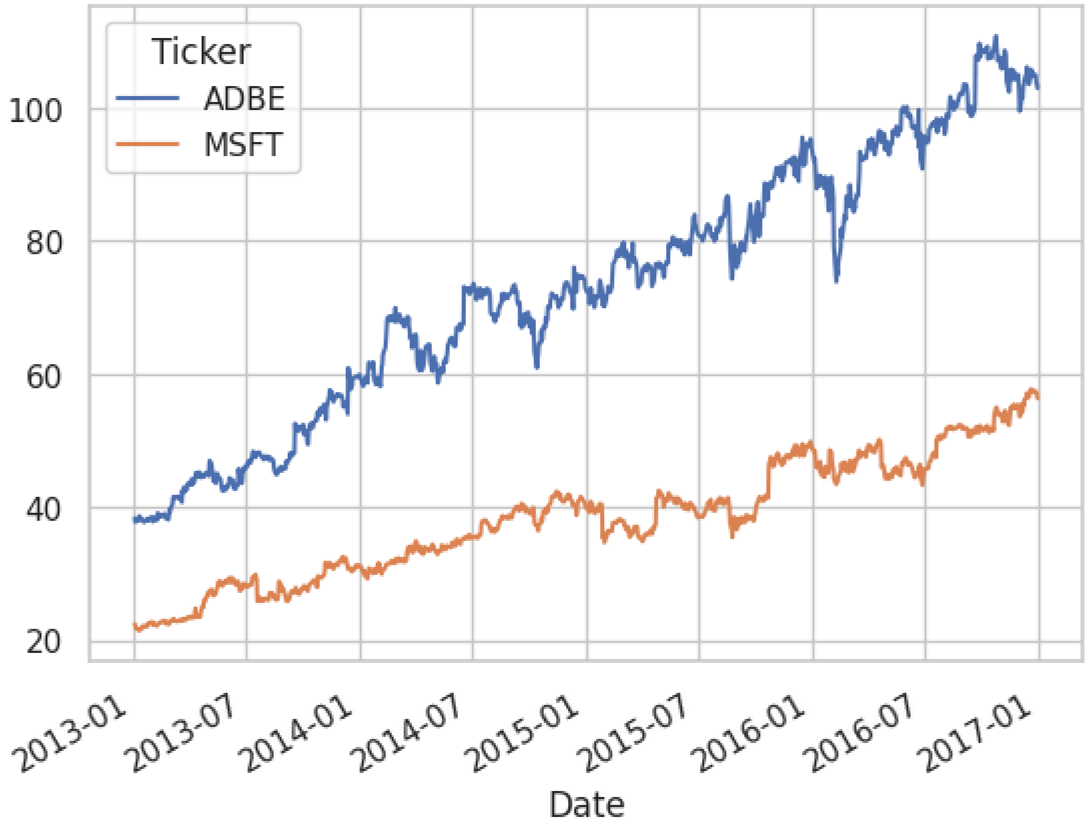
Report: Pairs Trading using Kalman Filters

Assignment 1

In this assignment the task was to implement a Pairs Trading Strategy using machine learning (ML) techniques on historical stock data. Pairs trading is a market-neutral strategy where two correlated stocks are traded. The general idea is to take a long position in one stock and a short position in the other when the price ratio of the two stocks deviates from its historical average.

Here the primary objective is to implement as pairs trading strategy using historical stock data of Adobe Inc. (ADBE) and Microsoft Corp. (MSFT). The strategy uses machine learning to predict the price ratio and generate buy/sell signals, aiming to maximize cumulative profit and loss (PnL).

The data was fetched using Yahoo Finance. The following are the two stocks used:



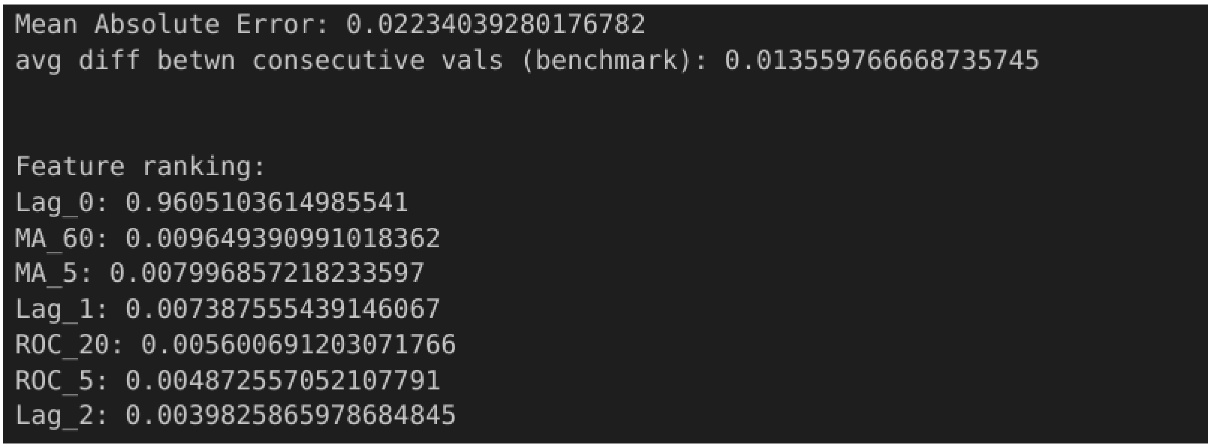
The pvalue of the cointegration of the two stocks was found to be 0.02 indicating that the stocks are fairly cointegrated.

Features were generated based on price ratios and technical indicators which included:

- Moving averages [5-day, 20-day, 30-day, 60-day]
- Rate of Change [5-day and 20-day period]

- Standard deviation [20-day, 60-day window]
- Lag features [1,2,3,4,5-day lag]

Using these features a random forest regressor was trained to predict the next days closing price. But this didn't give good results as it is apparent from the following results for the test data:



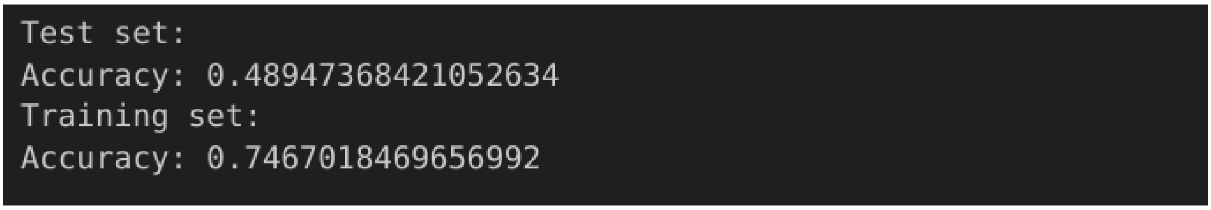
```
Mean Absolute Error: 0.02234039280176782
avg diff betwn consecutive vals (benchmark): 0.013559766668735745


Feature ranking:
Lag_0: 0.9605103614985541
MA_60: 0.009649390991018362
MA_5: 0.007996857218233597
Lag_1: 0.007387555439146067
ROC_20: 0.005600691203071766
ROC_5: 0.004872557052107791
Lag_2: 0.0039825865978684845
```

It performs worse than just using the current days price as the prediction for the next day in terms of mean absolute error.

On the training data it performs much much better than the benchmark (using the current days price as the prediction for tomorrow). This shows that the model is overfitting the training data. Reducing the features and the depth of the random forests also does not help much and gives similar results.

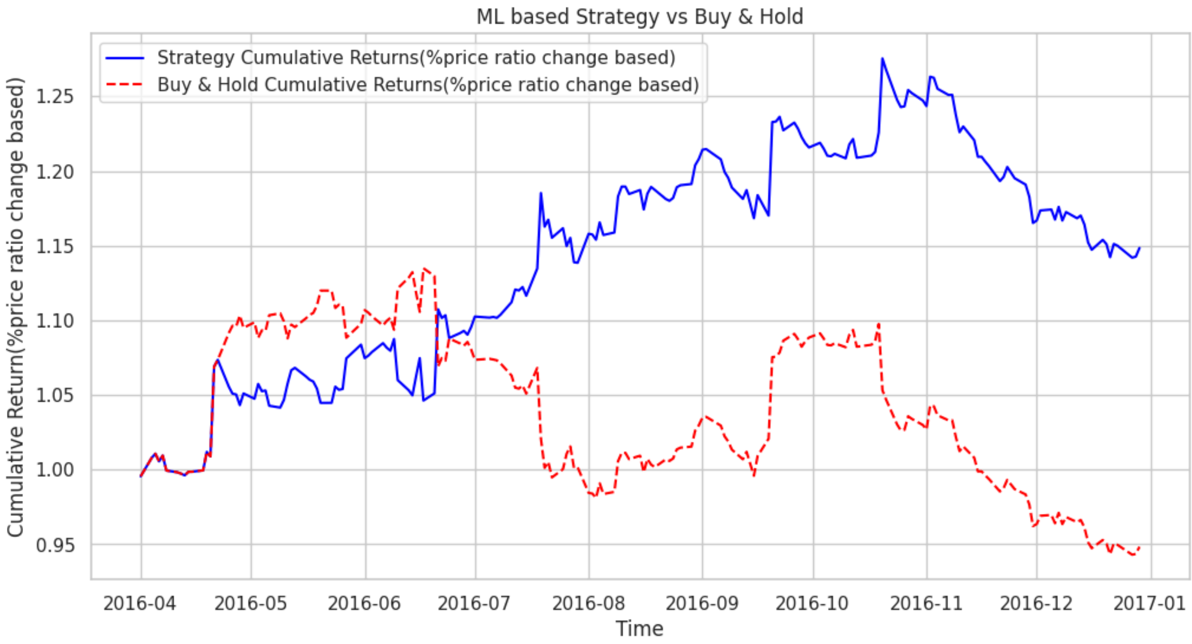
Therefore next thing i tried was trying to predict the direction of price movement using random forest classifier. The following results were obtained:



```
Test set:
Accuracy: 0.48947368421052634
Training set:
Accuracy: 0.7467018469656992
```

Here again the model is getting overfitted on the training data. But here the accuracy on the test data is 50.53% and hence is slightly better than random guessing(50%). Trading based on this with a large volume might give returns in the long run.

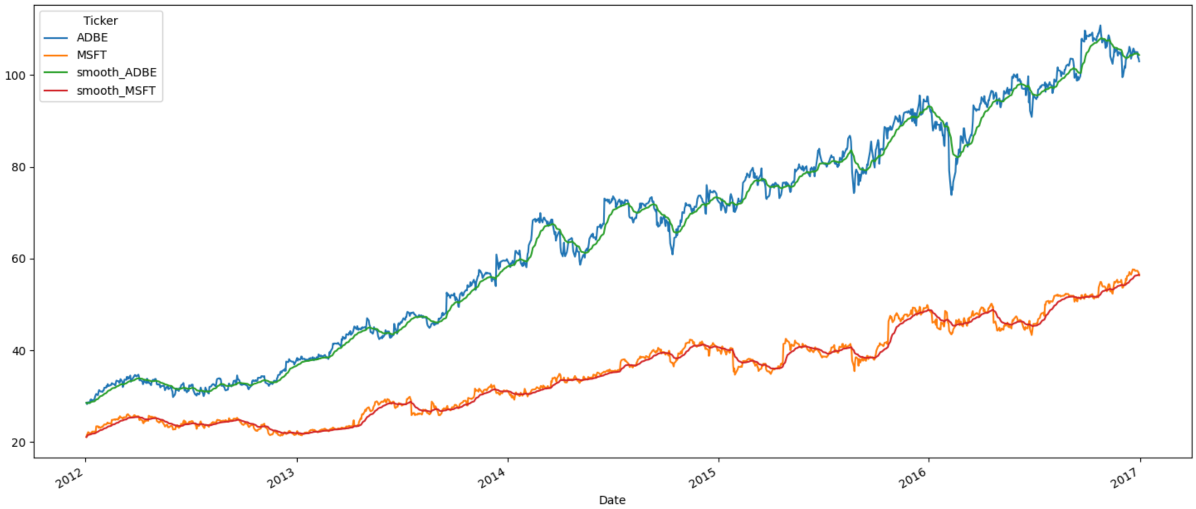
This model was used to make trades. If model predicts a price ratio increase, we buy the stock, otherwise we sell the stock. Here the returns are not calculated directly i.e. the absolute money returns. In turn the percent change in the price ratio of the two stocks is used as we are trading on the relative value of the stocks. The following result was obtained:



Here the ml based buying and selling is compared with the hold strategy which is just buying the spread(the ratio) and holding on to it. It basically denotes the returns we were going to get by default. We can see that the initially the ml based trading does worse than the baseline but towards the end it does better. Hence, even though not reliable, we can get some returns in the long run with these signals

Assignment 2:

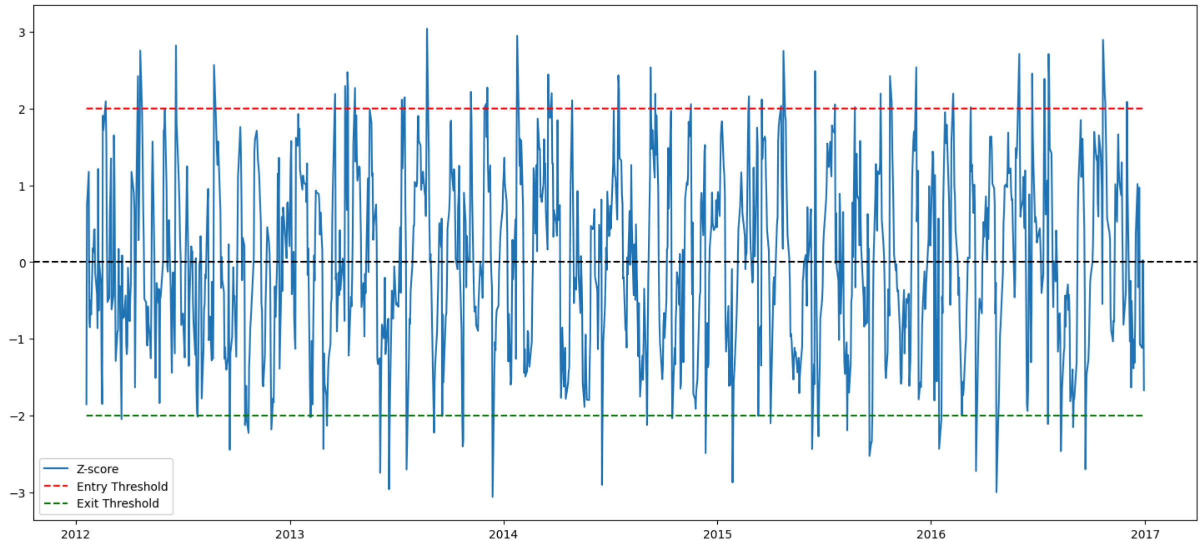
Here Kalman Filtering was used to dynamically estimate the relationship between two stocks and conduct rigorous backtesting to evaluate the strategy's performance.
Kalman filtering was used to smoothen the price movement of the two stocks:



We can observe that there is very little lag (compared to exponential moving avg) in the smooth versions obtained using kalman filtering. These smoothened price data was used to model the dynamic relationship between ADBE and MSFT again using the Kalman Filter. The parameters αt and βt were estimated in the linear regression model:

yt = αt + βt xt + ϵt , where xt is the price of ADBE and yt is the price of MSFT.
Then the spread was calculated as st = yt − (αt + βt xt).
Then zScore was calculated using this. The halflife was used as the rolling window for mean and the standard deviation.

Here trading signals are generated using the zscore of the spread which is dynamically calculated based on the previous data using kalman filter. Now since this spread is mean reverting, trading signals are generated by checking its deviation from the mean. A long entry is position is taken when the zscore of the spread moves from -entryZscore towards 0. And this position is exited when the zscore moves from -exitZscore towards 0. Similarly short entry is taken when the zscore of the spread moves from +entryZscore towards 0. And this position is exited when the zscore moves from +exitZscore towards 0.



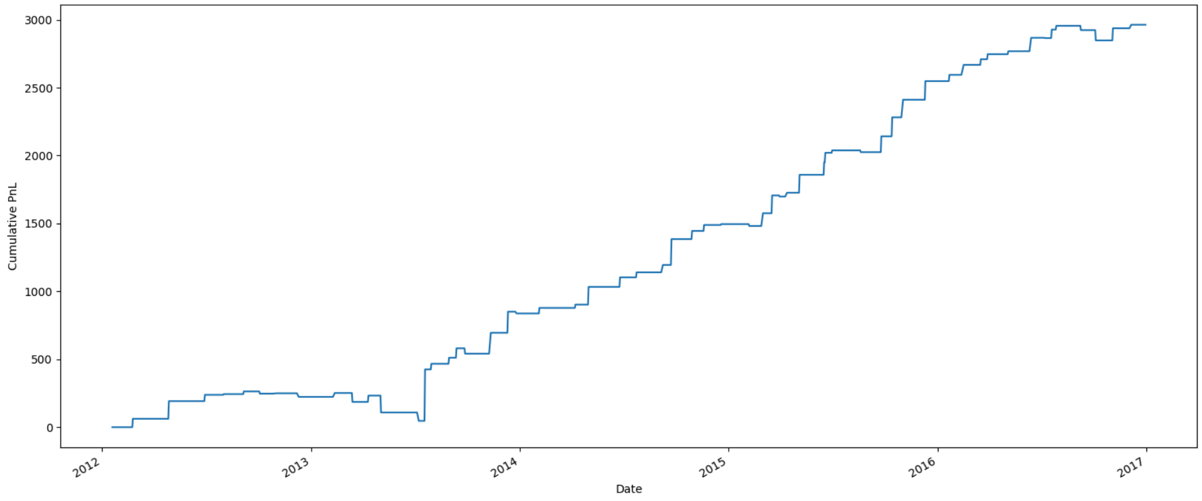
Similarly taking the entryZscore to be (4/3)*entryZscore [the 4/3 found empirically] and then if the above conditions again hold then the volume of the long or short positions is doubled. This ensures that if the spread deviates from the mean too much then we have a larger number of stocks in the long or short position as the spread is more likely to revert to mean now.

A transaction cost of 0.01% is added per transaction.

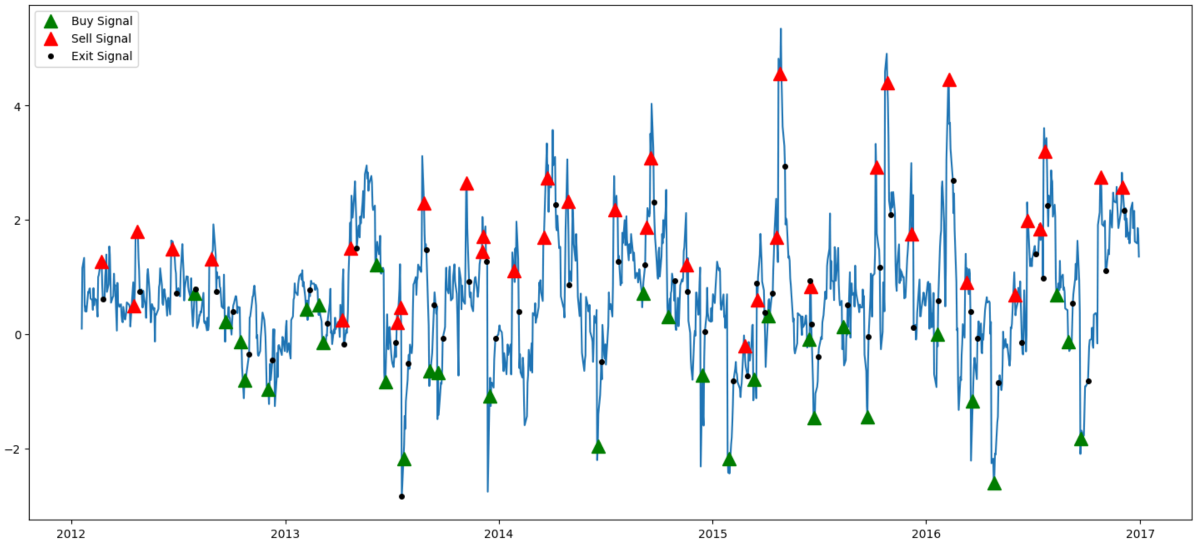
The trade_pnl is updated at every exit of position in the market. Hence the cumulative pnl is blocky as it gets updated only at certain intervals (exit of position in the market).

Here the entryZscore = 2 and exitZscore = 0

The following is based on a max trade volume of 100*2. So for the estimating % returns we can use 200*(avg price of the stocks) as the initial investment. That is 10,000 which yields 3000 over a period of 5 yrs(see graph below).



The following are the places where the long and short positions were taken:

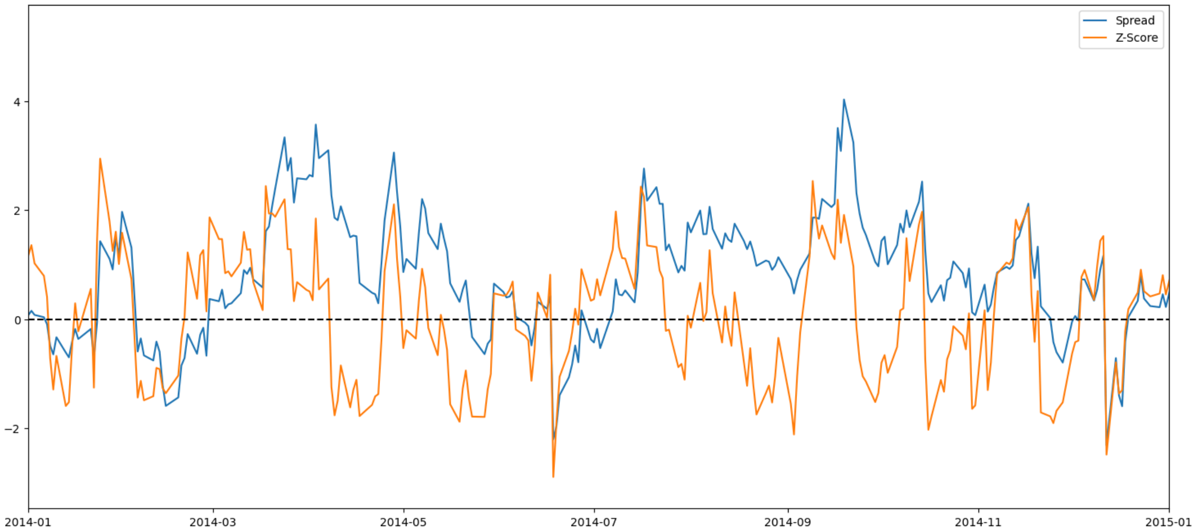


Here we can observe that the strategy in general manages to buy the spread at low price(spread) and sell high. which gives a sanity check on our calculations.

Final Project:

The final project build upon the assignment 2 which uses kalman filters for making trading decisions. Here some additional heuristics are used that mitigate the risks involved in pairs trading and achieve a good risk adjusted returns.

The following is a graph of Zscore of spread(calculated only using the closing prices of previous days) and actual spread:



The zscore is calculated using a window of 2*halflife of spread. Here the spread and zscore can be seen to be similar in terms of the +ve and -ve deviations from the mean. Hence the trades made are correct with respect too the spread as well.

Here again similar to assignment 2, trading signals are generated using the zscore of the spread which is dynamically calculated based on the previous data using kalman filter. Now since this spread is mean reverting, trading signals are generated by checking its deviation from the mean. A long entry is position is taken when the zscore of the spread moves from -entryZscore towards 0. And this position is exited when the zscore moves from -exitZscore towards 0. Similarly short entry is taken when the zscore of the spread moves from +entryZscore towards 0. And this position is exited when the zscore moves from +exitZscore towards 0.

Similarly taking the entryZscore to be (4/3)*entryZscore [the 4/3 found empirically] and then if the above conditions again hold then the volume of the long or short positions is doubled. This ensures that if the spread deviates from the mean too much then we have a larger number of stocks in the long or short position as the spread is more likely to revert to mean now.

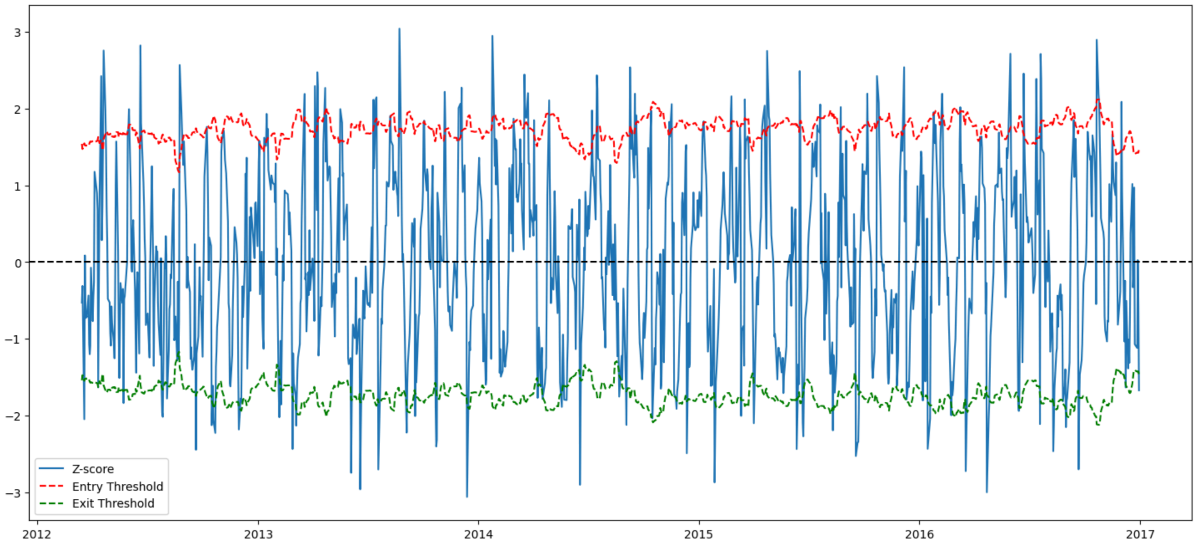
The long and short positions are taken with the hedge ratio as the beta value obtained form kalman filtering which changes dynamically.

A transaction cost of 0.01% is added per transaction.

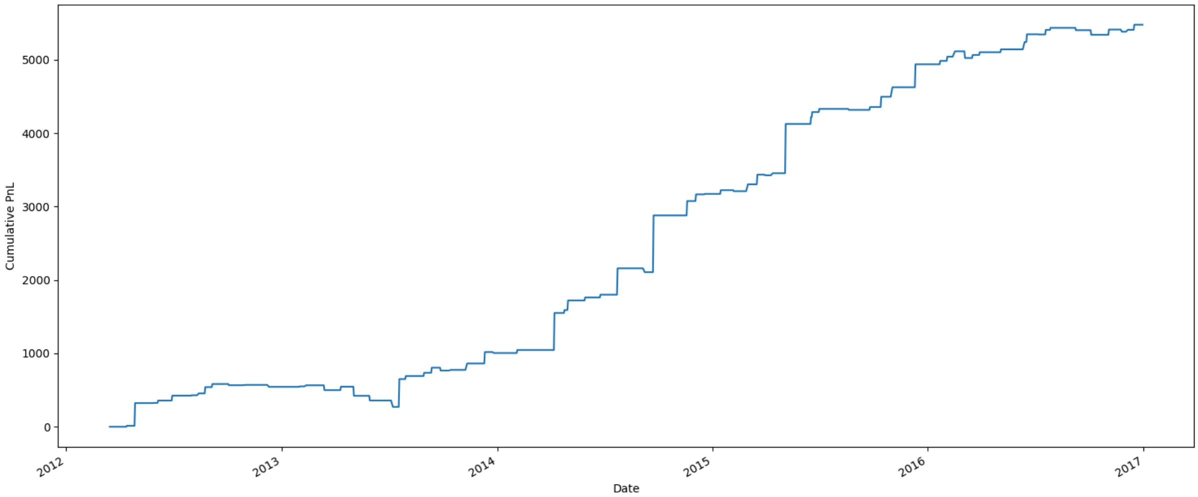
The trade_pnl is updated at every exit of position in the market. Hence the cumulative pnl is blocky as it gets updated only at certain intervals (exit of position in the market).

Here the entryZscore = (variance of 40 day window of the zscore)*1.5
and exitZscore = 0

This dynamically changing threshold of entryZscore helps in capturing the market volatility. When the variance is high we have higher threasholds for trading which decreases the amounts of trades in a volatile market and thus reduces the exposure to market risks. This also improves the pnl gained. The following graph shows the dynamically changing threshold:

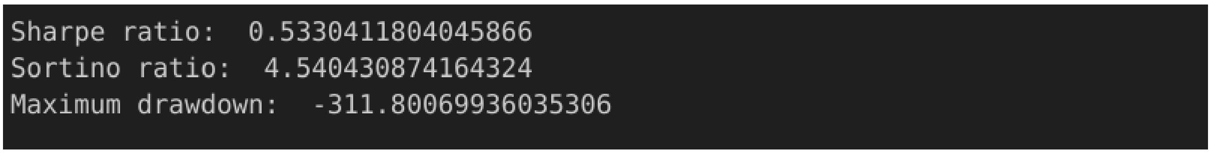


The following is based on a max trade volume of 100*2. So for the estimating % returns we can use 200*(avg price of the stocks) as the initial investment. That is 10,000 which yields 6000 over a period of 5 yrs(see graph below). With a static threshold of 2 for entryZscore gives a return of 3000 as opposed to 6000 now.



We can see that this strategy manages to get good profits.

The risk-adjusted returns for this strategy are:

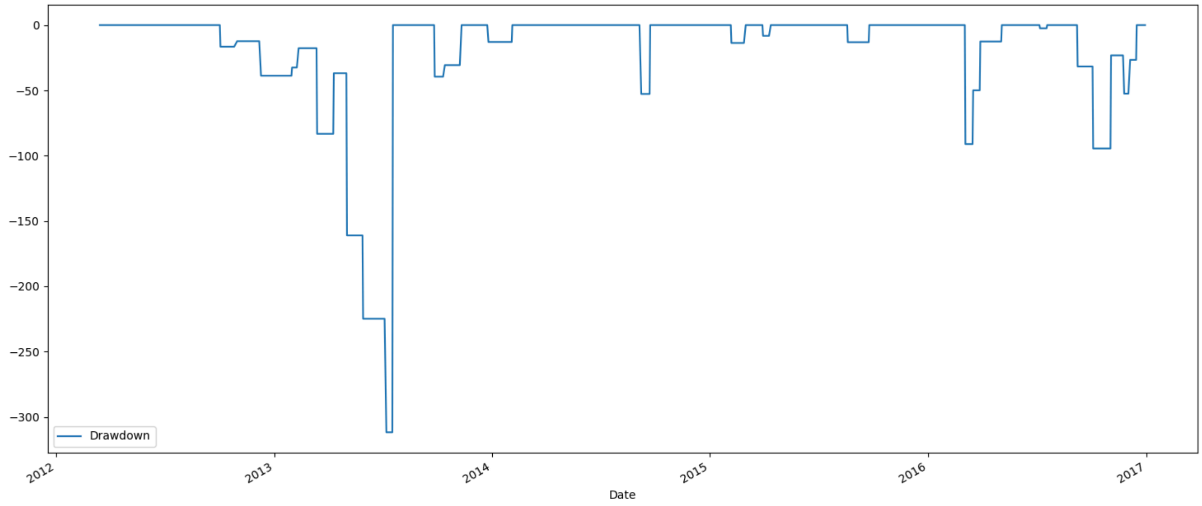


```
Sharpe ratio:  0.5330411804045866
Sortino ratio:  4.540430874164324
Maximum drawdown:  -311.80069936035306
```

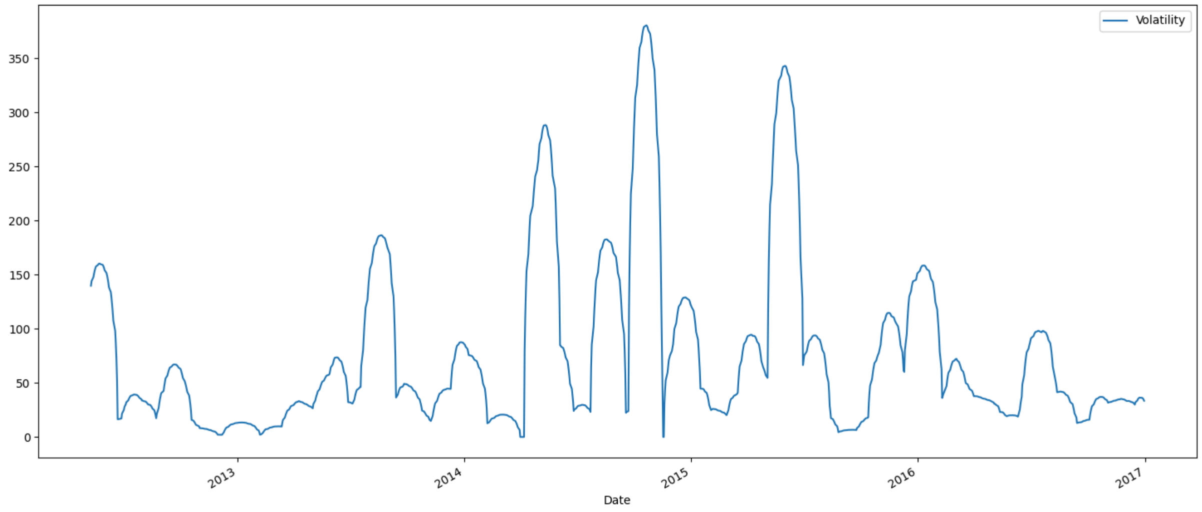
Here the sharpe ratio is 0.52 but that is due to way in which the PnL is calculated. The PnL is calculated when the position in the market is exited. Due the this the PnL is updated (the actual PnL gets reflected) at every exit and stays constant in between. This causes the percentage change in the cumulative PnL to be high at these points and hence increasing the variance. Most of the changes are positive though that is profit is earned when a position is exited. Using sortino which only reflects the negative returns we can see that that we have high risk adjusted returns

The drawdown graph again reflects the same thing with it being blocky due to the PnL being updated only at exits.

The max drawdown of 336 is reasonable considering the over all profit of 6000.

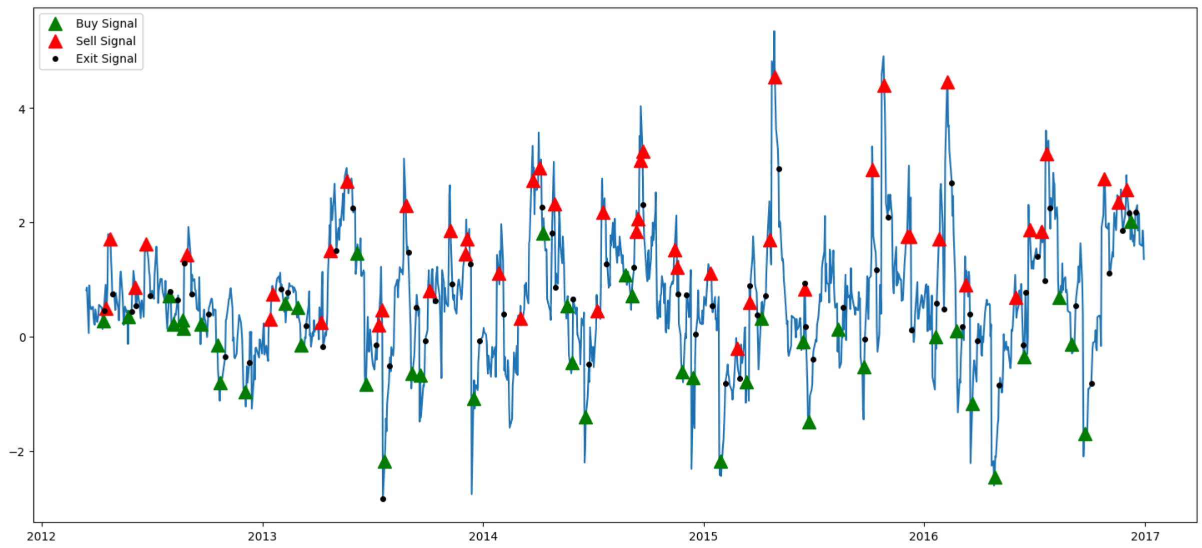


The following is the volatility graph for the PnL:



This is the volatility in the pnl using a 40 day window. The highest value of 400 standard deviation is reasonable considering the profit.

The places where the trades are made on the MSFT and ABDE stock data:



Here we can observe that the strategy in general manages to buy the spread at low price(spread) and sell high. which gives a sanity check on our calculations.

All the above analysis was done by fixing the tradable volume to 200. This can be different based in risk appetite and investment ability. This can also be dynamically changes using some rule to further get better risk adjustment and returns.
To summarize, we have achieved good risk adjusted returns by implementing kalman filtering for pairs trading. Using dynamically changing threshold for trading signals as well as using 2 different levels for long and short position with larger position taken for higher thresholds to increase profits and reduce risk.