Machine Learning Pairs Trading using Google Tensorflow

Andrew Brim

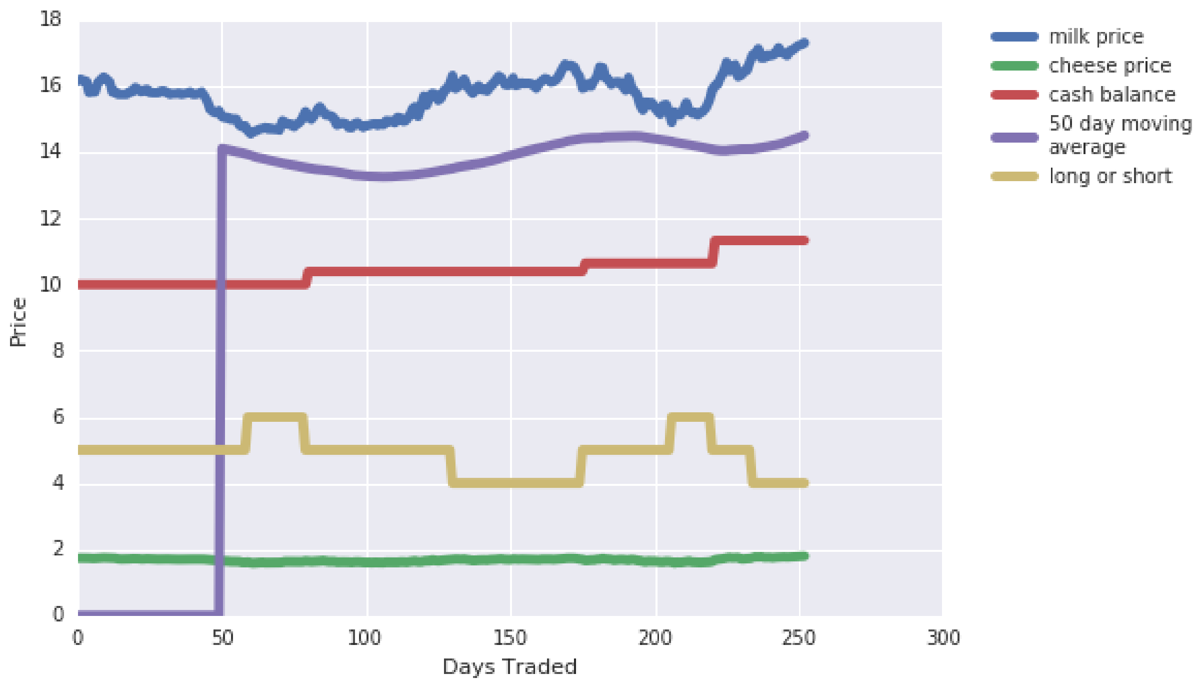
Dec 14th, 2016

1 Introduction

Many financial trading strategies are completely mathematical and/or statistics based. Wouldn't a Machine Learning System be a perfect fit, to create and use mathematical trading strategies? The idea of Machine Learning financial trading is nothing new [6]. This paper simply takes an approach using some of the latest machine learning techniques including Google's Deep Neural Net, Tensorflow [7]. It also applies a simple pairs trading, strategy to the cheese and milk futures markets to see if it can be profitable.

Pair trading looks at the relationship of financial instruments or commodities, in our case cheese and milk futures, and trades one based on the movements of the others. For example, in our cheese and milk scenario, when we see the prices of cheese and milk futures much closer than their average, we take a long position in the higher price and a short position in lower price, expecting that as the prices diverge back to the mean, then one or both positions will be profitable. If you understand the co-integration relationship, you can better predict price movements based on historical co-integrated behavior [4]. In other words, we will be able to watch the prices of milk and cheese futures, and predict in some instances, price movements based on their co-integrated relationship.

We will use Google Tensorflow to create a layer of perceptrons in a Neural Network. It's true that this is a very simple neural net, and perhaps not taking advantage of the full capabilities of Tensorflow. Still, we will show that Tensorflow will be very good at picking out possible pairs of commodities which are good candidates for pairs trading. Essentially each perceptron will test whether or not it sees co-integration, and at what level, in the pair it is assigned. The next layer will perform a trading test to see whether or not, a simple pairs trading strategy is profitable for that pair. Below is an example of Milk and Cheese future prices, which we detected as highly co-integrated by Tensorflow, then tested with a pairs trading strategy and found it to be profitable.



Above you can see the price of milk and cheese over the last 12 months, and the results of our pairs trading algorithm. The blue and green lines represent the prices of milk and cheese. The purple line shows the moving average of the price difference between milk and cheese, which is used to generate buy and sell signals. The red line is our cash balance beginning at $1000.00, scaled down 1/10 and then overlayed on the graph. The yellow line is a buy or sell signal, also overlayed on the graph to show buys (line up) and sells (line down). Essentially when we see the moving average line coming down, we buy expected the price to move back up to the average, and vice-versa when the moving average lines moves up.

The period of 12 months was to found to optimal by Tensorflow based on its Augmented Dickey-Fuller co-integration test value, and the trading strategy was then applied. The perceptron in Tensorflow uses the ADF tests for co-integration at a 95% confidence interval. Or at least reject the null hypothesis that co-integration does not exist.

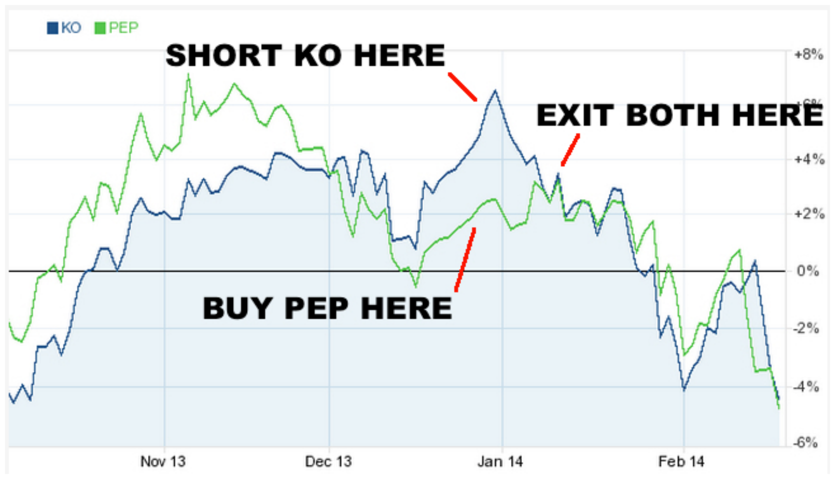
The technologies used include: Google Tensorflow, Google Datalab[8] to run Tensorflow and host [9]jupyter notebooks used for testing. python 2.7.9 on Linux Ubuntu. The milk and cheese prices were supplied by websole.barchart.com via json api.

2 Pairs Trading

As mentioned, Pair trading is where you watch the co-integration level between two (or more) financial instruments or commodities. If they are highly co-integrated and their prices drift too far apart, then it is likely that prices will correct.[1] You can short the high price, and long the low price and one or the other will be profitable. Essentially in pairs trading, you are observing deviations away from the mean of the difference of the two prices in the pair, and placing simultaneous long and short positions when they occur [2].

A good visual for a co-integrated pair is a drunk man walking a dog through a park. The drunk man performs a random walk through the park, and the dog follows performing a random walk of its own. While the two may appear random, they always move together because of the leash. We are looking for pairs that move together. In this case, cheese and milk. If cheese and milk prices are co-integrated, like the drunk and the dog, then we can expect prices that are far apart to converge back to a mean. Likewise if the prices are too close, they will diverge back to the mean. The mean gives us a baseline that we can use to determine trading opportunities.

In the example of milk and cheese, Let's say the average price of milk is around \$16.00 and the average price of cheese is \$1.75. This makes an average difference of \$14.25, a good baseline to make trading decisions. Here we will use a ceiling of 107% (or \$15.25) and a floor of 93% (or \$13.25). If the average difference of milk and cheese goes above \$15.25, then we say that these co-integrated prices are too far apart, and we expect them to converge back to \$14.25. If the average difference goes below \$13.25 then when expect the prices to diverge back to \$14.25. When we detect the prices are too far apart, we take a short position in milk, and a long position in cheese, expecting that when the difference in price converges back to mean, the price of milk will have gone done, and cheese up. This makes our strategy possible. This strategy is market risk neutral, because we don't care what the rest of the market is doing. We simply care about the current price difference between milk and cheese, the average price difference between milk and cheese, and the co-integration value.



[5] Above you can see a good example of prices diverging from the mean, creating a signal to short the top price, and go long the bottom price. In this example, as the higher price goes down and the lower price goes up, both positions become profitable [3]. Even if both positions are not profitable it is likely that at least one position will be, as the prices converge back to the price difference mean. This is where Machine Learning can help us. Testing for co-integration is statistical, and Tensorflow can allow us to do this more efficiently and on a large scale

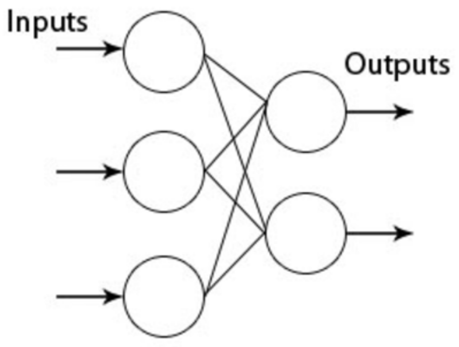
3 Machine Learning Approach

To pairs trade, we need co-integrated pairs. Preferably, as highly co-integrated as possible. To use the visual from earlier, we want to find a drunk/dog pair with a strongly elastic leash the we can dependably predict will continue to come back to the difference mean. This is where a Neural Net, like Tensorflow can help. A Tensor is simply an array of numbers. The neurons, or perceptrons, create a neural network where the inputs and ouputs to the neurons are all Tensors. Essentially, Tensorflow, is a framework to build an NN with numerous layers of perceptrons. The perceptrons can be programmed to perform any task. In our case the perceptrons will test for co-integration. Additionally they can take any Tensor as input, and they can easily scale as the input datasets become very large.

Here we will create the first layer in our Tensorflow NN to calculate co-integration. Essentially each perceptron will receive a pair of milk prices and cheese prices. Since all the prices are cheese and milk futures, for testing purposes, we used a variety of time frames between one month and four years. After each perceptron receives its data (a Tensor), it runs an Augemented Dickey-Fuller co-integration test. ADF at a 95% confidence will tell us, whether or not we can with 95% confidence reject the null hypothesis that co-integration does not exist. While it is true that this does not necessarily say the co-integration does exist. For our purposes here, we will consider this sufficient to move forward under the assumption that co-integration does exist.

The results from the perceptrons indicate that longest time frame tested, in this case 48 months, does exhibit the highest co-integration. However, once we get to 12 months the co-integration is a very small change as we move out to 48 months. In other words, 1 year's worth of milk and cheese prices has higher co-integration that any time frame shorter and very similar to any time frame longer. Therefore, for testing purposes, 1 years worth of cheese and milk prices will do well.

Below you can see a simple representation of our NN. The first layer of perceptrons takes as inputs cheese and milk prices. The output is then past to the second layer which runs a trade test. It should be said that here we are running the test for 1 year. Since most of the inputs are subsets of 1 year, running the trade simulation test for 1 year give us all the same performance results for any time frame shorter than 1 year. For our purposes here, this will suffice. We are intending to show machine learning can give us the tools to find good pairs for pair trading, and test see how well they perform in trading strategies. Further on, we will show that Tensorflow does work well for this purpose.



4 Experiment/Process

As mentioned, we are using Google Tensorflow running on Google Datalab to build a NN. Google Datalab is essentially a server which comes with all Google Machine Learning libraries, including Google Tensorflow. Technically you do not need to run Tensorflow within the Google Datalab environment, but it seems to run much better, since Google Datalab comes with all the necessary libraries for Tensorflow. After we install Google Datalab, we are ready to begin building our own Tensorflow environment.

Firstly, we need to get cheese and milk prices from websole.barchart.com via their json api. This data is then cleaned and stored in nd arrays. The data pulled from websole.barchart.com is requested in increments which will be tested for co-integration in our perceptrons in the NN. Initially we test for co-integration in cheese and milk for the time frames: 1 month, 2M, 3M, 4M, 5M, 6M, 9M, 12M, 24M, 36M and 48M. As we will see in the results section, the co-integration value approches -2.86 (between -5 to 5 is desired) as we get to 12 months. Longer than 12 months, and stays very close to -2.86 that as the time period increases. It seems as though 12 months is enough time to establish maximum co-integration for cheese and milk futures prices.

In Tensorflow a perceptron, or any neuron where the functionality is customized, is easily created. Essentially, we create a function to be performed. In our environment the function is appropriately named "perceptron". We then create a session for the perceptron to run. The inputs and outputs of the perceptron are ndarrays. Ndarrays are fast because they are compiled in c code, for efficiency. By requiring ndarray input and output, Tensorflow ensures performance as the NN gets larger. The inputs are ndarrays of milk and cheese prices for the given time frame. The output is an ndarray with the ADF values for the confidence levels 1%, 5% and 10%. An example perceptron output would be: [-3.44363, -2.8673966, -2.5698893]. This is interpreted as a 1% confidence value of -3.44, a 5% confidence value of -2.86, and a 10% confidence value of -2.56. Again, any value between -5 and 5 denotes we can reject the null hypothesis that co-integration does not exist. The example perceptron output suggests the pair in question can be treated as co-integrated.

After the pairs are tested for co-integration, we proceed with the trading test. A future work would be to create a multi-layer NN with multiple types of pairs, and trading tests comprising the layers of the NN. Here, for testing purposes, we simply perform a trading simulation test for 12 months worth of cheese and milk prices.

5 Trading Simulation

The trading simulation test uses real prices, and allows us to test if a simple pairs trading strategy would be profitable over the past year. The strategy keeps track of the moving average of the change between milk and cheese futures prices. As you can see from the graphic in the Introduction, the average price difference between cheese and milk is an estimated $14.00. The moving average of the difference, changes as the price changes and we keep track of this. Any time the price difference increases past 7% over the moving average, or decreases 7% below the average we simultaneously go long and short in the opposite direction, expecting the price to correct making our positions profitable.

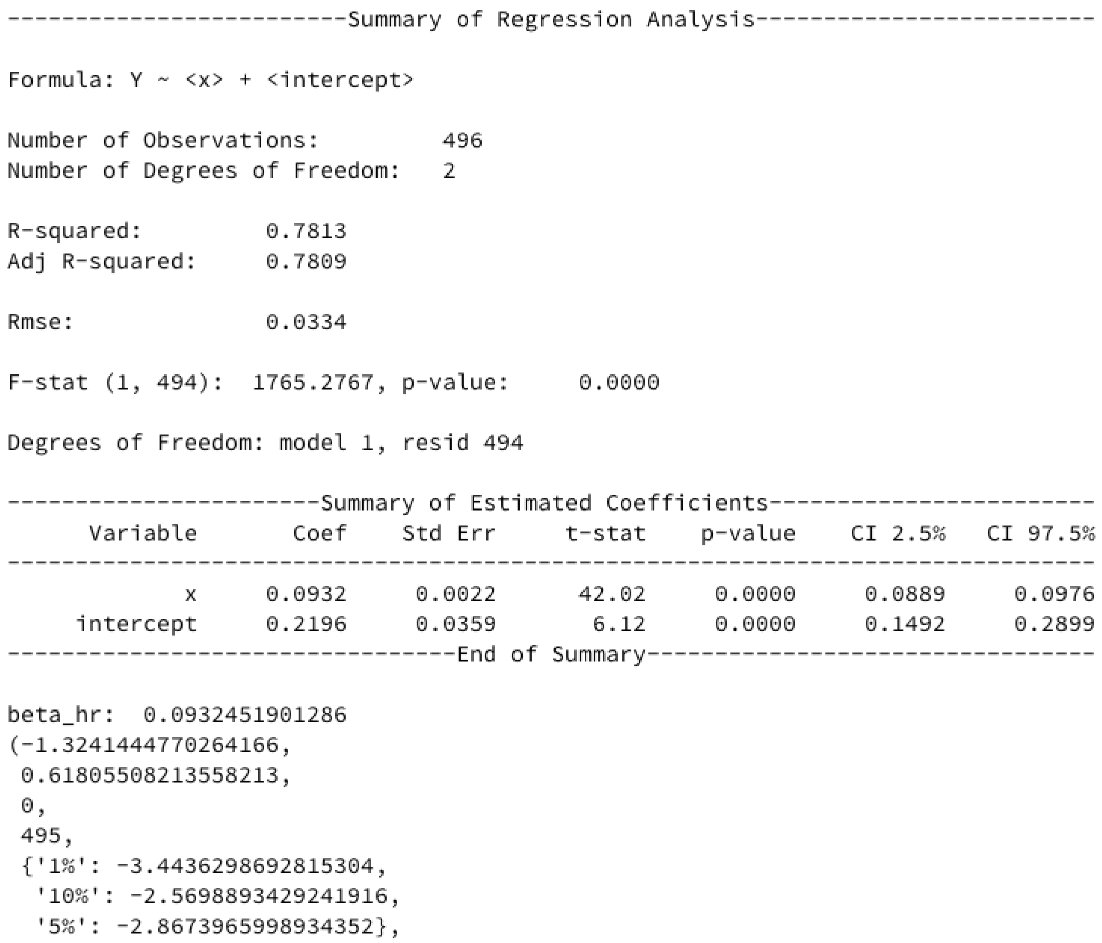
Looking at the graphic in the Introduction further we can make a few more observations. The yellow line is not a price, but a signal showing whether we are long or short milk. Cheese was excluded from this visual, as the price moves are so small. As you can see, when the price of milk is moving down initially, we enter in long position for milk, expecting the price to rise. The price does rise back to the moving average and we sell, creating profit as shown in our cash balance. The price of milk continues to rise, until it is above 7% over the moving average. Then we short milk, as shown by the yellow line. Once the price corrects, or return back to the moving average we buy milk, closing the short contract, and we can see our profit in the cash balance.

This cycle repeats again twice more, one long and one short position. As you can see we are market risk neutral, because we only worry about the price of this pair and their co-integrated movements. We don't worry about the movements of the market in general. Even if the market rises, the price of milk and cheese would both rise. If the market declines, the prices of milk and cheese would decline. This is fine, even expected, and as long as the co-integrated relationship stays in tact the strategy will be profitable. Obviously, if there is a systemic change in the relationship between cheese and milk, this could also change the co-integrated relationship. In a real-world trading scenario you would constantly test and re-test co-integration to verify the relationship is stable. In our 12 month simulation, this trading strategy produces a cash balance increase from $1000.00 to $1133, or an portfolio increase of 13%.

6 Results

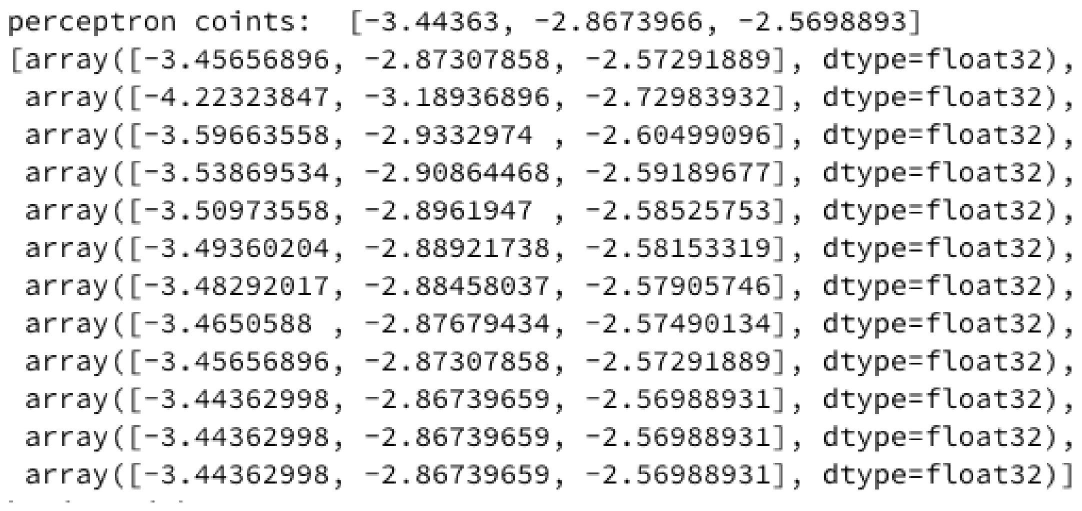
The scope of this project was to verfiy, that for cheese and milk futures prices, a simple Tensorflow NN can be used to detect highly co-integrated pairs, and even test that relationship with a simple pairs trading strategy. We have seen that a perceptron performing ADF tests for co-integration, can be effective and scalable. As the demand for testing more and more pairs grows, Tensorflow can easily create more perceptrons, and more layers in the NN to handle more data. In this experiment, we limited ourselves to a simple pairs trading strategy to verify the usability of the co-integrated pair in a trading environment. However, using Tensorflow, you could easily more customized types of trading simulation tests. Really, you could add any types of tests, or supplemental data you desired as inputs to the NN, and they would all scale to the size of your problem. It is apparent Machine Learning techniques and mathematically based trading strategies will continue to be a effective marriage.

Here is sample output from a perceptron performing a co-integration test on 12 months of cheese and milk futures prices. As you can see, as part of the ADF test, it is performing regression on the pair, then testing the residuals for a unit root. At the end of the output you can see the ADF test value for 5%, indicating we can with a 95% confidence interval, reject the null hypothesis that co-integration does not exist.



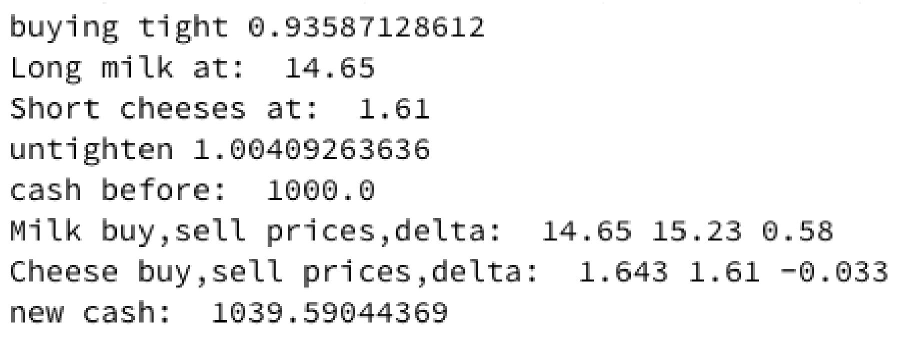
```
-------------------------Summary of Regression Analysis-------------------------

Formula: Y ~ <x> + <intercept>

Number of Observations:         496
Number of Degrees of Freedom:   2

R-squared:         0.7813
Adj R-squared:     0.7809

Rmse:              0.0334

F-stat (1, 494):  1765.2767, p-value:     0.0000

Degrees of Freedom: model 1, resid 494

-----------------------Summary of Estimated Coefficients------------------------
      Variable       Coef    Std Err     t-stat    p-value    CI 2.5%   CI 97.5%
--------------------------------------------------------------------------------
             x     0.0932     0.0022      42.02     0.0000     0.0889     0.0976
     intercept     0.2196     0.0359       6.12     0.0000     0.1492     0.2899
---------------------------------End of Summary---------------------------------

beta_hr:  0.0932451901286
(-1.3241444770264166,
 0.61805508213558213,
 0,
 495,
 {'1%': -3.4436298692815304,
  '10%': -2.5698893429241916,
  '5%': -2.8673965998934352},
```

Below you can see the ndarray output from Tensorflow. Each perceptron returns an ndarray consisting of 1%,5%, and 10% values. Here are the results from all the time frames tested: 1M, 2M, 3M, 4M, 5M, 6M, 9M, 12M, 24M, 36M, 48M. The middle value, 4th from the bottom, is -2.87. This is the value for 12 months. As previously mentioned it takes about a year's worth of prices to reach this value, and then it stays there from 1 year to 4 years. Indicating, for cheese and milk, 12 months is enough time to observe the highest co-integration within the pair.



```
perceptron coints:  [-3.44363, -2.8673966, -2.5698893]
[array([-3.45656896, -2.87307858, -2.57291889], dtype=float32),
 array([-4.22323847, -3.18936896, -2.72983932], dtype=float32),
 array([-3.59663558, -2.9332974 , -2.60499096], dtype=float32),
 array([-3.53869534, -2.90864468, -2.59189677], dtype=float32),
 array([-3.50973558, -2.8961947 , -2.58525753], dtype=float32),
 array([-3.49360204, -2.88921738, -2.58153319], dtype=float32),
 array([-3.48292017, -2.88458037, -2.57905746], dtype=float32),
 array([-3.4650588 , -2.87679434, -2.57490134], dtype=float32),
 array([-3.45656896, -2.87307858, -2.57291889], dtype=float32),
 array([-3.44362998, -2.86739659, -2.56988931], dtype=float32),
 array([-3.44362998, -2.86739659, -2.56988931], dtype=float32),
 array([-3.44362998, -2.86739659, -2.56988931], dtype=float32)]
```

The trading simulation output shows the long and short positional movements as the simulation detects trading opportunities. Here you can see the simulation detects a "tight" scenario, or the price difference is converging away from the moving average. It goes long the milk at price $14.65, and short cheese at $1.61. You can see in the logs, once the price difference moves back to the average the simulation "untightens" and the positions are closed resulting in a new cash balance of $1039.59, and a profit of $39.59.



```
buying tight 0.93587128612
Long milk at:  14.65
Short cheeses at:  1.61
untighten 1.00409263636
cash before:  1000.0
Milk buy,sell prices,delta:  14.65 15.23 0.58
Cheese buy,sell prices,delta:  1.643 1.61 -0.033
new cash:  1039.59044369
```

7 Future Work

The natural next step for this approach would be to get more pairs for testing. Pairs for trading can come from the stock market, bond market, commodities, FX currency pairs, derivatives markets including: rates derivatives, credit derivatives, and other futures and options. If you can think of a possible pair, and get inputs for the data, you could test it in a Tensorflow NN.

Additionally, there is much room for expanding the types of tests possible on those pairs. You could add more customized pairs trading strategies, or other tests entirely. For example, for stocks, you could add layers in the NN to receive as inputs the company's financial health information or media feeds. Again, if you can think it, you can create the perceptrons and layers to add it to a Tensorflow environment.

[1] Engle, Robert F., Granger, Clive W. J. (1987) "Co-integration and error correction: Representation, estimation and testing", Econometrica, 55(2), 251–276. done

[2] 2 Stocks For The Ultimate Pairs Trade
http://www.nasdaq.com/article/2-stocks-for-the-ultimate-pairs-trade-cm340936

done [3] "Pairs Trading: Correlation", Investopedia, Investopedia LLC 2016
http://www.investopedia.com/university/guide-pairs-trading/pairs-trading-correlation.asp

[4] "Guide to Pairs Trading" Investopedia, Investopedia LLC 2016 http://www.investopedia.com/university/gui
pairs-trading/

done [5] Image "pairs" Pairtrade Finder Professional Stock Trading Software
http://www.pairtradefinder.com/

done [6] A Pairs Trading Strategy for GOOG/GOOGL Using Machine Learning
http://cs229.stanford.edu/proj2015/028$_{r}eport.pdf$

done [7] Google Tensorflow, An open-source software library for Machine Intelligence https://www.tensorflow.org/get_started/

done [8] Google Datalab, An easy to use interactive tool for large-scale data exploration, analysis, and visualization
https://cloud.google.com/datalab/

done [9] Jupyter Notebooks, Formerly iPython Notebook,
https://jupyter.readthedocs.io/en/latest/index.html

8 Code

The code can be viewed at github:
https://github.com/abrim24/corrTrader/blob/master/fuzzyleader.ipynb