Introduction

The paper 'Nonlinear Forecasting of the Gold Miner Spread" investigates the relationship between the US gold mining industry and gold bullion. To do this, they look at the spread between the Market Vectors Gold Miners ETF (GDX) and GLD. GDX is composed of 31 mining stocks weighted by market cap, replicating the NYSE Arca Gold Miners Index (GDM), while GLD is the SPDR Gold Trust. Gold miners trade this spread as a hedge against inflation; GLD is the long leg and GDX is the short.

The paper seeks to test the hypothesis that there are short term fluctuations in the spread that can be exploited with appropriate models. Their data consists of daily closing prices for both ETFs, spanning 05/23/06 - 06/30/2011. Models tested include cointegration, ARMA, two variants of neural networks, and a genetic programming algorithm. They then backtested a trading system based on these models, all of which had annualized returns greater than 20%.

The success of this paper, as well as the ease of acquiring similar data for other assets, prompted us to look for another industry where the same principles might be applied. The United States Oil Fund (USO) and the S&P Oil & Gas Exploration and Production ETF (XOP) can be thought of as proxies for WTI crude price and oil production companies, respectively. USO is the most liquid crude ETP on the market. Because of this, we can expect the price to be reasonably close to the crude spot price. However, it uses front-month contracts to get exposure to crude, which can lead to contango in some situations. Another option, OIL, holds longer-dated contracts, but has a fraction of the volume of USO. Unfortunately, there is no way to get exposure to spot oil, as GLD gives us for the price of gold. XOP is an equal-weighted index of 63 oil production companies.

We hypothesize that the spread between these assets might be modeled in a similar way to what was done in the paper. We will focus on the cointegration approach - popularized by Engle and Granger - and a neural network, specifically a multilayer perceptron.

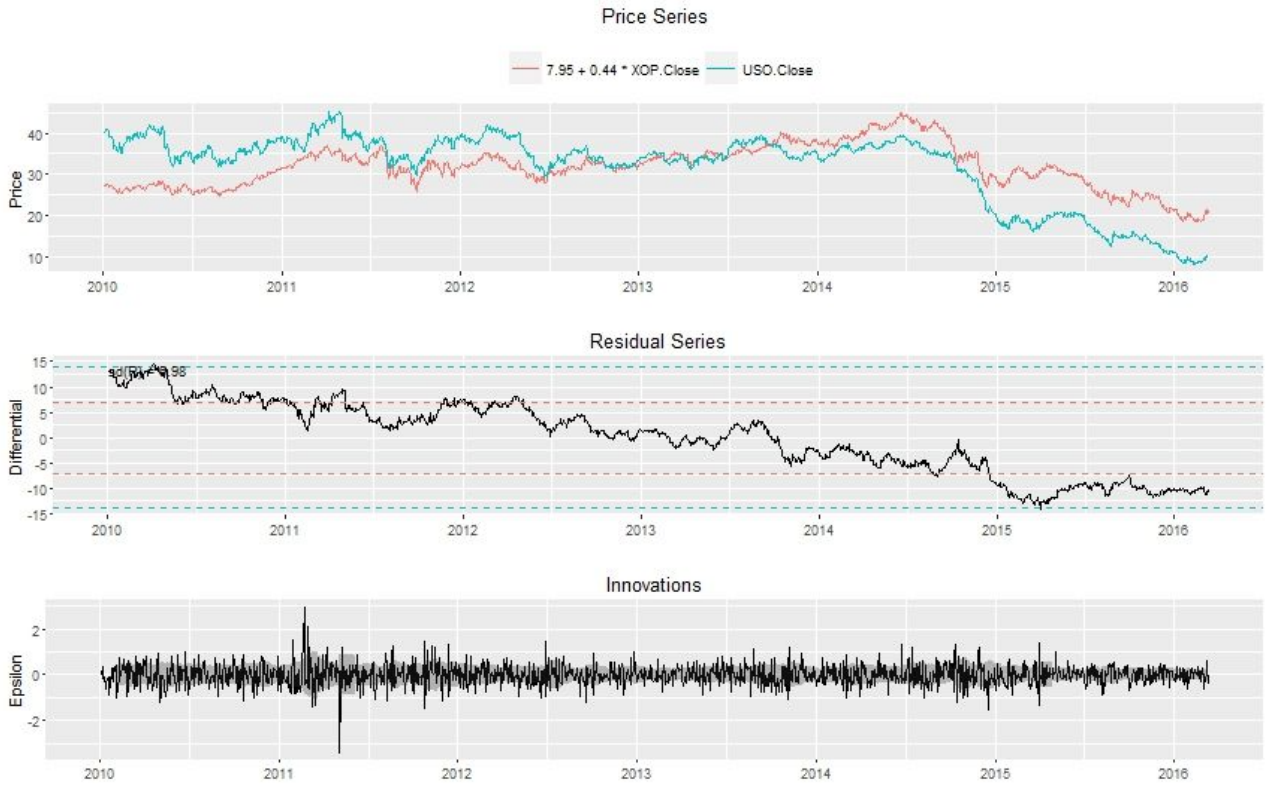
Tools and Data

Daily price series for USO and XOP was obtained from Quandl, a free provider of such data. Data spans 01/01/2010 - 03/01/2016. R Studio was used for data cleaning and analysis. R was chosen because of familiarity with the language and excellent support for many statistical procedures through packages. Code will be attached to the end of this document.

Cointegration

For price series A and B to be cointegrated, there exists a stationary time series that can be created from some linear combination of A and B. Once tested for, we can then take advantage of the properties of that residual time series, specifically the mean and the variance. Over time, we can expect the price to return to the mean if they diverge.

Cointegration is easily tested through the ‘egcm’ package in R. The Augmented Dickey-Fuller



test (ADF) is used, and gives us a p-value of 0.0068, well below an alpha threshold of 0.05. The plot below shows the two time series and their residual series.

We can close this part of the analysis, having successfully proven the existence of a stationary residual series. This was a promising first step in applying the techniques from the paper to a new spread.

Neural Networks

We move on to a more advanced nonlinear modeling technique described in the paper. The variant that we will be looking is the multilayer perceptron model (MLP).

An MLP can be broken down into three layers: the input (1 layer), hidden (=>1 layers), and output (1 layer) layers. Each of these layers has a configurable number of nodes. For example, the number of nodes in the input layer is determined by the number of input variables. For the hidden layer, the general consensus is that 1 layer usually fits the requirements of most problems, so that is what we will stick with. The number of nodes in this layer is usually recommended to be between the number of nodes in the input layer and the number of nodes in the output layer. The output layer has 1 node, since it will be the prediction of the spread price for that day.

The nodes are connected to each other through weighted activation functions that map the input of a neuron to the output. When we train the network, we adjust the weights that go into these functions. In our case, we used a 75/25 split for testing/training.

In the case of nonlinear machine learning systems, it is common to throw a bunch of inputs into the model and see which ones 'stick'. Nonlinear relationships, by definition, would be difficult to see just by looking at the time series plots of inputs. The inputs that ended up being used in the final model were 4 days of lagged spread returns, volume, and a proxy for the 'market' (SPY). Other inputs such as major energy ETFs, volatility, and macro ETFs were also tried, but did not have much of an impact. 4 hidden nodes were used.

Metrics such as mean squared error couldn't really be used in this case since we didn't create any other models with an MSE to compare with, but we could measure correct directional change (CDC). We averaged a 54% CDC: a little higher than the the 52.11% CDC that the MLP model from the paper recorded. This could be attributed to the fact that the timeframe used in the paper includes the financial crisis, a period where it might have been difficult to model the market.

Conclusion

We successfully applied two of the models described in the paper to a novel spread in the oil industry. However, the mechanics of this spread are a little different from the GDX/GLD spread; as mentioned at the beginning of the paper, it is not possible to simulate holding the spot price of oil as it is with gold through GLD. In the future, it might be a good idea to take into some of the effects of this (contango, deviations from the spot price).

Another remark is that recently, the price of oil has dropped to a level below what people considered the most pessimistic predictions a year ago. This has led to a large sigma deviation in the cointegrated residual series (visible in the chart above). Therefore, it is currently hard to say whether the models used are able to remain robust in changing market regimes.

A limitation that we had with this paper was the inability to find free intraday data for the securities that we targeted. Intraday price movement could prove to be a better input for our model. Further work might include optimizing the neural network (parameters, different variants) and implementing a trading strategy around these models, as was done in the paper, to compare metrics such as Sharpe ratio and annualized returns.

Overall, this was an educational exercise in taking a concept from the literature and applying it in a new direction. R provided us with the ability to prototype different models quickly and efficiently to give us some significant results in a short amount of time.