**Capstone 3 Final Report:**

**Constructing a futures trading strategy which combines custom asymmetric objective functions and direct position optimization**

Matt Kosik

**Problem Statement:**

The implementation of any trading strategy can be broken down into two separate processes: forecasting and position sizing. Forecasting involves predicting the future prices of a particular asset or combination of assets based on the certain information set known at the present. The position sizing function takes the forecasts as an input, combines them with investment goals (such as risk and return targets and available capital), and then outputs a buy or sell signal with an associated magnitude that represents the desired position size in the assets to be traded.

I am proposing a new procedure that improves upon both steps of this process. First, I will introduce and implement a custom asymmetric objective function in the forecasting step which incorporates real world consequences of errors into predictions. Secondly, I will incorporate a method that combines the forecasting and positing sizing steps into a single direct position optimization. This is akin to training forecasting and posting sizing models simultaneously rather than in a step wise manner. The overall trading strategy has highly attractive out of sample results across a variety of measures and looks promising for further research.

**Data:**

The data used in this project are all historical asset returns. I chose to create this trading strategy in the futures market because it has a significant amount of historical data, is heavily studied in academic literature, and is extremely liquid. All data can be found in the *pysystemtrade* python package.

I use daily returns data from the period 1-Jan-2000 to 1-Jan-2020 across the following futures markets:

FX:

*AUD, EUR, GBP, JPY, MXN*

Commodities:

*Corn, Soybean, Wheat, Crude Oil, Natural Gas, Lean Hog, Live Cow, Copper, Palladium, Platinum, Gold*

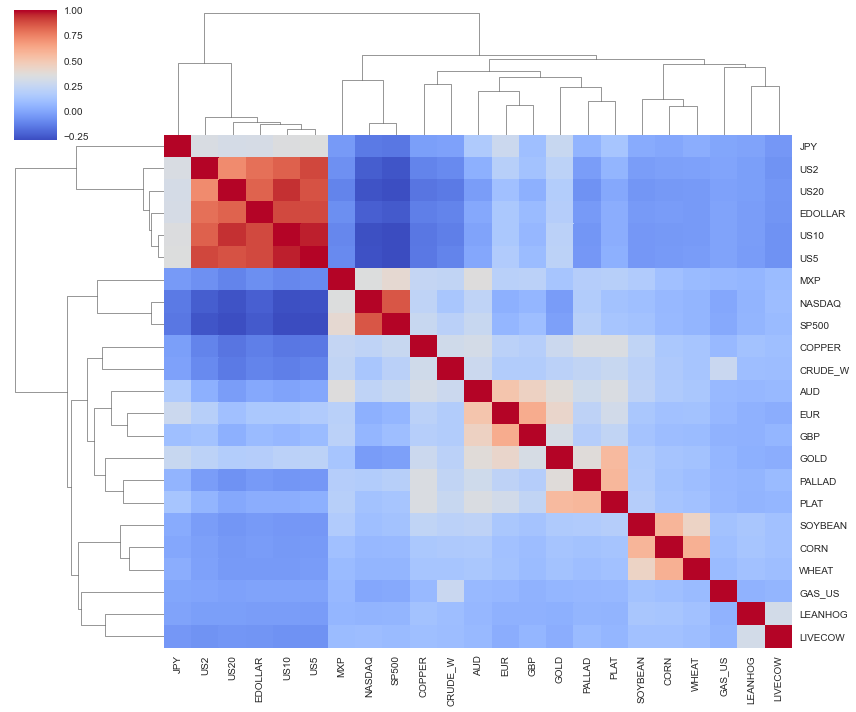
Equities:

*Nasdaq, S&P500*

Rates:

*Eurodollar, US2y, US5y, US10y, US20y*

As shown in the dendrogram correlation map below, there are small pockets of high correlation (within rate and equity products or similar commodities) but the assets chosen should generally provide strong diversification benefits.



As common in futures trading strategies, I will be scaling all returns to an ex-ante target annualized volatility of 15% prior to trading.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Future** | **Ann. Mean** | **Ann. Std.** | **Ann. Sharpe** | **Future** | **Ann. Mean** | **Ann. Std.** | **Ann. Sharpe** |
| **AUD** | 0.056216 | 0.144384 | 0.389349 | **MXP** | 0.057817 | 0.143863 | 0.401891 |
| **COPPER** | 0.0389 | 0.14395 | 0.270233 | **NASDAQ** | 0.103229 | 0.143178 | 0.720986 |
| **CORN** | -0.017191 | 0.143293 | -0.119973 | **PALLAD** | 0.079726 | 0.14506 | 0.549608 |
| **CRUDE\_W** | 0.091488 | 0.144402 | 0.633565 | **PLAT** | 0.0721 | 0.144754 | 0.498087 |
| **EDOLLAR** | 0.10207 | 0.144187 | 0.707903 | **SOYBEAN** | 0.071745 | 0.143631 | 0.499507 |
| **EUR** | 0.008412 | 0.144293 | 0.0583 | **SP500** | 0.088901 | 0.142701 | 0.622985 |
| **GAS\_US** | -0.040018 | 0.144611 | -0.276729 | **US10** | 0.112837 | 0.14426 | 0.78218 |
| **GBP** | 0.013024 | 0.1457 | 0.089392 | **US2** | 0.090295 | 0.142302 | 0.63453 |
| **GOLD** | 0.093991 | 0.142983 | 0.657355 | **US20** | 0.105059 | 0.144551 | 0.726799 |
| **JPY** | -0.071964 | 0.175916 | -0.409083 | **US5** | 0.106349 | 0.14388 | 0.739152 |
| **LEANHOG** | 0.052851 | 0.144687 | 0.36528 | **WHEAT** | -0.011832 | 0.144365 | -0.08196 |
| **LIVECOW** | 0.017509 | 0.144538 | 0.121138 |  |  |  |  |

Features:

To keep the model as simple as possible, I will only be using traditional trend and momentum features based on the historical price series.

*Vol normalized MACD(x,y) for x,y time-spans = (8,24), (16,48), and (32,96)*

*Vol normalized trend for lookbacks of: 1 day, 1 month, 3 months, and 6 months*

*Regression adjusted momentum: R2 \* beta, both from exponential regression of rolling past*

An example for gold:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **DATETIME** | **GOLD\_macd1** | **GOLD\_macd2** | **GOLD\_macd3** | **GOLD\_trend1** | **GOLD\_trend2** | **GOLD\_trend3** | **GOLD\_trend4** | **GOLD\_mom** |
| **12/25/2019** | -0.06304 | 0.17909 | 2.48661 | 1.40385 | 1.18672 | -0.13566 | 0.63625 | 0.00017 |
| **12/26/2019** | 0.10025 | 0.27515 | 2.60833 | 0.48150 | 1.25777 | 0.00000 | 0.83141 | 0.00061 |
| **12/27/2019** | 0.30332 | 0.38970 | 2.63404 | 1.35351 | 1.69217 | 0.73362 | 0.87295 | 0.00150 |
| **12/30/2019** | 0.48787 | 0.51105 | 2.75770 | -0.04374 | 1.54933 | 0.46654 | 1.25723 | 0.00138 |
| **12/31/2019** | 0.69820 | 0.64324 | 2.79122 | 1.42152 | 1.90869 | 0.27778 | 1.39691 | 0.00155 |

Timeframe:

I will be constructing the models to follow an expanding window training set and fixed test set. This strategy retrains and revalidates the models every 4 years to incorporate all available information up to that point in time. It will then predict a trading strategy for the following 4 years.

**Model:**

The first part of my proposed strategy is improving the forecasting step by incorporating custom loss functions. This is an important departure from most financial forecasting strategies which often use a symmetric Mean Squared Error (MSE) objective function. MSE has historically been used due to its neat mathematical properties which drastically simply the operational aspects of the calculation and maximize the likelihood of the model under certain assumptions. This is a perfectly fine solution for forecasting on its own but breaks down when you consider the decisions that are based on the forecasts.

This is best shown through a simple example. Assume your forecasting model predicts that a certain asset will return +1% over the investment period.

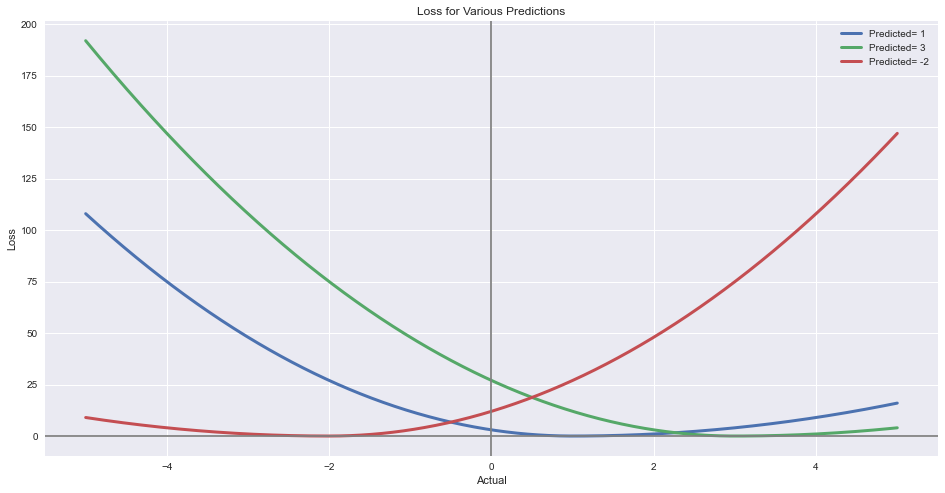
Now consider two scenarios:

1. The actual return is +3%
2. The actual return is -1%

In both scenarios, the squared error is (3% - 1%)2 = (-1%-1%)2 = +.04%. This symmetry is preferred from a forecasting standpoint but as investors, we obviously have a higher preference for the +3% return over the -1% return. Therefore, it is likely better to use an asymmetric objective function. For this project, I’ve simply used the below formula to keep the function smooth.

Where = estimated value, = actual value, and c = some constant

This results in the desired asymmetric behavior. As you can see with the blue line, if we predicted +1 as in our example above, the loss would be greater for -1 than +3.

While this result is much preferred to the traditional MSE approach, it significantly increases the operational workload. Fortunately, some of the intended models have python packages which allow for custom objective function definition.

The second part of my proposed procedure is to improve the position sizing step. Again, I will start by addressing what has traditionally been done and the current disadvantages. Once a forecast is complete, there needs to be some translation into position sizing. The most simple and common way this is currently done is to take the sign of the forecast and translate that into a long or short position. This +/- 1 is then scaled to match the desired volatility and reflect the investors bankroll.