Machine Learning in Portfolio Selection

Hila Malka 313312753 Ron Dagani 318170917

OLMAR

We chose to use OLMAR - On-Line Portfolio Selection with Moving Average Reversion – based on the article written by Bin Li and Steven C. H. Hoi.

Background:

The motivation for developing the method was the understanding that the empirical evidence show that stock's high and low prices are temporary and stock price relatives are likely to follow the mean reversion phenomenon. The existing mean reversion strategies are shown to achieve good empirical performance on many real datasets, but they often make the single-period mean reversion assumption, which is not always satisfied, and may cause a poor performance in some real datasets. Multi-Period without mean reversion has improved the performances. The idea is to take the benefits of both methods and create a new algorithm that will improve the results.

The algorithm:

The first step is initializing and defining w (window) and ϵ, for every day. Then, calculate the daily return and cumulative return.
After that, predict the next price relative vector with moving average:

$$\tilde{x}\,(w) = \frac{\frac{1}{w}\sum_{i=t-w+1}^{t} p_i}{p_t} = \frac{1}{w}\left(1 + \frac{1}{x_t} + \cdots + \frac{1}{\bigotimes_{i=0}^{w-2} x_{t-i}}\right)$$

and update the portfolio: $\mathrm{b_{t+1}} = \mathrm{b_t} + \lambda_{t+1}(\widetilde{\mathrm{x_{t+1}}} - \bar{\mathrm{x}}_{t+1}1)$

When $\bar{\mathrm{x}}_t, \lambda_{t+1}$ are also updated $\bar{\mathrm{x}}_{t+1} = 1^{\mathrm{T}}\frac{\tilde{\mathrm{x}}_{t+1}}{m}, \lambda_{t+1} = \max\{0, \frac{\epsilon - \mathrm{b_t}\cdot\tilde{\mathrm{x}}_{t+1}}{\|\tilde{\mathrm{x}}_{t+1} - \bar{\mathrm{x}}_{t+1}1\|^2}\}$

Finally, normalize b_{t+1}: $b_{t+1} = \underset{b \in \Delta_m}{argmin}\frac{1}{2}\|b - b_t\|^2 \;\; s.t. \;\; b \cdot \tilde{x_{t+1}} \geq \epsilon$

Our addition to OLMAR algorithm: After getting the weights we used only the extreme values weights, and put 0 for entries that are close to zero, to increase sparsity and follow the prediction trend of the algorithm:

Positive values grater than median positive got 1, and negative values smaller than negative median got -1. Eventually, divide by sum.

Experiments:

While working on this project we tried using deep learning – LSTM and CNN networks, and MCMC model. We weren't satisfied with the result. we read articles in the subject and found OLMAR. After tunning the hyper parameters, we got our final model as described above.
Hyper parameter tuning:
To test this model and pick the best hyper parameters, we tried fixing different sizes of ϵ and window:

$w \geq 3, \epsilon \geq 1$ as the paper recommended. We had to try all the combinations and check the model on different timeframes to understand if the model is robust as declared in the paper. we wanted the model to be able to deal with bear and bull market as in the training set – corona time etc.

We surprised to find that different terms required significantly different parameters that changed dramatically the results. We preferred to choose parameters that were suitable for periods with many changes in the market and get more conservative results than to aim for better results but risk not being suitable for the market during the test period. Chosen parameters: $w = 15, \epsilon = 9$.

For summary, the final tuned model showed good results compared to the baseline and to other models on most timeframes.