Quantinvest Competition Report

Augmenting economic theory on portfolio creation
with machine learning techniques

Krzysztof Kowalczyk

Mikołaj Ostrzołek

Theoretical background

Markowitz

In 1952 Harry Markowitz published a paper[1] giving foundation to the 'Modern Portfolio Theory'. It enabled scientist and investors to create optimal portfolios using statistical tools. Binding returns and risk he has shown the trade-off, but also that diversification can lower the risk without lowering the portfolio return. Having a narrow scope of assets (6) available, which are also strongly connected to the market cycle, limits gains from the diversification.

Efficient-Market Hypothesis

The notion of the unpredictability of the stock market has its foundations in Eugene Fama's paper[2] and further reviewed and confirmed by for example Burton Malkiel (2003)[3]. Our attempts to forecast future returns or variability for a long interval of one year will be then somehow contradicting this theory. Moreover, as we have less than twenty years in our sample and just one full business cycle it will be unlikely to find statistically significant models with a horizon of one year.

Exploratory data analysis

Distribution

Returns of bond and stock funds are quasi-normally distributed with fat tails what is common for financial time series. The distribution of the cash fund is positively skewed as negative interest rates are uncommon.

Correlation

Simple correlation matrix shows high positive correlation between stock funds, negative correlation between the developed markets bond fund and stock funds, developing markets bond fund is strongly positively correlated with stock funds. Rest of correlations is close to zero. As our data sample is fat tailed we will use Ledoit-Wolf Shrinkage[4] to avoid estimation error. After transformation we see near-zero correlation

[1] Markowitz, H. (1952). Portfolio Selection. *The Journal of Finance,7*(1), 77-91. doi:10.2307/2975974
[2] Fama, E. (1965). Random Walks in Stock Market Prices. *Financial Analysts Journal, 21*(5), 55-59. Retrieved from http://www.jstor.org/stable/4469865
[3] Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *Journal of economic perspectives*, *17*(1), 59-82.
[4] Ledoit, O., & Wolf, M. (2003). Honey, I shrunk the sample covariance matrix. *UPF economics and business working paper*, (691).

Additional data

To aid our analysis, we have gathered[5] additional data: interest rates, exchange rates, stock index values. In the end, most of this data was not useful for our machine learning models, as the features are heavily correlated with fund returns. While some of features are extremely relevant, they do not allow us to increase accuracy of long-term modelling.

Machine learning approach

Generating optimal portfolio

Our first machine learning experiment was to implement an end-to-end neural network that attempts to build an optimal portfolio. In our case, optimal portfolio allocation has the highest possible returns during 1 year following its construction, with volatility kept below 10%, as required for SRRI level 4[6].

Using a popular PyTorch[7] library, we have implemented a neural network consisting of LSTM and fully-connected layers, that given a time series of historical fund values was trained to return a vector of portfolio allocations (7-element vector in which each element represents the fraction of portfolio allocated to the corresponding fund). To ensure network output will be a set of weights that sum up to 1 we have used a softmax function on the network output. In order to train the network, we have implemented a variety of custom loss functions that penalized excessive volatility and low returns:

$$loss_1 = (\sigma_p - \sigma_t)^2 - r_p \cdot w_r$$
$$loss_2 = (\sigma_p - \sigma_t)^2 - exp(-r_p) \cdot w_r$$

where: σ_p - generated portfolio volatility, σ_t - targeted volatility - hyperparameter
r_p - generated portfolio returns, w_r - weight of return penalty - hyperparameter

Unfortunately, the model did not perform well during a validation period of the last 2 years in the dataset[8]. Our hypothesis was, that the period of 1 year ahead that we needed to forecast was long enough to introduce major shifts in economic market trends, which the model simply cannot take into account. Having discovered that end-to-end black-box approach is impossible, we have decided to apply machine learning techniques to aid the standard portfolio building process instead.

Covariance forecasting

Forecasting future covariance of fund values based on their historical behaviour and additional market data could a better input to the classical portfolio-building methods than a

[5] Source code: Additional Data
[6]Paszke, Adam and Gross, Sam and Chintala, Soumith and Chanan, Gregory and Yang, Edward and DeVito, Zachary and Lin, Zeming and Desmaison, Alban and Antiga, Luca and Lerer, Adam (2017). Automatic differentiation in PyTorch,
[7]Synthetic Risk and Reward Indicator, Committee Of European Securities Regulators, Date: December 2009, Ref.: CESR/09-1026
[8] Source code: LSTM (1st attempt)

standard Ledoit-Wolf covariance. To test whether this approach is viable, we have trained a LightGBM[9] regression model, predicting each value in the future covariance matrix individually. As an input, it received historical covariance of fund returns, calculated in daily, monthly and yearly time windows, along with time series features calculated on additional data sources. There model did not manage to predict covariance values with desired accuracy, which means either we did not have the right additional data to capture long-term changes in relative behaviour between funds, or that there is no way to forecast covariance on such a long period.[10]

Model Ensembling

Ensembling many models yields better results than any of the single estimators[11] for a variety of problems, and naturally it seemed like a valuable experiment to conduct. To do this, we have generated portfolios using 18 strategies build using PyPortfolioOpt[12] package, which allows for easy application of modern portfolio creation theory. All of the base portfolios use Ledoit-Wolf Shrinkage and historical returns as their input.

LightGBM ensembling

As a baseline, we decided to train a LightGBM model on a classification problem - predicting the likelihood of a base model having the top returns in the future. As an input, the ensembling (2nd-layer) model received allocation of each base portfolio along with metrics: expected annualized returns, expected volatility, sharpe ratio. The model was trained on the initial 64% of a dataset (sorted chronologically to prevent information leaks from the test set), and validated on a following 16% of a dataset for early stopping to prevent overfitting. The remaining 20% of the dataset was used to test the final model.

The test set accuracy of the model is initially quite high (40-50%), but drops quickly[13] as the test data starts following different trends than training data. Given that selecting the base estimator that achieved the best results most often in the history is expected to be 30% accurate, this approach quickly becomes impractical. The results are nonetheless promising and made us explore more sophisticated ensembling techniques, which use an neural networks to explicitly optimize for the selected portfolio performance (the classification of which model is the best one did not carry information about expected performance of the chosen portfolio)[14].

LSTM ensembling

The idea is simple: train a neural network that outputs portfolio allocations for the funds, exactly as in the “Generating optimal portfolio” chapter. We still want to optimize the same

[9] Ke, Guolin and Meng, Qi and Finley, Thomas and Wang, Taifeng and Chen, Wei and Ma, Weidong and Ye, Qiwei and Liu, Tie-Yan (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems 30 (NIPS 2017).*
[10] Source code: Covariance Forecasting
[11] Hendrik Jacob van Veen, Le Nguyen The Dat, Armando Segnini (2015). Kaggle Ensembling Guide. *https://mlwave.com/kaggle-ensembling-guide/*
[12] PyPortfolioOpt. *https://pyportfolioopt.readthedocs.io/*
[13] See Appendix A
[14] Source code: LightGBM Ensembling

thing, but this time we also feed the base estimator portfolios as an input to the network. We have also come up with a loss function that penalizes low returns less if the base models could not achieve high returns as well:

$$loss = exp(r_t - r_p)$$

where: r_p - returns from the generated portfolio, r_t - from the best-performing base portfolio

Unfortunately, we were unable to train the network to achieve a reliable test set performance. We have experimented with 3 different LSTM-based architectures, as well as with a simple fully-connected network, all with little success. Plotting loss function value over the validation[15] period reveals that for some days sub-optimal choice of portfolio can lead to a much worse performance than the optimal one, causing large spikes in loss function that none of the models could optimize for[16].

Final decision

The instability of ensembled models' predictions makes it even more plausible that we don't have a way to capture macroeconomic shifts driving the market changes in a period as long as 1 year. The fact that all of the features and base-level portfolio generators are heavily correlated does not help with ensembling either. Because of that, we have decided to settle on one of the traditional portfolio creation techniques, which appeared the most reliable historically.

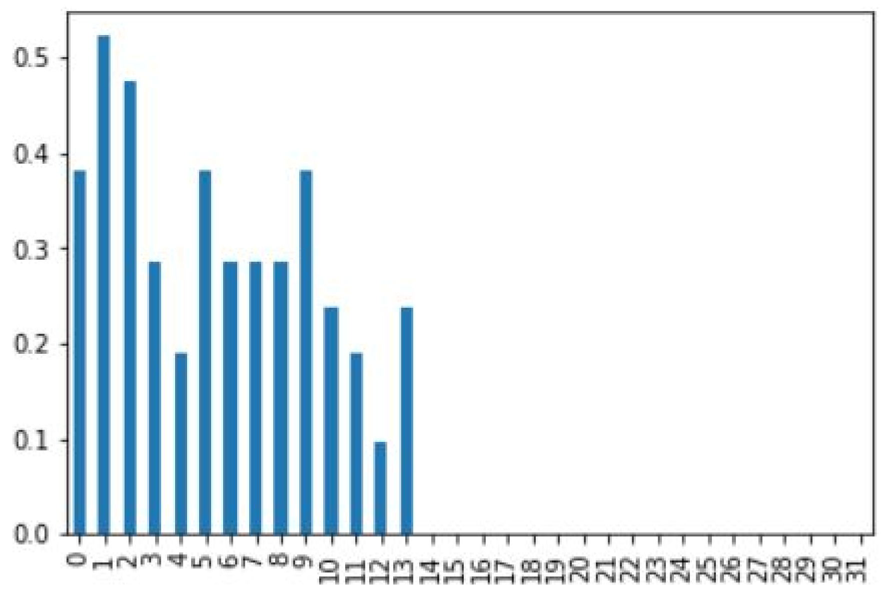
In most cases, the average portfolio of our 18 base-level portfolios has returns similar to the top one, and even during the economic crisis the average difference in performance was only 5 percentage points[17]. Our experiments proved that by constructing a 2nd-level portfolio generator that takes base-level portfolio allocations into account, we cannot account for this kind of uncertainty. Furthermore, choosing one of the base-level portfolio generators allows us for the complete interpretability of our model, which may be necessary, especially in the scenario in which both ensembled and base-level portfolios perform poorly.

Portfolio generator selection methodology

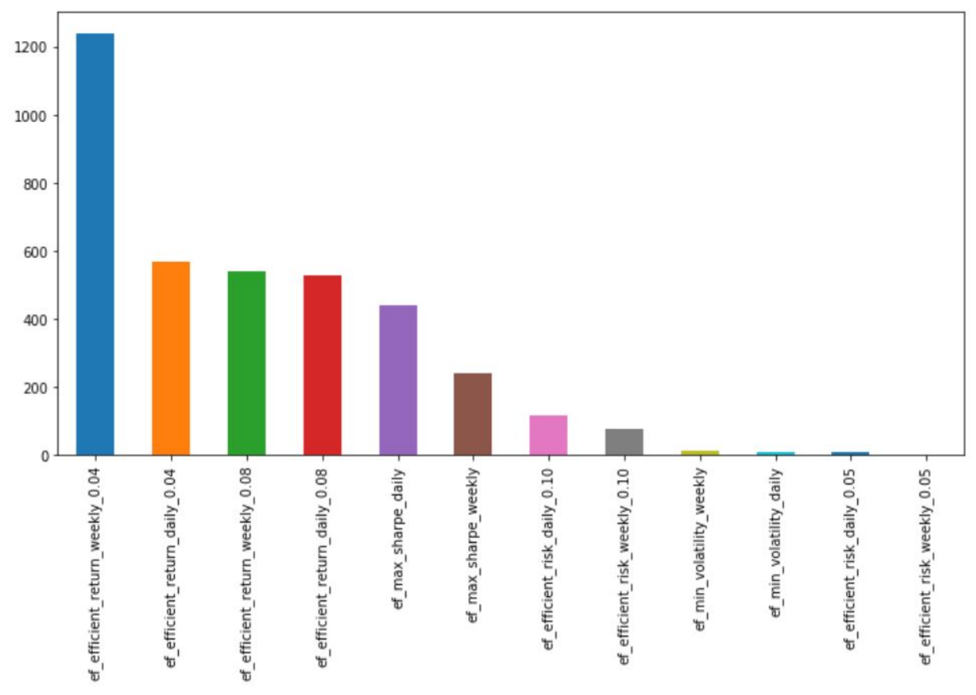
We have analysed performance of portfolios optimized in respect to a) maximum Sharpe ratio, b) minimum volatility, c) target risk (3%, 4%, 5%, 6%, 7%, 8%, 9%) and d) target returns(3%, 4%, 5%, 6%, 7%, 8%, 9%) We came to the conclusion that most optimal method for client with SRRI 3-4 is choosing the portfolio based on target returns equal to 5% and Efficient Frontier Optimisation using Ledoit-Wolf Shrinkage. Optimising portfolio with higher target returns leads to portfolios with risk exceeding SRRI 4, while optimising using method a), b) or c) results in allocation mainly in cash fund what is not optimal for the client as the expected returns are too low.

[15] See Appendix A
[16] Source code: LSTM Ensembling
[17] See Appendix A

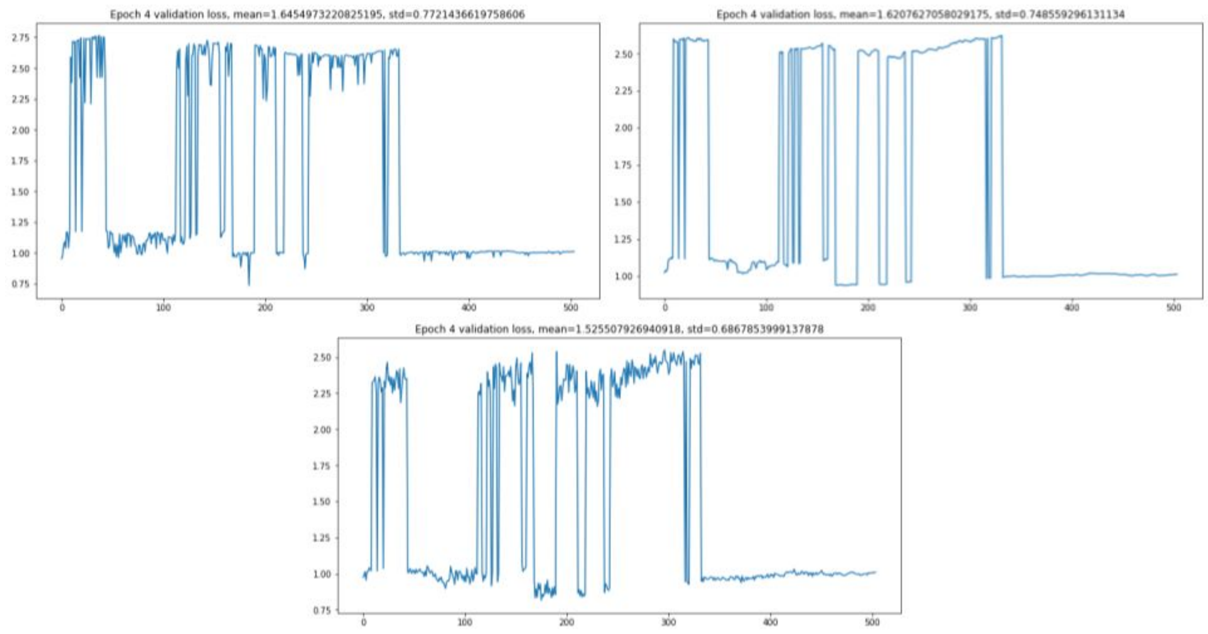
Appendix A: Selected results visualization



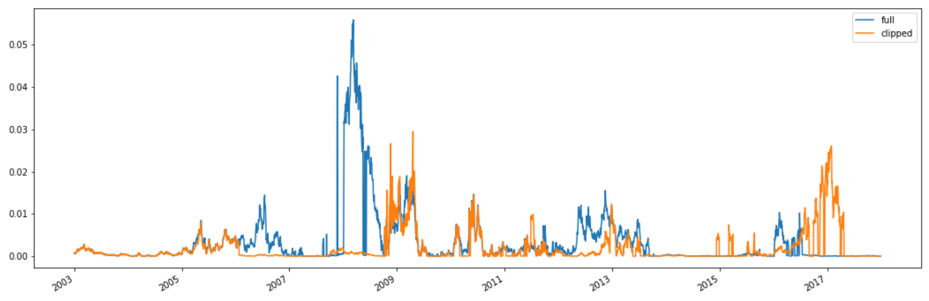
Model accuracy on the test set, X axis represents index of a month in the test set, Y axis represents average accuracy of the model prediction in that month



The base-level portfolio that targets 4% annualized return and minimizes volatility turns out to be the best in over 30% of time intervals.



Validation loss of 3 different 2nd-level models - only on some occasion the ensembled model performs better than all of the base-level portfolios (loss values below 1.0). Choosing sub-optimal weights (loss spikes) happens nearly as often as choosing optimal ones.



Variance of 18 base portfolios' returns (on a year following its construction).
Full - portfolios constructed from all available historical data,
Clipped - portfolios constructed from at most 5 years of historical data.

Appendix B: Experiments source code

All of the source code is available at: https://github.com/kowaalczyk/quant-invest