Vittorio Carlei[*], Oksana Ustenko[**], Gaetano Adamo[**], Donatella Furia[***]

CAN DIFFERENT LEARNING PATHS PRODUCE BETTER ESTIMATES IN EMPIRICAL ASSET PRICING VIA MACHINE LEARNING?

Received: 10 September 2022 / Accepted: 15 July 2023

Abstract.

This paper demonstrates how a portfolio composition technique can leverage the power of machine learning approaches in empirical asset pricing, using a very large set of identical Neural Networks (NN) and differentiating them only by the initial set of parameters. We implement the portfolio employing a trivial rule of ensemble, demonstrating how the variety generated by the initial conditions of the Neural Networks can produce better results than the average. This approach shed a new light on the potential application of ensemble methods to outperform a single NN involved in portfolio construction strategies, using more complex rules to extract the information discovered by the different training paths of identical NN.

JEL CODES: C45; G12; G17
KEYWORDS: EMPIRICAL ASSET PRICING; MACHINE LEARNING; PORTFOLIO CONSTRUCTION

1. Introduction

In this paper we outline the methodology that guides the portfolio management process at Qi4M. We perform the analysis focusing on the

[*] University of Chieti-Pescara, Pescara 65122, Italy, *Email address:* carlei@unich.it
[**] Qi4M, Rome 00144, Italy, *Email address:* ustenko@qi4m.com
[**] Qi4M, Rome 00144, Italy, *Email address:* adamo@qi4m.com
[***] University of Chieti-Pescara, Pescara 65122, Italy, *Email address:* donatella.furia@unich.it

problem of predicting expected returns. As a result, we obtain a model further used to develop an investment strategy that selects the equities to be included in a periodic portfolio. Using an ensemble rule, different from the commonly used average, for combining K estimates further contributes to better prediction and, subsequently, to the selection of best performing stocks. We then present the reader with experimental analysis, which is supported by historical pro-forma results.

When it comes to asset pricing research, one can evidence the extensive application of machine learning methods. For instance, time series asset pricing theory revolves around the ability to explain a satisfactory level of variability in a stock's future returns. This is a fundamental problem of prediction that in large has always been characterized by the ability of a researcher to select the most relevant predictors for a specific asset or time period that is being subject to analysis. In the previous work we showed in which way machine learning provides an optimal solution to this problem. Some important aspects of machine learning have been described as the ability to connect linear to nonlinear models through set functions, or to provide a high level of control in avoiding over-fit bias and false discovery.

The main characteristic of machine learning is that the rules governing the algorithms' functioning do not need to be explicitly coded. The model discovers rules on its own, looking at the *training set*, to then generalize the rule that will later drive its output construction. We showed how the adoption of two different *training* methodologies, supported by the creation of two different *training sets*, can significantly contribute to building performing portfolios.

Using machine learning and neural networks in particular, one inevitably encounters the problem of managing the randomness of results. Training algo- rithms for deep learning models are usually iterative in nature and thus require the user to specify some initial point from which to begin the iterations. Moreover, training deep models is a sufficiently difficult task that most algorithms are strongly affected by the choice of initialization. This makes the neural network unstable and unreliable, especially when sharing your code with others or show- casing your work (Zhuang et al., 2021). When looking at portfolio selection, we can potentially have different portfolios, as performances differ each time the algorithm is run. Randomness can occur in NN due to multiple reasons. Here we refer to random initialization of weights, biases and batch sampling. The literature suggests that random seeds can adversely affect the consistency of models resulting in counterfactual interpretations, when randomness is not managed.

Most of the authors who have raised the question of the application of machine learning in asset pricing have overlooked, or avoided, this point.

Random noise is crucial for getting NNs to work well: it allows neural nets to produce multiple outputs given the same instance of input and limits the amount of information flowing through the network, forcing the network to learn meaningful representations of data. Therefore, the characteristic randomness of the NNs can be used as a resource to improve the final prediction. Usually, in financial environment we want to achieve the model's stability, specifically volatility and returns that are stable overtime. We want to demonstrate how achieving the stability of the model using N different neural networks can actually lead to better portfolio selection. We will compare the standard average method on predictions with a single model, and another, less trivial ensemble rule of N neural networks.

2. Methodology

We begin by defining an investment universe that makes up the model's input. We consider highly liquid stocks that would allow us to interact on the market without price or volume frictions. As previously stated, the initial input selection of the model has been influenced by an arbitrary selection on our part, which is a product of conjunct research efforts. We will keep the description of such selection terse. The reader can refer to the previous work for details.

From this step, the model undergoes the training and testing phases, employing methodologies that will be presented in this section. The next step is to manage a basket of NN models with an ensemble, which is a product of 200 random seeds. The ultimate output of the model is an investment strategy in the form of a portfolio of equities, which repeatedly over-performs the universe's benchmark.

We will follow with a description of the financial theory that influenced the construction of the model, avoiding to present the reader with a profound description of the relevant models. We invite the reader to refer back to the bibliography, to gain a more thorough understanding of the topics mentioned in the discussion.

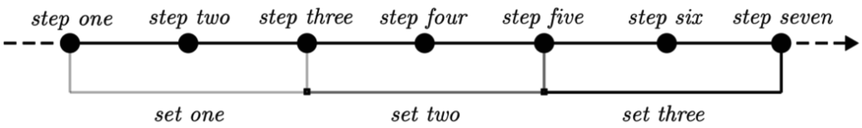
2.1 Prediction

The preliminary step in the formulation of our approach is to understand how to best design sub-samples for estimation and testing[1]. This process starts from setting a time-frame for the characteristic *walk-forward* we employ for training and testing both models.

The two methodologies cited before refer to two different methods we apply to perform the *walk-forward*. We essentially define the *steps* that constitute the *walk-forward*. Here, *steps* indicate the event of new information becoming available to the market. In our case, the *steps* indicate the update of a given company's set of selected fundamentals. With respect to the aforementioned *steps*, we can differentiate the two methods that from now on will be referred to as 'Veloce' and 'Lento'. We then use this set to predict stocks' returns at a future point in time.

The first method, 'Veloce', creates a fixed *steps-size* train set using the last 'few' *n steps* available in the information set. Once new information becomes available, it moves the rolling window forward to create a new train set that drops the first *step* and picks up the last. The number of steps chosen for the 'Veloce' method is n = 5 for our experiments. On the other hand, the second method, 'Lento', updates the training set by increasing its size to include new information available at the new *step*.

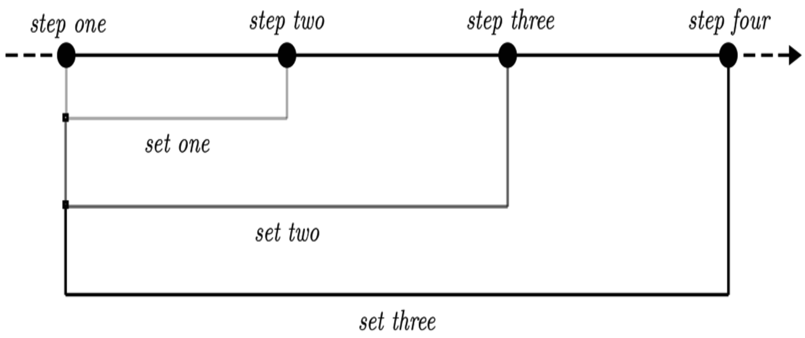
Fig. 1. Step size



The figure shows a fixed-length step in time. In this example the step length has been fixed to n = 2.

[1] Note that when constructing the *training set and test set*, one incurs two phenomena: missing data and noisy data. To overcome the problem of missing data, we apply the method suggested by Beaver et al. [2007]. At the same time, we adopt the approach proposed by Steege et al. [2012] to handle the noisy data.

Fig. 2. Step Incremental



To perform the prediction of stock returns we adopt two regressors: a feed- forward multi-layer perceptron (MLP), and a multivariate linear regressor(LR), which are then used to minimize the same objective function, the mean squared prediction errors (MSE). The multi-layer perceptron's training phase handles the dynamics to be picked up by the 'Lento' method, while those of 'Veloce' are better handled by the multivariate linear regressor. For the 'Lento' model, we train K MLPs that differ in initialization of weights and biases. So in the end we will have K + 1 signals (predictions), where K is the result of the 'Lento' approach and one of the 'Veloce' model.

The 'Lento' model gives us the following output:

$$L = \left(l^{s}_{t,k}\right)_{s,t,k} \tag{1}$$

indexed by:
$\forall\ t = 1, \ldots, T$ rebalancing date,
$\forall\ s = 1, \ldots, S$ stock in universe,
$\forall\ k = 1, \ldots, K$ random initialization of MLP,
where $l(s, t, k) \in R$

While, from the 'Veloce' model we get the following array of signals:

$$V = (v_t^s)_{s,t} \tag{2}$$

∀ $t = 1, \ldots, T$ rebalancing date
∀ $s = 1, \ldots, S$ stock in universe

By rebalancing date we mean the date on which training is performed with the walk-forward method described above, also intended as the step in defining training and test set. Therefore, at each rebalancing date we will have an out- of-sample prediction that corresponds with the next training step.

2.2 Ensemble

In this section we show how the 'Lento' and 'Veloce' output can be combined to build the ensemble rules (i.e. scores) that will guide the portfolio selection. We will specifically show two rules that differ in the use of the 'Lento' model. The first trivial one involves the average of the *k* scores of the 'Lento' model. The second one uses each score of the 'Lento' model as a market view in its own right.

Mean ensemble Whenever we employ a model whose result depends on the initial conditions, it is a good practice to train the model to vary the initialization parameter and then use the average of all the obtained results. This is the case with our 'Lento' model: it is constituted, in any given *t* and for each stock *s*, by *k* prediction scores (also known as *k* forecasting), one for each model initialization. The following equation illustrates how the array of signals is built:

$$L' = \left(\frac{1}{K}\sum_{k=1}^{n}\left(l_{t,k}^{s}\right)_{s,t,k}\right)_{s,t} = (l'^{s}_{t})_{s,t} \tag{3}$$

We subsequently build the final score by defining:

$$R = min(L', V) = \left(min(l'^{s}_{t}, v_t^s)\right)_{s,t} = (r_t^s)_{s,t} \tag{4}$$

The parameters we refer to, are defined as follows:
$K = 200$ random seeds,
$T = 42$[2] quarters,
$S = 980$[63] stocks on average.

The final portfolio is constructed using this rule, which means that for each t we take the first 30 stocks, sorted by the r score in a descending order.

Union ensemble Since we operate in the context of financial environment, different results produced by K NN be interpreted as different market views, instead of making an average prediction.

For each fixed initialization k we construct the vector:

$$R_k = \left(min\left(l'^{s}_{t,k}, v^{s}_{t}\right)\right)_{s,t,k} = \left(r^{s}_{t,k}\right)_{s,t,k} \tag{5}$$

From now on, we will refer to R_k with its descending enumeration i.e. the ranking. Using the series $(R_k)_{k=1,2...,K}$, that are K ordered lists of stocks at every fixed rebalancing date t, we want to construct a portfolio with at least P ($P = 20$) stocks, that are in the top of the ranking. We do this using the algorithm described in the following pseudo code that will run for every fixed rebalancing date t.

[2] The quantity of quarters (training steps) on which experimentation is carried out. The period under examination contains almost 11 years, i.e 42 quarters

[3] The number of stocks that make up the universe in each quarter (i.e. in each step, i.e. in each rebalancing date). It is a variable number, since it depends on the actual availability of the stock traded on market at each point of time.

Algorithmus 1 Unio

Input: *P* is the minimum stocks in the selected portfolio,
L is the number of stocks in the current portfolio,
S the number of stocks in the universe,
r^1 a parameter that represents a guide to select the stocks over the ranking

Output: Portfolio
Initialize*:*
r' = 1, L = 0
while $L < P$ or $r > S$ do

$$current\ portfolio = \left\{ s \in universe : s \in U_{k=1}^{K} \left\{ r_{s,t,k} \in R_k \, : \, r_{s,t,k} <= r' \right\} \right\}$$

$$r' = r' + 1$$

end whi*le*

In other words, for every fixed rebalancing date *t* we make a union of K ordered lists of stocks that are at least in the top 20[4] positions of the ranking.

3. Experimental evaluation

In this section we summarize our findings about different ensemble methods, constructed over K=200 runs of the model, with 200 different random seeds, and with identical model settings. We report ensemble performances in terms of financial indicators.

The first graph (Fig.3) represents the level of variability of performance as the random seed changes. By choosing one seed instead of another, we will potentially have very different portfolios, which makes the results of the model difficult to interpret. We also see how the two ensembles, mean and union, are placed in the sparsity of results. The mean ensemble helps produce the final result in a very trivial way and is a satisfactory solution for achieving the stability goal. Yet, as we can observe, it is not the best solution, unlike the union, which doesn't only solve the problem of randomness but also has a positive effect on the return and Sharpe ratio.

---

[4] To make the portfolio construction strategies, and the portfolios themselves, comparable, they must be composed, on average, of the same number of stocks. So:

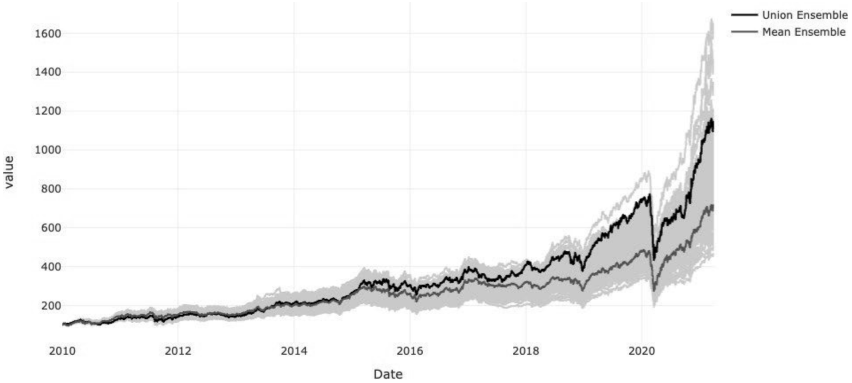
- For the Mean ensemble, on each rebalancing date we take the first 30 stocks, thus obtaining a portfolio consisting of exactly 30 stocks in each quarter;

-For the Union ensemble rule we set the minimum number of stocks for each of the K (K = 200 random seeds) models at 20. By definition, portfolios are built as a union of models, so that the final number in each quarter will be, on average, around 30.

Table 1. Out-of-sample financial performance

	Mean Ensemble	*Union Ensemble*
Return	19.01%	24.13%
Volatility	19.16%	20.84%
Sharpe Ratio	0.99	1.16
Max Draw-down	–43.99%	-43.72%

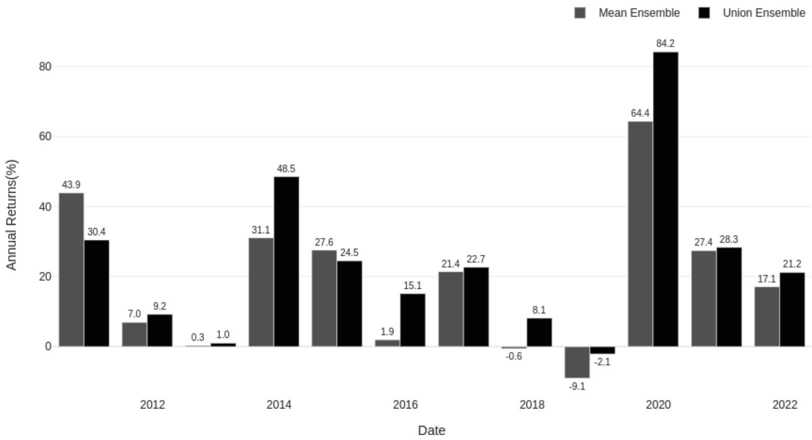
Fig. 3. NAV.



The figure shows performance in terms of cumulative product of investment of 1Euro from 2010 to 2021, of K=200 runs of the model with 200 different random seeds. In addition, mean and union ensembles are shown for comparison.

We noticed that in 5% of the cases, some of the random seeds exhibit, on average, better performance than the union ensemble. While it is true that some random seeds (in the long term) perform better, it is equally true that there is a priori no way to identify these random seeds. Since the random seed is usually not a tuning parameter, the goal of this study is to show the reader that there is a method that produces better performance than the usual mean ensemble.

Fig. 4. Annual returns



The figure shows performance in terms of percentage annual returns from 2010 to 2021. The graph compares a single random seed model and the mean ensemble to the union ensemble, whose performance is significantly better during the time of the investment.

4. Conclusion

In this article, we study how the inherent instability of neural models as a function of random seed can actually be used as a resource in the selection of portfolios in the field of machine learning.

We analyze the performance and robustness of the model in the form of financial performance. Our analysis strongly highlights how model stability problems and its effects on black box interpretation methods lead to different views of the financial market for different random seeds, and the ways they can be exploited to our advantage.

Only by varying the random seed and keeping all the parameters unchanged we introduce the variability necessary to make an ensemble of the predictors and with this obtain the possibility to significantly improve the performance of the final model.

We have proposed two solutions to overcome the problem of randomness. The first is the trivial rule of averaging, while the second one, the union ensemble, is more performing and brings added value to our final portfolio selection. We prove that our proposed method is significantly more efficient than using a single initialization or an average model, and greatly reduces the model's instability and increases the performance.

References

Beaver, W., McNichols, M., & Price, R. (2007), *Delisting returns and their effect on accounting-based market anomalies*, Journal of Accounting and Economics, 43(2-3), 341-368.

Bellgard, C., & Goldschmidt, P. (1999), *Forecasting across frequencies: Linearity and non-linearity*, in Forecasting across frequencies: Linearity and non-linearity, School of Information Technology, Bond University, 41-48.

Bender, J., Briand, R., Melas, R.A. (2013), *Subramanian, Foundations of factor investing*, MSCI Index Research.

Bishop, C.M. (1995), *Neural networks for pattern recognition*, Oxford University Press

Bishop, C.M., & Nasrabadi, N.M. (2006), *Pattern recognition and machine learning*, Springer, New York, 4(4), 738.

Breiman, L. (1996), *Bagging predictors*, Machine learning, 24, 123-140.

Butaru, F., Chen, Q., Clark, B., Das, S., Lo, A.W., & Siddique, A. (2016), *Risk and risk management in the credit card industry*, Journal of Banking & Finance, 72, 218-239.

Carlei, V., Terzi, S., Giordani, F., & Adamo, G. (2021), *Portfolio management via empirical asset pricing powered by machine learning*, in Decision Economics: Minds, Machines, and their Society. Springer International Publishing. N., 121-129.

Chan, M.C., Wong, C.C., & Lam, C.C. (2000*), Financial time series forecasting by neural network using conjugate gradient learning algorithm and multiple linear regression weight initialization*, Computing in Economics and Finance, 61, 326-342.

Chen, C. Guestrin, C. (2016), *Xgboost: A scalable tree boosting system*, Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794.

Dietterich, T.G. (2000), *Ensemble methods in machine learning*, in International workshop on multiple classifier systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 1-15.

Duffie, D. (2012), *Dark markets: Asset pricing and information* *transmission in over-the-counter markets*. Princeton University Press.

Eldan, R., & Shamir, O. (2016), *The power of depth for feedforward neural networks*, in Feldman V., Rakhlin, A., and Shamir, O., eds.,29th Annual Conference on Learning Theory, Proceedings of Machine Learning Research, 49, 907-940

Fama, E.F., & French, K.R. (1993), *Common risk factors in the returns on stocks and bonds*, Journal of financial economics, 33(1), 3-56.

Fama, E.F., & French, K.R. (2015), *A five-factor asset pricing model*, Journal of financial economics, 116(1), 1-22.

Freyberger, J., Neuhierl, A., & Weber, M. (2020), *Dissecting characteristics nonparametrically*, The Review of Financial Studies, 33(5), 2326-2377.

G. Brown, G. Kuncheva, L.I, (2010), "*Good" and bad" diversity in majority vote ensembles*, Proceeding of Multiple Classifier Systems, 1-15.

Giglio, S., & Xiu, D. (2021), *Asset pricing with omitted factors*, Journal of Political Economy, 129(7), 1947-1990.

Gu, S., Kelly, B., & Xiu, D. (2020), *Empirical asset pricing via machine learning*, The Review of Financial Studies, 33(5), 2223-2273.

Hansen, L.K., & Salamon, P. (1990), *Neural network ensembles*, IEEE transactions on pattern analysis and machine intelligence, *12*(10), 993-1001.

Harvey, C.R. Liu, Y. (2016), *Lucky factors, Technical report*, Duke University.

Heaton, J.B., Polson, N.G., & Witte, J.H. (2016), *Deep learning in finance*, arXiv preprint arXiv:1602.06561.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J.E., & Weinberger, K.Q. (2017), *Snapshot ensembles: Train 1, get m for free*, arXiv preprint arXiv:1704.00109.

Hutchinson, J.M., Lo, A.W., & Poggio, T. (1994), *A nonparametric approach to pricing and hedging derivative securities via learning networks*, The journal of Finance, 49(3), 851-889.

Khandani, A. E., Kim, A. J., & Lo, A. W. (2010), *Consumer credit-risk models via machine-learning algorithms*, Journal of Banking & Finance, 34(11), 2767-2787.

Koijen, R.S., & Van Nieuwerburgh, S. (2011), *Predictability of returns and cash flows*, Annu. Rev. Financ. Econ., 3(1), 467-491.

Kourentzes, N., Barrow, D.K., & Crone, S.F. (2014), *Neural network ensemble operators for time series forecasting*, Expert Systems with Applications, 41(9), 4235-4244.

Krogh, A., & Vedelsby, J. (1994), *Neural network ensembles, cross validation, and active learning*, Advances in neural information processing systems, *7*.

Krollner, B. Vanstone, B. and Finnie B, (2010), *Financial time series forecasting with machine learning techniques: A survey*, Paper presented at the European symposium on artificial neural networks: Computational and machine learning, Bruges, Belgium.

Lewellen, J. (2014), *The cross section of expected stock returns*, Forthcoming in Critical Finance Review, Tuck School of Business Working Paper, 2511246.

Madhyastha, P., Jain R. (2019), *On Model Stability as a Function of Random Seed*, CoNLL.

Moritz, B., & Zimmermann, T. (2016), *Tree-based conditional portfolio sorts: The relation between past and future stock returns*, Available at SSRN 2740751.

Opitz, D., & Maclin, R. (1999), *Popular ensemble methods: An empirical study*, Journal of artificial Intelligence research, 11, 169-198.

Rapach, D.E., Strauss, J.K., & Zhou, G. (2013), *International stock return predictability: What is the role of the United States?*, The Journal of Finance, 68(4), 1633-1662.

Rapach, D., & Zhou, G. (2013). Forecasting stock returns. In *Handbook of economic forecasting* (Vol. 2, pp. 328-383). Elsevier.

Sadhwani, A., Giesecke, K., & Sirignano, J. (2021), *Deep learning for mortgage risk*, Journal of Financial Econometrics, 19(2), 313-368.

Shefrin, H., & Statman, M. (1994), *Behavioral capital asset pricing theory*, Journal of financial and quantitative analysis, 29(3), 323-349.

Steege, F.F., Stephan, V., & Groß, H.M. (2012), Effects of noise-reduction on neural function approximation, in European Symposium on Articial Neural Networks.

Stein, R.M. (2002), *Benchmarking default prediction models: Pitfalls and remedies in model validation*, Moody's KMV, New York, 20305, 16.

Welch, I., & Goyal, A. (2008), *A comprehensive look at the empirical performance of equity premium prediction*, The Review of Financial Studies, 21(4), 1455-1508.

Yao, J., Li, Y., & Tan, C.L. (2000), *Option price forecasting using neural networks*, Omega, 28(4), 455-466.

Zhang, G.P., & Berardi, V.L. (2001), *Time series forecasting with neural network ensembles: an application for exchange rate prediction*, Journal of the operational research society, 52(6), 652-664.

Zhou, Z.H., Wu, J., & Tang, W. (2002), *Ensembling neural networks: many could be better than all*, Artificial intelligence, 137(1-2), 239-263.

Zhuang, D., Zhang, X., Song, S., & Hooker, S. (2022), *Randomness in neural network training: Characterizing the impact of tooling*, Proceedings of Machine Learning and Systems, 4, 316-336.