Portfolio Construction with Single Stage Optimization

Zoe Zhao, Utkarshbhanu Andurkar, Jinwei Zhu

Abstract

The traditional approach to portfolio construction uses a two-step process, where in the first step the expected return and volatility of the assets are estimated and in the second step a Markowitz's mean-variance portfolio optimization is used to compute the optimal asset allocation for the portfolio. In the two-stage approach, we optimize for two separate goals, minimizing the prediction errors of the expected return of the assets and maximizing Sharpe ratio of the portfolio, which may lead to sub-optimal asset allocation due to error accumulation. This case study explores an end-to-end deep learning based approach to mitigate the problem with the traditional two-stage approach.

1 Introduction

The cornerstone of Harry Markowitz's 1952 model is the optimization of a portfolio by selecting a combination of assets that minimizes the total risk for a given level of expected return, or equivalently maximizes the expected return for a given level of risk. This is achieved through mean-variance optimization, which can be formulated as an objective function with constraints, grounded in the concept of the efficient frontier.

The mathematical formulation of Markowitz's model is as follows. Consider a portfolio of n assets, and let w be a vector of weights w_i, where w_i represents the proportion of the portfolio's total value invested in asset i. Let R be a vector of expected returns R_i, and let Σ be the covariance matrix for the returns, with elements σ_{ij} representing the covariance between the returns of assets i and j.

The expected return of the portfolio $E[R_p]$ is given by the weighted sum of the expected returns of the individual assets:

$$E[R_p] = \sum_{i=1}^{n} w_i E[R_i] \tag{1}$$

The variance (risk) of the portfolio σ_p^2, which is the objective to be minimized, is a quadratic form given by:

$$\sigma_p^2 = w^T \Sigma w = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij} \tag{2}$$

The portfolio optimization problem can be stated as:

$$\begin{aligned} \min_{w} \quad & w^T \Sigma w \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i = 1 \\ & \sum_{i=1}^{n} w_i E[R_i] = \mu_p \end{aligned} \tag{3}$$

Here, μ_p is the desired expected return of the portfolio. The first constraint ensures that all the portfolio weights sum up to 1 (meaning all the budget is invested), and the second constraint sets the expected return of the portfolio to a certain level.

In matrix notation, this optimization is often subject to additional constraints, such as no short-selling:

$$w_i \geq 0 \quad \text{for all } i \tag{4}$$

The Markowitz's portfolio formulation is a quadratic programming problem because the objective function is quadratic and the constraints are linear. Solving this problem gives the set of weights that defines an efficient portfolio for the given expected return μ_p. By varying μ_p, one can trace out the efficient frontier, which represents the set of portfolios that maximize expected return for a given level of risk or minimize risk for a given level of expected return.

The efficient frontier is a parabola in the risk-return space, and only the upper part of this parabola is relevant to investors because it represents the portfolios with the maximum expected return for a given risk level.

In practice, the mean-variance optimization requires estimates of expected returns and the covariance matrix, which can be challenging to obtain accurately, leading to potential sensitivity in the optimal portfolio weights. This sensitivity and the assumption of normally distributed returns are some of the critiques of the mean-variance optimization, leading to the development of other models and methods in portfolio construction, such as the Black-Litterman model, which incorporates subjective views into the optimization process.

In summary:

- The Markowitz portfolio optimization problem deals with finding the right proportion of underlying assets in a portfolio such that you get maximum reward by taking minimum overall risk.
- Traditionally, portfolio construction problems are tackled in a two-step process. In the first step, the expected returns and volatility of the underlying assets are estimated using various techniques like regression models. In the second step using these estimates and Markowitz mean-variance portfolio optimization theory, the best proportion of each asset in the portfolio is computed such that the risk-reward function like Sharpe ratio is optimal.
- This approach has a major drawback. Any error in estimation in step 1 can lead to suboptimal portfolio allocation in step 2. Expected returns of the underlying assets are particularly difficult to estimate. Therefore, the two stage models can lead to error accumulation which result in suboptimal portfolio decisions.

2 Solution

2.1 Possible solutions in literature

There are many solutions to the issues with 2 stage optimization. Instead of optimizing to get best accuracy for expected returns (in step 1) and then using Markowitz theory to get optimal allocations, one can optimize to get the best model of expected returns which will lead to optimal asset allocation in the portfolio. Here learning is performed based on task loss instead of generic loss function. Prediction models are part of a larger process and probabilistic machine learning models are trained to capture task-based objective. Stochastic optimization models follow this approach. This is conceptually similar to the model free approach we are going to explore in the case study.

There are some solutions which improve the predict then optimize approach (2 step approach) with a framework called as 'Smart Predict then Optimize(SPO)' to leverage optimization problem structure to design better prediction models. The prediction based loss function is replaced with decision error by using convex surrogate loss function which is applicable to linear models. There is literature where generalization bounds are provided for the predict then optimize approach. These bounds can be better indicator to decide when to use the two step approach over the end to end approach.

An interesting line of research focuses on embedding the optimization problem (finding ideal portfolio allocation in our case) in the neural network which is referred as optimization or implicit layers. This approach allows to integrate the parameterized optimization problem as an individual layer in an end-to-end trainable neural network whose parameters are learned through propagation. This constitutes a part of model based approach that we are going to briefly summarize in the case study.

Some of researchers believe that the main focus of supervised learning system is to provide optimal predictions, not optimal decision making under uncertainty ie they prefer the two step approach. To mitigate the issues with the two step approach they introduce a link function to generate data-driven prescriptive predictions, and provide specific constructions for a great variety of supervised learning models with theoretical guarantees.

We draw the above information from the literature survey of *Uysal et al.*(2021)

2.2 Solution Explored In The Case Study

There are 2 solutions that we will explore in this case study. We will explore the model free end to end deep learning based approach in depth with the necessary math and a hands-on example on toy data. We will only briefly summarize the model based approach (with implicit layers) and will not go in entire mathematical derivation and hands-on example for it.

Using deep learning an end to end approach can be designed where we do not estimate intermediate parameters (future daily returns and covariance matrix), from inputs (past daily returns) we directly estimate the optimal future asset allocation, using a simple feed forward neural network with gradient descent as the optimization algorithm. *Uysal, A.S (2021)* refers this as model free approach.

The performance of this neural net can further be improved by implementing an implicit layer with risk budgeting portfolio optimization. This not only makes the neural network more interpret-able but it also improves the performance. *Uysal, A.S (2021)* refers this as model based approach.

Let us dive deep into these approaches.

2.2.1 Model Free Approach/ Simple Feed forward Neural Network

Let us understand the model free approach in detail. Following picture depicts the internal architecture of the neural network used for model free approach.

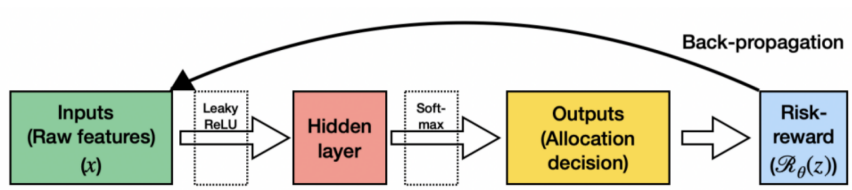

Figure 1: Computational graph of model-free approach [*Uysal, A.S (2021)*]

The structure consists of 1 input layer followed by a leaky relu activation. The activation function is followed by a single hidden layer. The hidden layer is followed by a softmax activation function. The output of the softmax activation are treated as allocation decision.

The role of input layer is to get inputs for the neural network. The inputs are daily returns of the assets under consideration (assets which constitute the portfolio).

Ideally we don't want to include assets which have negative returns in our portfolio. If we keep the negative returns as is, our model will learn to assign low or zero allocation to these assets. But even if an asset is giving negative daily returns for some period of time, there is always a possibility that the returns will bounce back and become positive. If we keep the negative returns as is in the input, our model will learn to predict an allocation of near 0 for these assets and to increase these allocation the asset need to consistently give positive returns ie our model will not react to changing market dynamics quickly. To remedy this situation, we use leaky relu, which does penalize the asset for having negative returns for a period of time, but it does so by assigning a negative number of smaller magnitude. This makes our model quick to react when the returns from those assets start turning positive.

The hidden layer scales its inputs by some weight and adds a bias to them. These weights and biases are learned from the data during backpropagation step.

The softmax layer normalizes the output provided by the hidden layer such that the sum of the outputs is now 1. We treat these numbers, which are between 0 and 1 (which sums up to 1), as allocation decision.

From these allocation decision we compute a risk reward function (return on the portfolio or sharpe ratio = return/variance). We try to maximize this risk reward function during the backpropagation stage. Alternatively we treat the negative of the risk reward function as loss and try to minimize it using gradient descent.

let us familiarize ourselves with the mathematics behind this neural network in the next section.

2.2.2 Mathematics behind the model

STEP 1 : Forward Propagation and Backpropagation for 1 Pass

- **Inputs**

$$X = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix}$$

$$E = \begin{bmatrix} E_{11} \\ E_{21} \end{bmatrix}$$

$x_i j$ are daily returns for asset i on day $t+1-j$.
The first row of the X matrix represents returns for Asset 1, the second row represents returns for Asset 2. The first column represents returns for day t, and the second column represents returns for day $t-1$.

The E matrix, on the other hand, has its first row representing Asset 1, the second row representing Asset 2, and this column represents daily returns for Day $t+1$.

- **Feed Forward**

  1. **Leaky ReLU**

  $$Y = \text{LeakyReLU}(X)$$

  As $\text{LeakyReLU}(X) = \max(X, 0) + \min(X, 0) \times 0.01$

  After this transformation we get a Y matrix, which we denote as

  $$Y = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}$$

  where,
  $Y_{11} = \max(x_{11}, 0) + \min(x_{11}, 0) \times 0.01$
  $Y_{12} = \max(x_{12}, 0) + \min(x_{12}, 0) \times 0.01$
  $Y_{21} = \max(x_{21}, 0) + \min(x_{21}, 0) \times 0.01$
  $Y_{22} = \max(x_{22}, 0) + \min(x_{22}, 0) \times 0.01$

  2. **Hidden Layer**

  $$Z = Y.W + B$$

  where W is the weight matrix/vector and B is the bias vector.

  Hidden Layer expanded

  $$\begin{bmatrix} Z_{11} \\ Z_{21} \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} + \begin{bmatrix} b \\ b \end{bmatrix}$$

  If we simplify (matrix multiplication and addition) and write the terms individually, we get

  $$Z_{11} = W_{11}Y_{11} + W_{21}Y_{12} + b$$

  $$Z_{21} = W_{11}Y_{21} + W_{21}Y_{22} + b$$

3. **Softmax Activation**

$$S = \text{Softmax}(Z)$$

$$\begin{bmatrix} S_{11} \\ S_{21} \end{bmatrix} = \frac{1}{e^{Z_{11}} + e^{Z_{21}}} \begin{bmatrix} e^{Z_{11}} \\ e^{Z_{21}} \end{bmatrix}$$

If we simplify and write the terms individually, we get

$$S_{11} = \frac{e^{Z_{11}}}{e^{Z_{11}} + e^{Z_{21}}}, \quad S_{21} = \frac{e^{Z_{21}}}{e^{Z_{11}} + e^{Z_{21}}}$$

We will treat these S values as allocation proportions for the portfolio of 2 assets.

4. **Loss function**

We will use negative of portfolio return for a certain allocation, as a loss function for the neural network.

$$L = -S^{\top}.E = -\begin{bmatrix} S_{11} & S_{21} \end{bmatrix} \begin{bmatrix} E_{11} \\ E_{21} \end{bmatrix}$$

$$L = -(S_{11}E_{11} + S_{21}E_{21})$$

- **Backpropagation**

We have 3 parameters to tune: W_{11}, W_{21}, b.

Using Gradient descent with learning rate α, we will have following set of equations for the backpropagation step.

$$W_{11} = W_{11} - \alpha \left(\frac{\partial L}{\partial W_{11}} \right)$$

$$W_{21} = W_{21} - \alpha \left(\frac{\partial L}{\partial W_{21}} \right)$$

$$b = b - \alpha \left(\frac{\partial L}{\partial b} \right)$$

We need to calculate the derivatives to update the weights.

The formulae for backpropagation are as follows -

- $$W_{11} = W_{11} - \alpha S_{11} S_{21} \left(Y_{21} - Y_{11}\right) \left(E_{11} - E_{21}\right)$$
- $$W_{21} = W_{21} - \alpha S_{11} S_{21} \left(Y_{22} - Y_{12}\right) \left(E_{11} - E_{21}\right)$$
- $$b = b$$

Refer to the appendix for detailed derivation of these formulae.

2.2.3 Training on Toy Data

Let us use a concrete example to solidify the concepts we learned in the previous section. The matrix X contains daily returns for 2 assets for yesterday and day before yesterday, while E vector contains daily returns for those two assets today.

$$X = \begin{pmatrix} 2 & 1 \\ 4 & -2 \end{pmatrix}, \quad E = \begin{pmatrix} 4 \\ 6 \end{pmatrix}, \quad W = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}, \quad b = 0.5$$

1. **LeakyReLU**
   As LeakyReLU$(x) = \max(x, 0) + \min(x, 0) \times 0.01$

   $$\text{LeakyReLU}(X) = Y = \begin{pmatrix} 2 & 1 \\ 4 & -0.02 \end{pmatrix}$$

2. **Linear Layer**
   $Z = Y \cdot W + b$

   $$Z = \begin{pmatrix} 2 & 1 \\ 4 & -0.02 \end{pmatrix} \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

   $$Z = \begin{pmatrix} 2 \\ 2.49 \end{pmatrix}$$

3. **Softmax**
   $S = \text{Softmax}(Z)$

   $$S = \begin{pmatrix} \frac{e^2}{e^2 + e^{2.49}} \\ \\ \frac{e^{2.49}}{e^2 + e^{2.49}} \end{pmatrix}$$

   $$S = \begin{pmatrix} \frac{7.38906}{19.45033} \\ \\ \frac{12.06128}{19.45033} \end{pmatrix}$$

$$\text{Therefore, } S = \begin{pmatrix} 0.3799 \\ 0.6201 \end{pmatrix}$$

For iteration 1:

$$\begin{aligned}
s_{11} &= 0.3799, \quad s_{21} = 0.6201 \\
y_{11} &= 2, \quad y_{12} = 1, \\
y_{21} &= 4, \quad y_{22} = -0.02 \\
E_{11} &= 4, \quad E_{21} = 6 \\
W_{11} &= 0.5, \quad W_{21} = 0.5, \alpha = 0.01
\end{aligned}$$

As

$$W_{11} = W_{11} - \alpha s_{11} s_{21} (y_{21} - y_{11})(E_{11} - E_{21})$$

$$\begin{aligned}
&= 0.5 - 0.01 \times 0.3799 \times 0.6201 \times 2 \times -2 \\
&= 0.5 + 9.4230396 \times 10^{-3} \\
W_{11} &= 0.5094
\end{aligned}$$

Similarly,

$$\begin{aligned}
W_{21} &= W_{21} - \alpha s_{11} s_{21} (y_{22} - y_{12})(E_{11} - E_{21}) \\
&= 0.5 - 0.01 \times 0.3799 \times 0.6201 \times (-1.02) \times -2 \\
&= 0.5 - 4.8058 \times 10^{-3} \\
W_{21} &= 0.4952
\end{aligned}$$

And

$$b = b = 0.5$$

Inference:

Let us predict what should be our allocation tomorrow with the weight we learned from the toy data. Now the X matrix will have daily returns for 2 assets for today and yesterday.

$$X = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix}, \quad W = \begin{pmatrix} 0.5094 \\ 0.4952 \end{pmatrix}, \quad b = 0.5$$

1. **LeakyReLU**

$$\text{LeakyReLU}(X) \Rightarrow Y = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix}$$

2. **Linear Layer**

$$Z = YW + b$$

$$Z = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix} \begin{pmatrix} 0.5094 \\ 0.4952 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$Z = \begin{pmatrix} 3.0281 \\ 5.0373 \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$Z = \begin{pmatrix} 3.5281 \\ 5.5373 \end{pmatrix}$$

3. **Softmax**

$$S = \text{Softmax}(Z)$$

$$S = \begin{pmatrix} \frac{e^{3.5281}}{e^{3.5281}+e^{5.5373}} \\ \\ \frac{e^{5.5373}}{e^{3.5281}+e^{5.5373}} \end{pmatrix}$$

$$S = \begin{pmatrix} 0.1182 \\ 0.8818 \end{pmatrix}$$

This means we should allocate 11.82% of our wealth to asset 1 and 88.18% of our wealth to asset 2 tomorrow.

**Here we used data for 2 days to train ie the rolling window had size of 2, In practice, we train on much larger data. Size of the rolling window is a hyper-parameter and Uysal, A.S (2021) recommends a rolling window size of 150 days*

3 Data Set description

Uysal, A.S (2021) tests the performance of the model free approach on both - a simulation dataset and real market data. Next subsections describe both of these datasets in some detail.

3.1 Simulation Dataset Description

Uysal, A.S (2021) simulates a seven-asset universe where the returns of the assets follow a multi-variate normal distribution, and is independently and identically distributed for each trading day.

To mimic real market environment, the distribution parameters are determined by the mean and covariance matrix of daily returns of seven ETFs from 2011 to 2021: VTI, IWM, AGG, LQD, MUB, DBC and GLD. The expected daily returns for the seven assets in the simulation are 0.059%, 0.013%, -0.011%, 0.022%, 0.056%, 0.017% , 0.017%, respectively.

3.2 Real Market Dataset Description

Uysal, A.S (2021) utilizes daily returns from seven Exchange-Traded Funds (ETFs) as proxies for representing conditions in the stock, bond, and commodity markets. These ETFs include VTI (Vanguard Total Stock Market ETF), IWM (iShares Russell 2000 ETF), AGG (iShares Core U.S. Aggregate Bond ETF), LQD (iShares iBoxx Investment Grade Corporate Bond ETF), MUB (iShares National Muni Bond ETF), DBC (Invesco DB Commodity Index Tracking Fund), and GLD (SPDR Gold Shares). The ETF performance statistics were analyzed over the time span from 2011 to 2021.

ETFs	Return	Volatility	Sharpe	MDD	Calmar Ratio	Return/Ave.DD
VTI	0.1410	0.1759	0.7677	0.3500	0.3849	4.1473
IWM	0.1248	0.2194	0.5418	0.4113	0.2767	1.7915
AGG	0.0335	0.0401	0.6995	0.0958	0.2913	2.1182
LQD	0.0537	0.0724	0.6650	0.2176	0.2209	2.1832
MUB	0.0417	0.0514	0.7052	0.1368	0.2642	1.8995
DBC	-0.0477	0.1611	-0.3273	0.6614	-0.0777	-0.1374
GLD	0.0204	0.1584	0.0951	0.4556	0.0330	0.0561

Figure 2: Annualized ETF performance statistics over the period 2011-2021 [*Uysal, A.S (2021)*]

4 Neural Network Training and Hyperparameters

Uysal, A.S (2021) presents the computational results over the last ten years of daily data, focusing on the period 2011-2016/12 for hyperparameter search and 2017-2021 for out-of-sample testing.

In the backtesting framework, the neural network models undergo re-training every 25 days, utilizing a look-back window of 150 days.

The in-sample period has been divided into training (2011-2014/12) and validation (2015-2016/12) phases to facilitate the hyperparameter selection process for the network.

Given the capabilities of shallow feed-forward neural networks as stipulated by the universal approximation theorem—namely, their ability to approximate any continuous function—a network consisting of 32 neurons in a single layer was chosen to avoid unnecessary complexity. This paper meticulously adjusted the learning rate (η) and the number of training steps (n), with a focus on balancing the speed of learning against the risks of overstepping or slow convergence.

The hyperparameter tuning spanned a range of learning rates $\eta \in \{50, 100, 150, 250, 300, 500\}$ and training steps $n \in \{5, 10, 25, 50\}$. The intention was to identify a hyperparameter set that would yield robust performance across both the training and validation datasets.

To select the hyperparameters, this paper first identified parameter sets that performed within the top half of both the train and validation sets. Then, it focused on those that demonstrated the best performance in the validation set, ensuring generalizability. It was observed that for portfolios optimized for the Sharpe ratio, a learning rate of 150 resulted in satisfactory performance with ten training steps. For the cumulative return function, the optimal parameter set was determined to be a learning rate of 300 with 25 training steps.

Risk parity is used as a benchmark to evaluate the performance of the model free approach.

5 Results for Model Free Approach

5.1 Results on simulated Data

The following strategies are executed on simulated data spanning 175 days, using model-free end-to-end learning. The selection of hyperparameters for the end-to-end neural networks is as follows:

- Number of neurons in the hidden layer: 32
- Learning rate: 10
- Number of steps: 50
- Rolling window for training purpose: 150 days
- Test window: 5 days

For every 5-day period, *Uysal, A.S (2021)* train the neural network with the data of 150 days immediately previous to the period of interest. They keep the same weights for the 5-day period, and repeat the same process for the next period. To test the robustness of performance, they run 100 seeds on both, and plot the best performing seed, worst performing seed, as well as the median and average results. They use portfolio sharpe ratio and portfolio return as objective functions to optimize.

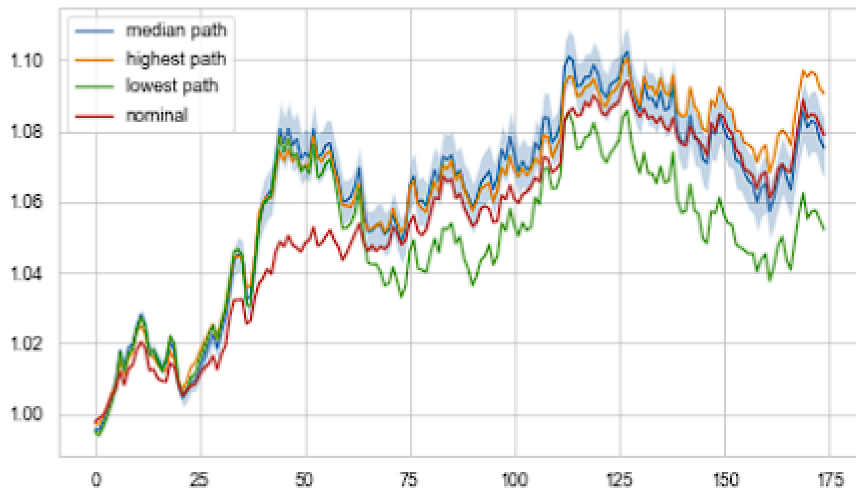

Figure 3: Computational result on simulated data when the tuning objective is chosen to be portfolio Sharpe ratio [*Uysal, A.S (2021)*]

We want to compare the performance of model-free method and the risk parity benchmark. We set up some hypotheses and test them with data provided by Uysal, A.S (2021). Geometric average return divided by average drawdown is used as the key measure of performance. Uysal, A.S (2021) train the neural network parameters to maximize the Sharpe ratio.

Hypothesis Testing

The average performance (in terms of geometric average return over average drawdown) of the model-free end-to-end method is no less than that of risk-parity strategy

$$H_0 : \hat{R}_{end-to-end} \geq R_{parity}$$

and

$$H_\alpha : \hat{R}_{end-to-end} < R_{parity}$$

Applying nominal risk parity on the simulated dataset leads to a geometric average return over average drawdown of 16.901. Whereas the model free end to end approach results in the metric of 7.877 with standard deviation of 1.522. The test statistic is

$$Z = \frac{7.877 - 16.901}{\frac{1.522}{\sqrt{100}}} = -59.29$$

The corresponding p value is 0.00001. Which means we can reject the null hypothesis.

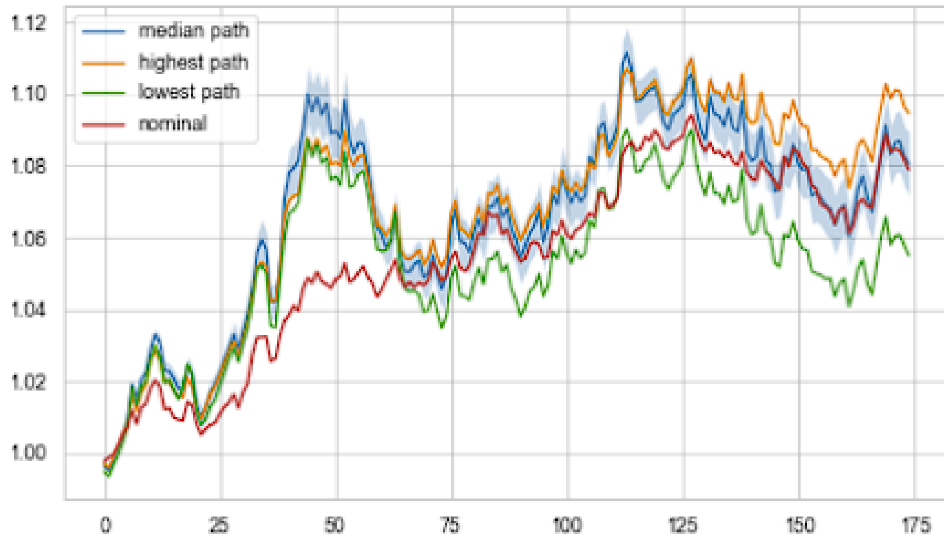

Figure 4: Computational result on simulated data when the tuning objective is chosen to be portfolio return [*Uysal, A.S (2021)*]

We have sufficient evidence to conclude that the end-to-end model free approach under-performs the risk parity, at 1% significance level.

5.2 Results on Real Market Data

	2017-2021/06			
Portfolio	Return	Volatility	Sharpe Ratio	Return/Ave.DD
Nominal RP	0.0609	0.0604	0.7906	3.2988
Model-free (cumulative return)	0.0040	0.1812	-0.0463	-0.0633
Model-free (Sharpe ratio)	0.0508	0.1194	0.3168	0.6190

Table 1: Portfolio Performance [*Uysal, A.S (2021)*]

As we can see from the table, the model free end to end approach, is not able to beat the risk parity benchmark. We need to improve its structure to improve the performance. It lacks any structure that guides the allocation, resulting in easy over-fitting with local structures that can be harmful when market dynamic shifts.

6 Conclusion

The model free end to end approach is not as effective as the benchmark risk parity optimization strategy. The model's inability to capture the market dynamics and poor approximation of allocation function can be due to its shallow structure. Increasing number of linear layers can help to improve its performance.

7 Improvements

In-order to improve the performance, *Uysal, A.S (2021)* enhances the structure of model free approach by adding an implicit layer (with risk budgeting portfolio optimization). They call it model based approach. Its structure is as follows -

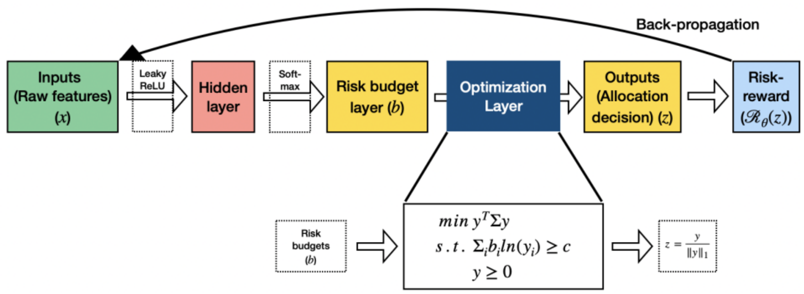

Figure 5: Computational graph of model-based approach [*Uysal, A.S (2021)*]

This new model was able to outperform risk parity benchmark in both simulation data and real market data.

7.1 Risk budgeting Portfolio Optimization

- A risk budgeting portfolio is one where the allocation is made so that the contribution of risk from each asset matches the pre-defined risk budget.

- Unlike mean-variance portfolio model, the risk budgeting portfolio problem does not require asset return forecasts as input, and it is robust misspecifications in covariance. This makes it immune from the errors in asset returns estimations leading to better performance

- The benefit of risk budgeting is the portfolio manager can include his subjective assessment in the calculations by defining or changing the risk budgets

- A special case of risk budgeting is risk-parity, where each asset has the same degree of risk contribution. Risk parity portfolios have gain increasing attention in recent years. Compared to the traditional mean-variance Markowitz approach, the risk parity portfolio does not depend on return estimations, and provide a more robust performance over different market circumstances, often leading to a higher Sharpe ratio.

- On the other hand, some critics point out that risk-parity could be sensitive to the underlying asset universe. Since risk-parity focuses solely on volatility, the portfolio can be hurt by assets with low or negative returns (let alone if an asset has negative return and low volatility). For example, when the bond market corrects (as the yield rises), the bonds will provide negative returns, yet as they have low risk associated with them, they will be included in the risk parity based portfolio. This is a major problem.

- According to the authors, A general risk budgeting portfolio can potentially mitigate the drawback by allocating less risk budget to the undesired assets.
- In the model based approach (with risk budgeting implicit layer), as the risk budgets are learned from the data(daily returns), the undesired assets will get assigned lower risk budgets and consequently - lower or near zero allocations. As the risk (co-variances and variance) changes, the risk budgets change. Which in turn cause the portfolio allocation to get altered.
- **Risk budgeting does not always outperform Markowitz approach**. The choice between risk budgeting and Markowitz portfolio optimization depends on the investor's preferences, risk tolerance, and the specific goals of the investment strategy.
- Some investors may prefer the simplicity and clarity of a risk budgeting approach. They may find the inclusion of subjective knowledge of portfolio manager (through the allocation of risk budgets) appealing.
- While others may find the diversification benefits and mathematical rigor of Markowitz optimization appealing. In practice, some investors might even use a combination of both approaches to tailor their portfolio management strategy to their specific needs.

8 Risks and Challenges

8.1 Associated Risks and Constraints

The single stage end to end approach uses more computational resources and is slower than the traditional approach(eg. linear regression + markowitz) and hence the traditional approach will be preferred in case of limited computational resources or time constraints. Even though the end-to-end approach is generally better, in the industry a traditional two-step approach is preferred.

8.2 Challenges Faced While Using AI in Industry

The major hindrance to the use of AI in the financial industry is the black box appearance of deep learning models. The lack of interpretability is a major concern when money is at stake. People are unwilling to bet their money on something they do not understand.

Another important problem is - how computationally expensive the machine learning model is compared to a simple regression model. People would not opt for machine learning models if a simple model behaves sufficiently good most of the time.

Interpretable AI and efficient model implementation are active area of research to mitigate the above issues.

9 Reference

Uysal, A.S., Li, X. & Mulvey, J.M. "End-to-end risk budgeting portfolio optimization with neural networks." *Ann Oper Res* (2023). (https://arxiv.org/abs/2107.04636)

Bahri Sales, J., Pakmaram, A., & Valizadeh, M. (2018). "Selection and Portfolio Optimization by Mean–Variance Markowitz Model and Using the Different Algorithms." *Financial Knowledge of Securities Analysis*, 11(37), 43-53.

Toy Code for model free approach https://colab.research.google.com/drive/1bMgKXGBvse9cgXVviR19YC8NV-bLekjP?usp=sharing

APPENDIX

A Calculation of gradients for backpropagation

What is $\frac{\partial L}{\partial W_{11}}$?

Given $L = -(S_{11}E_{11} + S_{21}E_{21})$, we have:

$$\frac{\partial L}{\partial W_{11}} = -E_{11}\frac{\partial S_{11}}{\partial W_{11}} - E_{21}\frac{\partial S_{21}}{\partial W_{11}}$$

(A) Calculation of $\frac{\partial S_{11}}{\partial W_{11}}$

For the softmax function, we use the chain rule:

$$\frac{\partial S_{11}}{\partial W_{11}} = \frac{\partial S_{11}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial W_{11}} + \frac{\partial S_{11}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial W_{11}}$$

Given $S_{11} = \frac{e^{Z_{11}}}{e^{Z_{11}}+e^{Z_{21}}}$, we find the partial derivatives:

$$\frac{\partial S_{11}}{\partial Z_{11}} = \frac{e^{Z_{11}}(e^{Z_{11}} + e^{Z_{21}}) - e^{Z_{11}}e^{Z_{11}}}{(e^{Z_{11}} + e^{Z_{21}})^2}$$

$$\frac{\partial S_{11}}{\partial Z_{11}} = \frac{e^{Z_{11}}e^{Z_{21}}}{(e^{Z_{11}} + e^{Z_{21}})^2} = S_{11}.S_{21}$$

As $Z_{11} = W_{11}Y_{11} + W_{21}Y_{12} + b$

$$\frac{\partial Z_{11}}{\partial W_{11}} = Y_{11}$$

$$\frac{\partial S_{11}}{\partial Z_{21}} = \frac{(e^{Z_{11}} + e^{Z_{21}})(0) - e^{Z_{11}}(e^{Z_{21}} + 0)}{(e^{Z_{11}} + e^{Z_{21}})^2}$$

$$\therefore \frac{\partial S_{11}}{\partial Z_{21}} = -\frac{e^{Z_{11}}e^{Z_{21}}}{(e^{Z_{11}} + e^{Z_{21}})^2} = -S_{11}.S_{21}$$

As $Z_{21} = W_{11}Y_{21} + W_{21}Y_{22} + b$

$$\frac{\partial Z_{21}}{\partial W_{11}} = Y_{21}$$

Substituting all the derivatives we calculated above in the equation,

$$\frac{\partial S_{11}}{\partial W_{11}} = S_{11}S_{21}Y_{11} + (-S_{11}S_{21})Y_{21}$$

$$\frac{\partial S_{11}}{\partial W_{11}} = S_{11}S_{21}(Y_{11} - Y_{21})$$

(B) Calulation of $\frac{\partial S_{21}}{\partial W_{11}}$

For the softmax function, we use the chain rule:

$$\frac{\partial S_{21}}{\partial W_{11}} = \frac{\partial S_{21}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial W_{11}} + \frac{\partial S_{21}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial W_{11}}$$

As $S_{21} = \frac{e^{Z_{21}}}{(e^{Z_{11}}+e^{Z_{21}})}$

$$\frac{\partial S_{21}}{\partial Z_{11}} = \frac{(e^{Z_{11}} + e^{Z_{21}})(0) - e^{Z_{21}}(e^{Z_{11}} + 0)}{(e^{Z_{11}} + e^{Z_{21}})^2}$$

$$\frac{\partial S_{21}}{\partial Z_{11}} = -S_{11}S_{21}$$

As $\frac{\partial Z_{11}}{\partial W_{11}} = Y_{11}$ and $\frac{\partial Z_{21}}{\partial W_{11}} = Y_{21}$

$$\frac{\partial S_{21}}{\partial Z_{21}} = \frac{(e^{Z_{11}} + e^{Z_{21}})(e^{Z_{21}}) - e^{Z_{21}}(0 + e^{Z_{21}})}{(e^{Z_{11}} + e^{Z_{21}})^2}$$

$$\frac{\partial S_{21}}{\partial Z_{21}} = S_{11}.S_{21}$$

Substituting all the derivatives we calculated above in the equation,

$$\frac{\partial S_{21}}{\partial W_{11}} = (-S_{11}S_{21})Y_{11} + (S_{11}S_{21})Y_{21}$$

$$\frac{\partial S_{21}}{\partial W_{11}} = S_{11}S_{21}(Y_{21} - Y_{11})$$

As: $\frac{\partial L}{\partial W_{11}} = -E_{11}\frac{\partial S_{11}}{\partial W_{11}} - E_{21}\frac{\partial S_{21}}{\partial W_{11}}$

$$\therefore \frac{\partial L}{\partial W_{11}} = -E_{11}S_{11}S_{21}(Y_{11} - Y_{21}) - E_{21}S_{11}S_{21}(Y_{21} - Y_{11})$$

$$\frac{\partial L}{\partial W_{11}} = S_{11}S_{21}(Y_{21} - Y_{11})(E_{11} - E_{21})$$

Now we can update the first weight parameter with the following equation -

$$W_{11} = W_{11} - \alpha\frac{\partial L}{\partial W_{11}}$$

$$W_{11} = W_{11} - \alpha \cdot S_{11}S_{21}(Y_{21} - Y_{11})(E_{11} - E_{21})$$

What is $\frac{\partial L}{\partial W_{21}}$?

As:

$$L = -(S_{11}E_{11} + S_{21}E_{21})$$

$$\frac{\partial L}{\partial W_{21}} = -E_{11}\frac{\partial S_{11}}{\partial W_{21}} - E_{21}\frac{\partial S_{21}}{\partial W_{21}}$$

(A)

$$\frac{\partial S_{11}}{\partial W_{21}} = \frac{\partial S_{11}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial W_{21}} + \frac{\partial S_{11}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial W_{21}}$$

As

$$S_{11} = \frac{e^{Z_{11}}}{e^{Z_{11}} + e^{Z_{21}}}$$

$$\text{and } \frac{\partial S_{11}}{\partial Z_{11}} = S_{11}.S_{21}$$

As

$$Z_{11} = W_{11}Y_{11} + W_{21}Y_{12} + b$$

$$\frac{\partial Z_{11}}{\partial W_{21}} = Y_{12}$$

$$also\ \ \frac{\partial S_{11}}{\partial Z_{21}} = -S_{11}S_{21}$$

As

$$Z_{21} = W_{11}Y_{21} + W_{21}Y_{22} + b$$
$$\frac{\partial Z_{21}}{\partial W_{21}} = Y_{22}$$

Substituting all these derivatives in the equation -

$$\frac{\partial S_{11}}{\partial W_{21}} = S_{11}S_{21}Y_{12} - S_{11}S_{21}Y_{22}$$

$$\frac{\partial S_{11}}{\partial W_{21}} = S_{11}S_{21}(Y_{12} - Y_{22})$$

(B)

$$\frac{\partial S_{21}}{\partial W_{21}} = \frac{\partial S_{21}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial W_{21}} + \frac{\partial S_{21}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial W_{21}}$$

As

$$\frac{\partial S_{21}}{\partial Z_{11}} = -S_{11}S_{21}$$
$$\frac{\partial S_{21}}{\partial Z_{21}} = S_{11}S_{21}$$
$$\frac{\partial Z_{11}}{\partial W_{21}} = Y_{12}$$
$$\frac{\partial Z_{21}}{\partial W_{21}} = Y_{22}$$

$$\therefore \frac{\partial S_{21}}{\partial W_{21}} = -S_{11}S_{21}Y_{12} + S_{11}S_{21}Y_{22}$$
$$\frac{\partial S_{21}}{\partial W_{21}} = S_{11}S_{21}(Y_{22} - Y_{12})$$

As:

$$\frac{\partial L}{\partial W_{21}} = -E_{11}\frac{\partial S_{11}}{\partial W_{21}} - E_{21}\frac{\partial S_{21}}{\partial W_{21}}$$
$$\frac{\partial L}{\partial W_{21}} = -E_{11}S_{11}S_{21}(Y_{12} - Y_{22}) - E_{21}S_{11}S_{21}(Y_{22} - Y_{12})$$
$$\frac{\partial L}{\partial W_{21}} = S_{11}S_{21}(Y_{22} - Y_{12})(E_{11} - E_{21})$$

Now we can update second weight parameter using following equation -

$$W_{21} = W_{21} - \alpha\frac{\partial L}{\partial W_{21}}$$
$$W_{21} = W_{21} - \alpha S_{11}S_{21}(Y_{22} - Y_{12})(E_{11} - E_{21})$$

What is $\frac{\partial L}{\partial b}$?

As:

$$L = -(S_{11}E_{11} + S_{21}E_{21})$$

$$\frac{\partial L}{\partial b} = -E_{11}\frac{\partial S_{11}}{\partial b} - E_{21}\frac{\partial S_{21}}{\partial b}$$

(A)

$$\frac{\partial S_{11}}{\partial b} = \frac{\partial S_{11}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial b} + \frac{\partial S_{11}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial b}$$

As:

$$\frac{\partial S_{11}}{\partial Z_{11}} = S_{11}.S_{21}$$

$$\text{and } \frac{\partial Z_{11}}{\partial b} = 1$$

,

$$\frac{\partial S_{11}}{\partial Z_{21}} = -S_{11}.S_{21}$$

$$\text{and } \frac{\partial Z_{21}}{\partial b} = 1$$

,

$$\frac{\partial S_{11}}{\partial b} = S_{11}S_{21}.1 + (-S_{11}S_{21}).1 = 0$$

(B)

$$\frac{\partial S_{21}}{\partial b} = \frac{\partial S_{21}}{\partial Z_{11}}\frac{\partial Z_{11}}{\partial b} + \frac{\partial S_{21}}{\partial Z_{21}}\frac{\partial Z_{21}}{\partial b}$$

As

$$\frac{\partial S_{21}}{\partial Z_{11}} = -S_{11}S_{21}$$

and

$$\frac{\partial Z_{11}}{\partial b} = 1$$

$$\frac{\partial Z_{21}}{\partial b} = 1$$

$$\frac{\partial S_{21}}{\partial Z_{21}} = S_{11}.S_{21}$$

Substituting all these derivatives in the equation -

$$\frac{\partial S_{21}}{\partial b} = -S_{11}S_{21}.1 + S_{11}S_{21}.1 = 0$$

As

$$\frac{\partial L}{\partial b} = -E_{11}\frac{\partial S_{11}}{\partial b} - E_{21}\frac{\partial S_{21}}{\partial b}$$

$$\frac{\partial L}{\partial b} = -E_{11}.0 - E_{21}.0$$

$$\frac{\partial L}{\partial b} = 0$$

Now we can find

$$b = b - \alpha\frac{\partial L}{\partial b}$$

$$b = b - \alpha.0$$

$$b = b$$

B Questions to test your understanding

1. What is the objective of Markowitz Mean Variance Portfolio Optimization?

A) Maximize returns

B) Maximize variance

C) Minimize variance

D) Minimize variance for given level of expected returns

2. What is the issue with 2 stage portfolio optimization process, which makes us prefer end to end approach?

A) It is computationally expensive

B) It is fast

C) It is slow

D) It can lead to error accumulation and subsequently suboptimal allocations.

3. What is the function of Softmax layer in the structure of model free neural network?

A) Make the neural network fast

B) Make the neural network slow

C) Normalize its inputs so that we can treat the output of softmax as allocation decision.

D) It has no role.

4. Which of the following is *not* a drawback of model free end to end approach?

A) It follows single stage optimization

B) It is not able to outperform the benchmark- risk parity

C) It is computationally expensive than simple linear regression

D) It is less interpretable

5. Which is *not* a major risk associated with using deep learning models for financial domain?

A) High computational costs

B) Low interpretability

C) Overfitting

D) Better performance

C Toy example to check your understanding

Fill in the blanks to check your understanding-

$$X = \begin{pmatrix} 2 & 1 \\ 4 & -2 \end{pmatrix}, \quad E = \begin{pmatrix} 4 \\ 6 \end{pmatrix}, \quad W = \begin{pmatrix} 0.9 \\ 0.1 \end{pmatrix}, \quad b = 0.3$$

1. **LeakyReLU**
   As $\text{LeakyReLU}(x) = \max(x, 0) + \min(x, 0) \times 0.01$

$$\text{LeakyReLU}(X) = Y = \begin{pmatrix} 2 & 1 \\ 4 & -0.02 \end{pmatrix}$$

2. **Linear Layer**
   $Z = X \cdot W + b$

$$Z = \begin{pmatrix} 2 & 1 \\ 4 & -0.02 \end{pmatrix} \begin{pmatrix} 0.9 \\ 0.1 \end{pmatrix} + \begin{pmatrix} 0.3 \\ 0.3 \end{pmatrix}$$

$$Z = \begin{pmatrix} \quad \\ \quad \end{pmatrix}$$

3. **Softmax**
   $S = \text{Softmax}(Z)$

$$S = \begin{pmatrix} \quad \\ \quad \end{pmatrix}$$

$$\text{Therefore, } S = \begin{pmatrix} \quad \\ \quad \end{pmatrix}$$

For iteration 1:

$$
\begin{aligned}
s_{11} &=, \quad s_{21} = \\
y_{11} &= 2, \quad y_{12} = 1, \\
y_{21} &= 4, \quad y_{22} = -0.02 \\
E_{11} &= 4, \quad E_{21} = 6 \\
W_{11} &= 0.9, \quad W_{21} = 0.1, \alpha = 0.01
\end{aligned}
$$

As

$$W_{11} = W_{11} - \alpha s_{11} s_{21} (y_{21} - y_{11})(E_{11} - E_{21})$$

$$= W_{11} \qquad\qquad\qquad\qquad =$$

Similarly,

$$
\begin{aligned}
W_{21} &= W_{21} - \alpha s_{11} s_{21} (y_{22} - y_{12})(E_{11} - E_{21}) \\
&= \\
W_{21} &=
\end{aligned}
$$

And

$$b = b = 0.3$$

Inference:

Let us predict what should be our allocation tomorrow with the weight we learned from the toy data. Now the X matrix will have daily returns for 2 assets for today and yesterday.

$$X = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix}, \quad W = \qquad\qquad, \quad b = 0.3$$

1. **LeakyReLU**

$$\text{LeakyReLU}(X) \Rightarrow Y = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix}$$

2. **Linear Layer**

$$Z = XW + b$$

$$Z = \begin{pmatrix} 4 & 2 \\ 6 & 4 \end{pmatrix} \begin{pmatrix} \quad \\ \quad \end{pmatrix} + \begin{pmatrix} 0.3 \\ 0.3 \end{pmatrix}$$

$$Z =$$

3. **Softmax**

$S = \text{Softmax}(Z)$

$$S = \begin{pmatrix} \quad \end{pmatrix}$$

This means we should allocate $\quad$ % of our wealth to asset 1 and $\quad$ % of our wealth to asset 2 tomorrow.