Final Project: Machine Learning in Portfolio Selection

Snir Lugassy 206312506, Anna Gurevich 322025123, Eden Nagar 312589815

August, 2023

Overview

The primary objective of this project is to develop and evaluate effective models for portfolio selection. Our approach utilized a neural network to directly estimate portfolio weights while using the Sharpe ratio as the objective function during training.

Data

We downloaded stock price data from Yahoo Finance between July 1, 2018, and June 30, 2023. The last month of June was considered as our unseen test dataset.

In our models, we were primarily interested in the relative returns of the stocks. Therefore, we kept only the “Adj Close” columns, representing the adjusted closing price of a stock on someday, and calculated the percentage of price change on consecutive days.

Experimental Approach:

We started our project with a comprehensive brainstorming session. During this, each team member suggested algorithms and techniques they believed would enhance our portfolio prediction capabilities. The proposed approaches were MVP, Boosted Trees, MLP, online learning, regression, and more. After brainstorming, we each took responsibility for specific methods—implementing and testing them on designated datasets. After this evaluation phase, we set up an additional meeting to share and discuss our findings. Armed with insights from each strategy, we combined and integrated several methods. Throughout the project, we maintained weekly check-ins to monitor progress.

Methods

We developed and evaluated several models to estimate a good portfolio in terms of Sharpe ratio and minimal variance. Eventually, we have decided to submit (3), following the best performance over our test data (unseen).

1. Linear models
   Predicting each stock price using a linear model and a constant time window for fitting the model, then using next-day prediction to build a high-return portfolio.
2. Minimap Variance Portfolio (MVP)
   Solving a system to find the minimum variance portfolio using previous returns and estimated covariance matrix
3. Neural network (Multi-layer Perceptron)
   Training a neural network to estimate the portfolio weights directly. Using the Sharpe ratio as the network's loss (objective) function.
4. 1D Convolutional neural network (CNN)
   Similar to the NN approach, but using a sequence of relative returns in the network inputs and applying one-dimensional convolutions over the stock returns.
5. 2D Convolutional neural network
   Same as before but applying a convolution on all the stocks returns.
6. Recurrent neural network
   Similar to the NN approach but using a sequence of relative returns in the network inputs and LSTM cells.
7. Online learning for choosing between different models
   Given several models, train the model to choose between the different portfolios.

Neural Network Training (Method 3)

We used a neural network to estimate profitable portfolio weights, which takes a vector of stock returns as an input and outputs the portfolio weights directly. Since we wanted to optimize the Sharpe ratio, outputting the weights directly allowed us to take the Sharpe ratio as an objective function and apply back-propagation, improving our goal.

The network architecture is a Multi-Layer Perceptron, where the layers are equal in size. Since we wanted the network to learn a complex interaction between the stocks returns, we avoided creating a bottleneck network in which information is being compressed and decompressed.

Our primary focus while training the network was enabling a high level of generalization using a small amount of training data. Therefore, we heavily applied augmentations over the inputs, using random noise, random zeros, and random negative market (flipping the market returns for all stocks). Moreover, we have added a dropout layer after each linear layer in the network, which helped lower the tendency to overfit.

Consider the relative returns of all stocks in time t, as x_t , and a neural network F with tunable weights θ. Forward-pass the inputs to receive weights estimation:

$$w_t = F(x_t)$$

The next day's returns were considered as ground truth and allowed us to calculate the true return and Sharpe ratio using the estimated portfolio:

$$r_t = w_t \cdot x_{t+1}$$

$$s_t = Avg(r_t) \,/\, Var(r_t)$$

Our objective is to maximize the Sharpe ratio, considering the network weights:

$$max_{\theta}(s_t)$$

This was achieved using backpropagation and the Adam optimizer.

Hyperparameters Optimizations

We conducted an extensive hyperparameter search to find the best-performing network, using the test data Sharpe ratio and variance as the maximization objective.

The following parameters were included in the search

- Network depth
- Learning rate
- Learning rate scheduling factor
- Batch size
- Dropout layer rate
- Using skip connection (true/false)
- Using batch normalization (true/false)
- Augmentations probabilities

In total, we trained 400+ different networks using a single GPU.