Empirical Asset Pricing via Machine Learning

1. Methodology

Three fundamental elements:

- First is the statistical model describing a method's general functional form for risk premium predict-ions.
- The second is an objective function for estimating model parameters.
- The third element in each subsection describes computational algorithms for efficiently identifying the optimal specification among the permutations encompassed by a given method.

Objective:

- Our objective is to isolate a representation of $E_t(r_{i,t+1})$ as a function of predictor variables that maximizes the out-of-sample explanatory power for realized $r_{i,t+1}$.

1.1 Sample splitting and tuning via validation

- Hyperparameters (i.e. tuning parameters) include, for example, the penalization parameters in lasso and elastic net, the number of iterated trees in boosting, the number of random trees in a forest, and the depth of the trees.

In particular, we divide our sample into three disjoint time periods that maintain the temporal ordering of the data.

- The first, or "training," subsample is used to estimate the model subject to a specific set of tuning parameter values.
- The second, or "validation," sample is used for tuning the hyperarameters. **Next, we calculate the objective function based on forecast errors from the validation sample, and iteratively search for hyperparameters that optimize the validation objective.**
- The third, or "testing" subsample, which is used for neither estimation nor tuning, is truly out of sample and thus is used to evaluate a method's predictive performance.

1.2 Simple linear

1.3 Penalized linear

- The simple linear model is bound to fail in the presence of many predictors, for example, predictors *P* approaches the number of observations *T*. It begins to overfit noise rather than extracting signal.
- Crucial for avoiding overfit is reducing the number of estimated parameters.
- This "regularization" of the estimation problem mechanically deteriorates a model's in-sample performance in hopes that **it improves its stability out of sample**.

1.4 Dimension reduction: PCR and PLS

- **Penalized linear models can produce suboptimal forecasts when predictors and highly correlated.** A simple example of this problem is a case in which all of the predictors are equal to the forecast target plus an iid noise term.
- **Advantage:** Forming linear combinations of predictors helps reduce noise to better isolate the signal in predictors and helps decorrelate otherwise highly dependent predictors.
- **Principal components regression (PCR)** consists of a two-step procedure. First, PCA combines regressors into a small set of linear combinations. Second, a few leading components are used in standard predictive regression. **A drawback of PCR** is that it fails to incorporate the ultimate statistical objective--forecasting returns.

1.5 Generalized linear

$$r_{i,t+1} - \hat{r}_{i,t+1} = \underbrace{g^{\star}\left(z_{i,t}\right) - g\left(z_{i,t};\theta\right)}_{\text{approximation error}} + \underbrace{g\left(z_{i,t};\theta\right) - g\left(z_{i,t};\hat{\theta}\right)}_{\text{estimation error}} + \underbrace{\epsilon_{i,t+1}}_{\text{intrinsic error}} .$$

- **Intrinsic error is irreducible.** It is the genuinely unpredictable component of returns associated with news arrival and other sources of randomness in financial markets.
- **Estimation error**, which arises due to sampling variation, is determined by the data. **It is potentially reducible by adding new observations**, though this may not be under the econometrician's control.
- Approximation error is directly controlled by the econometrician and **is poentially reducible by incorporating more flexible specifications** that improve the model's ability to approximate the true model.

1.6 Boosted regression trees and random forests

- The generalized linear model captures individual predictors' nonlinear impact on expected returns, **but does not account for interactions among predictors**.

Two "ensemble" tree regularizers that combine forecasts from many different trees into a single forecast.

- The first regularization method is "boosting", which recursively combines forecasts from many oversimplified trees.
- Like boosting, a random forest is an ensemble method that combines forecasts from many different trees. It is a variation on a more general procedure known as bootstrap aggregation, or "bagging".

1.7 Neural networks

Arguably the most powerful modeling device in machine learning, neural networks have theoretical underpinning as "universal approximators" for any smooth predictive association.

Our analysis focuses on traditional "feed-forward" networks

- These consist of an **"input layer" of raw predictors, one or more "hidden layers" that interact and nonlinearly transform the predictors, and an "output layer" that aggregates hidden layers into an ultimate outcome prediction**.
- One makes many choices when structuring a neural network, including the number of hidden layers, the number of neurons in each layer, and which units are connected.

- On the other hand, in small data sets simple networks with only a few layers and nodes often perform best.
- **Selecting a successful network architecture by cross-validation is in general a difficult task. It is unrealistic and unnecessary to find the optimal network by searching over uncountably many architectures.**

Nonlinear activation function

- Sigmoid, hyperbolic, softmaxa. This passage use rectified linear unit (ReLU)

Empirical Asset Pricing via Machine Learning

1. Methodology

In most general form, we describe an asset's excess return as an additive prediction error model:

$$r_{i,t+1} = E_t[r_{i,t+1}] + \epsilon_{i,t+1}$$

where $E_t[r_{i,t+1}] = g^*(z_{i,t})$ ⇒ P-dimensional vector

g^* ⇒ a flexible function of these predictors

Objective: max realized $r_{i,t+1}$

1.2 Simple Linear

$$g(z_{i,t};\theta) = z_{i,t}'\theta$$

Use a standard least square, or "l_2" objective function:

$$\mathcal{L}(\theta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}[r_{i,t+1} - g(z_{i,t};\theta)]^2 \quad (4)$$

1.2.1 Extension: Robust Objective functions

$$\mathcal{L}_w(\theta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} w_{i,t}[r_{i,t+1} - g(z_{i,t};\theta)]^2 \quad (5)$$

$w_{i,t}$

① we considerate sets $w_{i,t}$ inversely propotional to the number of stocks at time t.

② sets $w_{i,t}$ proportional to the equity market value of stock i.

Huber robust objective function, which can counteract the deleterious effect of heavy-tailed observation.

$$\mathcal{L}_H(\theta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} H[r_{i,t+1} - g(z_{i,t};\theta), \xi]$$

where $$H(x;\xi) = \begin{cases} x^2 & , \text{if } |x| \le \xi \\ 2\xi|x| - \xi^2 & , \text{if } |x| > \xi \end{cases}$$

1.3 Penalized linear

Penalized methods differ by appending a penalty to the original loss function:

$$\mathcal{L}(\theta;\cdot) = \mathcal{L}(\theta) + \phi(\theta;\cdot) \quad (7)$$

~~There are~~ "Elastic net" Penalty:

$$\phi(\theta;\lambda,\rho) = \lambda(1-\rho)\sum_{j=1}^{P}|\theta_j| + \frac{1}{2}\lambda\rho\sum_{j=1}^{P}\theta_j^2 \quad (8)$$

① The $\rho=0$ case corresponds to the lasso and use an absolute value, or "l_1", parameter penalization.

② The $\rho=1$ case corresponds to ridge regression, which uses an l_2 parameter penalization.

1.4 Dimension reduction: PCR and PLS

We reorganize the linear regression $r_{i,t+1} = z_{i,t}'\theta + \epsilon_{i,t+1}$ as

$$R = Z\theta + E \quad (9)$$

$R \to NT\times 1$ vector of $r_{i,t+1}$; Z is $NT\times P$ matrix of stacked predictors $z_{i,t}$

$E \to NT\times 1$ vector of $\epsilon_{i,t+1}$;

Dimension $P \longrightarrow$ much smaller number of K linear combinations

$$R = (Z\Lambda_K)\theta_K + \tilde{E} \quad (10)$$

- Λ_K is $P\times K$ matrix with columns $w_1, w_2, \ldots, w_K$
- Each w_j is the set of linear combination weights used to create the j-th predictive components.
- Likewise, θ_K is a $K\times 1$ vector rather than $P\times 1$.

PCR chooses the combination weights Λ_K recursively. The j^{th} linear combination solves:

$$w_j = \arg\max_{w} Var(Zw) \quad (11)$$

s.t. $w'w=1$, $Cov(Zw, Zw_l) = 0$, $l=1,2,\ldots,j-1$

The weight used to construct the j-th PLS component solve

$$w_j = \arg\max_{w} \; Cov^2(R, Zw)$$

s.t. $w'w = 1$, $Cov(Zw, Zw_l) = 0$, $l = 1, 2, \ldots, j-1$

1.5 Generalized linear

$$r_{j,t+1} - \hat{r}_{i,t+1} = \underbrace{g^*(z_{i,t}) - g(z_{i,t}; \theta)}_{\text{approximation error}} + \underbrace{g(z_{i,t}; \theta) - g(z_{i,t}; \hat{\theta})}_{\text{estimation error}} + \underbrace{\varepsilon_{i,t+1}}_{\text{intrinsic error}}$$

the simple linear form by adding a K-term spine series expansion of the predictors

$$g(z; \theta, p(\cdot)) = \sum_{j=1}^{P} p(z_j)' \theta_j \qquad (13)$$

- where $p(\cdot) = [p_1(\cdot), p_2(\cdot), \ldots, p_K(\cdot)]'$ is a vector of basis functions.
  - parameters are now a $K \times N$ matrix $\theta = (\theta_1, \theta_2, \ldots, \theta_N)$
  - we adopt a spline series of order two:
    $(1, z, (z-c_1)^2, (z-c_2)^2, \ldots, (z-c_{k-2})^2)$, where $c_1, \ldots c_{k-2}$ are knots

Our choice of penalization is specialized for the spline expansion setting and is known as the group lasso.

$$\phi(\theta; \lambda, K) = \lambda \sum_{j=1}^{P} \left(\sum_{k=1}^{K} \theta_{j,k}^2 \right)^{\frac{1}{2}} \qquad (14)$$

1.6 Boosted regression trees and random forest

the prediction of a tree T, with K "leaves" (terminal nodes) and depth L, can be written as

$$g(z_{i,t}; \theta, K, L) = \sum_{k=1}^{K} \theta_k 1_{\{z_{i,t} \in C_k(L)\}} \qquad (15)$$

- $C_k(L)$ is one of the K partitions of the data.
- The constant associated with partition k (denoted θ_k) is defined to be the sample average outcome in the partition.

We choose the most popular l_2 impunity for each branch of the tree:

$$H(\theta, C) = \frac{1}{|C|} \sum_{z_{i,t+1} \in C} (r_{i,t+1} - \theta)^2$$

- $|C|$ denotes the number of observations in set C.
- Given C, it is clear that the optimal choice of θ:

$$\theta = \frac{1}{|C|} \sum_{z_{i,t+1} \in C} r_{i,t+1}$$

1.7 Neural networks

Nonlinear activation function:

rectified linear unit (ReLU)

$$\text{ReLU}(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{otherwise} \end{cases}$$