4. Machine Learning

Contents

1 Learning objective:

- **Reading list:** Gu, Kelly, and Xiu (2020), Weigand (2019)
- Discuss and implement selected machine learning methods for return predictability, evaluate the outcome, and reflect on the implications for asset pricing

2 Suggested presentation 10 min

1. Empirical

Part I
Short version

3 Machine learning

3.1 Introduction

- Field of econometrics and asset pricing that's been explosive as of lately
- Useful field for asset pricing as it can
  - Handle enormous amounts of potentially relevant variables
  - Select among variables
  - Help us find an efficient model specification
- Drawbacks
  - Computational complexity
  - Doesn't tell us anything about economic intuition

3.2 Return predictability via ML

$$r^e_{it+1} = \underbrace{\mathbb{E}_t\left[r^e_{it+1}\right]}_{g^*(z_{it})} + \varepsilon_{it+1}$$

ML methods can help in learning the best specification and form of g^*
General goal is to improve with data, and to "learn" with respect to a goal.

- Hyperparameters
  - Not actual part of the model
  - Controls e.g. regularization to avoid overfit
  - "tuned", which is in learning
- Sample splitting
  - Three periods
  - Iterate between training and validation to "learn"
  - Test on OOS
  - Fixed, rolling, expanding

3.3 Machine learning techniques

Many methods, only dive into two of them, but briefly explain the progression.

- Linear method fails with many variables and overtfits to noise
- $\Rightarrow$ penalize objective function for non-zero variables, performing variable selection in some specifications
  - This removes variables with noise, but the variables that are kept are also noisy
- Averaging might make more sense

3.3.1 Dimension reduction

- Works by average, and therefore useful for correlated factors

$$R = (Z\Omega_K)\,\theta_K + \tilde{E}$$

Two methods:

- PCR
  - Two-step technique that aims to reproduce the predictor set with K components as:
  - $\omega_j = \underset{\omega}{\text{argmax}} \text{Var}\,[Z\omega]$
  - Clearly unsupervised
- PSL
  - Seeks maximal predictive association with the forecast target as per:
  - $\omega_j = \underset{\omega}{\text{argmax}} \text{Cov}\,[R, Z\omega]$
- With Ω_K estimated, the model is estimated by OLS and simple predictions, with K as the obvious hyperparameter.

3.3.2 Regression trees

- To include higher-order dimensionality, one might use GLM, but as this method can become unparsimonious from higher-order dimensionality being included combinatorically, one might instead use regression trees
- Fully nonparametric, handles non-linearities, including interactions
- Trees are formed by continually intersecting data based on certain criteria
- At extreme, all observations fit into one leaf/node $\Rightarrow$ regularization
- Random forest:
  - Ensemble method: combines multiple weak learners to create stronger learner
  - RF bootstraps samples of the data, fitting a tree to each bootstrap sample
  - Averages forecasts to stabilize performance and cancel out nice
  - Random aspect decorrelates trees by ensuring some degree of dropout

3.4 Performance assessment

- Obviously want to know how well the ML techniques perform
- As with classical return predictability, we might create an out-of-sample R^2 based on a benchmark such as the historical average. Gu et al argues that this is an artificially noise benchmark, and therefore uses benchmark of zero return

  $R^2_{OS} = 1 - \frac{\sum_{t=R_2+1}^{T} \sum_{i=1}^{N_{test,t}} \left(r^e_{it} - \hat{r}^e_{it}\right)^2}{\sum_{t=R_2+1}^{T} \sum_{i=1}^{N_{test,t}} \left(r^e_{it}\right)^2}$
- We can use diebold-mariano test with the forecast errors from different models to evaluate them against one another, which has relevance in terms of comparing e.g. nonlinear to linear models

- $d_{12t} = \frac{1}{N_{test,t}} \sum_{i=1}^{N_{test,t}} \hat{e}_{1it}^2 - \hat{e}_{2it}^2$
  Test $\mathbb{H}_0 : \mathbb{E}[d_{12t}] = 0$ through regression:
  $d_{12t} = \psi + \varepsilon_{12t}$
  Standard t-test on ψ with HAC standard errors to see if $\psi = 0$

$$t\left(\overline{d}_{12} = \frac{\overline{d}_{12}}{\sqrt{\text{Var}\left[\overline{d}_{12}\right]}} \xrightarrow{d} N(0,1)\right)$$

3.5 Variable importance

- Allows for a better peek inside what might otherwise be "black box"
- Allows for better connection to economic intuition through the variables
- Multiple ways to calculate:

1. Reduction in R_{OS}^2 from setting all values of a predictor to zero, with other estimates held fixed
2. SSD: $SSD_j = \sum_{t=R_2+1}^{T} \sum_{i=1}^{N_{test,t}} \left(\frac{\partial g(z;\theta)}{\partial z_j}|_{z_{it}}\right)^2$
3. Reduction in node impurity for trees
4. Partial-dependence based -> model agnostic

Placebo principle:

- Adds credibility to the model we've found if simulated variables do not change the importance ranking and are ranked lowest

Variable importance from Gu et al:

- Finds short-term price variables to be most important, which somewhat clashes with the EMH as momentum does, implicating either mispricing or behavior-based explanation
- Otherwise find liquidity variables to be important for stocks

Part II
Long version

4 Machine learning

4.1 Introduction

Machine Learning is a field of asset pricing and statistics that has been explosive as of lately. The field incorporates a large range of new methods that

- assist in finding proper specifications for predictability purposes
- handling the enormous amount of potential relevant variables and selecting among them
- while helping us find a parsimonious functional form

Of course, all these benefits do not come free of charge - Machine Learning is computationally very complex and demanding, and at the same time it doesn't tell us anything about economic intuition.

- Many variables can be correlated
  - Std. economic approaches will not work with a lot variables that are correlated.
  - ML techniques can help us with meaningful predictions using a lot of variables at the same time.
    - Encompassing variable selection and dimension reduction techniques
- Using machine learning we do not have the specify the functional form between expected return and a functional form taking some variables as input.
- ML is silent about economic mechanisms and equilibria.

4.2 Return predictability via ML

In general, an asset's excess returns can be expressed as

$$r^e_{it+1} = \underbrace{\mathbb{E}_t\left[r^e_{it+1}\right]}_{=g^*(z_{it})} + \varepsilon_{it+1}$$

Where the objective is to learn the best specification of the general function $g(\cdot)$, which machine learning techniques will help with.

The general goal of a machine learning model is to improve with data, or to "learn".

- For these purposes, all models rely on sample splitting and hyperparameters.
  - These are parameters that are not an actual part of the model, but used to control for example regularization in order to avoid overfitting.
  - Helps specifying the estimation procedure.
  - Often, these are the parameters that are "tuned" in order to "learn" the best model.

This is typically done over multiple periods

1. Training
   - Data or time period used to estimate (train) the model, given a particular choice of hyperparameters.
2. Validation period
   - trained model is used to form predictions, and the objective function from the forecast evaluation is computed.
   - Validation period is used to test the different choices of hyperparameters.
3. Testing

- Used to evaluate the methods' predictive ability

Hyperparameters are then chosen by iteration between the training and validation period to minimze the value of the objective function. The time periods can the be estimated on a fixed, rolling, expanding or hybrid scheme.

$$\underbrace{1, 2, \ldots, R_1}_{\text{training}}, \underbrace{R_1 + 1, R_1 + 2, \ldots, R_2}_{\text{validation}}, \underbrace{R_2 + 1, R_2 + 2, \ldots, T}_{\text{testing}}$$

4.3 ML techniques

Within ML there broadly exists 3 types of ML.

- **Supervised learning:**
  - Data contains both dependent and independent variables.
  - Models are trained to learn the relationship between those, typically used for prediction.
  - Example is OLS regression
- **Unsupervised learning:**
  - Data contains only independent variables.
  - Models are constructed as to find a structure in this, like grouping or clustering of data (example is PCA).
- **Reinforcement learning:**
  - The models learn how to take actions, given data, in an environment through feedback and rewards (example is to play chess).

Three fundamental elements:

- The model for $g^*(\cdot)$ - the general functional form for excess return predictions
- The objective function - mean squared prediction error, some with penalty terms for regularization
- Computational algorithms for optimal specification

4.3.1 Simple linear

This simple linear model is not a ML technique itself, but serves as reference point.
Linear in both variables and parameters as

$$g^*\left(z_{it}; \theta\right) = z_{it}'\theta$$

Objective function:

$$\mathcal{L}\left(\theta\right) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{it+1}^e - g\left(z_{it}; \theta\right)\right)^2$$

This model fails horribly when having many predictors, through inconsistency and inefficiency, while almost surely overfitting to noise rather than exctracting the signal.
Therefore, we want to lower the amount of variables $\Rightarrow$ enter penalized linear models

4.3.2 Penalized

Same model, but objective function is changed for minimization purposes.

$$\tilde{\mathcal{L}}\left(\theta; \lambda, \alpha\right) = \mathcal{L}\left(\theta\right) + \phi\left(\lambda, \alpha\right)$$

Penalty term in the elastic net:

$$\phi(\lambda,\alpha) = \underbrace{\underbrace{\lambda(1-\alpha)\sum_{j=1}^{P}|\theta_j|}_{\alpha=0\text{: LASSO}} + \underbrace{\frac{1}{2}\lambda\alpha\sum_{j=1}^{P}\theta_j^2}_{\alpha=1\text{: Ridge}}}_{\alpha=1/2\text{: ENET}}$$

where the α value decides whether we are talking about LASSO, Ridge or Elastic Net, which work through different mechanisms.

- LASSO = sets coefficients = 0
- Ridge = shrinks coefficients down, but never to exactly zero
- ENET = somewhere in between

4.3.3 Dimension reduction

If predictors are highly correlated, regularization through linear penalty term won't be enough as we only remove some noisy variables and leave many that still include noise. Here, averaging all variables would actually be better which dimension reduction techniques build upon. This is the idea behind dimension reduction.

We consider PCR (principal component regression) and PLS (partial least squares).

- PCR steps
  1. Unsupervised learning step using PCA to combine all regressors into small set of linear combinations that seek to extract common signal among the variables
  2. Supervised learning where common signals are used in a standard predictive regression
  - So in total, PCR regularizes by zeroing out low variance components.
  - Inherently unsupervised, as it doesn't incorporate the forecast objective, but purely condenses information among predictors
- PLS steps
  - Exploits the covariance between the target variable and the predictors, thus incorporating the forecast objective
  1. Estimate univariate return prediction coefficients via OLS for each predictor j, reflecting the partial sensitivity of returns towards the j'th predictor.
  2. Meaning that constructing components that maximize co-variation with the target variable.
  3. Average all predictors into single aggregate component with weights proportional to coefficients from prior step - weighting highest on the most important variable.

Dimension reduction technicals:
Model:

$$\boldsymbol{R} = \boldsymbol{Z\theta} + \boldsymbol{E}$$

Condense all predictors into K predictors as per

$$R = \left(\underbrace{Z\Omega_K}_{\text{Dimension-reduced set of predictors}}\right)\theta_K + \tilde{E}$$

Objective function:
PCR chooses weight Ω_K as per

$$\omega_j = \underset{\omega}{\text{argmax}} Var\left[Z\omega\right]$$

taking the K linear combinations of Z that most faithfully mimics the full predictor set.

PSL seeks the K linear combinations of Z that have maximal predictive association with the forecast target as per

$$\omega_j = \underset{\omega}{\text{argmax}} \text{Cov}\left[R, Z\omega\right]$$

Once Ω_K is estimated, the model is estimated by OLS and then forecasted simply. K is obviously the hyperparameter.

4.3.4 Generalized Linear Model

If the true DGP is not linear in variables, but might still be in parameters, we can expand the set of predictors by including higher order transforms in a number of various ways. This definetely proliferates paramters, requiring an additional layer of regularization.
Model is the same as the penalised linear regression model with additive inclusion of higher-order transformations of predictors as

$$g\left(z_{it};\theta\right) = z_{it}'\theta^{(1)} + z_{it}^{2'}\theta^{(2)}, \ldots, z_{it}^{p'}\theta^{(p)}$$

Where the objective function has the same format as the penalized linear model.

4.3.5 Regression trees

- The idea of regressions trees are (from the internet):
  - *What feature will allow me to split the observations at hand in a way that the resulting groups are as different from each other as possible (and the members of each resulting subgroup are as similar to each other as possible)?*

The generalized linear model will generally be poorly specified if allowed to include interactions, as it increases parameters combinatorially. Regression trees, on the other hand, can capture these types of non-linearities, being fully nonparametric.

Trees are designed to find groups of observations which behave similarly in terms of the predictors, this is done sequentially for the different predictors
However doing it too detailed we end up overfitting ⇒ boosting or random forest
Designed to find groups of observations that behave similarly.
Model:

$$g\left(z_{it};\theta, K, L\right) = \sum_{k=1}^{K} \theta_k 1_{\{z_{it} \in C_k(L)\}}$$

As these trees are very prone to overfit, we need to engage in regularization, here considering two ways that work through ensemble -> combining models and predicitons.

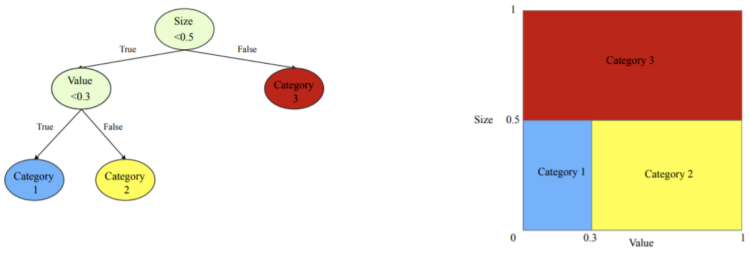

Figure 1: Caption

We will consider two ways to regularize regression trees:

Boosting:

- To overcome problem with overfitting ⇒ Boosting ensemble learning.
- Put different weights on the different trees.

Combines forecasts from many over-simplified trees to form stable and more accurate (less biased) predictions, by fitting a shallow tree, and using residuals to sequentially fit more shallow trees with succesively less weight/learning rate (ν) for a certain number of ensembles B. Suddenly, we have several tuning parameters to choose from. L (dept), ν (learning rate), B (number of ensamble trees).
Random forest (bagging):

- *A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.*
- We can also use random forests to decorrelate the correlation between the trees.
- The idea is:
  - Generate random bootstrap sample
  - For each bootstrap sample we construct a tree and only consider a subset of predictors (m variables)
  - Using this method will decorrelate the trees

Combines forecasts from many different trees through bootstrap samples of the data, fitting a separate tree to each, and averaging their forecasts. Individual trees made in this fashion will usually be deep and overfit, but averaging stabilizes the performance and cancels out the noise fitting.

4.3.6 Neural networks

- Quite complex and very uninterpretable.
- A feed-forward network works by having an input layer of raw predictors, 1+ hidden layers that interact and nonlinearly transform the predictors, and an output layer that aggregates hidden layers into an ultimate prediction.
-

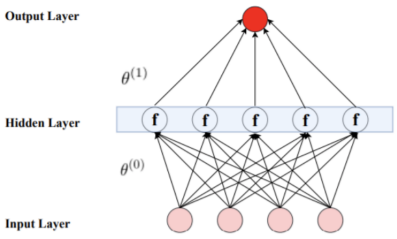

Figure 2: Caption

- The input layer contains the predictors.
- Each neuron applies a nonlinear activation function f that transforms the input before sending its output to the next layer.
- Neural networks includes estimating a lot of parameters $\Rightarrow$ increases exponentially with the number of layers.
- Their are a lot of choices left for the researcher when conducting neural networks.
- Last we have a output layer which consists of linear function from the last hidden layer, which is a function of our predictors.

4.4 Performance assessment

Gu et al. (2020)

- Provides a good illustration of how ML can be used to predict returns.
- Monthly equity returns from the CRSP tape that are listed on NYSE, AMEX, and NASDAQ.
- Using 102 predictors, excluding the 74 industry dummies, without any non-linear effects.

- Uses modified R^2 to assess predictability of ML techniques.
- The zero return benchmark beats OLS,
- Using ENet is better than OLS, while PCR and PLS improves performance further
- These are again beaten by Boosted trees and random forrest while neural networks are the best

When assessing the out-of-sample predictive performance of machine learning models, there are number of ways to look at it. As we learned in return predictability, looking at R^2_{OS} is very telling. This can then be compared to a natural benchmark such as the historical average, or a zero returns benchmark as with Gu et al, who argue that the historical average is too noisy, such being artificially easier to beat than it should be.
Pairwise testing of models can be done with a Diebold-Mariano test, where we take the forecast errors and construct the average loss differential across all stocks at time t

$$\begin{aligned} d_{12t} &= \frac{1}{N_{\text{test},t}} \sum_{i=1}^{N_{\text{test},t}} (r^e_{it} - \hat{r}^e_{1it})^2 - (r^e_{it} - \hat{r}^e_{2it})^2 \\ &= \frac{1}{N_{\text{test},t}} \sum_{i=1}^{N_{\text{test},t}} \hat{e}^2_{1it} - \hat{e}^2_{2it} \end{aligned}$$

Test $\mathbb{H}_0 : \mathbb{E}[d_{12t}] = 0$ by running the regression

$$d_{12t} = \psi + \varepsilon_{12t}$$

and performing a standard t-test on the constant ψ with HAC standard errors to evaluate whether $\psi = 0$, equivalently building the following test statistic:

$$t\left(\overline{d}_{12} = \frac{\overline{d}_{12}}{\sqrt{Var\left[\overline{d}_{12}\right]}} \xrightarrow{d} N(0,1)\right)$$

with T_{test} being the length of the testing period

$$\overline{d}_{12} = \frac{1}{T_{test}} \sum_{t=R_2+1}^{T} d_{12t}$$

4.4.1 Variable importance

Get the importance of variables in multiple ways:

1. Reduction in R^2_{OS} by setting all values of predictor j to zero, holding the remaining model estimates fixed.
2. SSD:
   - $SSD_j = \sum_{t=R_2+1}^{T} \sum_{i=1}^{N_{test,t}} \left(\frac{\partial g(z;\theta)}{\partial z_j}|_{z_{it}}\right)^2$
   - summarizing the sensitivty of the model fits to the changes in that variables
3. Reduction in node impurity for trees $\approx MSE$
4. Partial-dependence based associated variable importance which is model agnostic

- From the Gu et al. (2020) article, it is found that different sorts of momentum variables are the most important.

Placebo principle
To further validate actual variable importance, simualte irrelevant variables that share dynamics with the proposed relevant ones, but are fully unrelated to the object of interest. These shouldn't change the importance ranking of other actual variables, and should obviously be rated the lowest.

- This also seems to be the case for the Gu et al. (2020) article.

Gu et al variable importance
The study by Gu et al. (2020) find that recent price trends are most relevant, followed by liquidity variables. This can be considered relevant for the risk factor perspective where momentum is hard to explain and might have behavioral explanations.

For all the different ML techniques in use, the study find the following variables to be some of the most important:

- short-term reversal
- stock momentum
- momentum change
- industry momentum
- recent maximum return
- long-term reversal

Summary from slide 97:

- The book-to-market ratio is a critical predictor across all models.
- Market volatility has little role in any model.
- Linear and generalized linear models strongly favour bond market variables like the default spread and T-bill.
- Nonlinear models like RF and NN place great emphasis on exactly those ignored by linear models.

4.4.2 Marginal association / interaction effects / partial dependence

- Set all variables to their median values except the variable of interest
- One then varies the value of the variable under consideration from its minimum to its maximum (for characteristics this is between $[-1, 1]$) and compute the fitted value of returns.
- Plotting all fitted values implied from using values in the support of the variables allows one to assess the marginal association
- Expected returns are generally decreasing in size, increasing in past one-year returns, and decreasing in volatility.
- this can also be done for two variables simultaneously to investigate their interaction.
- There are clear interactions among size and short-term reversal, momentum, and total volatility, but none with accruals.

4.4.3 Pre-specified portfolios

- First, because all of our models are optimized for stock-level forecasts, portfolio forecasts provide an additional indirect evaluation of the model and its robustness.
- Aggregate portfolios tend to be of broader economic interest because they represent the risky-asset savings vehicles most commonly held by investors.
- The final advantage of analyzing predictability at the portfolio level is that we can assess the economic contribution of each method via its contribution to risk-adjusted portfolio return performance.
- Bottom-up forecasts for the market fails for the OLS-3 and the dimension-reduction approaches PCR and PLS. Regularized linear models show positive out-of-sample R2. Yet non-linear models substantially outperform the others.
- Consistent with our other results, the strongest and most consistent trading strategies are those based on nonlinear models, with neural networks the best overall.
- It is also possible to compute the Sharpe ratio gains from the ML forecasts. Here it is found that the SR is improved on the market for all methods, except for OLS-3 and the dimension-reduction techniques PCR and PLS. NN and RF always lead to SR improvements.

- Campell and Thompson (2008) shows:

$$SR^* = \sqrt{\frac{SR^2 + R^2}{1 - R^2}}$$

  They show that improvements in R^2 can into utility for a mean-variance investor.

4.4.4 Machine learning portfolios

- Rather than assessing forecast performance among prespecified portfolios, then they design a new set of portfolios to directly exploit machine learning forecasts.
- This is done by based by constructing a zero-net investment long short portfolio. I.e. selling a value weighted portfolio of the decile of stocks with the lowest forecasted returns and buying a value weighted portfolio of the decile of stocks with the highest forecasted return.
- Over a 30-year test period, they found that the NN3 performed in 11 of the years based on the predictive R^2.

Maximum drawdown: Maximum amount lost on the investment strategy.

$$MaxDD = \max_{0<t_1\leq t_2\leq T} (Y_{t_1} - Y_{t_2})$$

Strategy turnover: Average amount of portfolio balancing required

$$Turnover = \frac{1}{T}\sum_{t=1}^{T}\left(\sum_i \left|\omega_{it+1} - \frac{\omega_{it}\left(1 + r^e_{it+1}\right)}{\sum_j \omega_{jt}\left(1 + r^e_{jt+1}\right)}\right|\right)$$

Risk adjusted return: Built on Fama-French 2015 and Carhart 1997, the FF5+Mom risk-adjsuted return α is from the following regression:

$$r^e_{pt+1} = \alpha_p + \beta_p X_{t+1} + \varepsilon_{pt+1}$$

$$X_{t+1} = (MKT_{t+1}, SMB_{t+1}, HML_{t+1}, RMW_{t+1}, CMA_{t+1}, MOM_{t+1})$$

5 Conclusion of Gu et al. (2020)

- At the highest level, our findings demonstrate that machine learning methods can help improve our empirical understanding of asset prices
- Machine learning methods are most valuable for forecasting larger and more liquid stock returns and portfolios.
- Finding that all methods agree on a fairly small set of dominant predictive signals, the most powerful predictors being associated with price trends including return reversal and momentum.
- Nonlinear methods are the best performing methods, with neural networks mostly superior, and to a lesser extent regression trees
- Shallow learning in NN outperforms deep learning.