Evaluation of Machine Learning in Finance

Ze Yu Zhong

Wednesday 8th May, 2019

1. Problems in Empirical Finance
2. What is Machine Learning?
3. Why apply Machine Learning in Finance?
4. Model Specification
5. Simulation
   - Real World Observations
   - Simulation Design
6. Real Data
7. Model Evaluation
8. Results
9. References
10. Questions and Answers

Problems in Empirical Finance

Regressors can be:

- Non-stationary - information now does not contain information about the future
- Persistent - shocks in a series have effects that last for a long time
- Cross sectionally correlated - regressors may seem important but are actually the result of a different underlying regressor
- Endogeneous - omitted variable bias, etc

Problems in Empirical Finance

Data is not robust - structural breaks are evident in returns data, and many regressors that once performed well do not anymore

Extremely large number of potential factors (regressors) that is still increasing: over 600 documented in the literature

What is Machine Learning?

Statistical/Machine Learning refers to a vast set of tools for understanding data

Building statistical models for predicting outputs based on inputs

Find patterns in datasets

Examples of models: Ordinary Least Squares, LASSO Regression, Generalized Linear Models, Decisions Trees, Neural Networks

Why apply Machine Learning in Finance?

Well suited for prediction

Better equipped to deal with large dimensionality

Capable of capturing non-linear transformations humans cannot realistically find

Model Overview

Returns are modelled as an additive error model

$$r_{i,t+1} = E(r_{i,t+1}|\mathcal{F}_t) + \epsilon_{i,t+1} \tag{1}$$

where

$$E(r_{i,t+1}|\mathcal{F}_t) = g^*(z_{i,t}) \tag{2}$$

with $g^*(z_{i,t})$ representing the model approximation using the predictor set $z_{i,t}$

Sample Splitting

- Two main approaches to dealing with temporal data
  - Rolling window - training, validation, and test set lengths are fixed and move forwards in time
  - Growing window - training set grows in size, but validation and test set lengths are fixed and move forwards in time
- Hybrid approach was chosen for feasibility
- Define a training set, validate on the next year, forecast for the next year
- Increase training set by one more year and move the validation and test sets forward one year

Loss Functions

Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{j=i}^{n} |y_j - \hat{y}_j| \tag{3}$$

Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{j=i}^{n} (y_j - \hat{y}_j)^2 \tag{4}$$

Huber Loss

$$H(\epsilon_j = y_j - \hat{y}_j; \xi) = \begin{cases} (y_j - \hat{y}_j)^2, & \text{if} \quad |y_j - \hat{y}_j| \leq \xi; \\ 2\xi|y_j - \hat{y}_j| - \xi^2, & \text{if} \quad |y_j - \hat{y}_j| > \xi \end{cases} \tag{5}$$

Loss Functions

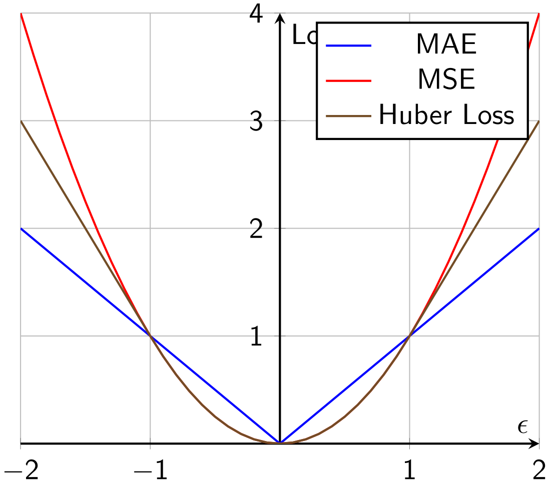


Figure: Illustration of MAE, MSE and Huber Loss when $\xi = 1$

Models Considered

Linear Models

Penalized Linear Models (Elastic Net)

Random Forests

Neural Networks

Linear Models

Linear Models assume that the underlying conditional expectation $g^*(z_{i,t})$ can be modelled as a linear function of the predictors and the parameter vector θ:

$$g(z_{i,t};\theta) = z'_{i,t}\theta \tag{6}$$

Optimizing this with respect to minimizing MSE yields the Pooled OLS estimator

Limitations:

Need to manually consider and specify non-linear interactions

Struggles with high dimensionality

Penalized Linear Models

Penalized linear models have the same underlying statistical model as simple linear models, add a new penalty term in the loss function:

$$\mathcal{L}(\theta;.) = \underbrace{\mathcal{L}(\theta)}_{\text{Loss Function}} + \underbrace{\phi(\theta;.)}_{\text{Penalty Term}} \tag{7}$$

Focus on the popular "elastic net" penalty (Zou and Hastie, 2005), which takes the form for the penalty function $\phi(\theta;.)$:

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

Regression Trees and Random Forests

Classification and regression trees are fully non-parametric models that can capture complex multi-way interactions. A tree "grows" in a series of iterations. With each iteration, a split ("branch") is made along one predictor such that it is the best split available at that stage with respect to minimizing the loss function. These steps are continued until each observation is its own node, or more commonly until the stopping criterion is met. The eventual model slices the predictor space into rectangular partitions, and predicts the unknown function $g^*(z_{i,t})$ with the average value of the outcome variable in each partition.

The prediction of a tree, $\mathcal{T}$, with K "leaves" (terminal nodes), and depth L is

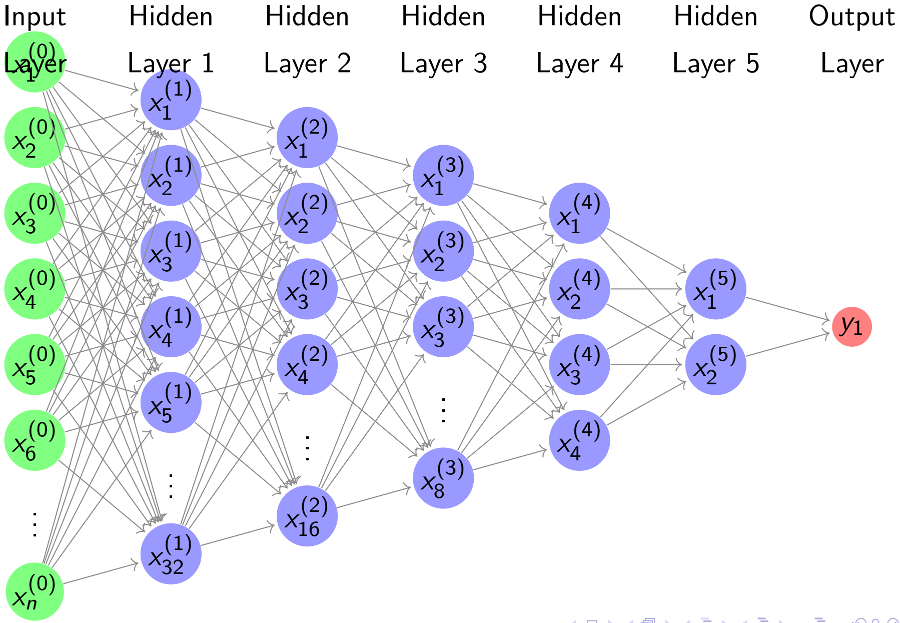
K

Neural Networks

Neural networks are arguably the most complex type of model available, able to capture several non-linear interactions through their many layers, hence its other name "deep learning." On the flipside, their high flexibility often means that they are among the most parameterized and least interpretable models, earning them the reputation as a black box model.

The scope of this paper is limited to traditional "feed-forward" networks. The feed forward network consists of an "input layer" of scaled data inputs, one or more "hidden layers" which interact and non-linearly transform the inputs, and finally an output layer that aggregates the hidden layers and transform them a final time for the final output.

Neural networks with up to 5 hidden layers were considered, each


Input Layer
Hidden Layer 1
Hidden Layer 2
Hidden Layer 3
Hidden Layer 4
Hidden Layer 5
Output Layer
$x_1^{(0)}$
$x_2^{(0)}$
$x_3^{(0)}$
$x_4^{(0)}$
$x_5^{(0)}$
$x_6^{(0)}$
$\vdots$
$x_n^{(0)}$
$x_1^{(1)}$
$x_2^{(1)}$
$x_3^{(1)}$
$x_4^{(1)}$
$x_5^{(1)}$
$\vdots$
$x_{32}^{(1)}$
$x_1^{(2)}$
$x_2^{(2)}$
$x_3^{(2)}$
$x_4^{(2)}$
$\vdots$
$x_{16}^{(2)}$
$x_1^{(3)}$
$x_2^{(3)}$
$x_3^{(3)}$
$\vdots$
$x_8^{(3)}$
$x_1^{(4)}$
$x_2^{(4)}$
$x_3^{(4)}$
$x_4^{(4)}$
$x_1^{(5)}$
$x_2^{(5)}$
y_1

Real World Observations

Though Gu et al. (2018) explore the performance of machine learning on simulated returns series, their design used factors are uncorrelated across i, and, in particular, that the factors which do not matter in the return equation are uncorrelated with those that matter. This is not what is observed in practice.

In particular, their specification lacks:

- Cross Sectional correlation among factors
- Stochastic Volatility in errors

Overall Simulation Design

Therefore, we simulate an extension: a latent factor model with stochastic volatility for excess return, r_{t+1}, for $t = 1, \ldots, T$:

$$r_{i,t+1} = g\left(z_{i,t}\right) + \beta_{i,t+1} v_{t+1} + e_{i,t+1}; \quad z_{i,t} = \left(1, x_t\right)' \otimes c_{i,t}, \quad \beta_{i,t} = (c_{i1,t}, c_i \tag{11}$$

$$e_{i,t+1} = \exp\left(\frac{\sigma^2_{i,t+1}}{2}\right) \varepsilon_{i,t+1}; \tag{12}$$

$$\sigma^2_{i,t+1} = \omega + \gamma_i \sigma^2_{t,i} + w_{i,t+1} \tag{13}$$

v_{t+1} is a 3×1 vector of errors, $w_{i,t+1}, \varepsilon_{i,t+1}$ are scalar error terms. The parameters of these were tuned such that the R squared for each individual return series was 50% and annualized volatility 30%.

Simulating Characteristics

A simulation mechanism for C_t that gives some correlation across the factors and across time was used. First consider drawing normal random numbers for each $1 \leq i \leq N$ and $1 \leq j \leq P_c$, according to

$$\overline{c}_{ij,t} = \rho_j \overline{c}_{ij,t-1} + \epsilon_{ij,t}; \quad \rho_j \sim \mathcal{U}\left(\frac{1}{2}, 1\right) \tag{14}$$

Then, define the matrix

$$B := \Lambda\Lambda' + \frac{1}{10}\mathbb{I}_n, \quad \Lambda_i = (\lambda_{i1}, \dots, \lambda_{i4}), \quad \lambda_{ik} \sim N(0,1),\ k = 1, \dots, 4 \tag{15}$$

which we transform into a correlation matrix W via

$$W = (\text{diag}(B))^{\frac{-1}{2}} (B) (\text{diag}(B))^{\frac{-1}{2}} \tag{16}$$

Simulating Macroeconomic Time Series

For simulation of x_t, a 3×1 multivariate time series, we consider a VAR model:

$$x_t = Ax_{t-1} + u_t, \quad u_t \sim N\left(\mu = (0,0,0)', \Sigma = \mathbb{I}_3\right) \tag{19}$$

Simulating Macroeconomic Time Series

Consider 3 different specifications for matrix A:

$$A_1 = \begin{pmatrix} .95 & 0 & 0 \\ 0 & .95 & 0 \\ 0 & 0 & .95 \end{pmatrix} \tag{20}$$

$$A_2 = \begin{pmatrix} 1 & 0 & .25 \\ 0 & .95 & 0 \\ .25 & 0 & .95 \end{pmatrix} \tag{21}$$

$$A_3 = \begin{pmatrix} .99 & .2 & .1 \\ .2 & .9 & -.3 \\ .1 & -.3 & -.99 \end{pmatrix} \tag{22}$$

Simulating Return Series

We will consider four different functions $g(\cdot)$:

$$(1)\ g_1(z_{i,t}) = \left(c_{i1,t}, c_{i2,t}, c_{i3,t} \times x_t'\right) \theta_0$$

$$(2)\ g_2(z_{i,t}) = \left(c_{i1,t}^2, c_{i1,t} \times c_{i2,t}, \operatorname{sgn}\left(c_{i3,t} \times x_t'\right)\right) \theta_0$$

$$(3)\ g_3(z_{i,t}) = \left(1[c_{i3,t} > 0], c_{i2,t}^3, c_{i1,t} \times c_{i2,t} \times 1[c_{i3,t} > 0], \operatorname{logit}(c_{i3,t})\right) \theta_0$$

$$(4)\ g_4(z_{i,t}) = \left(\hat{c}_{i1,t}, \hat{c}_{i2,t}, \hat{c}_{i3,t} \times x_t'\right) \theta_0$$

$g_1(z_{i,t})$ is a linear specification

$g_2(z_{i,t})$ and $g_3(z_{i,t})$ is a non-linear specification with interactions

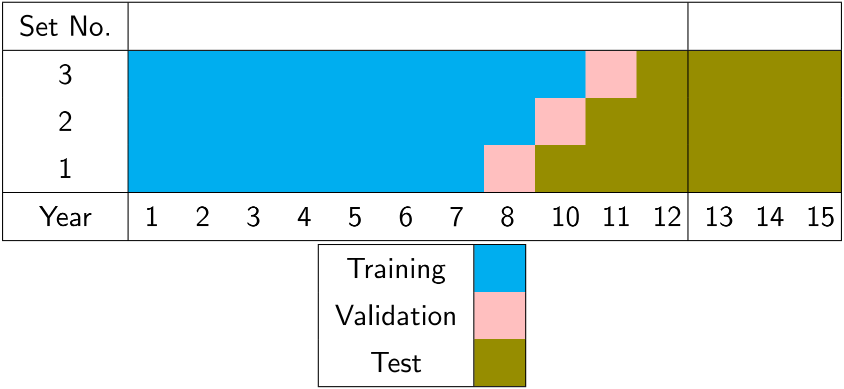
$g_4(z_{i,t})$ builds returns using $\hat{c}$, which are the unobserved characteristics without cross sectional correlation built in

θ^0 was tuned such that the cross sectional R^2 was around 25%, and the predictive R^2 5%.

Sample Splitting

$T = 180$ monthly periods corresponds to 15 years. The training sample was set to start from $T = 108$ or 9 years, a validation set 1 year in length. The last 3 years were reserved as a test set never to be used for validation or training.

Sample Splitting



Set No.														
3	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟥	🟫	🟫	🟫	🟫
2	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟥	🟫	🟫	🟫	🟫	🟫
1	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟥	🟫	🟫	🟫	🟫	🟫	🟫
Year	1	2	3	4	5	6	7	8	10	11	12	13	14	15

Training	🟦
Validation	🟥
Test	🟫

Figure: Sample Splitting Procedure

Data Source

CRSP/Compustat database for stock returns with stock level characteristics such as accounting ratios and macroeconomic factors will be queried.

Most previous studies have included decades of data in their training sample - this does not make much sense as several factors are different now

Only more recent data will be used, such as period before and after 2008 GFC

Out of Sample R Squared

Overall predictive performance for individual excess stock returns were assessed using the out of sample R^2:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in\mathcal{T}_3}(r_{i,t+1} - \widehat{r}_{i,t+1})}{\sum_{(i,t)\in\mathcal{T}_3} r^2_{i,t+1}} \tag{23}$$

where $\mathcal{T}_3$ indicates that the fits are only assessed on the test subsample, which is never used for training or tuning.

Diebold Mariano Tests for Predictive Accuracy

The Diebold-Mariano test (Diebold and Mariano (2002) and Harvey et al. (1997)) compares the forecast accuracy of two forecast methods. Different to the overall R squared metric because it tests whether or not the models' forecast accuracy is significantly different

Under the null hypothesis:

$$S_1^* = \left[\frac{n + 1 - 2h + n^{-1}h(h-1)}{n}\right]^{1/2} S_1 \sim N(0,1) \tag{24}$$

$$S_1 = \left[\hat{V}(\bar{d})\right]^{-1/2} \bar{d} \tag{25}$$

$$\hat{\gamma}_k = n^{-1} \sum_{t=k+1}^{n} (d_t - \bar{d})(d_{t-k} - \bar{d}) \tag{26}$$

$$V(\bar{d}) = n^{-1}\left[\; + 2\sum^{h-1} \right] \tag{27}$$

Variable Importance

The importance of each predictor j is denoted as VI_j, and is defined as the reduction in predictive R-Squared from setting all values of predictor j to 0, while holding the remaining model estimates fixed.

Despite obvious limitations, this allows us to visualize which factors machine learning algorithms have determined to be important.

Results

content...

References

Diebold, F. X., Mariano, R. S., 2002. Comparing predictive accuracy. Journal of Business & economic statistics 20, 134–144.

Gu, S., Kelly, B., Xiu, D., 2018. Empirical asset pricing via machine learning. Tech. rep., National Bureau of Economic Research.

Harvey, D., Leybourne, S., Newbold, P., 1997. Testing the equality of prediction mean squared errors. International Journal of Forecasting 13, 281–291.

Masters, T., 1993. Practical Neural Network Recipes in C++. Morgan Kaufmann, google-Books-ID: 7Ez_Pq0sp2EC.

Zou, H., Hastie, T., 2005. Regularization and variable selection via the Elastic Net. Journal of the Royal Statistical Society, Series B 67, 301–320.

Questions and Answers