Reimplement of Empirical Asset Pricing via Machine Learning

MATH6010Z Project 2
LAI Cong (20747850)
LIU Jinghui (20745644)
LU Qiaoyu (20736916)
ZHEN Mengnan (20749066)
Department of Financial Mathematics
Hong Kong University of Science and Technology
Nov. 14, 2021

Workload:

For coding part: LU Qiaoyu takes part of Linear models (including OLS, OLS-3, Lasso, Ridge and ENet-Huber). ZHEN Mengnan takes part of Dimension reducing models (including PCR and PLS). LIU Jinghui takes part of Tree methods (including GBRT and RF). LAI Cong takes part of Neutral Network (including NN1-NN5).
For report part: All team members take part of their own model's description and modeling process. LU Qiaoyu takes part of Abstract, Introduction, Model Comparison and Conclusion. ZHEN Mengnan takes part of Data Description. LIU Jinghui takes part of Introduction.

Abstract

In this report, we introduce how to conduct machine learning analysis on empirical asset pricing by replication paper. We will explain the modeling process and our model details. We evaluate the model performance by out-of-sample R-square scores and analyze the variable importance. The model complexity is also included. We compare each model by their test score and give a conlusion as well.

1. Introduction

Inspired by the paper, “Empirical Asset Pricing via Machine Learning” [1], we hope to replicate and explore some valuable methods in this project and expect to experience in handling reality financial data. Hence, we conduct a comparative analysis of machine learning methods for finance.

2. Data Description

We divide the 60 years of data into 18 years of training sample (1960–1977), 12 years of validation sample (1978–1989), and the remaining 30 years (1990– 2020) for out-of-sample testing. Each time we refit, we increase the training sample by 1 year. We maintain the same size of the validation sample, but roll it forward to include the most recent 12 months.

3. Modeling Process

3.1. Sample Splitting

Referring to the method of splitting samples in the paper, we split the 60-year data to train data, validation data and test data. For the needs of ‘recursive evaluation performance scheme’, the train data set includes 18-year data from 1960 as a start, while the validation data set is fixed as a 12-year data set. We realize this in our recursive loop. Each increment in the loop adds one-year data to the training data set, while the validation set window rolls to the next year, ensuring that for each year since 1990 can be set to the test data.

3.2. Missing Value Handling

For all linear models, there is a necessary precondition that to guarantee the linearity and additivity of the relationship between the target excess return and the features, as well as the statistical independence of the errors [2]. However, through the scatter plot analysis, the linear relationship between dependent and independent variables is inconspicuous. The OLS performance indicate that the precondition does not hold in this experiment. Hence, we try to offset the nonlinear effect by imposing some restrictions on the features. Towards linear model, considering that the 'recursive evaluation performance scheme' will require the train data to maintain most of features such that the constructed model is capable to make prediction on the validation and the test data, since some features may not be recorded entirely in the past years or they may appear recently, rather than deleting column of features during data preprocessing, the features with different proportions of missing values are processed differently.

For our dimension reduction models (PCA, PLS), the factors which had missing values more than 50%, the missing values were filled with 0. For factors that had missing values less than 50%, we filled the missing values with the mean of the factors.

For tree methods, we used the following steps to manage missing values. If the missing proportion is larger than 50%, it will be filled as zero. Otherwise, it is filled by the mode.

For the NN, for the variable which has more than 50% missing value, it will be dropped. For the variable which has less than 50% missing value, the missing value will be filled with mean of the variable.

4. Model Details

4.1. Simple Linear Regression

The simple linear regression model constructs a linear function between the target parameter and the predictors, which is estimated by Ordinary Least Squares. Although it is known that the OLS with all covariates do not perform well when dealing with high dimension data, we still implement it as a reference baseline.

According to the paper [1], the authors select three covariates (size, book-to-market value, and momentum) to be included in the OLS-3 model such that to restrict OLS to a sparse parameterization. We try to validate the multicollinear effect in high dimension data will drive OLS model to a poor performance by comparing the out-of-sample R-square scores of OLS and OLS-3.

4.1.1. OLS

The OLS model performs the worst due to noise influence. It's best out-of-sample percentage R-square is -0.62%, which is worse than other models' lowest level. So we may pay more attention to OLS-3.

4.1.2. OLS-3

For the simple linear regression model, OLS-3 is significantly superior to OLS, which represents excessive noise has a noteworthy negative impact on the predicted results. When penalizing the coefficients of features, models capture linear relationships between dependent and independent variables more easily. And in 'recursive evaluation loop', it shows that models trained for most recent years achieve high scores.

The OLS-3 focus on using three covariates, 'mom12m', 'bm' and 'mvel1', to predict the excess return. So, it avoids much negative effect from noise.

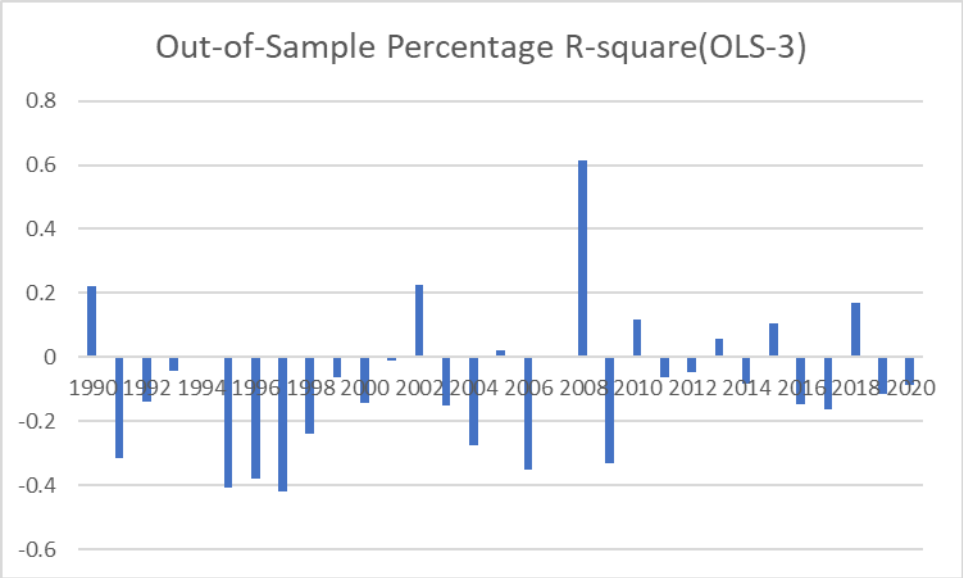

Table 1-Annual return forecasting R^2_{oos} of OLS-3

Table 1 show the out-of-sample percentage R-square derived by the test data sets. The OLS-3 model could get a relatively higher score up to 0.62%, while it also has a high probability of performing worse, down to -0.42%. This may because the three features have a larger influence towards some of the past years. By checking the coefficients of them, it is found that 'momentum' may has a larger influence than other two variables, which's average coefficient is up to 0.0079.

,4.2. Penalized Linear Regression

Before exploring Elastic Net, we choose to discover its foundation, that is Lasso and Ridge Regression. In order to improve the fitting accuracy, OLS would try to fit the noise as well, eventually resulting in overfitting, which catch the noise instead of the signal. But Lasso and Ridge provide a penalty towards coefficients of the features based

on OLS model. It is expected to see a gradual improvement during this process. The degree of penalty of these two regression methods depends on their parameter, Alpha. So, we add a parameter adjustment part in the evaluation loop to find out the parameter with best performance on validation data set and use it to make prediction on test data set.

4.2.1. Lasso Regression

Lasso produces a more exact model by constructing a penalty function that compresses some of the coefficients and deleting some unnecessary features by setting some the coefficients to be zero. Therefore, it may be a biased estimator for complex collinear data. However, when facing high dimensional data, it may compress excessively such that deleting some significant features.

The below Table 2 show the out-of-sample percentage R-square derived by Lasso. From 1993 to 2006, Lasso get a test score of 0. That is because in the first few evaluation windows, Lasso incorrectly reduces the feature numbers to 0 (Table 3). We use this result as a comparison baseline to check if Elastic Network performs better.

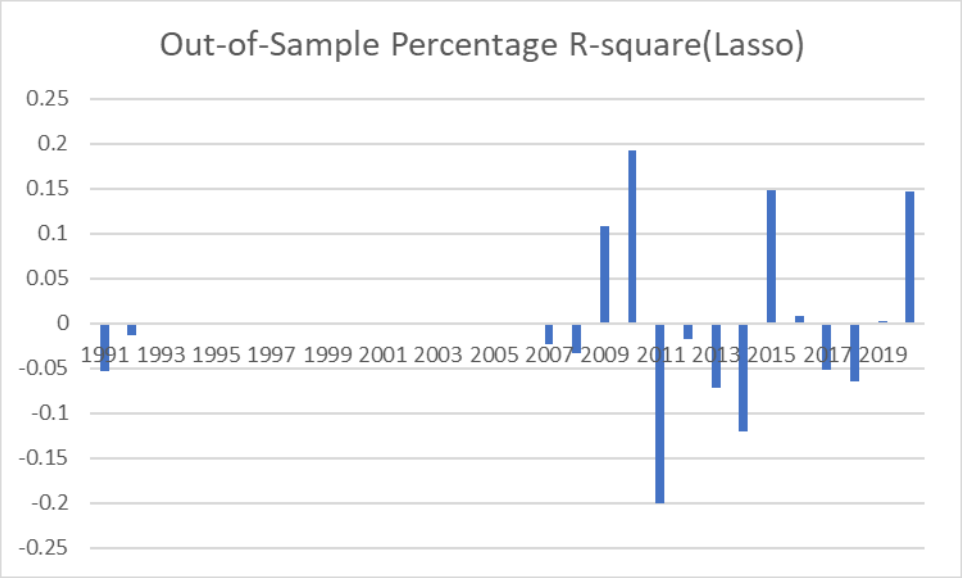

Table 2-Annual return forecasting R^2_{oos} of Lasso

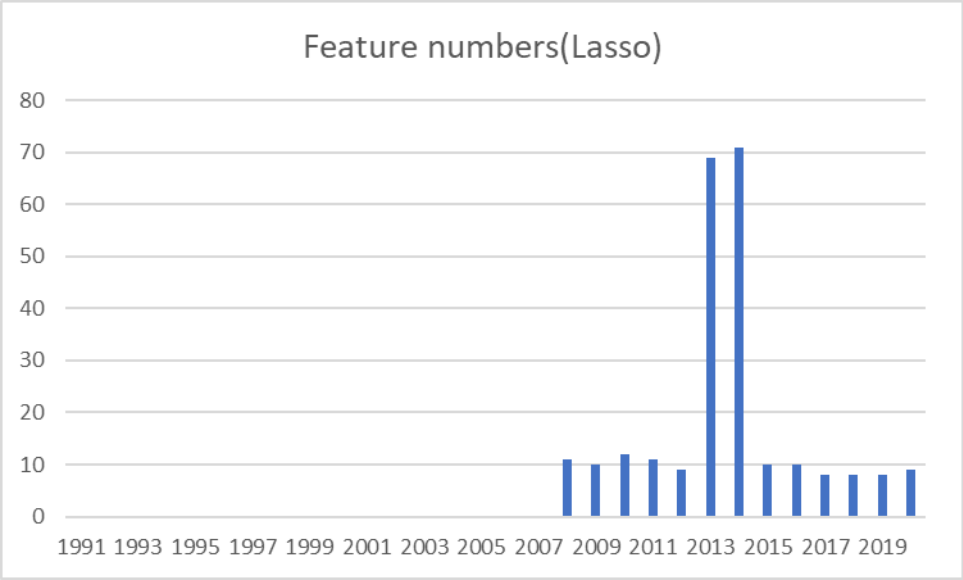

Table 3-Feature numbers of Lasso

4.2.2. Ridge Regression

Table 4 shows the Ridge regression case of compressing coefficients. It is obvious that the Ridge regression model does not show a good fitting effect. And the test scores range from -2.86% to -0.03%, which indicates that we need to adjust loss function of Elastic Net, otherwise it may be hard to reduce dimension.

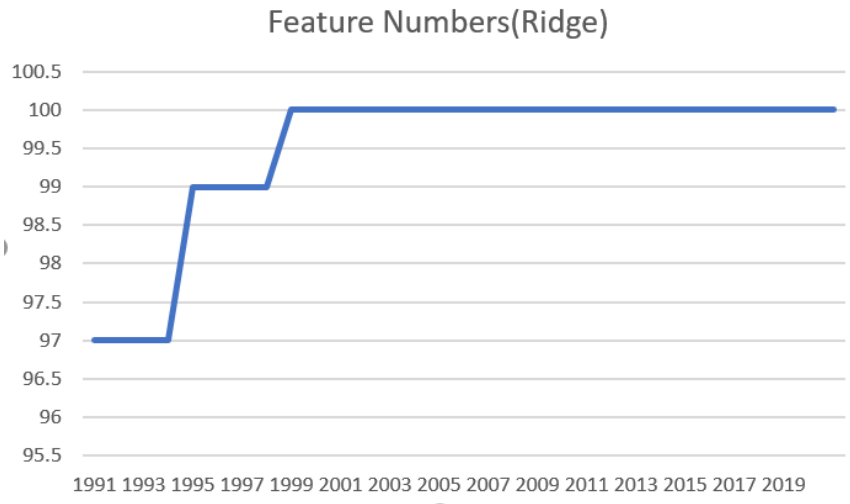

Table 4-Feature numbers of Ridge

4.3. Penalized Linear Regression-Elastic Network-Huber

In general, the Ridge regression can fit the model to a certain extent, but it is easy to distort regression results. Although Lasso regression can overcome this problem, it is too simple and does not conform to the reality. The Elastic Network regression achieves the purpose of Ridge regression to select features with high coefficients, and meanwhile,

it also deletes the features that have little influence on the dependent variables like Lasso regression. We expect the Elastic Net to be superior to other linear models. Besides, the authors adjust the loss function to be Huber robust loss function. Hence, we try to design a new class to utilize the robust loss function, instead of using functions from 'sklearn'.

There are two parameters to be adjusted, L2-ratio and L1-ratio. Here we use α and ρ to represent them respectively. Through the first evaluation window, we determined the rough range parameters and adjusted them in a loop around the below range.

Parameter	Description	Range
Alpha	L2-ratio	[0.15, 0.17, 0.19]
Rho	L1-ratio	[2.0, 2.2, 2.4]

Table 5-Parameter Description of Elastic Network-Huber

Meanwhile, Huber loss function also requires us to give a threshold, which is a cut point such that modify the error value properly. In our function, we try to set an appropriate threshold such that the robust loss function performs better than the standard loss function used in 'sklearn'.

From Table 6, the Elastic Net-Huber shows a better result than other linear models, which achieves test scores ranged from -0.09% to 0.23%. It is suspected that the parameter is fixed to the above range such that the model cannot automatically find out the best parameter for each evaluation window. Hence, we may use grid search to update our model in the future.

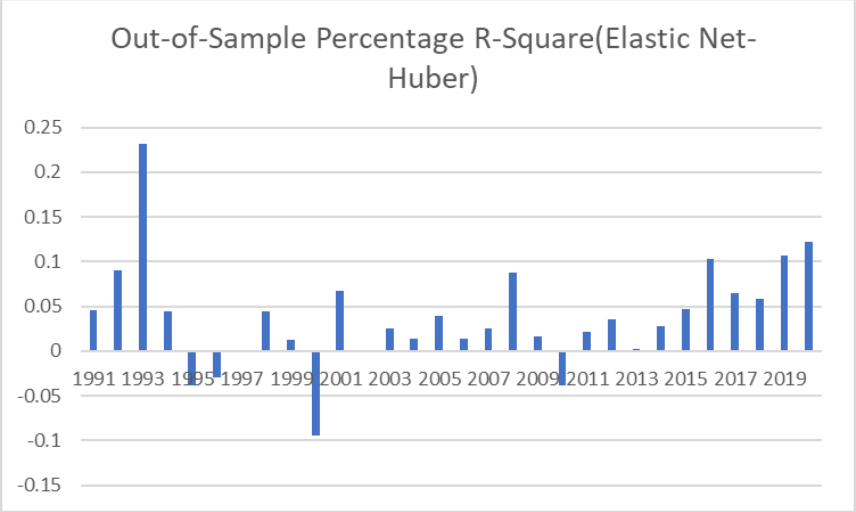

Table 6- Annual return forecasting R^2_{oos} of Elastic Net (Huber)

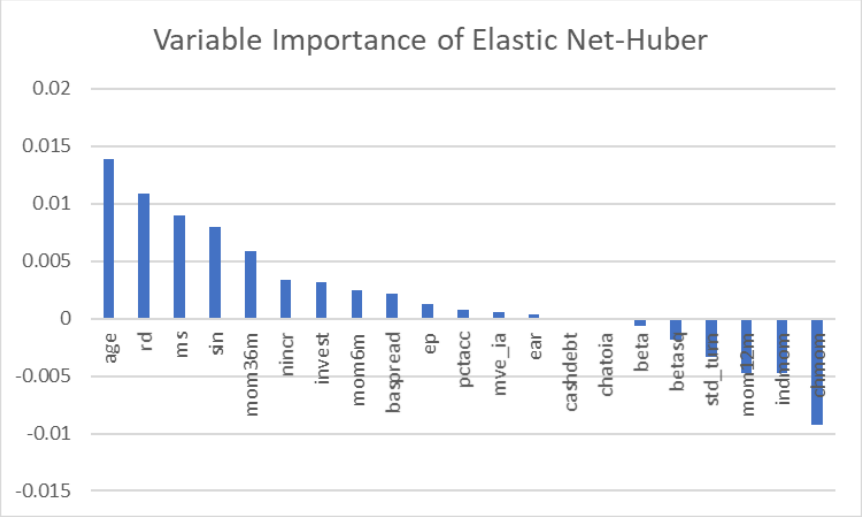

Table 7- Variable Importance of Elastic Net (Huber)

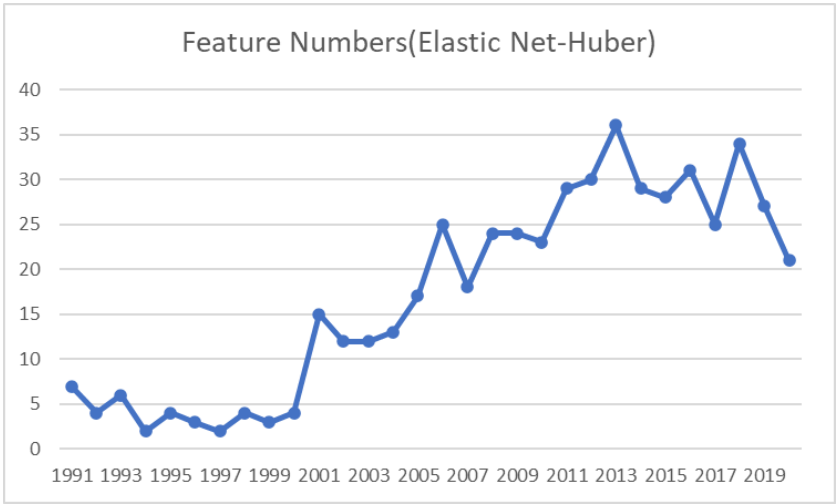

Table 8- Feature Numbers of Elastic Net (Huber)

We also check the variable importance and the model complexity of the Elastic Net-Huber (Table 7 and Table8). We use average coefficient values to display the variable importance of the model and delete some features with relatively small coefficients for convenience. From Table 7, ‘age’, ‘rd’, ‘ms’, ‘sin’, ‘mom12m’, ‘mom36m’ and other 16 variables have much influence on the Elastic Net-Huber model.

From the first few evaluation window results to the last ones, Elastic Net-Huber model shows a trend of increasing the numbers of features, which may implies that with the training data expanding, the model complexity is also rising.

4.4. Dimension Reduction

There are many reasons why dimensions need to be reduced sometimes.
In real machine learning projects, feature selection/dimensionality reduction is a must sometimes, because there are several problems in the data:

- Multicollinearity of data: there is correlation between feature attributes. Multicollinearity leads to the spatial instability of the solution, which leads to the weak generalization ability of the model.
- Due to the sparsity of high latitude spatial samples, it is difficult to find data features in the model.
- Too many variables will hinder people to find rules of the model;
- Considering only the effect of a single variable on a target attribute may ignore the underlying relationships between variables.
- The purpose of feature selection/dimension reduction is to:
- Reduce the number of characteristic attributes
- Ensure that feature attributes are independent of each other

Of course, sometimes the eigenmatrix is too large, resulting in a large amount of calculation and long training time, we used PCR and PLS models for dimension reduction.

4.4.1. PCR

Principal Component Regression (PCR) is a common unsupervised learning method. We used Principal Component Analysis firstly, which contains orthogonal transformation to transform the observed data represented by linearly dependent variables into a small number of data represented by linearly independent variables, which are called principal components. The number of principal components is usually smaller than the number of original variables, so principal component analysis belongs to dimension reduction. Principal component analysis is mainly used to discover the basic structure in data, that is, the relationship between variables in the data. Then, we used the independent variables that were transformed into some components by means of principal component analysis to build the regression model.

In the project, we define the minimum proportion threshold is 80%. That is, the number of dimensionality reduction is automatically determined according to the sample characteristic variance, where the components represent the variance of principal components.

Parameter	Description	Range
components	The number of principal components to be retained	1-100

Table 9- Parameter Description of PCR

The complexity of PCR model is shown below:

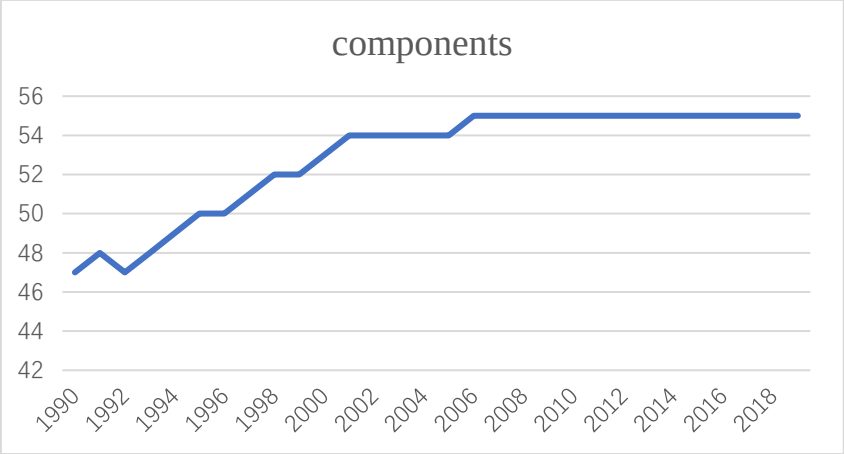

Table 10- Model complexity of PCR

Using the components selected, we could obtain the annual return forecasting R_oos^2 of PCR model, which is shown in the chart below.

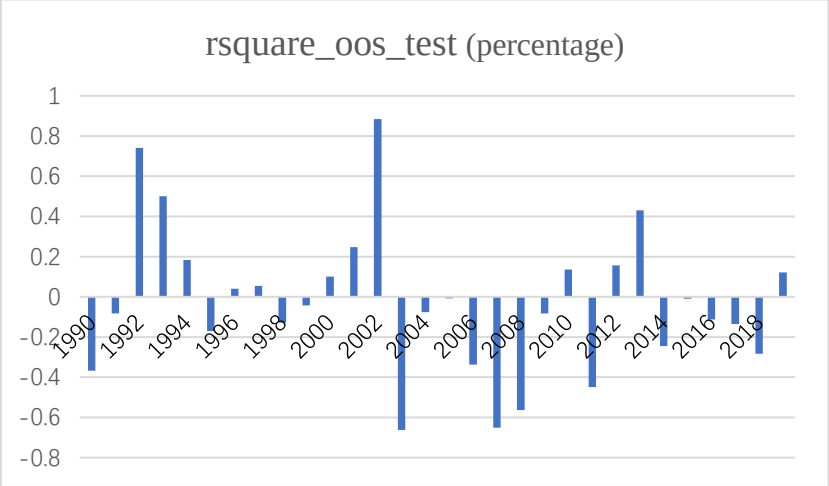

Table 11-Annual return forecasting R^2_{oos} of PCR model

Next, we analyzed feature importance of PCR model with the coefficients in the regression model of each factor, finding that the factor: age, contributes a lot to prediction. What's more, the factors: pchgm_pchsale, pchsale_pchrect also can be regarded as the importance features to prediction.

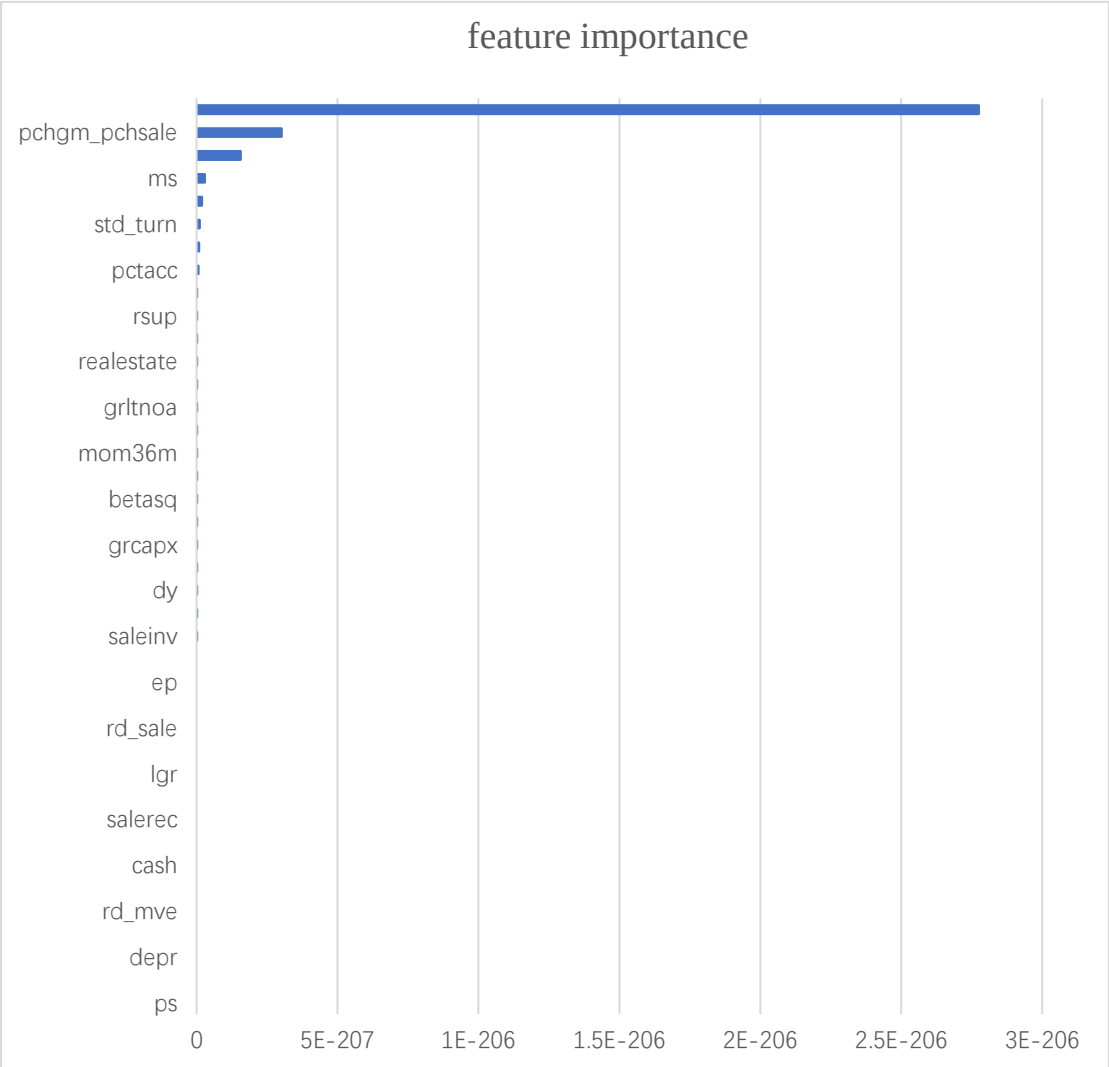

Table 12-Variable Importance of PCR model

4.4.2. PLS

Partial least squares regression (PLS) is a statistical method that is related to principal component regression, but instead of finding the hyperplane with the maximum variance between response variables and independent variables, it projects the prediction variables and observed variables into a new space by projection to find a linear regression model. Because both data X and Y are projected into the new space, the PLS family of methods are known as bilinear factor models.

Partial least squares regression is used to find the fundamental relationship between two matrices (X and Y), that is, an implicit variable method to model the covariance structure in these two Spaces. The partial least squares model will try to find multidimensional directions in the X space to explain the multidimensional directions with the largest variance in the Y space. Partial least squares regression is particularly suitable when the prediction matrix has more variables than the observed, and when the value of X is multicollinearity. By contrast, standard regression is ineffective in these cases (unless it is Tikhonov regularization).

In the project, we used out-of-sample R^2 to find optimal components. Then chose the parameter which had the greatest R_oos^2 every time to predict out-of-sample testing data.

Parameter	Description	Range
components	The number of principal components to be retained	[20,40,60,80]

Table 13- Parameter Description of PLS

The complexity of PCR model is shown below:

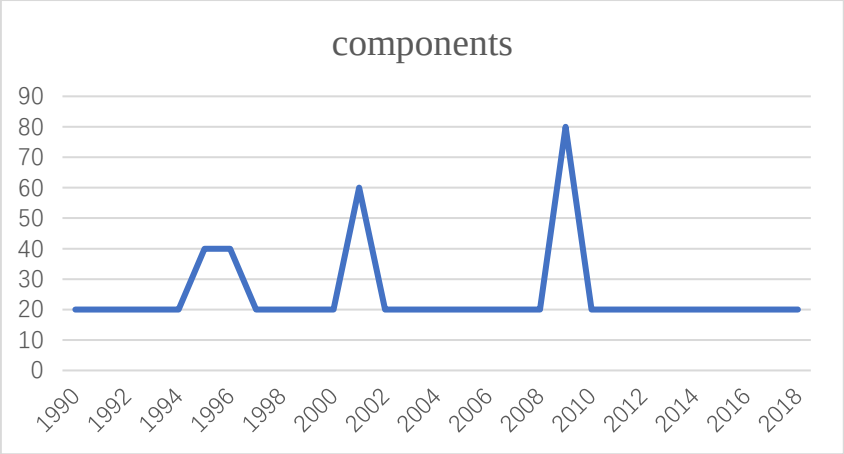

Table 14- Model Complexity of PLS

Using the components selected, we could obtain the annual return forecasting R_oos^2 of PCR model, which is shown in the chart below.

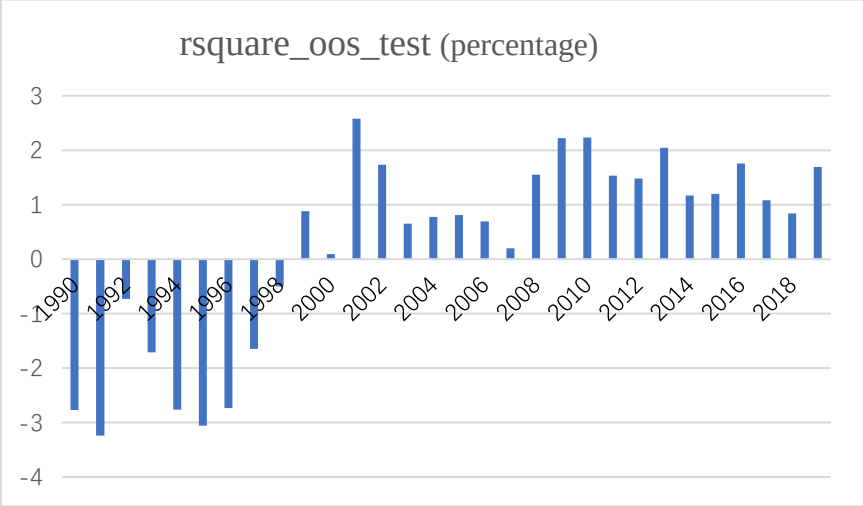

Table 15- Annual return forecasting R^2_{oos} of PLS

4.5. Trees Methods

Regression trees have become a popular machine learning approach for incorporating multiway predictor interactions. Unlike linear models, trees are fully nonparametric and possess a logic that departs markedly from traditional regressions. At a basic level, trees are designed to find groups of observations that behave similarly to each. A tree "grows" in a sequence of steps. At each step, a new "branch" sorts the data leftover from the preceding step into bins based on one of the predictor variables. This sequential branching slices the space of predictors into rectangular partitions and approximates the unknown function with the average value of the outcome variable within each partition.

Advantages of the tree model are that it is invariant to monotonic transformations of predictors, that it naturally accommodates categorical and numerical data in the same model, that it can approximate potentially severe nonlinearities, and that a tree of depth L can capture (L−1)-way interactions. Their flexibility is also their limitation. Trees are among the prediction methods most prone to overfit, and therefore must be heavily regularized. In our analysis, consider two "ensemble" tree regularizes that combine forecasts from many different trees into a single forecast.

4.5.1. Gradient Boosting Regression Tree

The first regularization method is "boosting," which recursively combines forecasts from many oversimplified trees. 19 Shallow trees on their own are "weak learners" with minuscule predictive power. The theory behind boosting suggests that many weak learners may, as an ensemble, comprise a single "strong learner" with greater stability than a single complex tree. The details of our boosting procedure, typically referred to as gradient boosted regression trees (GBRT).

We used Huber loss function and aimed to find optimal hyperparameters among learning rate and max depth.

Parameter	Description	Range
learning_rate	The weight reduction factor of each weak learner is also called step size. The smaller the value is, the more iterations of the weak learner are needed.	[0.01, 0.03, 0.05, 0.1]
max_depth	The depth of the tree. The larger the value is, the easier it is to over fit; The smaller the value is, the easier it is to underfit.	[3, 4, 5, 6]

Table 16- Parameter Description of GBRT

To find the sensitiveness of each parameter, we conduct sensitiveness test, following shows the result. The colored values are out of sample R square of corresponding model. The plot indicated 0.01 should be optimal learning rate.

	0.01	0.03	0.05	0.1
3	-0.02136	-0.03743	-0.04381	-0.04598
4	-0.01712	-0.02136	-0.0238	-0.03755
5	-0.02263	-0.04348	-0.05312	-0.06346
6	-0.01583	-0.02502	-0.0268	-0.00877

Table 17- Parameter Sensitiveness of GBRT

In terms of rolling test, the maximum of out of sample R square is still low, 0.004734, with learning rate = 0.01 and max_depth = 3.

We also analyzed feature importance of model with the minimum out of sample R square, finding that few features contribute a lot to prediction. “mvel1” is the market size which contributes the most.

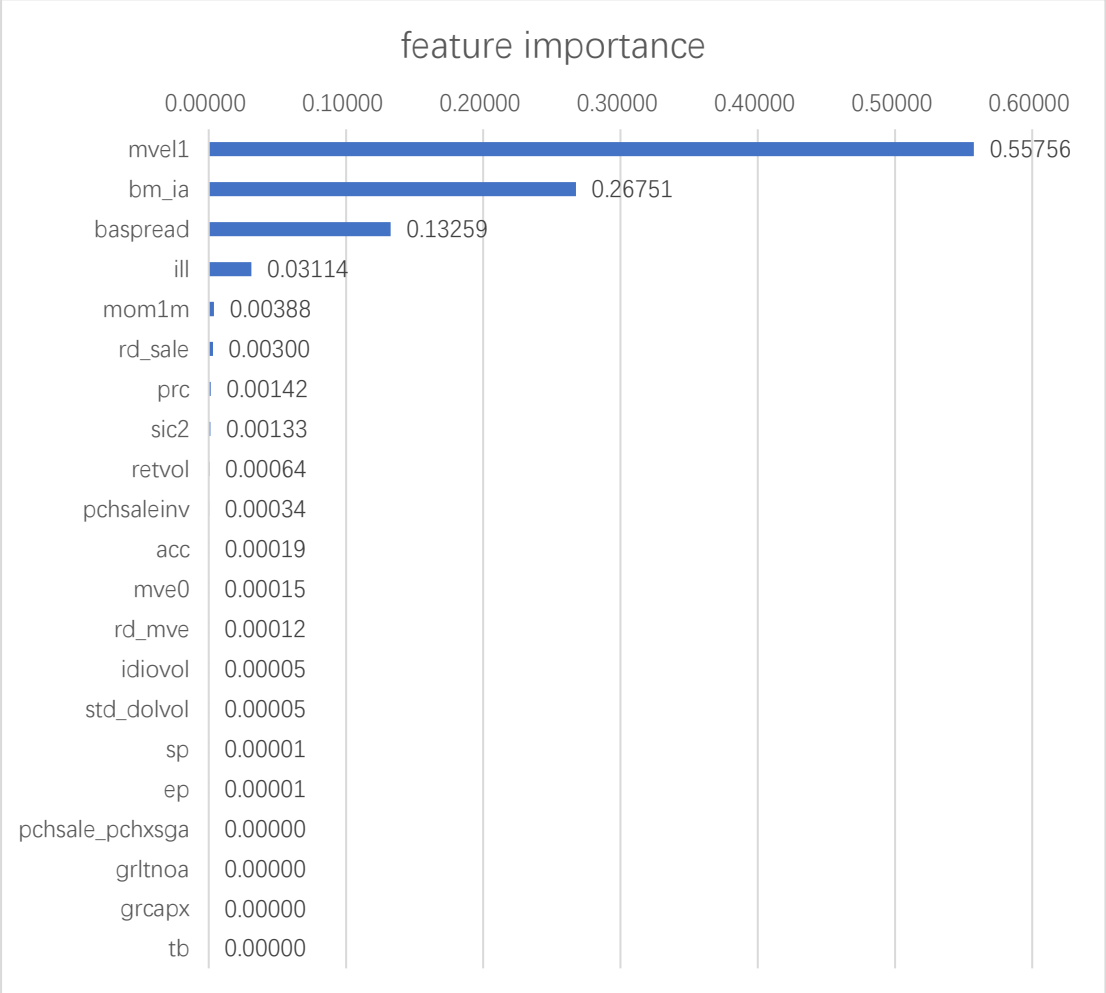

Table 18- Variable Importance of GBRT

“bm_ia” is “Industry-adjusted book to market”. “ill” is illiquidity of stock.

4.5.2. Random Forest

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". The idea of random forests is to improve the variance reduction of bagging by reducing the correlation between trees, without increasing the variance too much.

We found optimal hyperparameters of number of estimators, minimum samples split and max depth.

Parameter	Description	Range
n_estimators	The number of decision trees. Generally, the more, the better, but the worse the performance. At least 100 can achieve acceptable performance and error rate.	[300,400,500,600]
min_samples_split	When dividing nodes according to attributes, the minimum number of samples for each partition.	[20,50,100,200]
max_depth	Maximum depth of tree	[3,4,5,6]

Table 19- Parameter Description of RF

To find the sensitiveness of each parameter, we conduct sensitiveness test, following shows the result. The plot indicated 50 should be optimal min_samples_split, while optimal max depth in [4,5,6], setting number of decision trees as 300.

	20	50	100	200
3	-1.16348	-1.25617	-1.49725	-1.27372
4	-0.92944	-0.82956	-1.30821	-1.56029
5	-1.0369	-0.56746	-1.55699	-1.53424
6	-1.05955	-0.70683	-1.55955	-1.46309

Table 20- Parameter Sensitiveness of RF

In terms of rolling test, the maximum of out of sample R square is negative, -0.5675, with max_depth = 5 and min_samples_split = 50. We laid the bad performance to potential difference of data cleaning, comparing to the result in origin thesis.

Next, we analyzed feature importance of model with the minimum out of sample R square, finding that few features contribute a lot to prediction. It shows similar pattern as GBRT, but important features changed and number of them increased, which is shown as the following plot. The top important features are ep (Earnings to price),

mom36m (36-month momentum), idiovol (Idiosyncratic return volatility), mve_ia (Industry-adjusted size), covind (Convertible debt indicator), agr (Asset growth), betasq (Beta squared), sin (Sin stocks) and grcapx (Growth in capital expenditures).

4.6. Neural Network

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. Generally, Neural Network has one input layer, one output layer and hidden layers.

In our project, we choose five Neural Network with hidden layers from 1 to 5, "ReLU" as active function, and 1-R_OOS^2 as loss function. However, due to limited time, it is hard to adjust the learning rate. Almost 155 models' training process are less than 20 epochs, which may lead to negative effect.

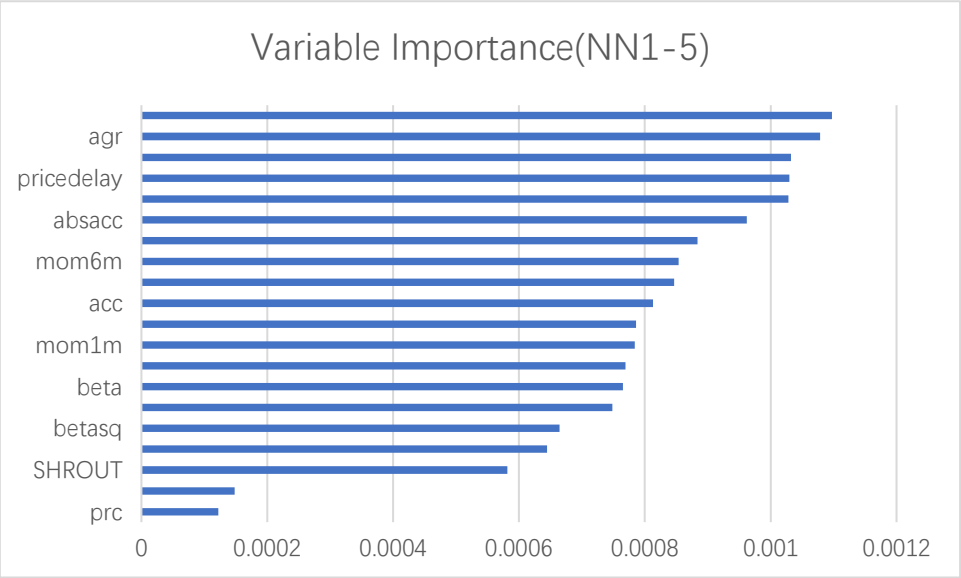

Table 20- Variable Importance of NN1-5 (Using absolute value)

Table 20 shows the average variable importance of NN1-5. As expected, 'mom6m' and 'mom1m' and some other variables are significant for most of our models. NN1-5 also consider some other features that are not put into other models, like 'agr'.

5. Model Comparison & Conclusion

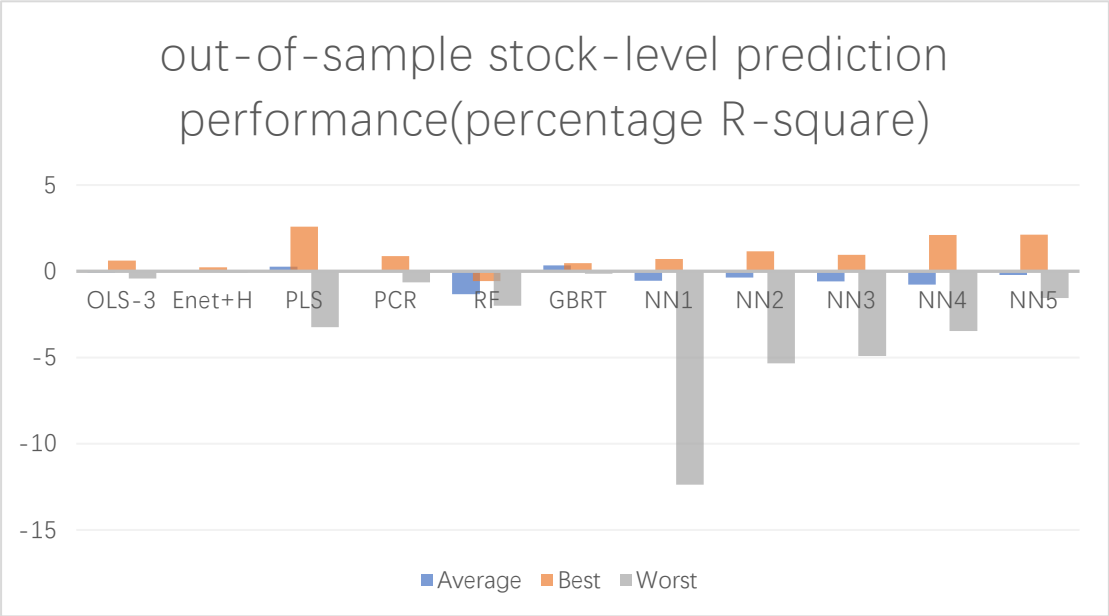

Table 22- Model comparison by out-of-sample R-square

Table 21 shows the comparison of model performances. It is expected that Neutral Network has a good performance, whose best score is up to 2.12%. However, it may be unstable. From NN1 to NN5, the increasing best worst score drives the average score up, which implies NN5 may be superior to NN1 in such cases. Not surprisingly, the linear models (OLS-3 and ENet+H) performs worse, which may be caused by dropping too much necessary features, and ignoring the interaction effect between the features. Dimension Reducing models, PLS and PCR have a significant prediction effect compared to the linear models. While tree models do not reflect the prediction effect intuitively.

In this experimental process, we realize that there is huge difference between the authors' models and our models. We rethink about our modeling process and summarize some limitations about our models:

It is speculated that the time gap between the training data set and the test data set is too long, regarding 12 years data as validation set, so that the trained model may not be able to capture the new trend and recent characteristics, which may lead to a lower test score. It is hard to be sure whether large amounts of data from decades ago will make sense for training models to predict the future. Besides, due to the limited memory and program speed, we cannot use grid search to find out the best parameter needed, such that reduce the likelihood of scoring higher. Additionally, the functions adjusted with Huber robust loss function may ignore some details, resulting differences between our models and the authors'.

We may improve our models from the above aspects in the future.

6. References

[1] Gu, S., Kelly, B., and Xiu D. (2020). Empirical Asset Pricing via Machine Learning. The Review of Financial Studies 33 (2020) 2223–2273.

[2] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.