Paper Replication: Empirical Asset Pricing via Machine Learning

Jialin ZENG(20807909)
Tianyu JIN(20808264)
Tiankai HU(20808238)

HKUST
Department of Mathematics

2022.5

https://hkust.zoom.us/rec/share/SXDnKvAyoou5P7ae_skErz-PyiHZDQWQcd3HktA4xdMeF4PPPJAE-CoTpGy2_qFp.H3qEp83ohLGAYpNp

Contents

1. Workload Contribution
2. Data Preprocessing
3. Linear Regression:OLS,OLS-3 and Elastic Net
4. Dimension Reduction:PCR and PLS
5. Gradient Boosting Regression Tree(GBRT)
6. Neural Networks
7. Conclusion

Workload Contribution

Workload Contribution

- Jialin ZENG:Data Preprocssing and Dimension Reduction:PLS and PCR.
- Tiankai HU: Linear Regression(OLS, OLS-3,Elastic Net) and Gradient Boosting Regression Tree.)
- Tianyu JIN: Neural Networks from NN1 to NN5.
- Acknowledgements to Chuqi Chen for helping with Linear Regression.

Data Preprocessing

Data Split

We select 60 years of data from 1960-2019 as the entire data, including :

- 18 years(1960-1977) training sample
- 12 years (1988-1989) of validation sample
- 30 years(1990-2019) of test sample

The data includes three parts: firm characteristic, SIC code and the macroeconomic predictors.

Data Merging

The macroeconomic predictors are given in a different table than the first two parts. Before merging three parts of data together, we use formulas to **calculate out 8 macroeconomic predictors** specified in the paper following instruction of the paper.

Predictors Setting

Due to limited computational resources, we do not calculate Kronecker product between firm predictors and macroeconomic predictors to construct 920 covariates. We believe in neural networks and tree models, such nonlinear interactions between predictors can also be guaranteed. Therefore, we drop the unmentioned predictors and keep the original **103 predictors as final predictors**.

Regression Goal

Our regression goal, according to the paper, is to predict the risk premium $r_{i,t+1}$ given predictors $z_{i,t}$:

$$r_{i,t+1} = E_t(r_{i,t+1}) + \epsilon_{i,t+1}$$

, where

$$E_t(r_{i,t+1}) = g^*(z_{i,t}).$$

However, this $r_{i,t+1}$ is not directly given in the data. According to the paper's instruction, we calculate this **return in excess of risk-free rate** $r_{i,t+1}$ through substracting RET by tbl .

Missing values Filling

The merging data includes a lot of missing values, which will cause bad influence to regression result without filling properly. The filling method we adopt is as follow:

- Calculate the **cross-sectional** median instead of all-time median to fill the missing characteristic so that **no future information will be included** in the present statistics.
- For the missing value after this filling, we **fill them by zero**.

.

Recursive Performance Evaluation Scheme

We adopt the **recursive performance evaluation scheme** mentioned in the paper:
When evaluating every new year in the test sample, we let the training sample increase by one year, the validation sample roll it forward by 1 year while keeping the fixed size of 12 years.

Linear Regression:OLS,OLS-3 and Elastic Net

Linear Regression

The purpose of linear regression is to find a simple linear relationship between the independent and dependent variables.
In this section, we will show the results using **OLS, OLS3, and Elastic Net methods**.

Huber Loss

Instead of using standard least squares, we use **Huber robust objective function**, which is defined as

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

where

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

OLS

The OLS is the **simplest linear model and the least effective**.
Our goal is to use a simple linear model to predict the excess return.
The performance of OLS has shown below.

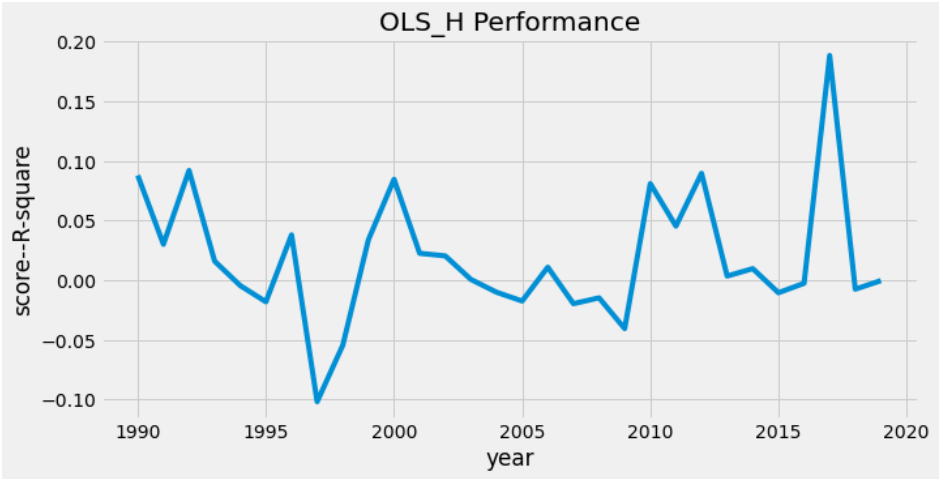

Figure: Annual return forecasting R^2_{oos} of OLS.

OLS-3

To do a simple linear regression, the OLS-3 only focuses on **using three covariates,'mom12m', 'bm'and'mvel1'**, to predict the excess return.The average R^2_{oos} of OLS-3 is -0.0863.

Variable Importance

We can use a simple linear model to make some prior estimates. So in the OLS model, we **calculate the correlation coefficient** of each feature to represent the variable importance.

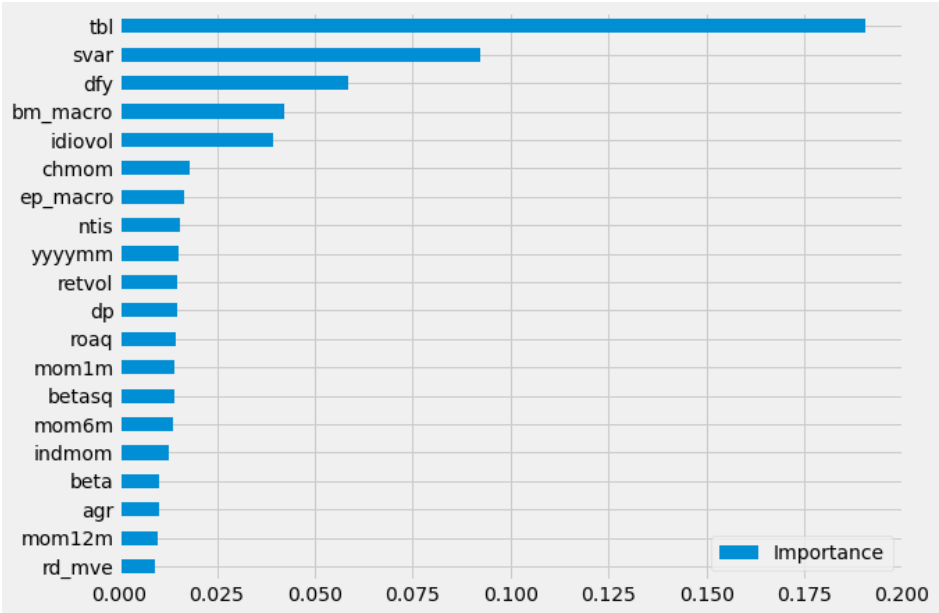

Figure: Top-20 most influential variables for OLS.

Penalized Linear Regression-Elastic Network

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

$$\mathcal{L}_H(\theta;\cdot) = \mathcal{L}_H(\theta) + \phi(\theta;\cdot).$$

Here we use the "elastic net" penalty, which takes the form:

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

Hyperparameter Setting

We use validation datasets to choose **the optimal hyperparameters where** $\rho = 0.5$ and $\lambda = 0.05$.
The performance of Elastic Net is shown below.

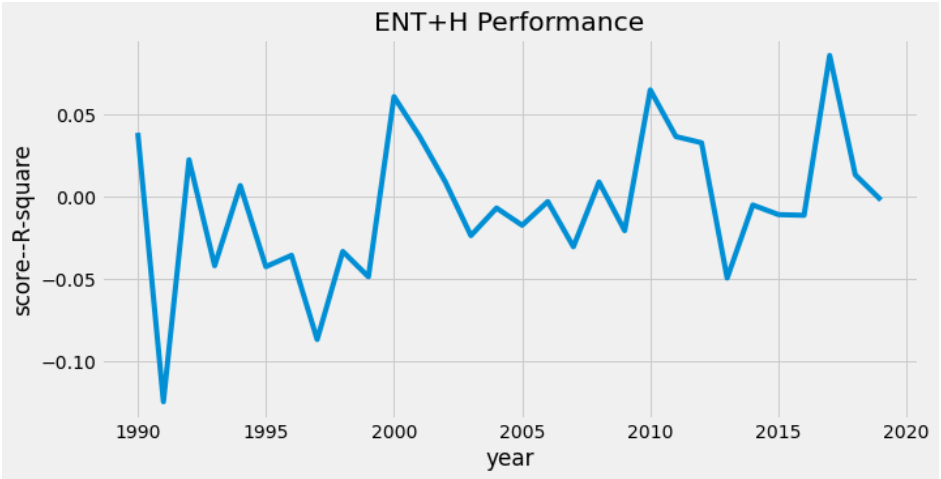

Figure: Annual return forecasting R^2_{oos} of ENET.

Variable Importance

Here to calculate the Variable Importance, **the reduction in panel predictive R2** from setting all values of predictor j to zero while holding the remaining model estimates fixed is used.

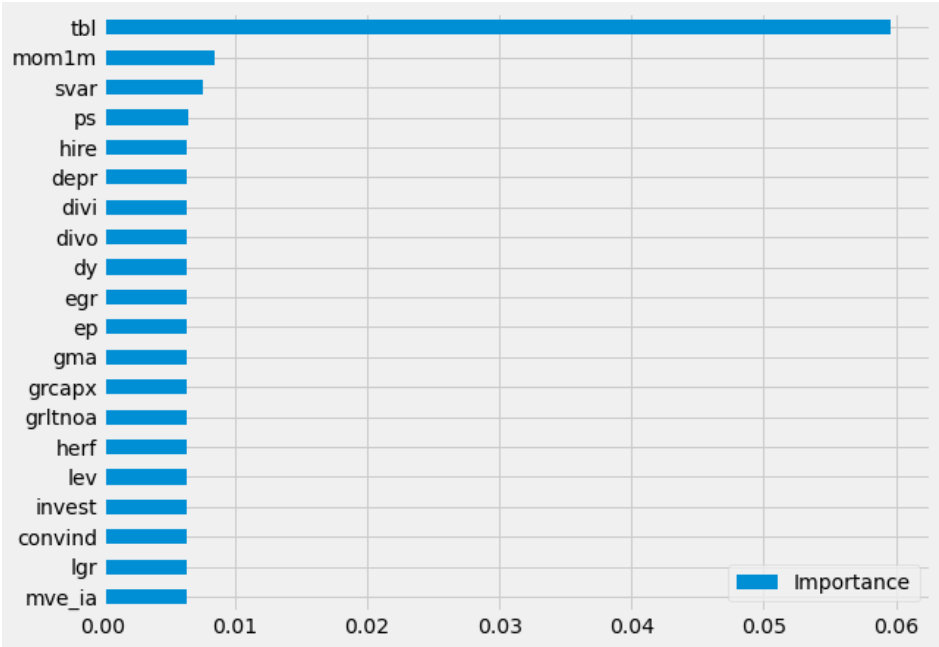

Figure: Top-20 most influential variables for ENET.

Dimension Reduction:PCR and PLS

Model Formulation

Both these two methods averages noise by **combing predictors into low-dimensional components**. The forecasting model for both two methods can be written as:

$$R = (Z\Omega_K)\theta_K + E$$

,
where Ω_K is $P \times K$ matrix with columns $\omega_1, \omega_2, \ldots, \omega_K$ and $Z\Omega_K$ is the dimension-reduced version of the original predictor set.

Model Formulation

- PCR looks for each ω_k recursively that maximize the variance of $Z\omega$, so j^{th} linear combination solves

$$\omega_j = argmax_{\omega} Var(ZW)$$

s.t.

$$\omega^{'}\omega = 1, cov(Z\omega, Z\omega_l) = 0, l = 1, 2, \ldots, j-1.$$

- PLS looks for K-dimensional projections of Z that **maximize predicative association with the final target R**, so the j^{th} linear combination solves

$$\omega_j = argmax_{\omega} Cov^2(R, ZW)$$

s.t.

$$\omega^{'}\omega = 1, cov(Z\omega, Z\omega_l) = 0, l = 1, 2, \ldots, j-1.$$

Time Varying R^2 Performance for PCR

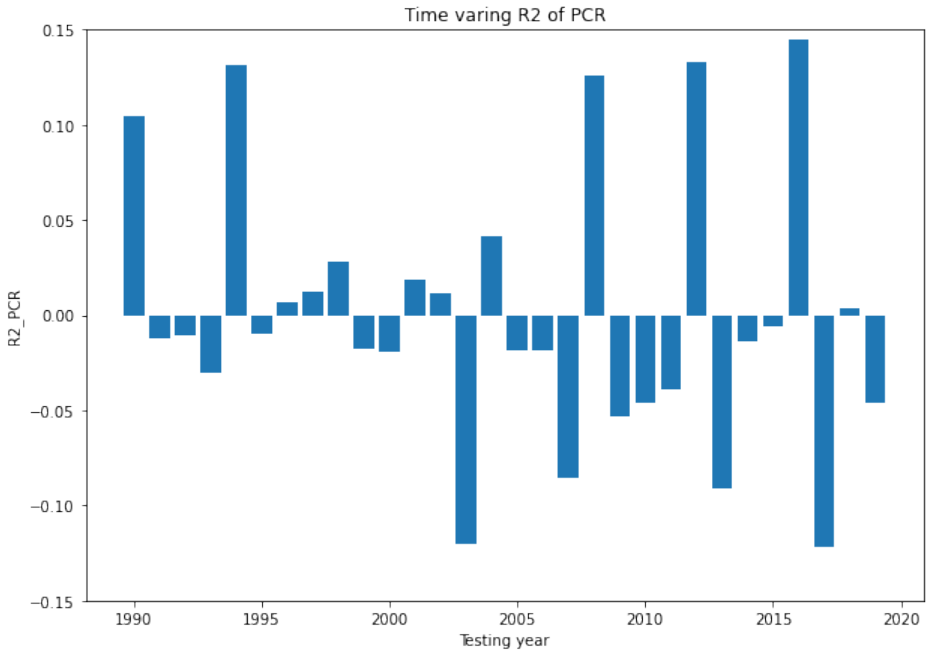

Figure: Time varying R^2_{oos} with mean 0.01904 for PCR

Time Varying R^2 Performance for PLS

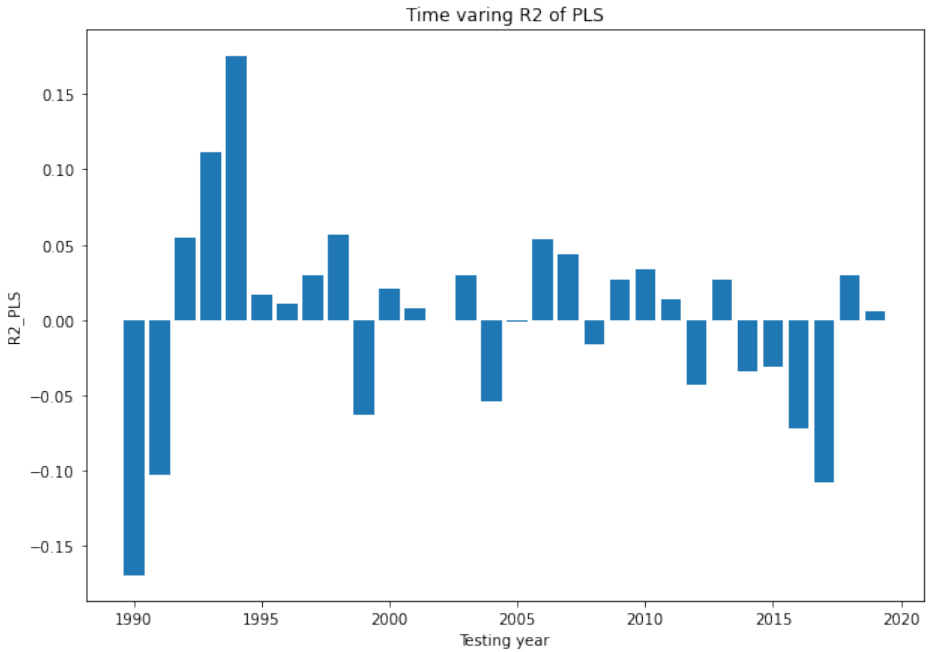

Figure: Time varying R^2_{oos} with mean 0.01531 for PLS

Observations

- In general, both methods have **better and more stable** predictive power in the **earlier testing years**.
  –Stock market is highly time-dependent.
- Performance of **PCR seems to outperform PLS a little in the long run**.
  –PCR focuses more on capturing covariance structure of predictors rather than correlation between forecast return which seems to have little power to predict future pattern.

Model Complexity for PCR

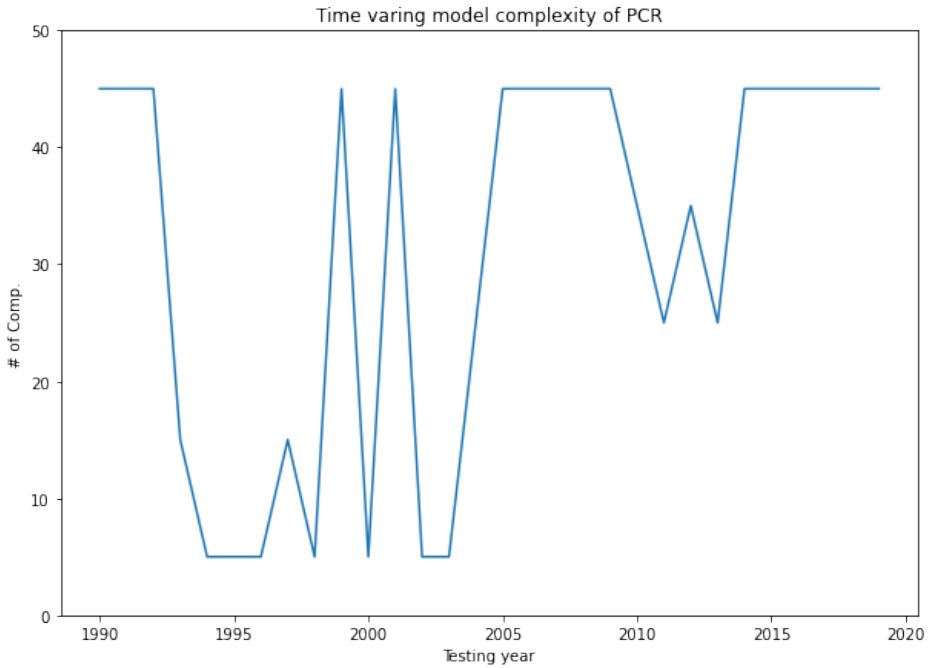

Figure: Time varying model complexity for PCR

Model Complexity for PLS

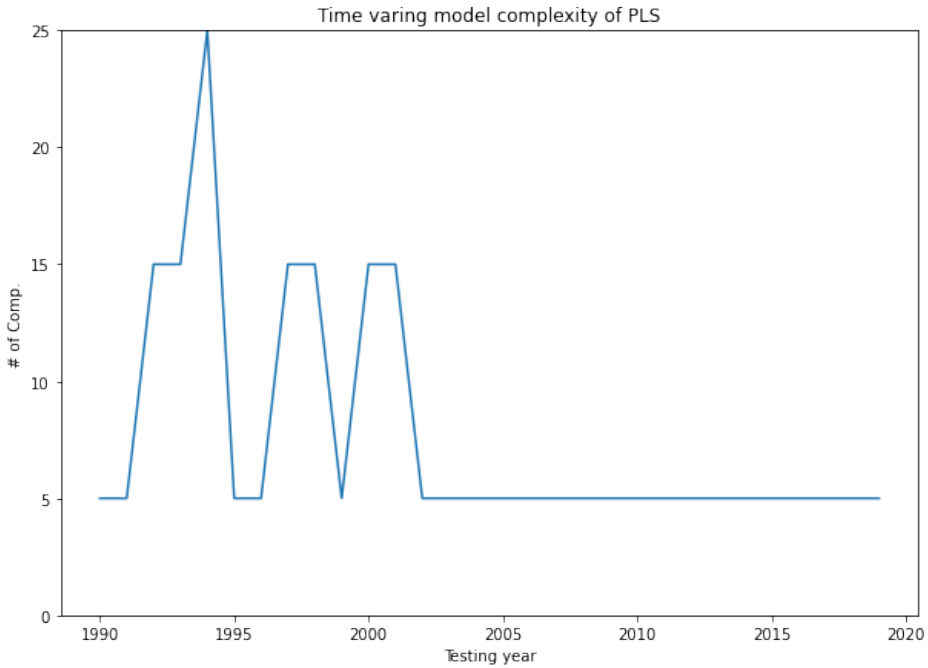

Figure: Time varying model complexity for PLS

Hyperparameter Settings and Observations

- For each testing year, we select the best number of components K by validating on the validation sample. Due to computational cost, though we only validate component numbers from $[5, 15, 25, 35, 45]$.
- The result pattern is similar to the one given in paper. That is, **PCR** typically use **25 to 45** components in its forecast while **PLS** falis to stablize in the early stage but eventually settles on about **5**.

Variable Importance for PCR

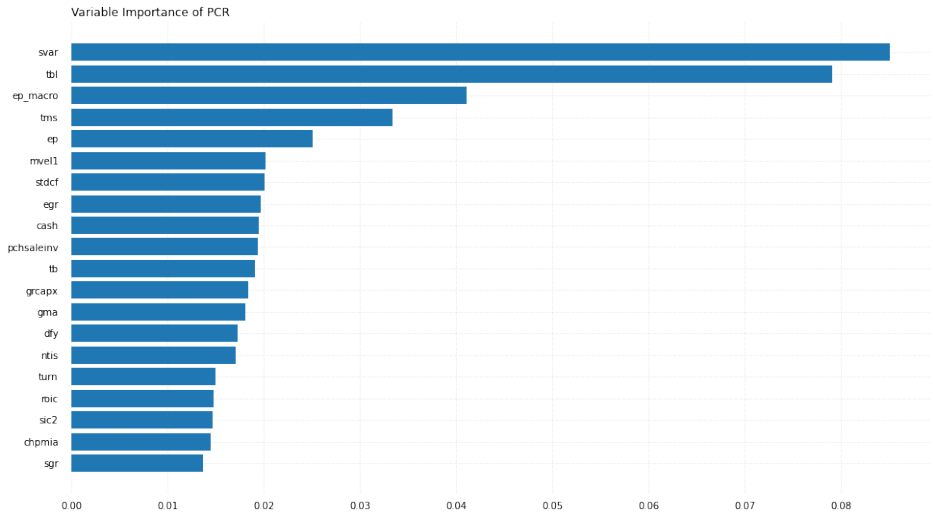

Figure: Top-20 most influentil variables for PCR

Variable Importance for PLS

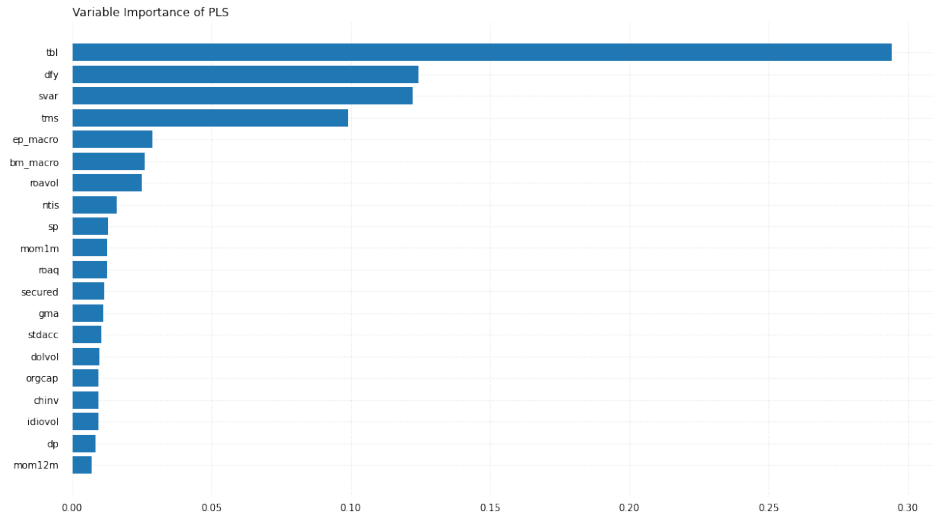

Figure: Top-20 most influentil variables for PLS

Observations

- We measure each predictors's importance using the R^2 **reduction** from setting all values of this specified predictor to zero, while holding the remaining model estimates fixed. We only calculate the R^2 reduction for the testing year with best R^2 performance and corresponding best component numbers due to limited computational power.
- we can see that the importance magnitude is **highly skewed towards many macroeconomic predictors such as tnl, dfy and some firm predictors like mom1m**. This is plausible since we independently merge firm predictors and macroeconomic ones. Moreover, the nature of dimension reduction method also leads us to discover just a few leading components.

Gradient Boosting Regression Tree(GBRT)

Regression Tree

Regression trees are the most popular machine learning methods, which are intuitive for decision-making. Unlike linear regression, regression tree is a **nonparametric method** and quite different from traditional regressions. Our target is to divide the observations into small groups. For each group, they behave similarly.

Gradient Boosting Regression Tree(GBRT)

Boosting is a regularization method that **converts many weak learners into a strong one**. Combining with regression trees, we have gradient boosting regression tree(GBRT).
GBRT recursively considers results from trees with a simple structure to get a strong learner with higher stability than a single complex tree.

Hyperparameter Setting

Here, we use the Huber loss as our loss function. The hyper -parameters we need to train are the weight reduction factor of each weak learner (step size) and the maximum depth of the individual regression estimators.

Performance of GBRT

We use validation datasets to choose the optimal hyperparameters where $max\_depth = 3$ and $learning\_rate = 0.01$ and we also use Huber loss.

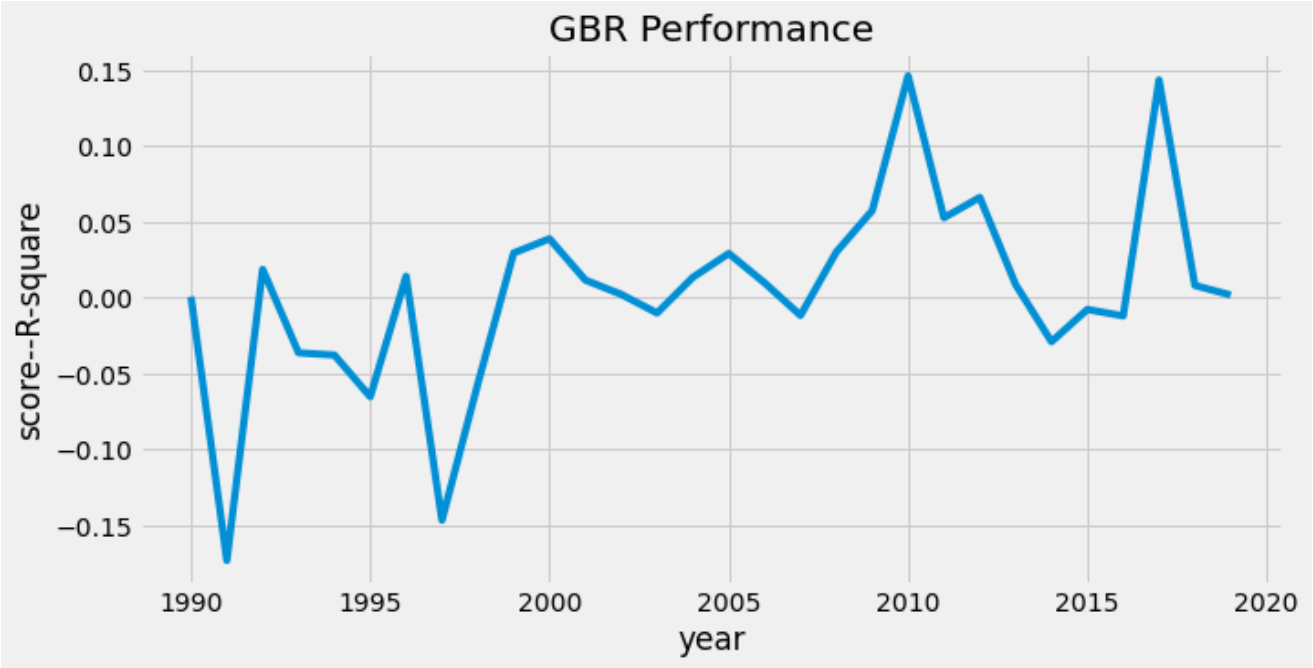

Figure: Annual return forecasting R^2_{oos} of GBRT.

Variable Importance

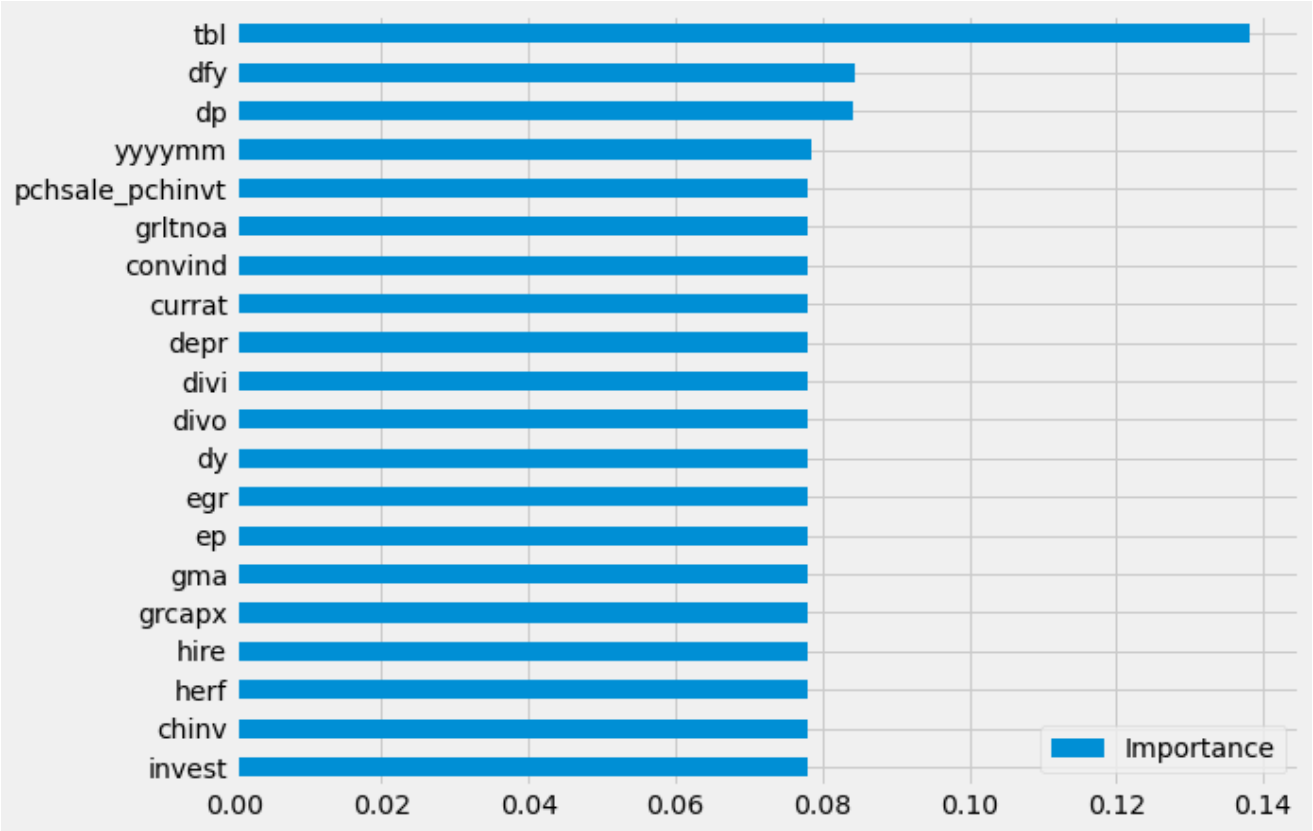

Figure: Top-20 most influential variables for GBRT. The average R^2_{oos} of GBRT is 0.0030.

Neural Networks

Neural Networks

The structure of neural networks used in this paper are based on traditional "feed-forward" networks.
NN1: a single hidden layer of 32 neurons.
NN2: two hidden layers with 32 and 16 neurons.
NN3: three hidden layers with 32, 16 and 8 neurons.
NN4: four hidden layers with 32, 16 ,8 and 4 neurons.
NN5: five hidden layers with 32, 16, 8, 4 and 2 neurons.

R-square scores

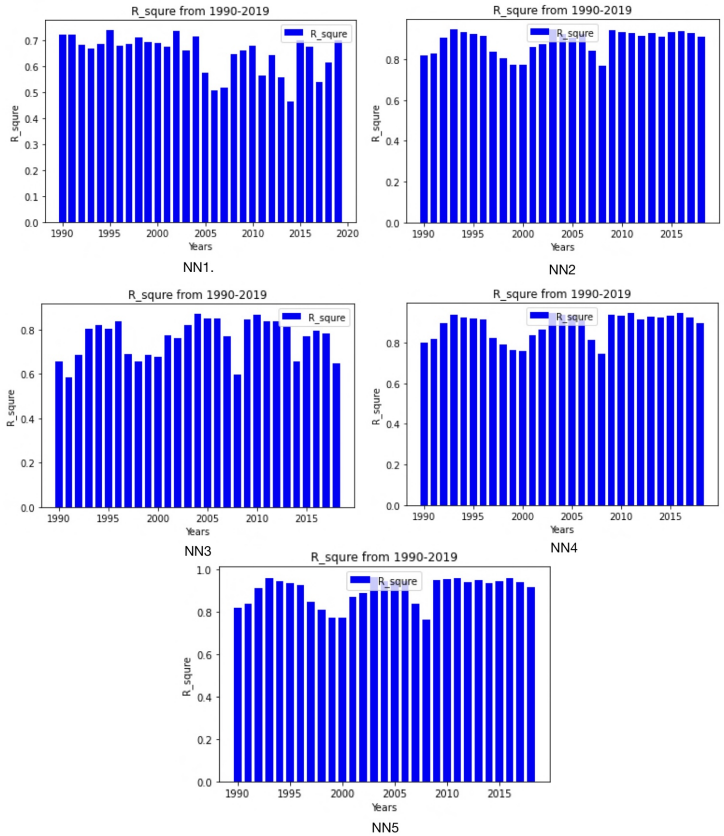

Figure: R-square scores for NN1-5(1990-2019)

Variable Importance

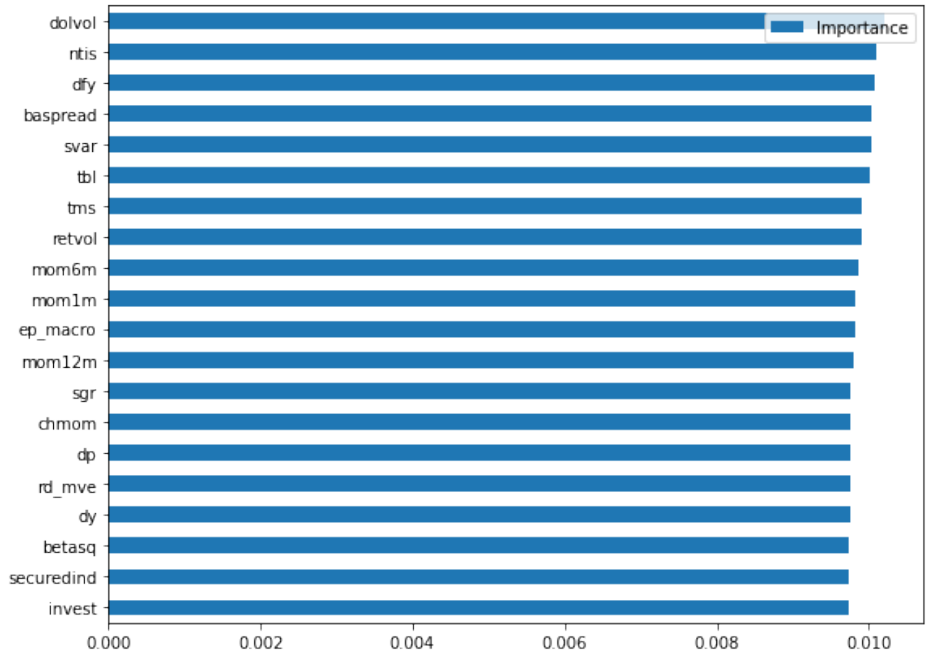

Figure: Variable Importance for NN1-5

Conclusion

Result Analysis

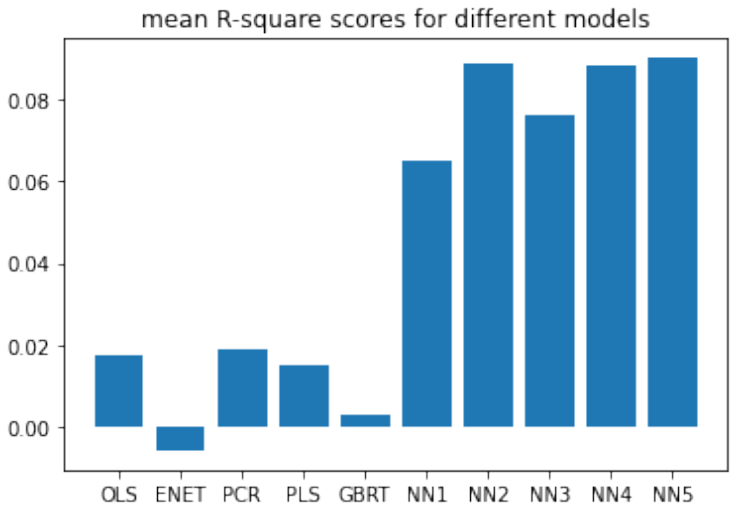

Figure: Mean R^2_{oos} comparison of different models

As expected, Neural Networks perform best among all models. PCR, OLS-3+H, and PLS are also effective in this task, but their scores are much smaller than NN1-5. ENet has the worst R-square score, which means that it is not effective in this prediction task.

Conclusion and future Improvements

We have also obtained several interesting and helpful observations:

- OLS-3 outperforms the OLS model, indicating that we can **averages out noise by reducing the number of predictors**.
- PLS and PCR can help with dimension reduction and **pick out several more important components**.
- Linear interactions between predictors is not enough. Non-parametric models such as neural networks can better **boost performance by adding nonlinear interactions**.
- **Shallow neural networks seems to outperform** deep ones. It may be due to insufficient data and large proportion noise.
- We discover in general macro predictors like **dfy and tbl** have larger impact on the prediction. For the firm predictors, **momentum variables such as mom6m and mom1m** are more influential.

Future Improvements

Future improvements could be **a combination of linear interactions and nonlinear ones**. We can first select several important components to reduce dimension, and then use more powerful neural networks to make final predictions.

Thank You!